SYDE 462 – Voice Assistive System for Recently Visually Impaired Individuals in a Kitchen Setting

Team 13: Anshuman Patnaik, Linda Wang, Edrick Wong, Justin Wong

Abstract – MIRU aims to be a natural-feeling voice assistive system that operates in the kitchen to help visually impaired individuals locate displaced objects in the kitchen. Continuing from the developments in SYDE 461, the team continues the parallel development of the user interface and object detection components of the project. For the user interface, feasibility testing was done with different voice assistants to determine the best solution in the context of the project. In terms of the object detection, the MobileNet SSD model that was trained on the MS-COCO dataset was retrained in attempt to have a more accurate detection specifically tailored to kitchen objects. On top of these efforts, deployment strategies for the end-to-end system were investigated. From these three components, the full end-to-end system is developed, leveraging Dialogflow and the Google Assistant platform for the front-end solution and MobileNet SSD for a custom object detection model that is optimized for the kitchen setting. The following paper highlights the outcomes of the project, summarizes relevant findings, and presents conclusions and next steps for the project.

Keywords – Technologies for Visually Impaired, Indoor Navigation, Object Detection

INTRODUCTION

The project focuses on improving the kitchen experience for visually impaired individuals. Specifically, the target users are blind individuals who once had vision but are now visually impaired. The target group lives in an average North American household shared with other tenants. The household consists of a normal kitchen and the users dislike common aids that would label them as blind individuals. Lastly, they prefer to perform tasks individually, seeking as little help as possible from other household members. Key findings from user research will be discussed in the summary of engineering analysis and design methods.

TapTapSee and Microsoft's SeeingAI are existing mobile phone application solutions aimed to help the visually impaired navigate their surroundings better [1, 2]. Microsoft SeeingAI is more powerful of the two and is able to read short snippets of text, recognize objects and people, perform depth estimation, and describe scenes in front of a user [2]. In the mobile application, the user points the phone at their target and presses a button on their phone to take a picture. After the application finishes analysing the photo it verbalizes information about the photo. However, both mobile applications require the use of a hand and interviewees indicated that this is not practical in the kitchen.

The proposed solution uses existing technologies including a USB camera, Google Home, Google Assistant, and Tensorflow framework. In a nutshell, a user can ask Google Assistant, an interactive intelligent assistive system, for the location of an object. Google will fetch visual

information from the camera, send it to a cloud-based service involving object recognition and location determination via a trained object detection model. Finally, inquiries are surfaced back to the user via the Google Home as audio feedback. All the technology needed to capture visual information, recognize and determine objects via computer vision, interpret speech commands, etc. is available but will be integrated and packaged together into a natural and empowering kitchen experience for the user.

PROJECT SCOPE AND OBJECTIVES

User interviews with visually impaired individuals and staff that work directly with these individuals from institutions such as the Canadian National Institute for the Blind (CNIB) and American Foundation for the Blind (AFB) presented three main findings that helped solidify the problem space and establish requirements for the solution.

First, the majority of blind individuals fall into two groups: individuals that openly accept any adaptation that would improve their daily experience or individuals that reject noticeable adaptations that would clearly distinguish them from sighted individuals. This project's target users identify with the latter group. It was also revealed from the interviews that many blind individuals who live with sighted individuals do not have kitchen adaptations or blind kitchen aids because other household members tend to find these adaptations aesthetically displeasing or annoying. This suggested that the project solution should not make the user feel visually impaired and it should not negatively affect the aesthetics or feel of the kitchen.

Secondly, few target users have kitchen aids like the liquid level detector [3]. They were considered expensive for a solution that only solves one particular problem, often required maintenance (e.g. replacing batteries), or were aesthetically displeasing to their sighted housemates. As such, some of these interviewed blind individuals reported that they often resort to asking family members for assistance even though it is against their preference as they find it annoying themselves to constantly rely on others. However, without assistance, they found simple tasks could require heavier cognitive resources or effort, which could be exhausting, annoying and potentially dangerous if they were not careful. This suggested that the solution had to be accessible, affordable and should reduce the cognitive workload of a user operating in the kitchen.

Lastly, most of the interviewed users stated that they did not struggle with locating objects so long as they were placed in the same location. This way, their cognitive map of the kitchen does not frequently change. Often times, however, other sighted individuals (e.g. family members or visiting friends) misplace kitchen objects, which makes it very difficult for blind individuals to locate a specific item again as their cognitive map of the kitchen is not aligned with reality. This could be especially dangerous in common kitchen situations such as the case where glass might have been shattered and the user is searching for an item that might have been accidentally knocked down. This suggested that the solution should help users update their mental model of the kitchen space.

SCOPE

To address the full scope of the problem space, the project was broken down into three major phases. The goal of Phase 1 was to determine the feasibility of different object identification and location solutions as well as the feasibility of different audio feedback solutions. Phase 2's objective was to separately design and validate the object detection system on a localized system as well as designing and validating the types of audio feedback. Finally, the goal of Phase 3 is to integrate and validate the object recognition and audio feedback subcomponents into a natural-feeling end-to-end solution.

Phase 1 and 2 were completed during SYDE 461 and Phase 3 was completed in 462. To tackle the object identification and location problem of Phase 1, two distinct solution tracks are proposed and evaluated: computer vision object detection and radio-frequency identification (RFID) sensor pairs. For the audio feedback subsystem, the approaches proposed and evaluated include speech, sonification or a combination of both, as well as the use of single or multiple audio sources on either the Google Assistant or Amazon Alexa. Through literature reviews, user testing and low-level prototyping, it was determined that computer vision was going to be explored as the tool of choice for object recognition whereas allocentric speech feedback on a single audio source would be best suited for audio feedback. To ensure timely completion of Phase 2, the scope was simplified to analysing a small subset of kitchen items in a restricted environment, for example, a bottle on a table top.

