AN EMPOWERING MOBILE AID APPLICATION TO HELP VISUALLY IMPAIRED PEOPLE NAVIGATE AND EXPLORE PLACESAROUND THEM USING LOCATION TRACKING AND TEXT DETECTION

Yiyao Zhang[1], Anne Yunsheng Zhang[2] and Yu Sun[3]

[1]Bonita High school, 3103 D St, La Verne, CA 91750
[2]Ramona middle School, 3490 Ramona Ave, La Verne, CA 91750
[3]Computer Science Department, California State Polytechnic University, Pomona, CA 91768

ABSTRACT

In Section 1, we provide a comprehensive overview of the research findings concerning the challenges encountered by visually impaired individuals, which significantly impact their daily lives, and discussing potential solutions to address their needs. Section 2 delves into the obstacles we encountered during the development of our application, along with the strategies we implemented to overcome them. In Section 3, the functionality of our application is outlined, focusing on how it addresses the issues highlighted in Section 1. This section also details the specific systems within our app, including the sources and packages utilized in their development. Section 4 presents an in- depth analysis of potential blind spots in our application, supported by data and graphical interpretations. This section also discusses the precision and accuracy of our findings. Section 5 involves a comparative study of alternative solutions that address similar problems as our application , highlighting the distinct features and advantages of our approach. Finally, Section 6 summarizes the limitations of our current application and outlines planned enhancements to address these shortcomings in future updates, thereby improving the app's overall effectiveness.

KEYWORDS

Blindness, Firebase, iOS, Android

1. INTRODUCTION

We aim to address the challenges visually impaired individuals face in daily life, including mental health, communication, navigation, and reading, to enhance their integration into various environments and connection with the broader world. The prevalence of blindness is escalating, surpassing the previous year's rates. In 1990, the global blind population stood at 31 million, with projections estimating a significant increase to 115 million by 2050. Notably, three Asian regions, comprising 51% of the global population, account for 62% of the worldwide blindness population [2]. Beyond the escalating numbers, the mental health issues affecting the visually impaired have not seen improvement. Individuals grappling with visual impairment commonly experience mental health challenges, including depression, nervousness, anxiety, stress, and, in some instances, even suicidal thoughts [4]. Research indicates that nearly one-third of the blind exhibit mild depressive symptoms, while 10.7% to 45.2% report moderate-to-severe depressive symptoms [5]. These challenges intensify during social interactions, as blind individuals contend with anxiety arising from difficulties in perceiving nonverbal cues, adapting to unfamiliar environments, and self-consciousness about their visual impairment. Concerns about societal perceptions, negative reactions, and judgment contribute to heightened stress, severe anxiety, and, in extreme cases, suicidal tendencies. Living with blindness extends beyond visual impairment, encompassing unjust limitations in social and travel opportunities [6]. Communication hurdles arise from the inability to visualize conversations and interpret verbal cues relying on sight, leading to misinterpretations, heightened self-awareness, and social isolation. Moreover, barriers to safe navigation emerge due to inadequate accessibility features such as poorly placed signage, lack of wayfinding information, and absence of audio guidance [7]. Blind individuals face unjust obstacles impeding their autonomy, potentially exposing them to hazards. The impact is especially profound on those born blind, including children, young adult women, and older individuals. Young children, deprived of firsthand visual experiences, face challenges navigating the world independently. Those losing their sight soon after birth lack visual memories, potentially affecting their mental health. Young adults with vision loss are particularly vulnerable, lacking the experience and independence of older adults who have adapted to vision impairment [8]. Vision loss fundamentally influences behavior, introducing uncertainties about trust and relationships when visual cues are absent [9]. In conclusion, younger and less experienced visually impaired individuals face greater risks in traveling and navigating compared to others. Given that they cannot always rely on guidance from others, it is crucial to provide them with access to technology specifically designed to aid in overcoming these challenges , thereby enhancing their independence and safety.

One of the three solutions is to guide the visually impaired person to move around. This solution is to try to make it easier for visually impaired individuals to navigate around by telling them the amount of steps to reach the destination by voice message. This solution's shortcoming is that it can't tell the user whether there is an obstacle in front or not. Our applications can identify the obstacle that is in front of them and tell them what it is. Also can tell them they're surrounded by the environment. The second solution is to help blind people to navigate in indoor or outdoor scenarios. This solution is also trying to accomplish making the blind people travel around better by telling them if there is an obstacle detached or not by audio. The shortcoming of this solution is that it can't tell the environment around the user. Our application is combined with GPS with AI-powered scanning to make the user safely navigate around. The last solution is to provide location-specific audio insights. This solution is trying to accomplish auditory descriptions about the user's immediate environment. The shortcoming of this solution is that it can't help the scan texts. While our app can identify texts, signs, and menus and read aloud it to the user.

