FACULTY OF ENGINEERING
DEPARTMENT OF AI ENGINEERING

FINAL REPORT

EVI (Eye of Visually Impaired)

GROUP MEMBERS

Zeynep Meriç AŞIK
Emre BELİKIRIK
Meriç DEMİRÖRS

CONTENTS

1. PURPOSE OF THE PROJECT

Visually impaired individuals encounter many obstacles throughout their lives. The most difficult of these obstacles is the inability to perceive the objects and obstacles around them and therefore have difficulty moving or doing a job. The EVI project we are presenting is a project designed to help visually impaired individuals with these difficulties in their lives. To achieve this, it uses a large language model along with multiple image processing models to give audio feedback to the user.

Whether navigating crowded streets, moving through unfamiliar indoor spaces, or being outdoors in open spaces; visually impaired individuals face numerous situations that endanger their safety and hinder their independence. Considering that visually impaired individuals perceive and recognize the environment through sound unlike healthy individuals, it is necessary to present the environment they belong in to these individuals as a verbal solution.

Traditionally visually impaired individuals rely on outside help such as canes, guide dogs or a second person to help them. Although these aids are important, they affect the environment around the individual; they do not provide a real-time solution that is always available and can be explained verbally to the person. Despite this external help, it is incomparably more difficult for visually impaired individuals to do even the simplest tasks than for a healthy person.

Visually impaired individuals unfortunately face many obstacles and limitations that affect their lives. Helping people experiencing these difficulties, ensuring their safety and transforming them into free individuals who are not dependent on other aid is an important step in social equality.

Whether it is objects at head level, benches on the sidewalk, or dangerous objects blocking the walkway, visually impaired individuals look to those around them for help to avoid these obstacles. This significantly restricts their mobility, ultimately limiting their personal freedom and putting them at higher risk of accidents, injuries and even social isolation.

Additionally, the lack of real-time information about their environment prevents visually impaired individuals from fully experiencing and engaging with the world around them, depriving them of opportunities such as exploring new places or participating in outdoor activities.

Therefore, finding a solution to this problem is extremely important. By strengthening the ability of visually impaired individuals to perceive and understand their environment; their safety, independence and overall quality of life can be significantly improved.

1.1. Target User Audience

323 million (4.1%) people worldwide and 1.04 million (1.4%) people in Turkey are visually impaired [1, 2], and considering the usability of the project, it is obvious that it will make the lives of many individuals easier if it is successful.

A world in which visually impaired individuals can navigate safely, easily enter unfamiliar environments, and make informed decisions about their environment will increase the welfare of society and visually impaired individuals.

Addressing this issue not only improves the lives of visually impaired individuals, but also promotes a more inclusive and accessible society. It creates an environment to break down barriers and enable every individual, regardless of their visual abilities, to fully participate in the world around them.

1.2. Products and Services Related to the Project Idea

If we take a look at the previous solutions on this subject, after examining different articles, we see that the most used object detection/classification/localization models are mostly YOLO variants [3, 4, 5, 6, 7]. In addition, there are projects developed using built-in systems instead of deep learning models in systems defined to be used only in certain areas.

Almost all of the methods touched upon the importance of building a system on wearable devices such as phone cameras or head cameras in order to make the systems they have established usable by everyone, and projects suitable for this have been researched and examined [3, 10]. There are also projects that use depth sensors [4] and round-trip time (ToF) cameras to localize the scene from cameras.

In these methods, auditory [3, 4, 6, 7, 8, 9] and/or tactile [7, 8, 9] feedback is used to inform the user.

Additionally, after doing research on models that perform depth perception from RGB images and reviewing the articles, it was seen that a Unet-style encoder-decoder [11, 12, 13, 14] structure was generally used. In addition, there are also models using different techniques such as transformer [11, 12, 13, 14].

Many separate articles that are directly related to the subject or are far from the subject that can only give an idea at a certain point have been examined, the majority of the articles are articles of projects aimed at helping visually impaired individuals or individuals with another disability. Only the articles most relevant to the topic are listed in the "References" section.

1.3. Our Solution Suggestion

The proposed project proposes a solution involving machine learning that uses multiple models together to help people in need.

Information about the environment is obtained by using a camera that constantly monitors the environment in front of the individual and then feeding the visual data captured by this camera into vision models.