SUMMARY OF ENGINEERING ANALYSIS AND DESIGN METHODS

CHOOSING AUDIO FEEDBACK

For Phase 2, it is important to validate the instructions that are provided to the users; that is, the instructions are understandable and accurate but do not introduce extraneous cognitive loads such as signal fatigue or split attention effects. Fundamentally, the difficulties of search tasks for blind individuals is a lack of level one perceptual situational awareness, which is, as described by Endsley, the ability to perceive elements in the current situation [4]. If a user were to enter a space where their cognitive map of the room is not congruent with the present state of the room, it is very difficult for them to perceive details about the location of certain objects in the room. To establish the appropriate audio feedback, the different methods tested were allocentric, egocentric and allocentric paired with sonification. Allocentric feedback is used when the location of an object is given with respect to another object. Egocentric feedback is used when the location of an object is given with respect to the subject [5]. Sonification is the use of non-speech audio which involves the use of earcons which are "abstract, synthetic sound patterns with variations on rhythm, pitch, timbre, register and dynamics" to deliver information [6]. A single audio source and simulated multiple audio sources to heighten spatial awareness were also tested.

The Sustained Attention to Response Task (SART) engineering evaluation model was used to measure the effectiveness of the different alternatives of audio feedback to improve situational

awareness (and to ultimately decide on an alternative). With SART, participants rate their perception of demand and supply of their cognitive resources and their understanding of the situation after they have completed the task [7]. Results from SART are then used to calculate the user's situational awareness during the task.

User tests were performed by blindfolded sighted individuals mainly because the team was not able to get ethics clearance in time. However, several similar research studies such as the University of Lisbon's research in blind people's information scanning with concurrent speech as well as Stanford University's research in converting real-time visual recognition results to three-dimensional (3D) audio also performed tests with blindfolded, sighted individuals as a starting point [8, 9]. During SART, the participant was in a familiar kitchen but the testers moved an object (e.g. a chair) to another part of the room so that it was different compared to the user's original cognitive map of the kitchen. The kitchen had a speaker(s) placed in an audible location. First, the user asked for the location of the desired item and electronic audio feedback was returned to the user with the item's location. Then, the participant either attempted to locate the object or asked the system more questions. To mock the dynamic conversations that occurred between the user and the system, testers typed in dynamic text responses to a text-to-speech application and played them on the speaker. The task was completed when the user either found the desired object or if they decided to give up. A/B testing between pure speech and a combination of speech and sonification auditory feedback was used to determine which was the better alternative. A/B testing between single and multiple audio sources will also be used to determine the best alternative.

Key results from testing are shown in Appendix A. A combination of allocentric and sonification feedback (without the outlier cases) led to the least number of questions asked, shortest time to find the target and highest measured situational awareness. Users felt that allocentric feedback updated their mental map most effectively and efficiently because it positions the target object relative to stationary landmarks that the user is already well-aware of. Sonification was useful to pinpoint the exact location of the target. However, building an engine to generate dynamic sonification feedback required significant time and research. Further, the difference between pure allocentric speech and this combination were not too drastic. As a result, this alternative was deemed out of scope and the team decided to move forward with a pure allocentric audio feedback instead.

CHOOSING THE USER INTERFACE

During interviews, users expressed a desire for smart home systems that could assist them with tasks in the kitchen. Rather than build a new smart assistant from scratch, the team decided to build an application for an off-the-shelf smart assistant. The most adopted smart home assistants that are cheap and support a large development community to use are the Amazon Alexa and Google Home Assistant [10, 11]. Ideally, the solution will support both the Google Home and the Amazon Alexa but the team chose to move forward with one of the two for the purposes of developing an MVP. From the user interviews, the group also noted that several users were already leveraging Alexas and phone assistants in the kitchen as timers and other simple aids. As a result, it was determined that the existing presence of these items with

users could yield a better adoption of the product in the long run due to a lower perceived cost and users' established familiarity with a major component of the system.

A weighted decision matrix was used to determine which would be the best alternative for the project. The six criteria were development speed, cost to purchase, integration and hosting, development collaboration, cost to use, and ecosystem. The matrix can be found in Appendix B. Identical basic proof-of-concept applications were prototyped on both platforms to determine values for the decision matrix. From the decision matrix results, it was evident that the Google Assistant was the better alternative so it was used as the front-end client for the remainder of the project.

CHOOSING OBJECT DETECTION MODEL

One possible method of computationally identifying objects is through computer vision. Object detection in the context of computer vision is concerned with two major problems, localization and identification. Localization refers to knowing where in the frame the object is whereas identification refers to knowing what the object is [12]. Mobilenet SSD, whose architecture is shown in Appendix C, was chosen as the detection network for the backend of the system because of its efficiency with respect to size and speed [13]. SSD is a method for detecting objects that encapsulates all computation into a single network, making it easy to train with standard backpropagation algorithms and straightforward to integrate [14]. Additionally, for inferencing, shown in Appendix D, SSD maintains a better mean average precision (mAP) than other state-of-the-art object detection models while running in real-time on a GPU [14]. Furthermore, as shown in Figure 1, MobileNet SSD achieves comparable results with other similar models while maintaining a good tradeoff between accuracy, for both localization and classification, and computational complexity [13].