We are proposing an application designed for the visually impaired, enabling them to scan texts like menus, posters, and signs using the camera, and it also features GPS location and navigation functionalities. Our solution enables blind individuals to explore places by setting specific distance parameters and audibly present information about the place the user likes. Organized by distance range and then by categories such as stores, restaurants, hospitals, churches, and more.The GPS location/navigation helps blind individuals understand the places available to them. App reads aloud information about a location, including opening hours, popular dishes, distance, prices, and more. This would be a more holistic solution than other methods as it combines GPS with AI-powered scanning to provide both distant location awareness and immediate environment understanding for the blind. AI scanning can help blind people scan objects in front of them, such as poles, people, objects, etc. It will help them navigate better. AI scanning canalso define objects such as plants, buttons on remote controls, lamps, and various objects in people's homes, restaurants, and shopping malls. As we developed the app, we continued to refine its features, putting ourselves in the shoes of a blind person: "What would we need if we were them?” This empathy-driven approach led us to introduce real-time scanning to prevent accidents. We are continually working to improve the app to ensure maximum convenience and usefulness for blind users. " So we made a plan. While blind people navigate, they can use the scan feature to tell them what's ahead so they don't trip over it. In this way, we continue to transform the application into an application that is more convenient for blind people to use.

In Section 4, our focus revolves around rigorously testing the connectivity between the server, data sources, and our application, specifically within predefined distance ranges to ensure optimal performance. Experiment A is designed to assess the accuracy of data reception and transmission between our app and the server. Given the potential challenges arising from the use of two different sources, variations in connection quality are anticipated. To conduct Experiment A, we initiate our simulator, set a close distance range, and patiently observe as shops for each category emerge. Subsequently, we meticulously verify if all the anticipated shops are accurately displayed. The noteworthy outcome of Experiment A underscores its reliability, with an impressive 90% accuracy out of 100%. In Experiment B, our objective is to ascertain whether all the shops within the user-set distance range are appropriately displayed. To facilitate a comprehensive examination, we initiate this experiment with a short distance range, making it more manageable for testing purposes. We systematically inspect the displayed results to identify any discrepancies, ensuring that only relevant shops are showcased. The salient finding from Experiment B is the absence of inaccuracies in the displayed information within the specified distance range, reaffirming the precision and reliability of our application. These meticulous experiments not only validate the robustness of our server connections but also affirm the accuracy of information retrieval and display within designated distance parameters, ensuring the seamless functionality of our application.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. Gathering Information

Our primary challenge lies in accurately and reliably collecting data from various locations around the user. The accuracy of the data and the reliability of its source could lead to potential issues that require our attention. For example, if users are in an area where our information is not accessible, then our app may become ineffective for them because it relies heavily on location-specific data. For the accuracy of the data we could solve it by using the information from Google and Yelp, that provides a large amount of precise information, which can greatly enhance our dataset.

2.2. Scanning Feature

Our second major challenge involves enhancing the scanning feature of our application, which is crucial for text recognition and interpretation. This feature is vital for assisting blind users, acting as a 'second eye' by converting visual information into a format they can understand. Without the ability to read aloud the text captured by the camera, the feature would be ineffective. To resolve this issue, we could use a Flutter package called “google mlkit text recognition”. This package could help interpret whatever text and signs the user takes a picture of.

2.3. Identify a Wide Range of Objects

The third challenge we are addressing revolves around enhancing the camera's capabilities to identify a diverse array of objects. By seamlessly integrating cutting-edge computer vision and machine learning technologies, our aim is to empower the camera with real-time recognition capabilities for a wide spectrum of common objects. This task presents significant complexity due to the imperative of achieving high accuracy levels, set at a minimum threshold of 30%. Challenges are further compounded by variations in environmental conditions and the need for extensive and unbiased training datasets. In our pursuit of resolving this intricate challenge, we are exploring the utilization of a Flutter PyTorch package. This package serves as a powerful tool for running model classifications, enhancing the camera's ability to discern and categorize a multitude of distinct objects. Additionally, we are implementing object model detection, incorporating bounding box prediction methodologies. This multi-faceted approach is poised to not only elevate the accuracy of object recognition but also enhance the adaptability of the system to diverse and dynamic real-world scenarios. In essence, our strategy involves leveraging state-of-the-art technologies within the Flutter PyTorch package, coupled with innovative object model detection techniques, to navigate the intricacies of achieving accurate and real-time identification of a broad spectrum of objects under varying conditions.