The outputs produced by vision models that collect information about objects in the environment and their distance to the individual are then fed into the large language model (LLM). This model analyzes input and responds accordingly, communicating important information to visually impaired users, such as available paths to follow or potential hazards to avoid.

What makes this system a feasible solution is that, unlike guide dogs, it is accessible to everyone, unlike canes, it reflects the environment as a whole to the individual, and it is always accessible, unlike a second person helping them. Thus, it is aimed to eliminate the difficulties that may arise from navigating both in places where it is difficult to move and in areas that the individual does not know.

The project aims to make the lives of visually impaired individuals easier by providing them with real-time information about their environment. The environmental awareness it provides to individuals increases their safety, supports their independence and ultimately improves their overall quality of life.

The solution proposal we have presented within the scope of the project has similarities and

differences compared to previously developed sample projects. When the solutions so far are examined, although similar approaches have been followed in recognizing obstacles in the environment, apart from the difference in combining and using different models in different ways compared to the past, the operation of an LLM model in the innermost layer of the system is the biggest difference and the biggest originality between the previous idea suggestions.

When previous solutions are examined, there are common and similar aspects in many points, from the use of smartphone cameras to the use of YOLO variant object detection/ classification/ localization models and models that create depth maps from RGB images, but a system where these methods are combined and used as explained above has not been seen in the observed research. The project proposal shows originality in terms of the way the models used are used.

The most prominent point among the original points is that, unlike previous methods, there is no guidance around the rules determined by the system itself. Instead, there is an LLM model that evaluates the inputs given to it from a logical perspective and provides a response to the user accordingly.

2. USER SCENARIOS & REQUIREMENTS ANALYSIS

2.1. User Scenarios

2.1.1. User scenario: University student passing through the university garden and parking lot and reaching the main building

1. Let's assume that the user is sitting in the garden of the university and trying to reach the main building.
2. The user will open the application on his phone (the phone will automatically connect to the server and start sending pictures) and move through the garden towards the main building, and at the same time, the user will receive feedback about the environment such as "there is a tree this far ahead", "there are people walking this far to the right, turn left a little". User will be able to leave the area without encountering any obstacles.
3. Considering that the user is not completely disconnected from the environment and has general knowledge and assumptions about his/her surroundings, we can think that he/she arrives at the parking lot after advancing with the help of a cane in cases where application support alone will not be sufficient, such as stairs.
4. To cross the street in the parking lot, he/she will listen to the surroundings, turn his phone left and right, and if he does not receive any obstacle warning from the application, he/she will cross the street safely.
5. Since he is knowledgeable about the environment, he/she will reach the main building without getting stuck in obstacles with the help of the application's feedback about the obstacles and his cane.

2.1.2. User scenario: Reaching the tables in the garden from the building lobby

1. Let's assume that the user is sitting on the seats in the building lobby and wants to reach the tables in the pergola in the garden.
2. 2. The user opens the application on his phone (the phone will automatically connect to the server and start sending pictures), gets up from the couch, walks to the door with the help of a cane, exits the door and begins to follow the blind paths on the ground.
3. While following the paths, user can check whether there is a person or animal in front of him/her with the help of his/her phone. User receives audio feedback whether he/she will hit one of the pots or garbage bins in the garden and adjusts his/her direction accordingly.
4. After reaching the area where the camellias are located with the help of the cane and the application, he/she will turn to the direction of the camellias and start moving in that direction, using his/her knowledge and assumptions about its surroundings.

5. As he/she approaches the camellias, he/she can choose one of the tables and sit down with the help of a cane, by receiving feedback from the application such as "there is a table this far", "there is a chair this far to the right".

2.1.3. User scenario: Walking outside in the park

1. Let's assume that the user wants to take a walk in a park that contains obstacles.
2. The user will open the application on his phone (the phone will automatically connect to the server and start sending pictures) and move through the park from the starting point.
3. Meanwhile, user will receive audio feedback about how far away it is from people, benches, cats, dogs or other animals, garbage bins and similar obstacles that he/she may encounter and adjust its route accordingly.
4. In this process, the user can obtain information about the objects around him by turning his phone left and right or by using general environmental information with his cane.
5. Thus, he will be able to proceed on the path and complete his walk without hitting any objects or people.