Framework Resolution	Model	mAP	Billion Mult-Adds	Million Parameters
SSD 300	deeplab-VGG	21.1%	34.9	33.1
	Inception V2	22.0%	3.8	13.7
	MobileNet	19.3%	1.2	6.8
Faster-RCNN 300	VGG	22.9%	64.3	138.5
	Inception V2	15.4%	118.2	13.3
	MobileNet	16.4%	25.2	6.1
Faster-RCNN 600	VGG	25.7%	149.6	138.5
	Inception V2	21.9%	129.6	13.3
	Mobilenet	19.8%	30.5	6.1

Figure 1: COCO object detection results comparison (mAP reported with AP at IoU=0.5:0.05:0.95 [13]

TRAINING OBJECT DETECTION MODEL

The MobileNet SSD network was trained on 1387 training images from a custom dataset of four types of kitchen items: bottles, cups, bowls, and kettles. The network was trained once for 700,000 epochs which took approximately 155 hours on a GeForce GTX 980 with fine tuning (ie. the trained MobileNet SSD model listed by the paper was used as the starting weights [14]).

It should be noted that the mAP for 462 test images were taken at 200,000, 300,000, 500,000 and 700,000 epochs.

Most neural networks are trained using loss or objective functions for the network error, traditionally using backpropagation. For MobileNet SSD, the objective function is defined as the weighted sum of localization loss and confidence loss [14]. The total loss was calculated taking every 10,000 epochs during training and is shown in Figure 2. A base learning rate of 0.004 was used which the same as when the network was trained on the MS-COCO dataset [15]. In literature, the learning rate decay is noted to be 0.95 every 800k steps [15]. However, since the loss was oscillating, the learning rate decay was set to 0.95 every 10,000 steps to adjust the learning rate so that there are smaller weight adjustments as the epochs increase.

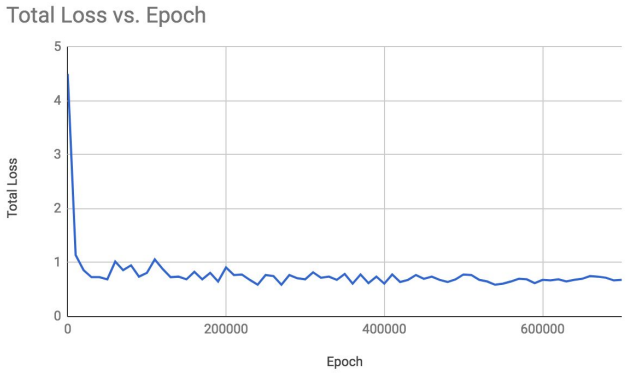


Figure 2: Total Loss versus Epoch

SUMMARY OF SOCIAL, ECONOMIC, AND ENVIRONMENTAL IMPACTS

Looking at the different types of social impacts within the social impact assessment framework, the project is most relevant to lifestyle impacts and quality of life impacts [16]. Social impacts relate to the way individuals interact with their family and friends. From the user interviews that were conducted, one of the biggest issues that visually impaired individuals face is using a shared kitchen space along sighted friends and family members. The goal of MIRU is to bridge the interaction between sighted friends and families and the visually impaired individual. A positive impact of MIRU is that it provides the visually impaired individual an autonomous level of environmental awareness that was not available to them before, and at the same time, leverages a platform that can be used by others for other purposes. On top of this MIRU was designed such that the interactions feel natural and attempts to minimize the amount of attention that the systems brings.

Since MIRU analyses live videos of the kitchen, privacy is an important social issue to consider. The prototype runs the object detection locally, so the video component is not sent online. However, moving forward, the video will need to be processed online for scalability reasons. The privacy policy will have to be extremely clear to the user and no information is being

stored online after analysis. On top of this, another social issue is that not all users will want to adopt this. Insofar as the project aims to minimize the ostracism associated with asking for help, it might still bring unwanted attention.

In terms of economic impacts, the system was intentionally built on the Google Home platform because it is well integrated into households and is a cheap and accessible solution that is flexible and can be updated. This avoids the need to purchase new hardware for new functionality, which both decreases the cost and environmental damage from producing more hardware. However, the downside of this solution is that it has to be constantly powered on as there are service fees associated with the camera system and extra hardware for video processing.

DESIGNED SOLUTION

The designed solution consists of a Google Home as the user interface and an SSD Mobile-net object detection framework. The Google Home accepts a user's request that specifies an object of interest and passes it to the backend object detector component. The object detection framework analyzes the frame passed by the webcam for the purposes of the demo and returns a list of detected objects, bounding boxes for each of the boxes and their associated probabilities. These results from the neural net are used in conjunction with information about reference objects to build an allocentric response for the user. This response is passed back to the Google Home that relays and audio response back to the user. A summarized functional diagram is shown in the following Figure 3, while a detailed functional diagram is included in Appendix E.

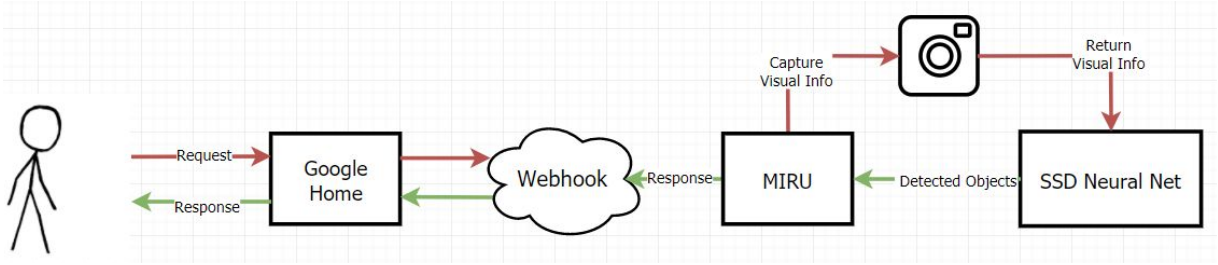


Figure 3: Summarized functional diagram

In addition to the above components, a third component that adds reference objects was also built by the group. From user interviews conducted previously, it was noted that most visually impaired individuals relied on landmarks to help them navigate through their homes. Combined with the decision to provide allocentric feedback to the user, it was determined that reference objects were necessary for the design. Currently, a backend process is used to input these reference objects without a user interface. These reference objects primarily consist of objects within a kitchen that are relatively stationary such as toaster ovens and stoves. As seen in the functional diagram in Appendix E and above, there are several components that make up the end to end application. In the development of the application, the group designed each component as independent processes that communicate with each other using queues [17, 18].