3. METHOD ANALYSIS

The application is intricately structured around three key components: a robust text recognition system, an intuitive explore nearby destinations system, and an efficient object recognition system. Upon launching the application and navigating to the home screen, users are presented with two distinctive options: "Scanning Feature" and "Context Screen." Within the "Context Screen," users encounter a sophisticated distance range system complemented by diverse categories such as restaurants, shops, hospitals, churches, banks, and more. Utilizing the 'speech to text' and 'text to speech' Flutter packages, the application facilitates seamless voice interaction. The 'text to speech' package enables the app to audibly convey information to users.

To personalize their experience, users can express preferences through a long press on the home page, leading them to the context screen where they can dynamically adjust their preferred distance range. Google API then populates places based on the specified range, providing users with basic information about each destination, which is read aloud. Users have the option to delve deeper into details about a particular place by responding affirmatively with a simple 'yes.' The home screen also features a distinctive "Scanning Feature" that encompasses both text and object recognition capabilities. The text scanner leverages Google's MLKit Text Recognition, swiftly interpreting scanned content. Simultaneously, object recognition employs Flutter's Py Torch for model classifications, identifying a myriad of objects in the user's surroundings. The application efficiently harnesses Flutter's 'camera package' to access the camera, capturing desired text or objects with precision. This comprehensive design ensures a user-friendly experience, offering advanced functionalities in both voice-enabled exploration of nearby destinations and the versatile scanning feature for text and object recognition.

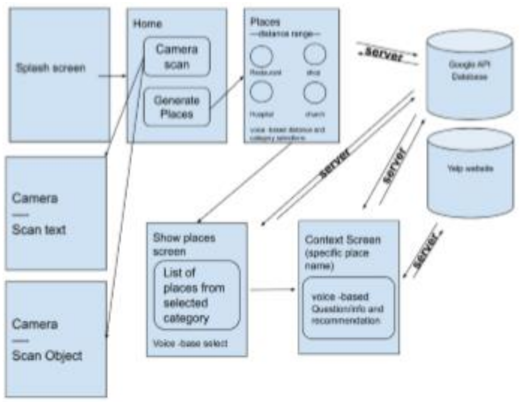
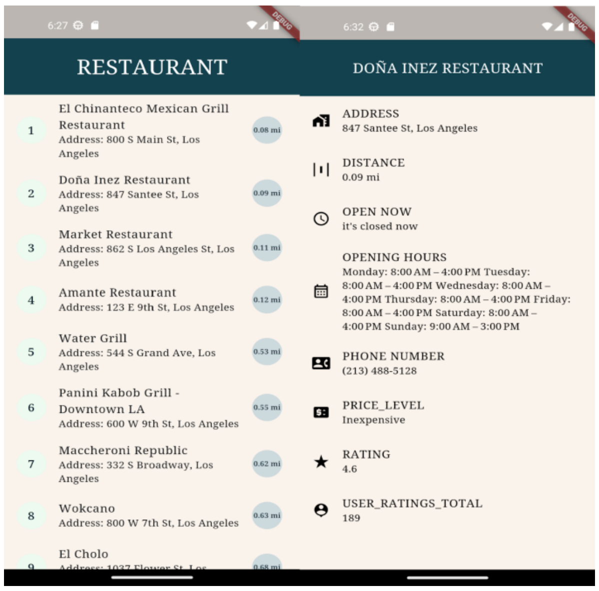


Figure 1. Overview of the solution

One of the central components in this application is the location generation system. Within the app, users define a distance parameter, prompting our server to source location data from both 'Google API' and 'Yelp' within the specified range. This system operates through a Python Flask server interfacing with the 'Google API' and 'Yelp', facilitating data requests and processing the returned results."



6:27

RESTAURANT

1 El Chinanteco Mexican Grill Restaurant
Address: 800 S Main St, Los Angeles — 0.08 mi

2 Doña Inez Restaurant
Address: 847 Santee St, Los Angeles — 0.09 mi

3 Market Restaurant
Address: 862 S Los Angeles St, Los Angeles — 0.11 mi

4 Amante Restaurant
Address: 123 E 9th St, Los Angeles — 0.12 mi

5 Water Grill
Address: 544 S Grand Ave, Los Angeles — 0.53 mi

6 Panini Kabob Grill - Downtown LA
Address: 600 W 9th St, Los Angeles — 0.55 mi

7 Maccheroni Republic
Address: 332 S Broadway, Los Angeles — 0.62 mi

8 Wokcano
Address: 800 W 7th St, Los Angeles — 0.63 mi

9 El Cholo
Address: 1037 Flower St, Los — 0.68 mi

6:32

DOÑA INEZ RESTAURANT

ADDRESS
847 Santee St, Los Angeles

DISTANCE
0.09 mi

OPEN NOW
it's closed now

OPENING HOURS
Monday: 8:00 AM – 4:00 PM Tuesday: 8:00 AM – 4:00 PM Wednesday: 8:00 AM – 4:00 PM Thursday: 8:00 AM – 4:00 PM Friday: 8:00 AM – 4:00 PM Saturday: 8:00 AM – 4:00 PM Sunday: 9:00 AM – 3:00 PM