2.1.4. User scenario: Going grocery shopping

1. Let's assume that the user wants to go to a grocery store within walking distance.
2. Meanwhile, user needs to leave the building where his/her house is and reach the market by staying on the sidewalk on the street.
3. The user will open the application on his phone (the phone will automatically connect to the server and start sending pictures) and start following the route he knows before, all the way to the market.
4. Meanwhile, the distance of other pedestrians walking on the pavement, trees, garbage cans or bags, shop signs on the ground and similar objects that may come across will be provided by the application as an audio feedback.
5. In this process, the user can turn the phone in the direction he feels/hears an object or person approaching, and can change direction according to any obstacle warning and avoid the obstacle.
6. Thus, he/she can safely go from home to the market with notifications sent from the application, taking into account the situations he hears and feels with his cane.

2.1.5. User scenario: Reaching the apartment from the site entrance

1. The user passes through the entrance gate of the site thanks to the currently open application.
2. The application detects buildings, roads and important areas within the site and guides the user by transmitting this information to his phone.
3. When the user approaches the entrance of the apartment, he/she learns the location of the entrance door by listening to the voice narration on his phone.
4. When the user enters the apartment, he/she learns the roads and stairs inside the apartment with the voice guidance of the application.
5. When the user wants to reach the floor where the apartment is located, he/she finds the elevator or stairs thanks to the voice narration of the application.
6. When the user approaches the door where the apartment is located, he/she can more easily understand which direction the door is with the feedback of his/her phone and reach home safely.

2.1.6. User scenario: Getting to the bus stop from home

1. User aims to leave his home and go to the bus stop.
2. When user leaves his house, he/she opens the application and scans surroundings with his/her phone to get information about his surroundings.
3. The application recognizes obstructive objects around the user's home and

provides guidance by transmitting this information to the user.

4. When the user wants to reach the bus stop, he/she listens to the voice narration on his phone and follows the correct directions.
5. The user carefully passes the sidewalks, pedestrian crossings and intersections that may come across on the road with the voice guidance of the application.
6. When he approaches the bus stop, he/she moves towards the bus stop safely with the voice narration on his phone.

2.2. Requirements Analysis

The desired solution to the common problem of millions of people, which is also mentioned under the heading of Target User Audience (1.1.), is to create an application that will help visually impaired individuals move comfortably in their environment. In this era when almost everyone has smartphones, the requirements for developing a project that helps the user by sending information about his/her surroundings from his/her smartphone to a computer running in the background, analyzing the information sent on that computer, and transmitting the results to the individual via audio are as follows:

- This project is important for visually impaired individuals to be able to move around independently of others in their environment and to provide an inclusive and individual-guiding feedback about the environment even when external assistance such as a cane/guide dog is not available, and it is necessary considering the wishes and needs of these individuals.
- A smartphone with a camera is required to capture the visual data of the environment, and since auditory feedback to the individual will be made through this device, a device that preferably has a speaker output or can provide output through headphones is required.
- In order to send visual data from the smartphone to the computer, data transmission must be provided via APK over the phone, and text transmission must also be provided so that the outputs obtained on the computer can be transmitted to the phone. For this reason, in terms of the usability of the project, an environment where data communication can be fast and complete is important for the project to show optimal performance.
- Since the computer that will be working in the background will pass the information transmitted to it through different image processing models and feed the outputs obtained to the LLM in a suitable format, it must perform all these operations quickly and send the result obtained from the LLM to the visually impaired individual's smartphone without any delay. A computer is required to perform these operations and information transfers in near real time, in accordance with the working logic of the application.
- In addition to the application used to provide the best assistance to the user in the most reasonable time, an external assistance such as a cane/guide dog, which can help the individual provide the information transmitted to him/her via auditory means, will assist in the operation and operation of the application.
- Considering the environmental characteristics required to ensure the best performance in the project, the appropriate usage areas of the system are environments where models can provide accurate and fast information. For this reason, environments where communication with the computer at the back will be clear in order to ensure fast and complete data transfer are the optimal environments in the project. In environments where models will run slowly and/or incorrectly, the usability of the project will decrease.
- It would be more suitable for the project if the APK that the user will connect to the system over the phone in order to receive the most up-to-date service should be the latest version APK.