This modularity would potentially allow the group to draw system boundaries anywhere in the application to create a personalized version of the application depending on the user's privacy preferences and cost. For example, using cloud-based message queuing systems such as RabbitMQ, Amazon SQS, or Google Pub/Sub, the group would be able to deploy the object detection components into the cloud [19, 20, 21]. As a consequence, the user would only need to setup a camera that could relay pictures on the frame which is a significantly smaller hardware requirement for the user compared to having the entire application running locally as currently prototyped by the group.

The Google Home development was implemented by the group using existing Google APIs. The neural network architecture and relevant code was used off-the-shelf as included in the main TensorFlow library [15]. The group implemented their own data pipeline modelled after sparse examples to fine tune the existing SSD model to detect specific kitchen objects (bottles, cups, kettles and bowls). Finally the response generation logic that took into account reference objects and detected objects to provide actionable insights to the users was developed by the group.

To train the model, the team developed a suite of scripts to support the data pipeline. The group modelled the data pipeline of existing pipelines used in object detection [22, 23, 24, 25]. The group also collected approximately 800 images in a similar format to an existing kitchen dataset [26]. Leveraging an open source tool to label the images, LabelImg, the group also manually annotated bounding boxes for the images collected [23, 27]. The group also built logic to augment the size of the dataset by randomly horizontally and vertically flipping some images in the dataset and calculating the new bounding boxes [14].

For the purposes of the technical demonstration, a continuous visualization of the object detection framework is also included. The logic facilitating this continuous visualization was primarily from existing visualization applications and was modified by the group to be integrated with the remainder of the application [15].

SUMMARY OF DESIGN VALIDATION

HOME ASSISTANT USER INTERFACE

In order to validate the audio response from the home assistant, user testing of the end to end system was performed to determine two key metrics: situational awareness and emotional design.

To test the SA of the system, a modified version of the SART protocol from previous user testing was used. Using the same SART calculations as mentioned in the Choose Audio Feedback section, it was determined that the SA of participates using MIRU is higher than the SA of the participants not using it. The participants using MIRU had a score of 11.56 whereas the participants not using it had a score of 8.75. On top of this, the time of completion was also

measured. The average time of completion with MIRU is 44 seconds whereas the time of completion without the system is 3 minutes with two participants failing to locate the object.

To test the emotional design of MIRU, a semantic differential survey was given to the participants after interacting with MIRU. A set of six semantic differential pairs with seven gradations were used: Robotic/Natural, Direct/Indirect, Trustworthy/Not Trustworthy, Friendly/Unfriendly, Annoying/Pleasant, Flexible/Inflexible. Based on the positive feature in the pairs, Table 1 is a normalized value of how MIRU performs.

Table 1: MIRU performance based on a set of six semantic differential pairs

Naturalness	Directness	Trustworthiness	Friendliness	Pleasantness	Flexibility
0.64285714	0.80952381	0.80952381	0.78571429	0.70238095	0.60714286

MIRU performs very well in most categories. Naturalness and flexibility are comparatively lower; however, they still scored about 0.5, which is at an acceptable level. In order to improve this, more complicated custom user flows will need to be implemented and tested with users in the future.

OBJECT DETECTION VALIDATION METHODS

When the user asks for where an item is, the object detection model must be accurate so that the user is directed in the right direction. To do this, there must be a high intersection over union and mean average precision. The model must also be fast enough so that the user is not waiting a long time for a response and to create a more natural experience.

The testing for object detection model accuracies for the predicted bounding box accuracy will be evaluated using the Intersection over Union (IoU) method. To use this evaluation, a ground truth bounding box and a predicted bounding box from the model are required [28]. The ground truth bounding box will be a hand-labeled bounding box for where the object in the image is. IoU is calculated by the area of intersection over the area of union of the two bounding boxes:

$$IoU = \frac{A_G \cap A_P}{A_G \cup A_P} \quad (1)$$

where A_G is the area of the ground truth and A_P is the area of the predicted bounding box. MobileNet SSD have IoU values between 0.5 and 0.95 when tested on the MS-COCO [13], which is a good baseline validation for this model.

The testing for the accuracy of object identification will be evaluated using mAP. The average precision is computed as the number of true positive predictions over the number of true positive and false positive predictions for each class [28].

$$AP(c) = \frac{TP(c)}{TP(c) + FP(c)} \quad (2)$$

The mean average precision is the average precision for all classes averaged over all classes [28].

$$mAP = \frac{1}{|classes|} \sum_{c \in classes} AP(c) \quad (3)$$

MobileNet SSD has 19.3% mAP, shown in Figure 1, measured at IoU from 0.5 to 0.95 with 0.05 step sizes when tested on MS-COCO [13], where a higher IoU values mean more accurate bounding box proposals.

Speed is another important component to test for the object detection model. The speed is measured as the duration required for the model to locate and identify objects in an image during test-time [15]. In addition, for the purpose of the continuous visualization, the framerate is important to ensure a smooth presentation.

OBJECT DETECTION VALIDATION RESULTS

During training, mAP values were reported using the test data set at various steps in Table 2. First, the average precisions (AP) were calculated over all the classes for each IoU threshold from 0.5 to 0.95 taking step sizes of 0.05. AP values at IoU of 0.5 and 0.75 are also shown in the table. The AP values increased from 0.5 to 0.75 for each result and the IoU at 0.75 is closest to the mAP for all the steps except 200,000, which shows that 0.75 is a good localization threshold [15]. After the APs are calculated, the mean sof the APs are taken to get the mAP. As shown from the table, mAP increased from 200,000 to 300,000 epochs and then stayed from 300,000 to 700,000 epochs. This is consistent with the total loss graph in Figure 2. Since the model was trained on 4 classes, the mAP values are expected to be higher than the mAP of 19.3% when the model was trained on MS-COCO, which contains 80 classes [13]. Based on the results, the hypothesis was met.