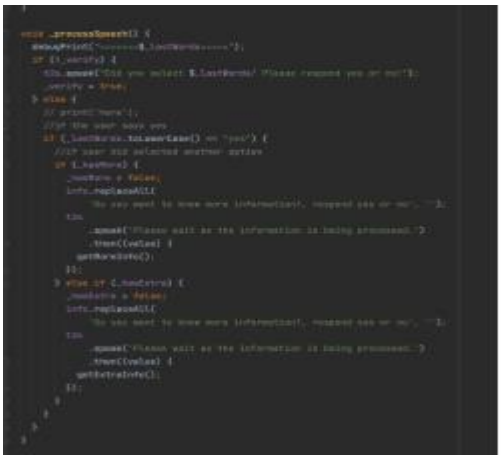
PHONE NUMBER
(213) 488-5128

PRICE_LEVEL
Inexpensive

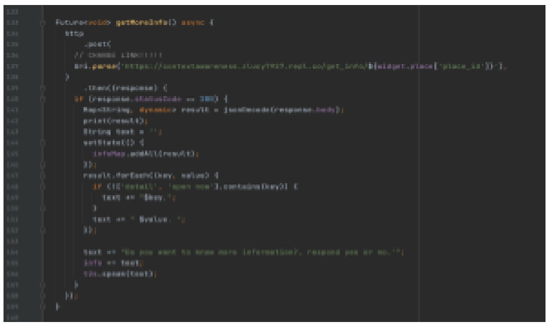
RATING
4.6

USER_RATINGS_TOTAL
189

Figure 2. Screenshot of the project 1

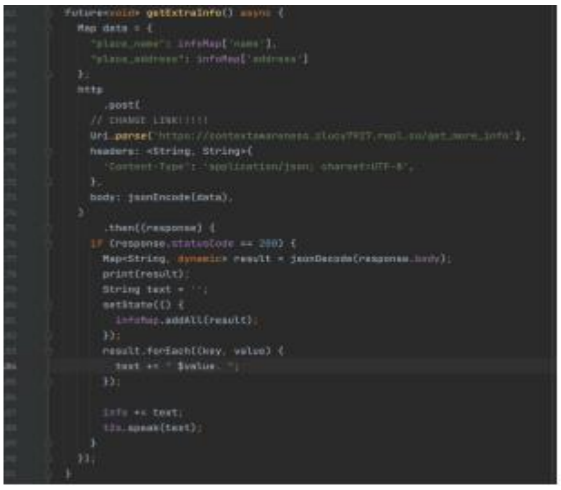


```
void _processSpeech() {
  debugPrint("--------$_lastWords------");
  if (!_verify) {
    tts.speak('Did you select $_lastWords? Please respond yes or no!');
    _verify = true;
  } else {
    // print('here');
    //if the user says yes
    if (_lastWords.toLowerCase() == "yes") {
      //if user did selected another option
      if (_hasMore) {
        _hasMore = false;
        info.replaceAll(
            'Do you want to know more information?, respond yes or no', '');
        tts
            .speak('Please wait as the information is being processed.')
            .then((value) {
          getMoreInfo();
        });
      } else if (_hasExtra) {
        _hasExtra = false;
        info.replaceAll(
            'Do you want to know more information?, respond yes or no', '');
        tts
            .speak('Please wait as the information is being processed.')
            .then((value) {
          getExtraInfo();
        });
      }
    }
  }
}
```



```
Future<void> getMoreInfo() async {
  http
      .post(
    // CHANGE LINK!!!!!
    Uri.parse('https://contextawareness.[illegible].repl.co/get_info/${widget.place['place_id']}'),
  )
      .then((response) {
    if (response.statusCode == 200) {
      Map<String, dynamic> result = jsonDecode(response.body);
      print(result);
      String text = '';
      setState(() {
        infoMap.addAll(result);
      });
      result.forEach((key, value) {
        if (!['detail', 'open now'].contains(key)) {
          text += "$key, ";
        }
        text += " $value. ";
      });

      text += "Do you want to know more information?, respond yes or no.'";
      info += text;
      tts.speak(text);
    }
  });
}
```



