Eyes-Free Environment Detection for Visually Impaired People

Spencer Ng, Joyce Passananti, and Hannah Zheng
University of Chicago
CMSC 23400: Mobile Computing
March 23, 2021

INTRODUCTION

Despite recent strides towards inclusion in technology, there is still a large gap between mobile services geared towards the general public and those available to the visually impaired community. Current mobile applications on the market in this realm mostly serve to perform a singular task, which is inefficient and inconvenient for visually impaired users. Our project aims to work towards resolving this issue by creating a centralized, eyes-free application to help blind or low-vision individuals accomplish many of their day-to-day needs.

Our implementation focuses primarily on the identification of groceries and medication through auditory reading of labels and expiration dates. However, the application is designed to be scalable so that its scope can be easily expanded for increased functionality. Features that can be added include integration with It is controlled solely through audio and haptic cues, which our evaluation indicates to be learnable and usable.

BACKGROUND

Over 285 million people across the world are visually impaired, and mobile assistive technologies in phones have become significantly more prominent in the past decade [6]. Since the advent of smartphones, mobile systems for the visually impaired community should also be completely eyes-free to ensure users do not have to interact with small text on a touchscreen, and early wearable solutions have used simulated 3D audio, synthesized speech, and gesture recognition as effective interaction techniques for navigating the environment [5]. However, while these navigation and screen reader systems are useful to visually impaired users, many have expressed concerns over discreetly using them in front of others, self-consciousness when speaking aloud or hearing audio feedback from phones, and having the ability to use them when one hand is holding an object. Studies also indicated that visually impaired users prefer to use only their mobile phone compared to having additional wearables [12].

Our environment detection system thus seeks to have an interface that exclusively targets mobile phones, relying only on common sensors accessible by all users. It also can be operated with one hand, reduces the degree of motion required by input gestures, and minimizes audio input and output. Moreover, haptic feedback replaces visual and auditory output when indicating that user input is received, as past experiments have demonstrated how it supersedes the need to glance at text on screens [4]. Haptic interactions are also preferred to create micro-level prompts and responses that require immediate action, whereas audio interaction is more easily misunderstood and better suited for higher-level instructions that require more detail than a fixed number of patterns [15].

Prior work in mobile environment detection for visually impaired people include hardware-based text reading systems utilizing text-to-speech and off-the-shelf Optical Character Recognition (OCR) methods [1, 10], in addition to object detection and recognition methods on smartphones [13, 11]. However, the former do not have the flexibility of being on mobile phones, and existing object recognition methods have the limitation of needing to create a database of exact image matches to perform keypoint extraction locally rather executing general serverside feature extraction. Object detection methods also require users to shake the object of interest, which lacks the discretion that visually impaired users desire. Our work packages the intuitive gesture and audio-haptic interfaces for user interaction alongside the environment detection features of the aforementioned systems to create an eyes-free application that is both easy to use and performs accurate classification.

Finally, new design paradigms in assistive education technology and home automation indicate a desire for audio-based interfaces that combine multiple functionalities, rather than discrete functions in various prototypes. Systems should also be simple, scalable, and customizable on a user level, dependent on the user's preferences and degree of disability [9, 8]. Our design seeks to do this by providing modular audio-based commands that could be easily expanded upon, multiple functionalities using either the front or rear-facing camera, and flexible haptic feedback alongside or in lieu of audio prompts.

APPLICATION DESIGN & IMPLEMENTATION

Our application was developed with the Expo platform and React Native framework, using libraries and sensors compatible with both Android and iOS: the microphone, accelerometer, and camera. Our backend is set up with Google Cloud, leveraging Google Cloud APIs for object identification, speech-to-text, text recognition, and text-to-speech.