3. SYSTEM FUNCTIONS

3.1. APK

- Works with Flutter module.
- Uses audioplayers module to play sounds.
- Uses camera module to access camera images.
- It uses the http module to communicate with app.py over the internet.
- It uses Flutter TTS module to give voice feedback to the user.
- Processes and sends images captured from the camera to app.py via the "/process_image" path.
- It receives the texts sent by app.py via the "/get_llm_output" path and provides voice feedback to the user.

The interface of the application has been adjusted to be most suitable for visually impaired individuals, as simple and easy to use as possible. Users start and stop the application with a button that allows switching between 2 different states. The interface images of the application in 2 states are as follows:

Fig. 1. Rest mode interface

It goes into active mode with the “Start” button and goes into rest mode with the “Stop” button.

Fig. 2. Active mode interface

When the application is first launched, the user is notified that the application is ready for use with a “Press to Start” audio callback. By pressing the "Start" button, the application becomes active and gives the user a voice response saying "Checking for obstacles, be cautious", indicating that the application has started working, started taking images from the camera, and will soon send instructions appropriate to the user's situation from the server. The exchange of images and instructions continues between the application and the server until the user presses the "Stop" button, and the application provides a voice response to the user as soon as it receives the instructions. When the user presses the "Stop" button, the "Press to Start" voice message informs the user that the application has entered a resting state, that it is no longer receiving images from the camera or waiting for instructions from the server, and that it will no longer provide voice feedback.

3.2. Server (app.py)

- It uses Flask module to communicate with the user.
- It receives the images sent by the APK from the "/process_image" path, processes them and saves them in a shared disc for user.py to read.
- It reads and processes the LLM text saved in another shared disc by user.py and sends it to the APK via the "/get_llm_output" path.
- It uses the OpenCV module to process the image and save it to disk.
- Uses numpy module to process images.
- It uses base64 module to encode and decode images.
- It uses the os module to check the updates of the files.

3.3. YOLO

- Using the ultralytics module, object detection, classification and localization are performed on input images.
- By using the OpenCV module, extra information is provided to the developer about the objects detected through the outputs of the model.

For a more detailed explanation, let's examine the model outputs with 2 separate input images:

In the image on the left, the YOLO model detected 2 motorcycles and 1 person, but for the image on the right, no objects were detected.

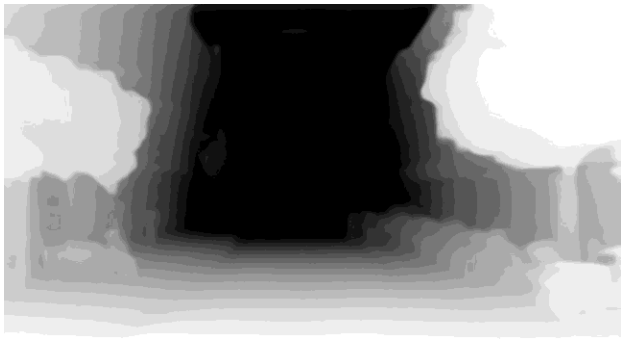
YOLO output:
[((10, -30), 'person', 1),
((20, -40), 'motorcycle', 2),
((30, 40), 'motorcycle', 3)]

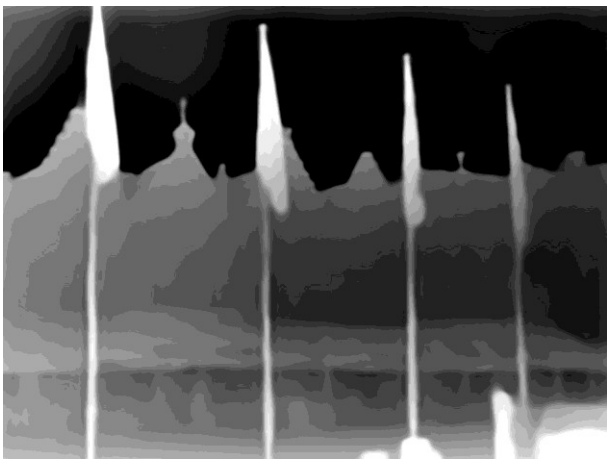
YOLO output:
[]