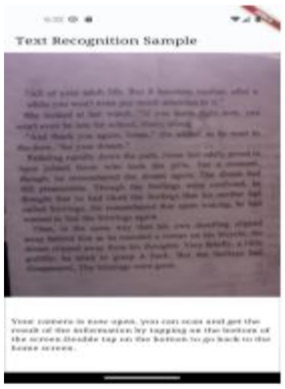
```
Future<void> getExtraInfo() async {
  Map data = {
    "place_name": infoMap['name'],
    "place_address": infoMap['address']
  };
  http
      .post(
    // CHANGE LINK!!!!!
    Uri.parse('https://contextawareness.[illegible].repl.co/get_more_info'),
    headers: <String, String>{
      'Content-Type': 'application/json; charset=UTF-8',
    },
    body: jsonEncode(data),
  )
      .then((response) {
    if (response.statusCode == 200) {
      Map<String, dynamic> result = jsonDecode(response.body);
      print(result);
      String text = '';
      setState(() {
        infoMap.addAll(result);
      });
      result.forEach((key, value) {
        text += " $value. ";
      });

      info += text;
      tts.speak(text);
    }
  });
}
```

Figure 3. Screenshot of code 1

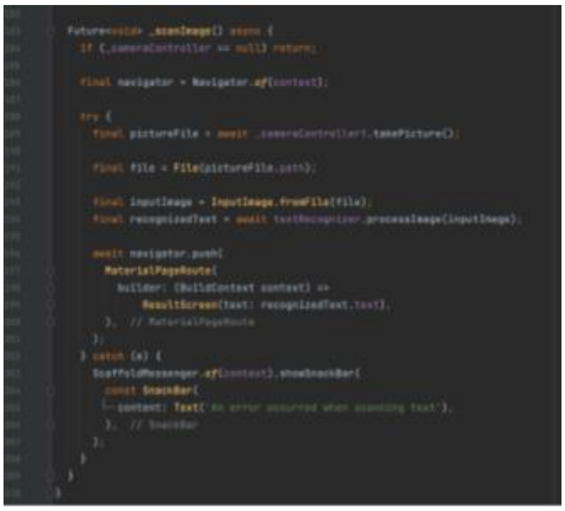
To ensure users receive comprehensive information based on their selections, we employ a tiered process. Initially, we deploy a process speech, acting as an intuitive touchpoint to authenticate and affirm users' decisions, The decision that like 'do you want to know more information'. Once their choices are acknowledged, and the user showcases curiosity for further details, we seamlessly activate the get More Info function. This particular function by communicating with our dedicated Python server, which subsequently extracts data from Google API. Desire even more in-depth details following that, we've provisioned the get Extra Info function. Upon invocation, this function sends specific queries to our server, which dives deep into Yelp's vast database to extract data. This ensures that users are presented with a rich tapestry of detailed information, catering to their varied and evolving requirements.

Another component that we established is “The Text Recognition System". This system allows the users to scan text on menus, signs, books, etc. In order to make this system, we used “google mlkit text recognition” This package could help interpret whatever text, signs, and books the user takes pictures of.



Your camera is now open, you can scan and get the result of the information by tapping on the bottom of the screen.Double tap on the bottom to go back to the home screen.

Figure 4. Screenshot of project 2



```
Future<void> _scanImage() async {
  if (_cameraController == null) return;

  final navigator = Navigator.of(context);

  try {
    final pictureFile = await _cameraController!.takePicture();

    final file = File(pictureFile.path);

    final inputImage = InputImage.fromFile(file);
    final recognizedText = await textRecognizer.processImage(inputImage);

    await navigator.push(
      MaterialPageRoute(
        builder: (BuildContext context) =>
            ResultScreen(text: recognizedText.text),
      ), // MaterialPageRoute
    );
  } catch (e) {
    ScaffoldMessenger.of(context).showSnackBar(
      const SnackBar(
        content: Text('An error occurred when scanning text'),
      ), // SnackBar
    );
  }
 }
}
```

Figure 5. Screenshot of code 2

The code sample shows that we have used many different functions, packages and files. We used the processImage() function from google mlkit text recognition package to process the captured text in the image. The Text Recognition System runs when the user taps on the bottom of the screen to capture a picture of the text. The screen will have instructions on how to use the text recognition system, for example, double tap on the bottom to go back to home. All of the introduction will be audio and text. The system will try to capture, process, and generate text results based on the image taken, but if it fails, it will pop up an error message: “An error occurred when scanning text”. After the user takes the picture, the system will take time to load first, then it will pop up a result screen (which contains the audio read aloud and text display).The third component that we established is “The Object Recognition System”. This system allows us to define objects with the camera feature. We are using flutter pytorch to run model classification. And have an object model detection that uses bounding box prediction. These things allow our camera object-defined feature to work properly.