Table 2: mAP reported using testing data set taking mean AP at IoU=0.5:0.05:0.95, and AP values at IoU=0.5 and IoU=0.75

Epochs	mAP(%)	AP at IoU = 0.5(%)	AP at IoU = 0.75(%)
200,000	89.7262	92.2883	92.7004
300,000	96.5052	95.7054	96.1744
500,000	96.5076	95.7054	96.1744
700,000	96.5126	95.7054	96.1744

Appendix F and Appendix G show some examples of correct classifications and misclassifications of the trained model on the test dataset. The white bounding boxes represent the prediction. From the results, it can be seen that there is a large overlap between the ground truth and prediction. For the correct classifications, the model has high confidence for the first prediction and much lower confidence for the four other predictions. On the contrary, for misclassifications, the model has similar confidences for the top two predictions, which are most misclassifying cup and kettle as bottle and bowls as cups. The substantial difference

between the measured mAP and reported mAP in the MobileNet paper is an indication that the model most likely overfits due to the limited dataset size.

The inferencing is on an Intel Core i7-4700MQ 8 core CPU at 2.40GHz. The average runtime of detecting and identifying objects takes about 58.1 milliseconds with a standard deviation of 2.8 milliseconds; the 99th percentile was 70s, calculated for 1000 requests. The frame rate of the visualization is about 11 frames per second.

LIMITATIONS OF DESIGNED SOLUTION

There are limitations relating to both the user interface and the object detection. On the user interface side, two important limitations noted during testing by the users are the inflexibility and robotic feel of the voice response. The inflexibility is because the flow of the conversation relies on the Dialogflow architecture. The robotic feel is limited by Dialogflow's natural language processing and Google Home's voice synthesis services. The types of responses that MIRU can interpret is fundamentally dictated by Dialogflow's ability to parse the input and extract the important points. The naturalness of the voice synthesis is also limited by the abilities of Google Home's provided synthesizers. Another limitation is that the scope of the project was restricted such that the system can only detect one target object. For example, the system cannot handle having two water bottles in the scene. In addition, the group did not actually test the MVP with visually impaired; rather, sighted users were blindfolded.

At a high-level, a key limitation of the design is the critical requirement that the desired object must be visible to the camera. During the initial user testing, it was determined that users also sometimes misplace items within cabinets. Currently, there is no workflow integrated that allows the camera to scan the inside of the cabinets when the user opens it. In addition, the current iteration of the system has no memory; it does not store the results of previous scenes or a consistent working map of the environment. In addition, even though using computer vision for object detection was the paradigm of choice, the group integrated only one neural network architecture (Mobilenet SSD) due to usability and time constraints. Finally, from the testing results described in the validations section and from user testing, it was determined that the trained model overfit the data and would not generalize well to other environments.

CONCLUSIONS AND RECOMMENDATIONS

From the final round of user testing, it can be concluded that the developed system does in fact reduce the cognitive workload and users enjoyed the overall application. While the system scored lower in term sof naturalness and flexibility, they were still at an acceptable level. In addition, the object detection model performed better than expected when imposing a limit on the number of classes detected. While the trained model did overfit the data, the higher accuracy allowed for a reliable MVP that was used to validate the entire system as a whole.

To deal with some limitations on the user interface side, an important next step would be to establish a workflow to deal with multiple detected objects in the scene. For example, if there are two water bottles, the system should be able to have a natural conversation with the user and prompt for distinguishing features between the bottles to guide the user appropriately. In addition, as a more robust test before making any further iterations, the system should be tested on visually impaired individuals.

On the object detection side, it is recommended that a bigger kitchen dataset is produced and used to train the model, as a potential method to reduce overfitting. In addition other popular object detection frameworks such as Faster R-CNN and YOLOv2 should be explored further to yield a better comparison.

In addition, to reduce further errors, increase accuracy and increase the size of the overall dataset, it is recommended that a workflow be developed to allow users to notify the system of identification errors which can be used to retrain the network.

At a high-level user interface perspective, it is recommended that an accompanying application on the phone be supported to allow the users to configure landmarks appropriately. In addition, the depth perception algorithms should be iterated to provide more accurate feedback for the users. It is highly likely that more information about the environment will need to be collected in order to provide more accurate depth perception. Thus, the next iteration should focus on a standardized approach to help the user configure information about the working environment.

ACKNOWLEDGEMENTS

We would like to thank Prof. Alex Wong for his continued support and guidance for our project. We would also like to thank Prof. John Zelek for providing hardware for our prototyping efforts.

REFERENCES

[1] Taptapseeapp.com. (2017). TapTapSee - Blind and Visually Impaired Assistive Technology - powered by CloudSight.ai image recognition API. [online] Available at: http://taptapseeapp.com [Accessed 4 Nov. 2017].

[2] Microsoft.com. (2017). Seeing AI | Talking camera app for those with a visual impairment. [online] Available at: https://www.microsoft.com/en-us/seeing-ai/ [Accessed 4 Nov. 2017].

[3] Indicator, 2. and Indicator, L. (2017). 2 Stage Liquid Level Indicator - FREE Shipping. [online] Rehabmart.com. Available at: https://www.rehabmart.com/product/2-stage-liquid-level-indicator-28843.html[Accessed 4 Nov. 2017].

[4] M. R. Endsley. (2000). Theoretical Underpinnings of Situation Awareness: A Critical Review. Situation Awareness Analysis and Measurement, pp.3.