3.4. Depth Model

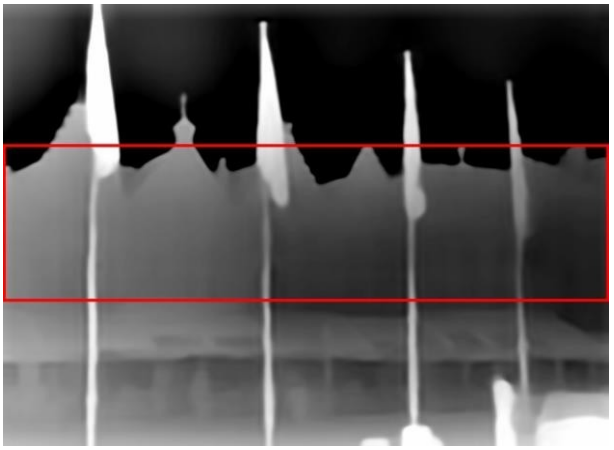
- Using the torch module, depth detection is made on input images with the “ZoeDepth” depth detection model.
- Depth is determined by paying attention to the objects detected in the output of the YOLO model or, if no output is given, to the user.

For a more detailed explanation, let's examine the model outputs with 2 separate input images:





If we show the areas where depth will be determined according to the output of the YOLO model:



When the outputs of the YOLO and DEPTH models are combined to create a prompt, the scene information we obtain is as follows:

[(5, -40, 2, "motorcycle"),
(3, 40, 3, "motorcycle")]

[(5, -30, id=0)]

Depending on whether the YOLO model detects an object or not, according to the confidence score given by the model to the object, and according to the depth detected by the DEPTH model, detections that are above the threshold value (i.e. those that are considered dangerous) are included in the scene information. Scene information in the form of (3, 40, 3, "motorcycle") carries "meter distance", "left right degree information", "id" and "object name" information respectively (in scenes without YOLO detection, "object name" information will be missing).

3.5. LLM

- After the outputs of the YOLO and DEPTH models are combined, they are combined with prompts depending on the situation.
- By using Ollama and langchain modules, short and situation-summarizing output texts are obtained from the “llama2” large language model to the input prompts.

If we were to show the prompts to be given to the models upon combining the scene information given above with ready-made prompts, the following examples can be given:

“There is a system to help blind people, this system captures the obstacles ahead the user via camera and calculates how far and how many angles left or right are this obstacles from user. You should guide the user. Now I will provide the outputs of this system and you will give clear instructions based on where are the obstacles and guide the user with short sentences, don’t output anything except guidance sentences. Here is the captured information of the scene(output only short sentences to guide user):

motorcycle at 5 meters ahead at half left
motorcycle at 3 meters ahead at half right user
should go forward

Now write a short sentence to guide the user.”

“There is a system to help blind people, this system captures the obstacles ahead the user via camera and calculates how far and how many angles left or right are this obstacles from user. You should guide the user. Now I will provide the outputs of this system and you will give clear instructions based on where are the obstacles and guide the user with short sentences, don’t output anything except guidance sentences. Here is the captured information of the scene(output only short sentences to guide user):

obstacle at 5 meters ahead at half left user
should go forward

Now write a short sentence to guide the user.”

After determining that LLM models do not work on numbers with the desired accuracy, the "left and right degree information" is converted into textual directions such as full left, half left, slightly left, forward, slightly right, half right, full right. And if no obstacle is detected by the models, the prompt is determined as “no obstacle detected, go forward” and is sent directly to the user, bypassing the LLM model.

Let's show the LLM outputs for the images on the left and right, respectively:

“Please proceed forward with caution, there is a motorcycle ahead at 5 meters and 3 meters to your left and right.”

“Please move forward slightly to avoid the obstacle at 5 meters ahead at left.”

Sample outputs such as these are sent back to the user over the internet to provide voice feedback to the user.