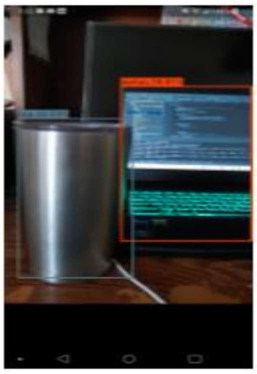
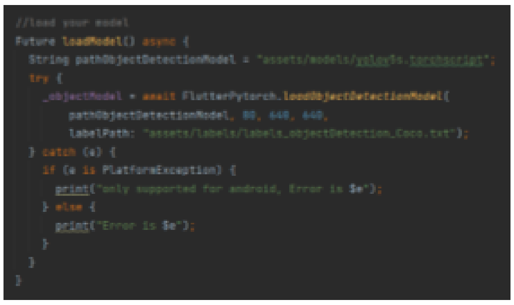
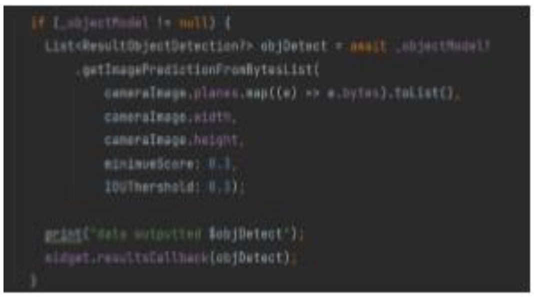


Figure 6. Screenshot of project 3

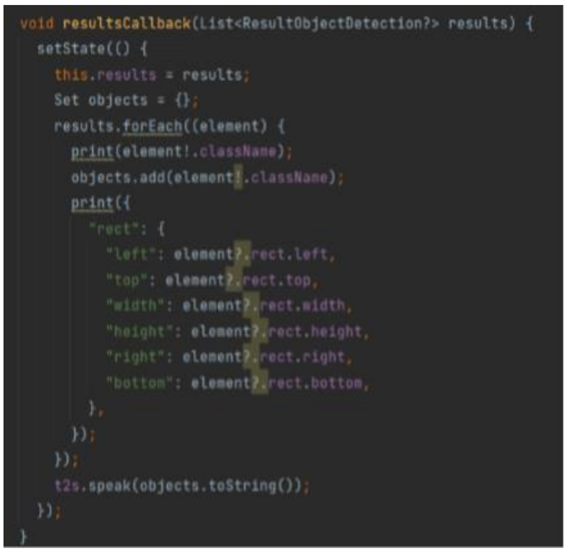


```
//load your model
Future loadModel() async {
  String pathObjectDetectionModel = "assets/models/yolov5s.torchscript";
  try {
    _objectModel = await FlutterPytorch.loadObjectDetectionModel(
        pathObjectDetectionModel, 80, 640, 640,
        labelPath: "assets/labels/labels_objectDetection_Coco.txt");
  } catch (e) {
    if (e is PlatformException) {
      print("only supported for android, Error is $e");
    } else {
      print("Error is $e");
    }
  }
}
```



```
if (_objectModel != null) {
  List<ResultObjectDetection?> objDetect = await _objectModel!
      .getImagePredictionFromBytesList(
          cameraImage.planes.map((e) => e.bytes).toList(),
          cameraImage.width,
          cameraImage.height,
          minimumScore: 0.3,
          IOUThershold: 0.3);

  print('data outputted $objDetect');
  widget.resultsCallback(objDetect);
}
```



```
void resultsCallback(List<ResultObjectDetection?> results) {
  setState(() {
    this.results = results;
    Set objects = {};
    results.forEach((element) {
      print(element!.className);
      objects.add(element!.className);
      print({
        "rect": {
          "left": element?.rect.left,
          "top": element?.rect.top,
          "width": element?.rect.width,
          "height": element?.rect.height,
          "right": element?.rect.right,
          "bottom": element?.rect.bottom,
        },
      });
    });
    t2s.speak(objects.toString());
  });
}
```

Figure 7. Screenshot of code 3

In the first code sample, the camera's ability to recognize the model proves, using the YOLOv5 model. The YOLOv5 model is mostly used for creating the pre-trained model. Our application will attempt to load the pre-trained model. However, if this process was unsuccessful, it will promptly issue a detailed error message to alert the user. An IOS phone user will receive "only supported for android, Error is (the error)", an Android phone user will receive "Error is (the error)". For object labeling we use the "labels_objectDetection_Coco.txt" file located in the Labels folder. The second code sample is calculating the prediction of the object accuracy. Our application requires a minimum accuracy of 30% to reliably recognize an object. The third code sample receives data from calculator prediction, and visually shows by drawing boxes around the object that are detected. Additionally, the objects are then being organized into a set to notify the users about the identified items.

4. EXPERIMENT

4.1. Experiment 1

A blind spot in our program that we want to test out is the "Context Screen" that displays specific information about a place. It is important that this part of our program works well because this page is getting information from two places: Google API Database and Yelp Website. The accuracy of various aspects, including address, name, popular dishes, and opening hours, is paramount. We need the information all to match each other so the information about each location will be precise and accurate.