[5] Nmr.mgh.harvard.edu. (2018). Allocentric vs. Egocentric Spatial Processing. [online] Available at: http://www.nmr.mgh.harvard.edu/mkozhevnlab/?page_id=308 [Accessed 17 Mar. 2018].

[6] G. Kramer, B. Walker, B. Bonebright, P. Cook. (2010). Sonification Report: Status of the Field and Research Agenda. Faculty Publications, Department of Psychology, pp. 1,2.

[7] Endsley, M. R. (2000). Direct measurement of situation awareness: Validity and use of SAGAT. In M. R. Endsley & D. J. Garland (Eds.), "Situation Awareness Analysis and Measurement" (pp. 147-173). Mahwah: Lawrence Erlbaum Assoc.

[8] Guerreiro, J. (2015). The use of concurrent speech to enhance blind people's scanning for relevant information. ACM SIGACCESS Accessibility and Computing, (111), pp.21-22.

[9] R. Jiang et al (2016). Let Blind People See: Real-Time Visual Recognition with Results Converted to 3D Audio. Stanford, pp 5-6.

[10] Voicebot. (2018). Amazon Echo & Alexa Stats - Voicebot. [online] Available at: https://www.voicebot.ai/amazon-echo-alexa-stats/ [Accessed 17 Mar. 2018].

[11] Sarah Perez. (2017). Voice-enabled smart speakers to reach 55% of US households by 2022, says report. [online] Available at: https://techcrunch.com/2017/11/08/voice-enabled-smart-speakers-to-reach-55-of-u-s-households-by-2022-says-report/ [Accessed 17 Mar. 2018].

[12] R. Girshick et al. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. arXiv:1311.2524v5.

[13] A. Howard et al. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. arXiv: 1704.04861v1

[14] W. Liu et al. (2016). SSD: Single Shot MultiBox Detector. arXiv: 1512.02325v5

[15] J. Huang et al. (2017). Speed/accuracy trade-offs for modern convolutional object detectors. arXiv:1611.10012v3.

[16] Comprehensive Guide for Social Impact Assessment. (2006). Centre For Good Governance.

[17] Python. (2018). Multiprocessing - Process-based “threading” interface. [online] Available at: https://docs.python.org/2/library/multiprocessing.html [Accessed 4 Nov. 2017].

[18] Sebastian Raschka. (2014). An introduction to parallel programming using Python's multiprocessing module. Available at: http://sebastianraschka.com/Articles/2014_multiprocessing.html [Accessed 9 Jan. 2017].

[19] RabbitMQ. (2018). RabbitMQ. [online] Available at: https://www.rabbitmq.com/#features [Accessed 26 Feb. 2017].

[20] AWS. (2018). Amazon Simple Queue Service (SQS). [online] Available at: https://aws.amazon.com/sqs/ [Accessed 26 Feb. 2017].

[21] Google Cloud Platform. (2018). Google Cloud Pub/Sub: A Google-Scale Messaging Service. [online] Available at: https://cloud.google.com/pubsub/architecture [Accessed 26 Feb 2017].

[22] Tsung-Yi Lin et al. (2015). Microsoft COCO: Common Objects in Context. arXiv:1405.0312v3

[23] Dat Tran. (2017). How to train your own Object Detector with Tensorflow's Object Detector Api. [online] Available at: https://towardsdatascience.com/how-to-train-your-own-object-detector-with-tensorflows-object-detector-api-bec72ecfe1d9 [Accessed 21 Nov. 2017].

[24] Priya Dwivedi. (2017). Building a Toy Detector with Tensorflow Object Detection API. [online] Available at: https://towardsdatascience.com/building-a-toy-detector-with-tensorflow-object-detection-api-63c0fdf2ac95 [Accessed 8 Jan. 2018].

[25] Daniel Sonntag et al. (2017). Fine Tuning deep CNN models on specific MS COCO categories. arXiv:1709.01476v1

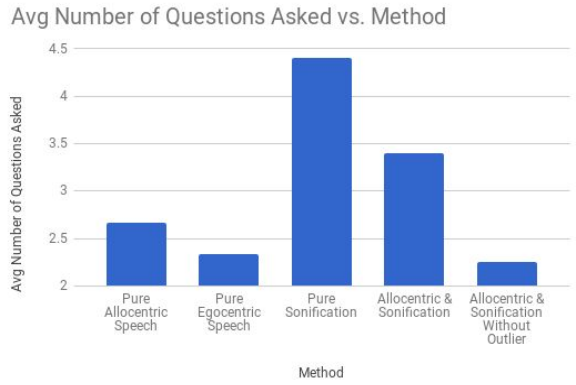
[26] Carnegie Mellon University School of Computer Science. (2011). CMU IKEA Kitchen Object Dataset. [online] Available at: http://www.cs.cmu.edu/~vmr/datasets/ikea_kitchen/ [Accessed 8 Jan. 2018].

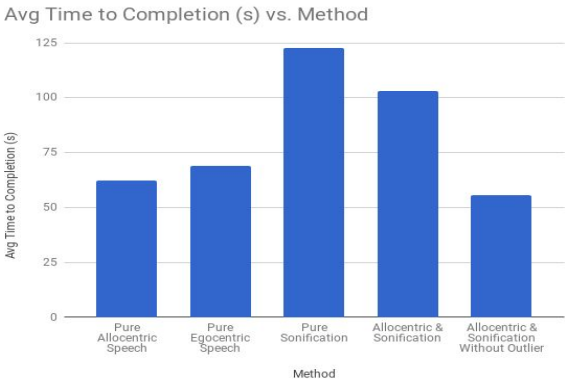
[27] Tzutalin. (2015). LabelImg. [online] Available at: https://github.com/tzutalin/labelImg [Accessed 21 Nov. 2017].