4. SYSTEM ARCHITECTURE, RELATIONSHIP AND DETAILS OF ELEMENTS

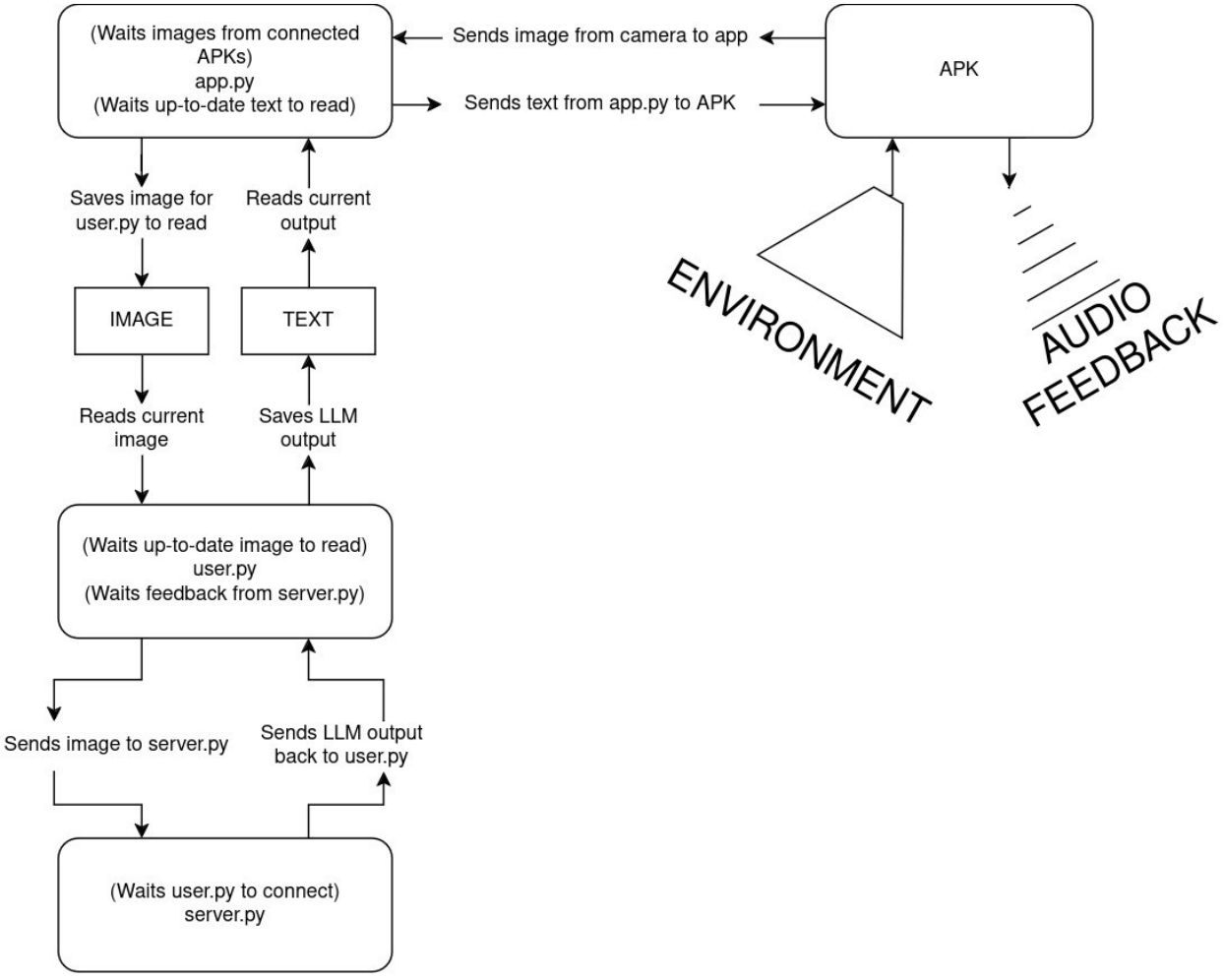


Fig. 3. System architecture

The system architecture consists of 4 main parts, app.py, user.py and server.py parts are controlled by the distributor; The user is only interested in the APK part. If we explain the operation of the system step by step:

1. APK sends the image captured from the phone camera to app.py over the internet
2. When app.py receives an image, waiting for connections from APKs, it processes the image and saves it on the shared disk for user.py's use.
3. Waiting for an updated image to be saved by app.py, user.py reads the image saved to disk and shares the image with server.py via RTSP connection in accordance with the TCP protocol.
4. server.py, which is waiting for a connection from user.py, starts a separate subprocess after receiving the image via RTSP and displays the LLM output obtained during the official processes and transactions, forwards it again to user.py over the same connection
5. user.py, which receives the LLM output from server.py, saves the LLM output to the shared disk for app.py to read.
6. Waiting for user.py to save an updated LLM output to disk, app.py reads the LLM output from disk and sends it to APK over the internet.
7. After receiving the APK LLM output awaiting text return from app.py, it gives voice feedback over the phone to warn the user.