We will do the following steps to test our information accuracy. First, we are going to tap on the bottom portion of our app, then set up the distance limit. Next, choose a random category. After choosing the category, the system will say the amount of category is available within that range. Finally, after choosing a pacific location/place, the information about that place will pop up. If we want all of the information then we will say "yes" and all of the facts will pop up. (Because at first the app will only show the basic information.) After all of the data pops up then we can check our data with the data on Google API and Yelp website. The experiment is set up this way because setting up like this we can easily figure out the difference/errors between our app's data and their data.

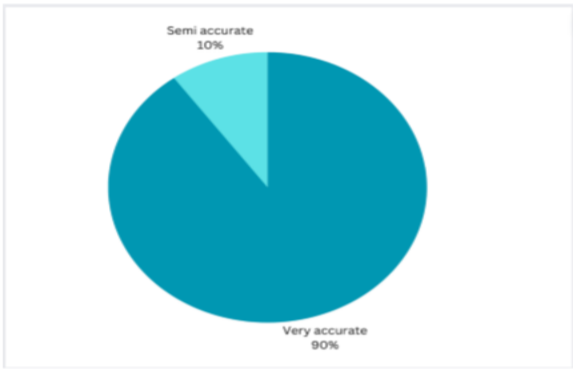


Figure 8. Figure of experiment 1

Our data accuracy ranges from a minimum of 10% to a maximum high of 90%. The reason why 90% is very accurate, because we are receiving data from google API and yelp website. They are both highly dependable platforms in the digital landscape. So, it stands the reason that the quality of data we receive from these trusted sources would be reliable, and as a result there will be less chance of errors or unreliable data. And the reason why 10% is semi accurate is because the app may not always retrieve the information, this can be attributed to either occasional server disconnection with the app or with Google API and the Yelp website. We are confident in the data that we collected, because we understand how the server operates: it will either get accurate information from the database or return nothing at all. It is reliant on established sources like Google API and Yelp and unexpected data is going to be highly unlikely.

4.2. Experiment 2

Another blind spot that we want to test out is the distance limit. Because we want the blind people to know how far they need to travel to reach their destination by walking or driving. The distance limit is important to our app because the maximum range needs to be set in order for places to generate and be read aloud for the user to choose from.

First, we will set a distance limit, then we will choose a category such as a restaurant, shop, hospital, or church. Next, check if there are any places in the results that are farther than the maximum range. Finally, we are going to check if there are any places that didn’t pop up or aren't mentioned within the area when the app is reciting the information. We will also check if there are any other categories showing up that are not related to the category that we have chosen – for example, if we choose a shop, then restaurant, church, and hospital shouldn’t be showing up. We did the experiment like this because we want to make sure when the blind people are using it, it won’t have any false or missing information.

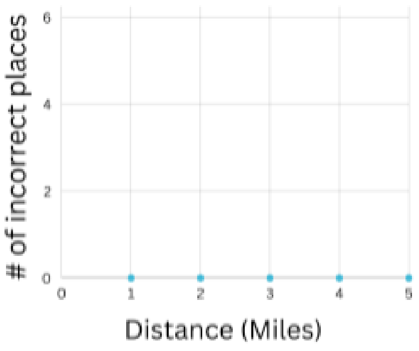


Figure 9. Figure of experiment 2

In this graph it doesn't have a minimum and maximum range. Because when we test the distance range it has an accuracy of 100%, no matter how far the distance is. For this system we also use the help of Google API, it helps us eliminate the shops that are out of the range. Google API is a very reliable source, we depend on it for this distance range system and the system that allows users explore places nearby.

5. RELATED WORK

The Lazarillo App, created by Luiz Fernando, is a georeferencing application that guides the visually impaired person to move around [10]. It automatically maps the routes traveled by the user and creates a graphical representation that can be used later. It provides guidance on the direction and the amount of remaining steps to reach the destination through voice messages [11]. The Lazarillo App by Luiz Fernando focuses more on navigation. While we provide users with an understanding of distant locales alongside an acute awareness of their immediate environment. By juxtaposing both, our innovative approach aims to furnish a comprehensive navigational experience for the blind community.

Daniel Vera researchers propose a system based on this technology that helps blind people to navigate in indoors or outdoors scenarios. The prototype is portable assuring that can be used anytime and anywhere. The system is composed of wireless sensors that can be used in different parts of the body. The sensors detect an obstacle and inform the user with an audible warning providing a safety walk to the users [12]. As Daniel Vera's system is intended to be used for obstacle detection and safe navigation, while our method is more holistic, it combines GPS with AI-powered scanning to provide both distant location awareness and immediate environment understanding for the blind.