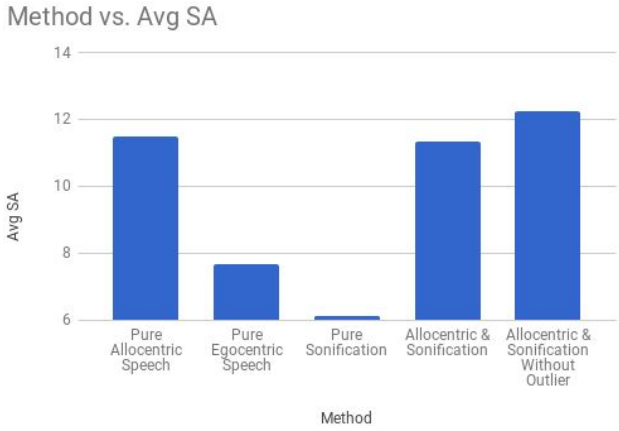
[28] M. Everingham et al. (2010). The PASCAL Visual Object Classes (VOC) Challenge. http://homepages.inf.ed.ac.uk/ckiw/postscript/ijcv_voc09.pdf

[29] R. Munroe. (2006). Filler Art. [online] Available at: https://xkcd.com/157 [Accessed 4 Nov. 2017].

APPENDIX A
AUDIO FEEDBACK TEST RESULTS







APPENDIX B

WEIGHTED DECISION MATRIX FOR GOOGLE ASSISTANT VS AMAZON ALEXA

Criteria directly related to prototype development were weighted more than other criteria. Development Speed is an approximation of the time it will took to develop the same front-end application in the platform. It was found that there was noticeably more overhead to develop with Amazon than it was to develop with Google. Cost to Purchase refers to the MSRP of the smart assistant. Integration and Hosting refers to the ease at which the platform could be hosted and integrated with the back-end computer vision solution. Development collaboration refers to the platform's support for collaborative work. It was found that Google supports multiple developers to work on a project concurrently whereas Amazon fixes a project to one developer account. Cost to Use refers the variable cost associated with using the platform. Amazon offers 1 million free queries per month whereas Google has no charge. Weights and scores for each criteria were assigned with 0 being the lowest score and 5 being the highest score. Ecosystem refers to both smart assistants' availability. Google Assistant comes native on Android phones as well as Google Home products whereas Amazon Alexa is only supported on their Echo products.

	Weight	**Amazon Alexa**	**Google Assistant**
Development Speed	6	3	5
Cost to Purchase (fixed)	2	5	4
Integration and Hosting	5	3	4
Development Collaboration	3	0	4
Cost to Use (variable)	4	4	5
Ecosystem	1	3	5
Sum		6*3+2*5+5*3+3*0+4*4+1*3 **= 62**	6*5+2*4+5*4+3*4+4*5+1*5**= 95**
Rank		2	1
Continue		No	Yes

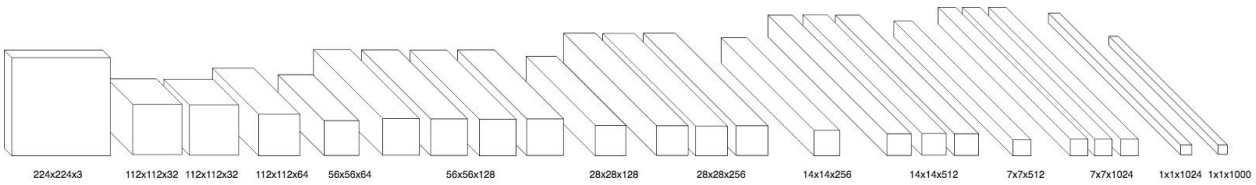
APPENDIX C

MOBILENET AND SSD ARCHITECTURE

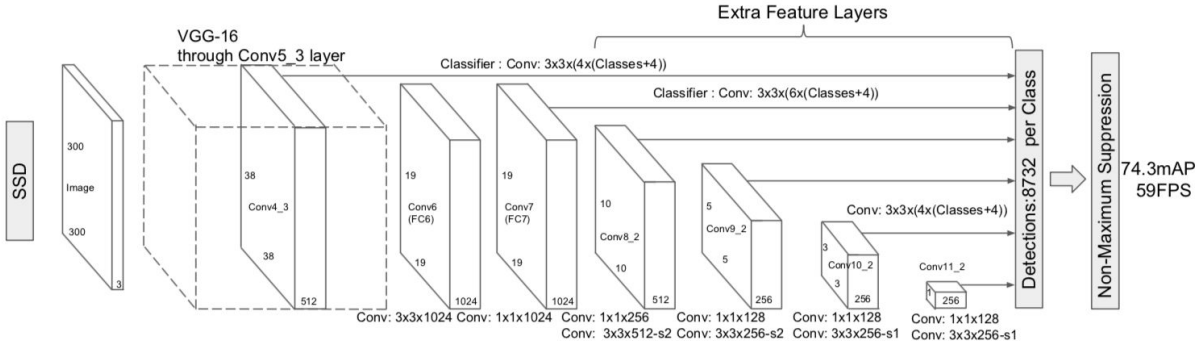
The Mobilenet architecture consists of an input image and uses depthwise and pointwise convolution filters to capture features at different hierarchical levels. The features are then passed into a classifier.

Table 1. MobileNet Body Architecture

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5\times$ Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$



SSD model predicts possible bounding boxes for each class.