Expo Application

The application is made up of a simple UI design, alternating between a home screen and a camera display. Upon first use, the user is prompted both visually and auditorily to allow for camera and microphone permissions, which is followed by an auditory explanation of the instructions for how to use the app. The home screen is shown by default, and when the camera screen is inactive for more than 8 seconds, the display

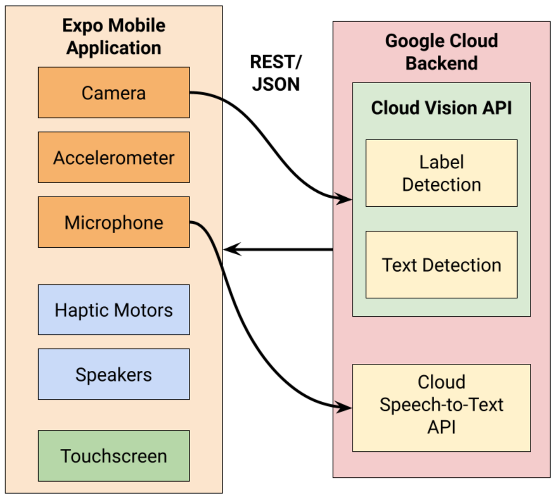


Figure 1. System design block diagram with mobile phone input sensors (orange), device output methods (blue), and cloud-based endpoints (yellow).

is reverted to the home screen in order to minimize battery consumption.

The user can indicate for the application to begin listening for commands by either shaking the phone or double tapping the screen, which allows for an intuitive and flexible trigger. We made sure to implement significant flexibility with the 'shaking' trigger using the accelerometer sensor from the phone, so that each user would be able to shake in their desired direction or speed and be able to activate the listening functionality of the application.

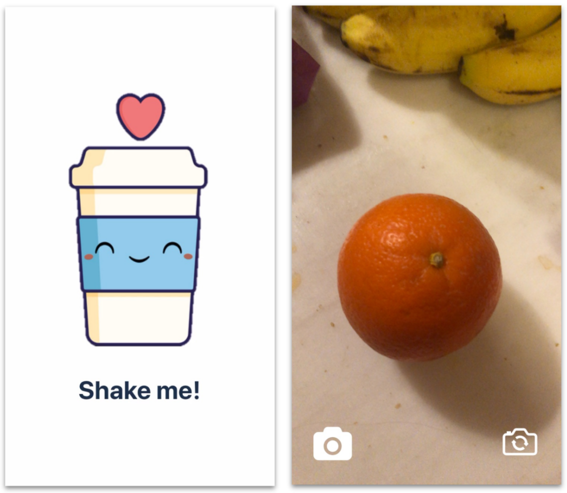


Figure 2. The default home screen (left) and the camera display (right) of the application.

Upon activation of the listening trigger, the basic commands that the user can use include: flipping the direction of the camera, identifying the current direction of the camera, reading text, classifying objects, and repeating the instructions. When the command has been identified, the phone will vibrate to signal that the command is being processed and executed.

The two primary actions among these commands are the reading of text and classification of objects from the camera. When the user shakes the phone or double taps the home screen, the application will switch to the camera display. Then, if the user prompts for either reading text or identifying an object, the app takes a picture and sends the picture to the Google Cloud backend for processing. The phone will pulse after taking the picture, in order to indicate to the user that the image has been taken and is being processed for recognition. Upon successful recognition of the text/image, an automated voice will utilize the text-to-speech API to announce the processed text or image, thus communicating the desired information to the user.

Google Cloud Integration

The application leverages Google Cloud APIs for speech-to-text, object identification, text recognition, and text-to-speech.

We first use speech-to-text in order to parse the commands inputted by the user. Once we have scanned the words in the command for the trigger words, we perform the desired action. We implemented recognition of synonyms in the commands in order to provide the user with greater flexibility in vocabulary. For example, users can flip the camera by either saying "flip" or "turn". Since our commands are designed to be relatively unique from each other, we were able to implement the recognition of synonyms without overlap between different action commands.

When the action desired is object identification, we take a picture using the camera, then send the picture into the object identification API through the appropriate REST endpoint. The API then returns a JSON array of possible identifications. We initially would simply select the first item in the JSON array as the identified object. However, upon performing basic testing, we found that many objects were identified to be broad categories such as "greens" or "food". In order to establish a finer level of granularity in the identification, we implemented an algorithm that would parse out vague words and return more specific identifications from the JSON array. We found that this allowed us to have much more success in identifying objects, with the level of specificity we envisioned.