BlindSquare, crafted by Ilkka Pirttimaa, through the utilization of advanced GPS and map data, it efficiently delivers auditory descriptions about the user's immediate environment, it also can be paired with other navigation apps for turn-by-turn directions, BlindSquare's primary focus is to provide situational awareness. BlindSquare aims to enhance situational awareness, while its primary function is providing location-specific audio insights [13]. Our proposed application takes a more holistic approach. It seamlessly combines GPS navigation with cutting-edge AI scanning capabilities. This fusion not only aids in general navigation but also offers users a deep understanding of their immediate surroundings, bridging the gap between mere location information and comprehensive environmental awareness.

6. CONCLUSIONS

The core objective of this application is to empower visually impaired individuals by providing them with a tool to enhance their daily navigation and interaction with their surroundings. Despite the promising strides the app has made, it is evident that ongoing research and development efforts are imperative to elevate its performance across various intricate systems, specifically the text recognition, object recognition, and explore nearby destination functionalities. In the domain of text recognition, there is a clear need for heightened accuracy to precisely identify texts, ensure their correct sequencing, and seamlessly read them aloud for the user. The object recognition system, driven by artificial intelligence, requires enhancements to accurately identify objects and determine their proximity to the user. Simultaneously, the explore nearby destination system necessitates improvements in our app's server, fostering a more robust connection for efficient information transfer from sources like Google API and Yelp Website to the server and subsequently to our app [14]. One noteworthy limitation we currently face is in the realm of AI navigation, where further advancements are yet to be realized. Acknowledging this limitation is a crucial step toward transparency and sets the stage for potential future developments in this area. To address these challenges and propel the application forward, we are considering enhancements to the YOLOv5 model. This model, instrumental in creating a pre-trained model for our object recognition system, holds the potential for further refinement. Additionally, improvements in AI capabilities are on the horizon, aiming to enhance the accuracy of object calculations [15]. In summary, while the application has demonstrated notable progress, the commitment to continuous research and development remains paramount. The identified areas for improvement, encompassing text and object recognition, server connectivity, and AI navigation, underscore our dedication to providing a robust and effective tool for the visually impaired community.

REFERENCES

[1] Riazi, Abbas, et al. "Outdoor difficulties experienced by a group of visually impaired Iranian people." Journal of current ophthalmology 28.2 (2016): 85-90.

[2] Ackland, Peter, Serge Resnikoff, and Rupert Bourne. "World blindness and visual impairment:

despite many successes, the problem is growing." Community eye health 30.100 (2017): 71.

[3] Skaggs, Steve, and Chris Hopper. "Individuals with visual impairments: A review of psychomotor behavior." Adapted Physical Activity Quarterly 13.1 (1996): 16-26.

[4] Thylefors, B., et al. "Global data on blindness." Bulletin of the world health organization 73.1 (1995): 115.

[5] Demmin, Docia L., and Steven M. Silverstein. "Visual impairment and mental health: unmet needs and treatment options." Clinical Ophthalmology (2020): 4229-4251.

[6] Burton, Harold. "Visual cortex activity in early and late blind people." Journal of Neuroscience 23.10 (2003): 4005-4011.

[7] King, Andrew J. "What happens to your hearing if you are born blind?." Brain 137.1 (2014): 6-8.

[8] Senra, Hugo, et al. "Psychologic adjustment to irreversible vision loss in adults: a systematic review." Ophthalmology 122.4 (2015): 851-861.

[9] Wang, Shu-Wen, and Kathrin Boerner. "Staying connected: re-establishing social relationships following vision loss." Clinical Rehabilitation 22.9 (2008): 816-824.

[10] Guillén, Claudio. "La disposición temporal del Lazarillo de Tormes." Hispanic Review (1957): 264-279.

[11] Rose, Richard C., Eric I. Chang, and Richard P. Lippmann. "Techniques for information retrieval from voice messages." Acoustics, Speech, and Signal Processing, IEEE International Conference on. IEEE Computer Society, 1991.

[12] Vera, Daniel, Diego Marcillo, and Antonio Pereira. "Blind guide: Anytime, anywhere solution for guiding blind people." Recent Advances in Information Systems and Technologies: Volume 2 5. Springer International Publishing, 2017.

[13] Kumar, PV Arul, et al. "A study of added sic powder in kerosene for the blind square hole machining of cfrp using electrical discharge machining." Silicon 14.4 (2022): 1831-1849.

[14] Këpuska, Veton, and Gamal Bohouta. "Comparing speech recognition systems (Microsoft API, Google API and CMU Sphinx)." Int. J. Eng. Res. Appl 7.03 (2017): 20-24.

[15] Fang, Yiming, et al. "Accurate and automated detection of surface knots on sawn timbers using YOLO-V5 model." BioResources 16.3 (2021): 5390.