All these processes continue in a loop, waiting for each other/current data.

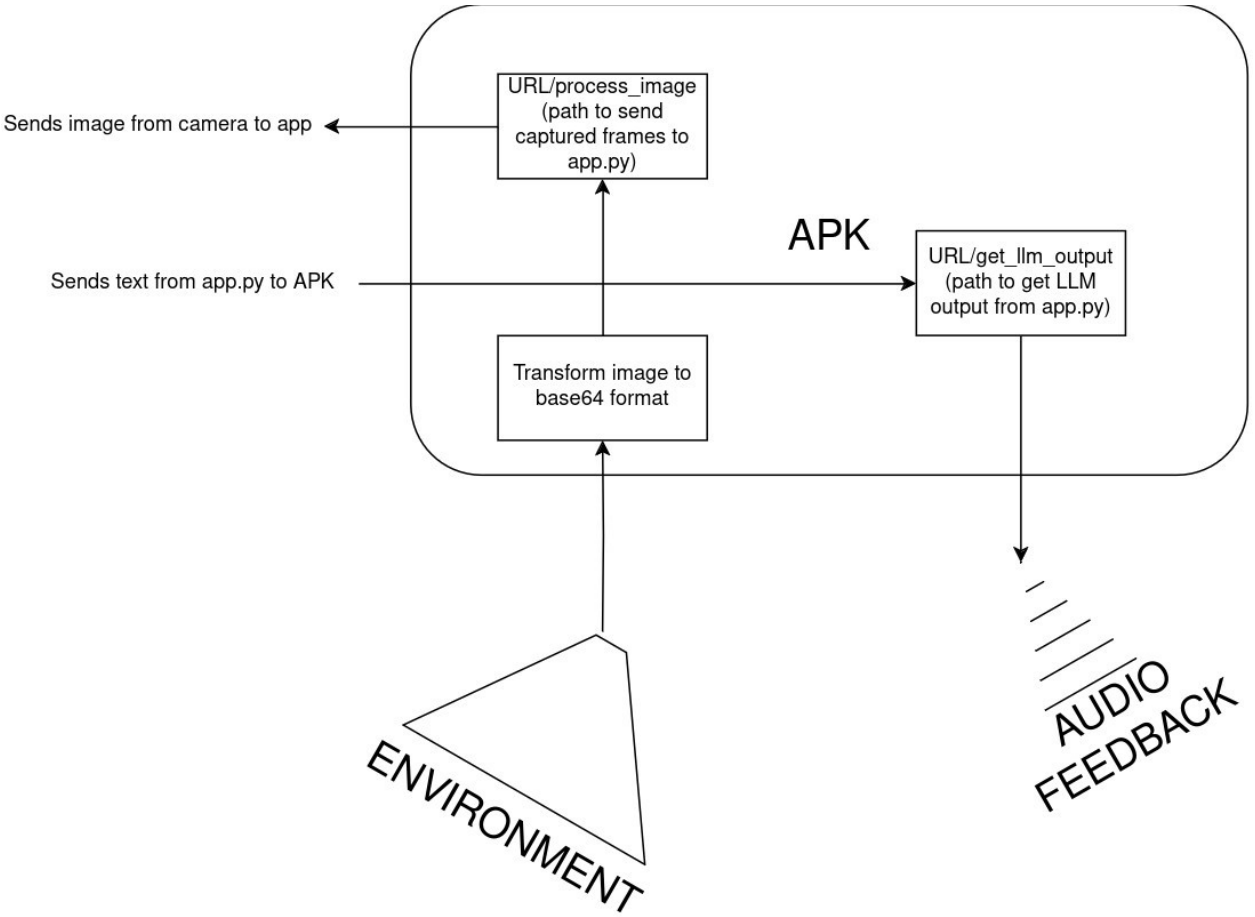


Fig. 4. APK-Backend Relation

If we examine the APK part in more detail, the following steps are followed when sending images and receiving text using the modules and packages mentioned above:

1. Image is taken from the phone camera.
2. The received image is converted to base64 format in order to transmit it more efficiently over the internet.