When the action desired is text identification, we similarly take a picture using the camera, then send the picture into the text identification API through the appropriate REST endpoint. The API then returns a string of the detected text from the picture.

We use the text-to-speech API to verbally identify the contents of the picture taken by the user. When the object or text identification API returns with the proper label, we pass the label into the text-to-speech API so that an automated voice can read the identification aloud to users.

Design Considerations

Battery Consumption

A vital part of creating a successful application is minimizing battery consumption. We evaluated the different components of our application in order to distinguish what the most significant battery drains would be. We determined that the camera display, microphone usage, and accelerometer readings created the largest energy costs.

In order to reduce these costs, we set timeouts for both the camera and microphone to be used. When the user has not made any interaction with the camera display for longer than 8 seconds, the app switches back to the default home screen to minimize amount of time spent displaying the camera. In addition, when the user is inputting voice commands, the app uses the microphone to listen for only 5 seconds before processing the commands. This allows the user enough time to say one of the trigger words in the commands while minimizing battery power consumption

We also implemented duty cycling with the accelerometer in order to avoid constantly reading from the accelerometer - this additionally helped reduce the energy cost from the application, while still maintaining accurate functionality in detecting shaking of the phone.

User Privacy

We wanted to consider any possible user privacy issues in designing our application. The primary concern would be in the requirement for users to interact with the app through voice commands and auditory feedback. However, given that visual cues are not effective for visually impaired users, there seemed to be no other option for communicating with the user besides through audio.

Users concerned with privacy could use headphones so that the output audio from the application would remain confidential to just them. This would allow outside observers to only hear the commands spoken by the user, which are discrete and nonspecific, while the object and text identification would be kept private to just the user. With this being a convenient workaround in offsetting user privacy concerns, we felt sufficiently justified that there were no significant breaches of the user's privacy in using the app.

Another possible concern is that the identified objects and text could be stored, thus creating exposure in the user's privacy. However, we do not store the information in the app - once the identification information is returned from Google Cloud, read aloud, and shown on the screen, when the timeout period passes, the information is no longer stored anywhere on the application. This means that the user should feel no concerns with their private information being recorded or exposed.

Latency

Backend server latency has a high impact on the user experience of applications, and high latency decreases the likelihood of users continuing to use a platform [2]. Our design thus seeks to minimize the delay between when the Expo application sends encoded image or audio data to the Google Cloud server and when it displays a response to the user.

To help achieve this, we attempted to minimize the amount of data sent to the cloud for processing. We perform compression on images captured from the camera before they are sent to the server for classification. We experimented with various compression levels, such that the application has an appropriate tradeoff between classification accuracy and response latency.

Moreover, audio is recorded with a frequency of 44.1 kHz and a bitrate of 128 kbps using only one audio channel. We also compress audio through the Advanced Audio Coding (AAC) standard on Android phones. This was sufficient for achieving high accuracy in speech recognition while minimizing the size of the data packets sent to the server.

Because maintaining the visibility of the system's status while minimizing audio clutter is a key design principle of mobile computing applications [3], we opted to output a short haptic vibration to confirm that the phone was processing audio commands after recording, while a similar vibration and camera noise indicated that an image was taken and being processed. This set up user expectations that the system would output the result of image recognition or speech detection approximately one second later.

Input/Output Methods

In early stages of testing we noticed two areas of weakness with regard to user input: 1) users did not always remember commands clearly and 2) commands were not recognized with 100% accuracy.

To address the first issue, we implemented recognition of synonyms for voice commands, for example allowing users to flip the camera by either saying "flip" or "turn". The most concerning issue with the second issue was the wait period between the user giving a command and realizing it had not been recognized. This confusion is due to the process time of image labeling for read/identify commands, as even when recognized there is often a long pause before output is ready. To fix this we implemented haptic feedback on recognition of voice commands, so the user feels a vibration that signifies an image has been captures and is being analyzed.