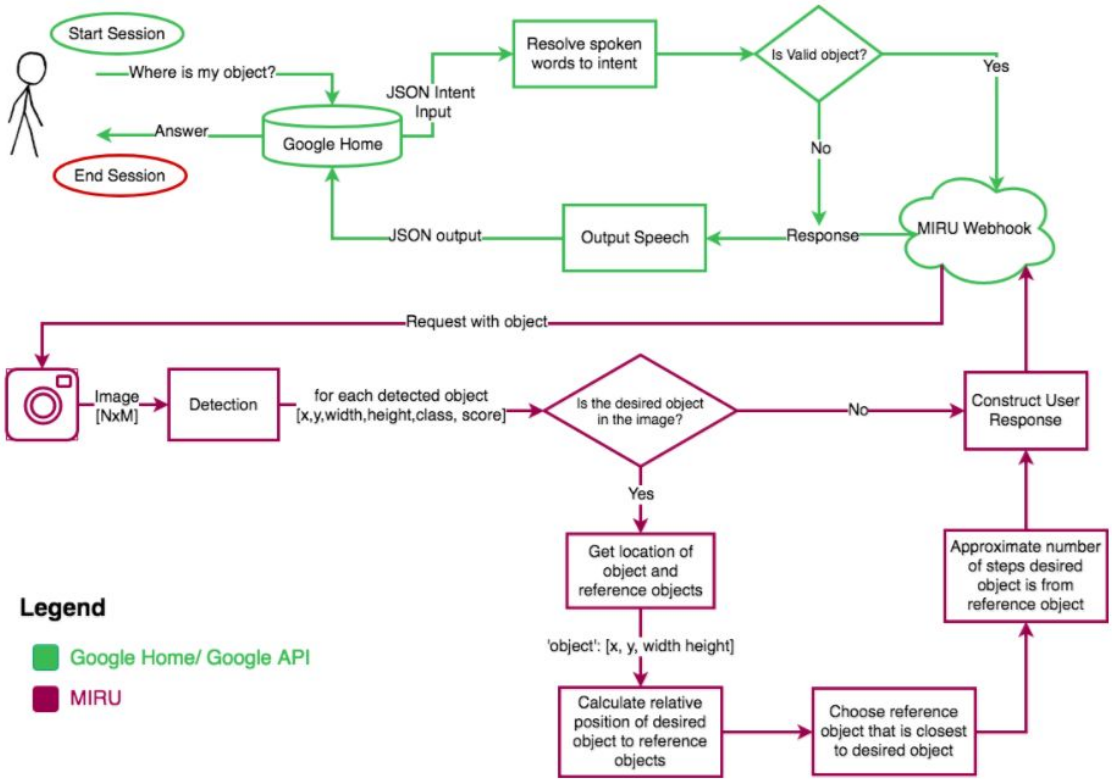


APPENDIX D
SPEED VERSUS ACCURACY

SSD has a better speed and accuracy trade off than the other models in the comparison.

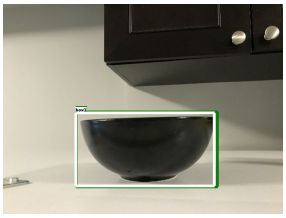
Method	mAP	FPS	batch size	# Boxes	Input resolution
Faster R-CNN (VGG16)	73.2	7	1	~ 6000	$\sim 1000 \times 600$
Fast YOLO	52.7	155	1	98	448×448
YOLO (VGG16)	66.4	21	1	98	448×448
SSD300	74.3	46	1	8732	300×300
SSD512	76.8	19	1	24564	512×512
SSD300	74.3	59	8	8732	300×300
SSD512	76.8	22	8	24564	512×512

APPENDIX E
DETAILED SYSTEM DIAGRAM



APPENDIX F
CORRECT CLASSIFICATIONS

Below are examples of correct classifications for each class. The white bounding boxes indicate the predicted bounding box, whereas the coloured bounding boxes are the ground truth from the manual annotations. The top 5 predictions for each class is shown below along with their respective confidence score that it is that class. For correct classifications, there is a large confidence difference between the top predictions and the others.

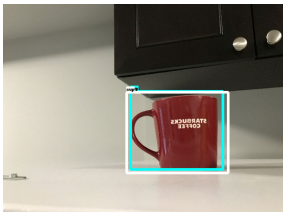


Bowl

Bowl	100%
Kettle	0.133%
Kettle	0.080%
Cup	0.050%
Kettle	0.047%

Kettle

Kettle	100%
Bottle	14.4%
Cup	0.105%
Cup	0.068%
Cup	0.034%



Cup

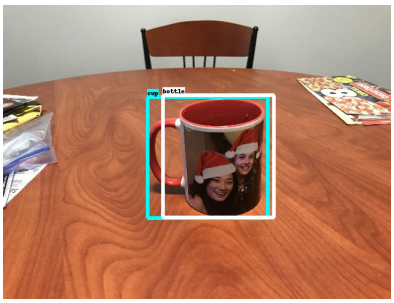
Cup	99.98%
Bottle	1.375%
Bowl	0.212%
Kettle	0.039%
Bowl	0.030%

Bottle

Bottle	99.68%
Kettle	0.089%
Kettle	0.028%
Kettle	0.004%
Bowl	0.004%

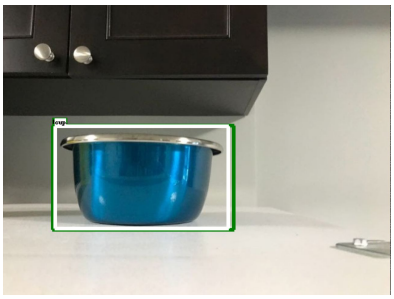
APPENDIX G
MISCLASSIFICATIONS

Below are examples of common misclassifications. For the misclassifications, there is a small difference between the top two predictions and the correct class is always in the second prediction.



Cup

Bottle	99.628%
Cup	88.673%
Bowl	0.711%
Bottle	0.335%
Kettle	0.060%



Bowl

Cup	99.988%
Bowl	90.039%
Kettle	1.396%
Bottle	0.194%
Bowl	0.059%

Kettle

Bottle	99.893%
Kettle	99.829%
Cup	0.027%
Bowl	0.009%
Bottle	0.005%