To maintain consistency with our goal of a completely eyes free interface, we chose to convey all output verbally in addition to labels on the screen. This includes a "listening" prompt once the user shakes the phone as well as text to speech translation of all image and text labels.

Notifications accompanying the output are through haptic feedback, alerting the user when results have been processed and on any state change such as flipping the camera or transitioning between listening modes.

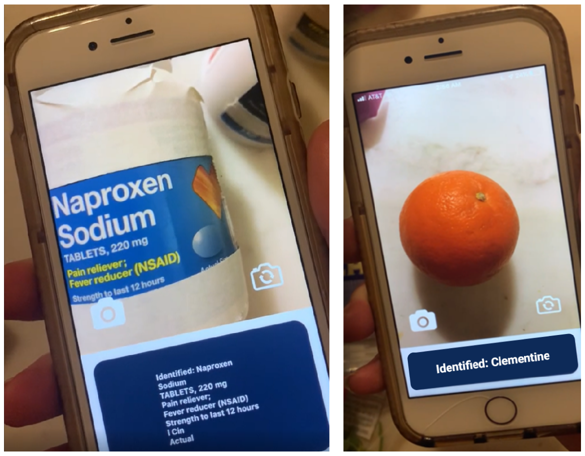
EVALUATION & RESULTS

In order to mimic the limitations of visually impaired users, we evaluated our application by blindfolding unfamiliar test subjects and having them try to perform identification tasks with the app. Our tests showed positive results, with indications of easy discoverability and successful functionality.

Eyes-Free Ease of Use

To evaluate the general usability we tested the app on 9 different users. For each procedure the participant was introduced briefly to the service, then shown an example of it being used. Following a brief period in which they were free to familiarize themselves with voice and movement controls, we had participants blindfold themselves to test it more effectively. While testing in a grocery store was not possible, in each scenario a shelf with a variety of food was set up to mimic such a setting. The testing consisted of two sections: one for identifying objects and the other for reading text.

First participants were asked, using any method, to identify three distinct objects which most did intuitively through the "identify" command. In one situation the "read" command was attempted, which proved difficult at first until the participant chose to flip the camera and hold the object above the phone, decreasing background noise. Within the first 10 attempts all users were able to distinctly identify 3 objects, with an overall accuracy of 76.5% image recognition.



```
Identified: Naproxen
Sodium
TABLETS, 220 mg
Pain reliever;
Fever reducer (NSAID)
Strength to last 12 hours
I Cin
Actual
```

```
Identified: Clementine
```

Figure 3. Identification of medicine bottle labels (left) and grocery items (right). Note that the text labelled on the bottom is for supplementary purposes to confirm accuracy of identification, as the text would not be easily visible for visually impaired users.

Next participants were asked to identify 2 objects by reading the label out and find the expiration date of another. This proved much more difficult than the previous test as writing in the background was also detected and read out. 7 out of 9 participants chose to flip the camera to better read objects they'd removed from the shelves. While this proved to be an effective solution, an issue we noticed was that other objects were occasionally knocked askew or off the shelves in the process. Text recognition proved to be very accurate with the main source of error being crinkled or folded bags that hid lettering, however by holding objects up the weight of the bags smoothed them out.

Identifying expiration date was the hardest goal for participants to achieve by far, as in general there is no standardized placement for all objects and the date is easily lost among other packaging text. 5 out of 9 participants achieved success with this task by feeling around for bottles on which they expected the expiration date to be on the top. As that was the example given to them while introducing the feature, it is hard to measure how intuitive this approach was, yet it still proved effective.

Another aspect we tested on is whether or not users remembered the list of available commands and could recall them to double check camera direction, flip camera, or read out instructions in addition to reading and identifying objects. All users performed well for this criteria and all remembered the commands, with some knowing the variations of the commands as well (i.e. "turn" can be used instead of "flip" for the same result).

System Latency

The Google Cloud APIs are a critical component of the application's functionality, such that significant latency could serve as a bottleneck to the user experience of our system. We thus measured the latency of each call to the Cloud Vision (aggregating both label detection and text detection) and Cloud Speech-to-Text APIs when evaluating our system, with results in Table 1.

API Endpoint	Median (ms)	$P_{95\%}$ (ms)
Cloud Vision ($n = 187$)	1057	1997
Speech-to-Text ($n = 135$)	624	1006

Table 1. Median and 95th percentile response latency of Google Cloud services during evaluation.

Prior research has indicated that response times around 500 ms are often unnoticeable to users, though users tend to feel delays in software applications after 1000 ms of latency [2]. Combined with haptic confirmation that voice commands are being processed, we believe that the median response time of the Speech-to-Text API (624 ms) does not significantly affect the user experience.

However, the Cloud Vision API has a median latency of 1057 ms and a 95th percentile latency of 1997 ms, which was generally noticeable during our evaluation studies and should be reduced in the future to improve the responsiveness of the system. This could be done through increased image compression before sending images to the evaluation server or device-based feature extraction (e.g. local object and text detection with server-side classification) to crop out unnecessary elements in images [7]. In addition, if this system was specialized to certain retail locations with a more limited list of items for classification, transfer learning models could be deployed to edge-based Internet of Things devices located within stores to provide more relevant results to users while minimizing the cost of sending data to a remote server and classifying images [14].

CONCLUSIONS & FUTURE WORK

A key area for improvement in the application is in providing users with a sense of spatial awareness. Visually impaired users could struggle to correctly point the camera at the intended object, so it would be helpful to implement functionality to guide the camera movement. When an insignificant identification is made, the application could inform the user of a possible misdirection and suggest a direction to move in through haptic or audio feedback. The lack of spatial awareness presents a significant issue that visually impaired users could face, so implementing a feature such as the one described could greatly improve the usability of the application.

There are many ways in which the features of our application can be expanded for greater functionality and centralization of common tasks for visually impaired users. The mobile app could provide notification summaries from messages or social media platforms such as Instagram and Facebook; it could also be integrated with Siri or Google Assistant for easy activation. Features could also be added to read a grocery list to users while shopping or calculate a tip from scanning a receipt.

Ultimately, we believe that our application provides a promising platform for allowing people in the visually impaired community to have greater convenience and ease in performing many of their day-to-day tasks. We hope that with the developing advances in technology, underrepresented groups such as visually impaired people can receive more much-needed aid so that any deficiencies perceived in their quality of life due to their disabilities can be supplemented by helpful technology similar to the application we have developed.

REFERENCES

[1] R. Ani, E. Maria, J. J. Joyce, V. Sakkaravarthy, and M. A. Raja. 2017. Smart Specs: Voice assisted text reading system for visually impaired persons using TTS method. In *2017 International Conference on Innovations in Green Energy and Healthcare Technologies (IGEHT)*. 1–6. DOI: http://dx.doi.org/10.1109/IGEHT.2017.8094103

[2] Ioannis Arapakis, Xiao Bai, and B. Barla Cambazoglu. 2014. Impact of Response Latency on User Behavior in Web Search. In *Proceedings of the 37th International ACM SIGIR Conference on Research Development in Information Retrieval (SIGIR '14)*. Association for Computing Machinery, New York, NY, USA, 103–112. DOI: http://dx.doi.org/10.1145/2600428.2609627

[3] Enrico Bertini, Silvia Gabrielli, Stephen Kimani, Tiziana Catarci, and Giuseppe Santucci. 2006. Appropriating and Assessing Heuristics for Mobile Computing. In *Proceedings of the Working Conference on Advanced Visual Interfaces (AVI '06)*. Association for Computing Machinery, New York, NY, USA, 119–126. DOI: http://dx.doi.org/10.1145/1133265.1133291

[4] Stephen Brewster, Faraz Chohan, and Lorna Brown. 2007. Tactile Feedback for Mobile Interactions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '07)*. Association for Computing Machinery, New York, NY, USA, 159–162. DOI: http://dx.doi.org/10.1145/1240624.1240649

[5] Stephen Brewster, Joanna Lumsden, Marek Bell, Malcolm Hall, and Stuart Tasker. 2003. Multimodal 'eyes-Free' Interaction Techniques for Wearable Devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '03)*. Association for Computing Machinery, New York, NY, USA, 473–480. DOI: http://dx.doi.org/10.1145/642611.642694

[6] Lilit Hakobyan, Jo Lumsden, Dympna O'Sullivan, and Hannah Bartlett. 2013. Mobile assistive technologies for the visually impaired. *Survey of Ophthalmology* 58, 6 (2013), 513–528. DOI: http://dx.doi.org/https://doi.org/10.1016/j.survophthal.2012.10.004

[7] J. Hauswald, T. Manville, Q. Zheng, R. Dreslinski, C. Chakrabarti, and T. Mudge. 2014. A hybrid approach to offloading mobile image classification. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 8375–8379. DOI: http://dx.doi.org/10.1109/ICASSP.2014.6855235

[8] Rabia Jafri, Asmaa Mohammed Aljuhani, and Syed Abid Ali. 2015. A Tangible Interface-based Application for Teaching Tactual Shape Perception and Spatial Awareness Sub-Concepts to Visually Impaired Children. *Procedia Manufacturing* 3 (2015), 5562–5569. DOI: http://dx.doi.org/https://doi.org/10.1016/j.promfg.2015.07.734 6th International Conference on Applied Human Factors and Ergonomics (AHFE 2015) and the Affiliated Conferences, AHFE 2015.

[9] Barbara Leporini and Marina Buzzi. 2018. Home Automation for an Independent Living: Investigating the Needs of Visually Impaired People. In *Proceedings of the 15th International Web for All Conference (W4A '18)*. Association for Computing Machinery, New York, NY, USA, Article 15, 9 pages. DOI: http://dx.doi.org/10.1145/3192714.3192823

[10] Z. Liu, Y. Luo, J. Cordero, N. Zhao, and Y. Shen. 2016. Finger-eye: A wearable text reading assistive system for the blind and visually impaired. In *2016 IEEE International Conference on Real-time Computing and Robotics (RCAR)*. 123–128. DOI: http://dx.doi.org/10.1109/RCAR.2016.7784012

[11] K. Matusiak, P. Skulimowski, and P. Strurniłło. 2013. Object recognition in a mobile phone application for visually impaired users. In *2013 6th International Conference on Human System Interactions (HSI)*. 479–484. DOI: http://dx.doi.org/10.1109/HSI.2013.6577868

[12] Hanlu Ye, Meethu Malu, Uran Oh, and Leah Findlater. 2014. Current and Future Mobile and Wearable Device Use by People with Visual Impairments. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. Association for Computing Machinery, New York, NY, USA, 3123–3132. DOI: http://dx.doi.org/10.1145/2556288.2557085

[13] C. Yi, Y. Tian, and A. Arditi. 2014. Portable Camera-Based Assistive Text and Product Label Reading From Hand-Held Objects for Blind Persons. *IEEE/ASME Transactions on Mechatronics* 19, 3 (2014), 808–817. DOI: http://dx.doi.org/10.1109/TMECH.2013.2261083

[14] K. Zhang, S. Leng, Y. He, S. Maharjan, and Y. Zhang. 2018. Mobile Edge Computing and Networking for Green and Low-Latency Internet of Things. *IEEE*

Communications Magazine 56, 5 (2018), 39–45. DOI: http://dx.doi.org/10.1109/MCOM.2018.1700882

[15] Xiaochen Zhang, Xiaoyu Yao, Yi Zhu, and Fei Hu. 2019. An ARCore Based User Centric Assistive Navigation System for Visually Impaired People. *Applied Sciences* 9 (2019), 989.

GROUP MEMBER CONTRIBUTIONS

Spencer Ng worked on integrating the Google Cloud API components into the application functionality, while Joyce Passananti established the user interface components of the app. Hannah Zheng worked on application debugging and testing.