3. Data in base64 format is transmitted to the "process_image" path of the internet address to be connected to app.py.

1. Text is received from the "get_llm_output" path of the internet address to which app.py will be connected.
2. Audio feedback is given from the phone's audio output device.

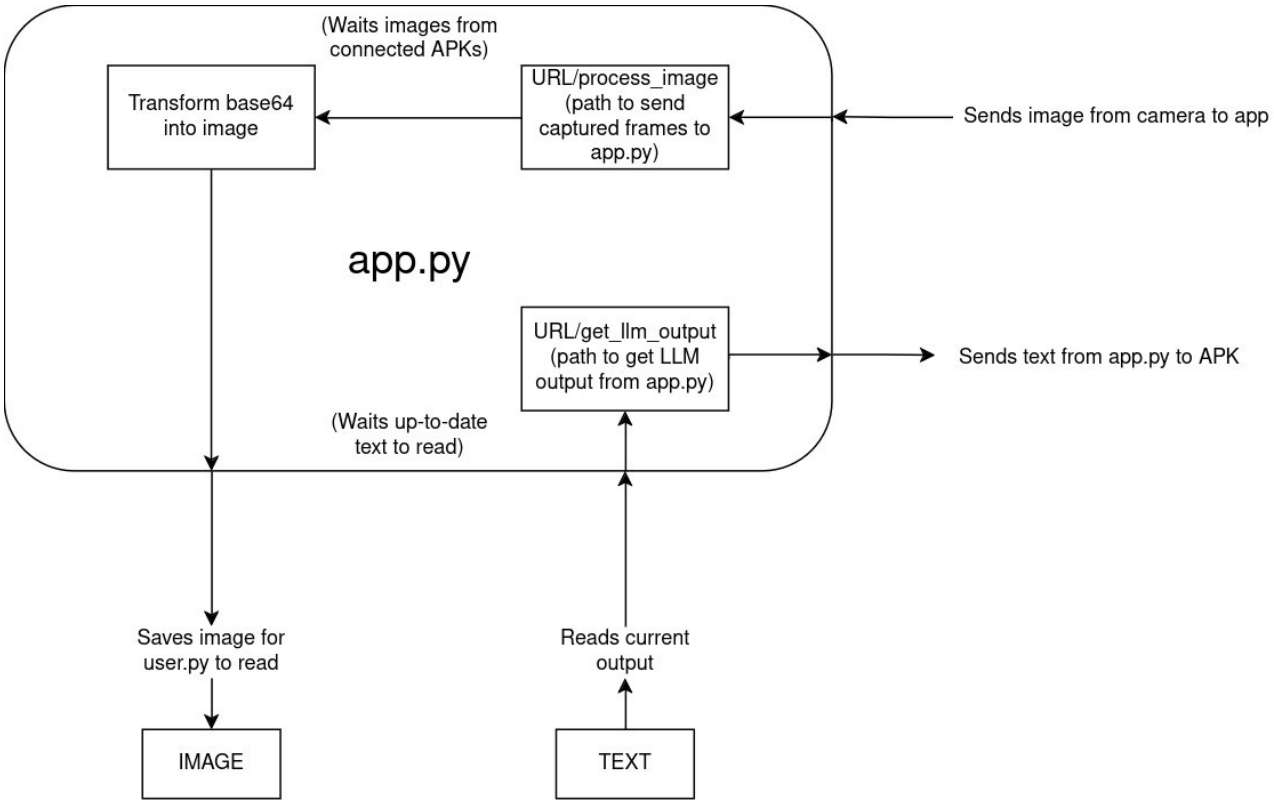


Fig. 5. APK Server-Backend Relation

If we examine the app.py part in more detail, the following steps are followed when receiving/saving images and reading/sending text using the modules and packages mentioned above:

1. An image is received in base64 format from the "process_image" path of the internet address contacted via the internet.
2. The image in base64 format is converted to RGB format.
3. The image in RGB format is saved to the public disk for use by user.py.

1. The last modification date of the text file on the shared disk is checked to see if an up-to-date LLM output has been recorded by user.py.
2. The file on the disk is read and sent to the APK via the "get_llm_output" path of the internet address.

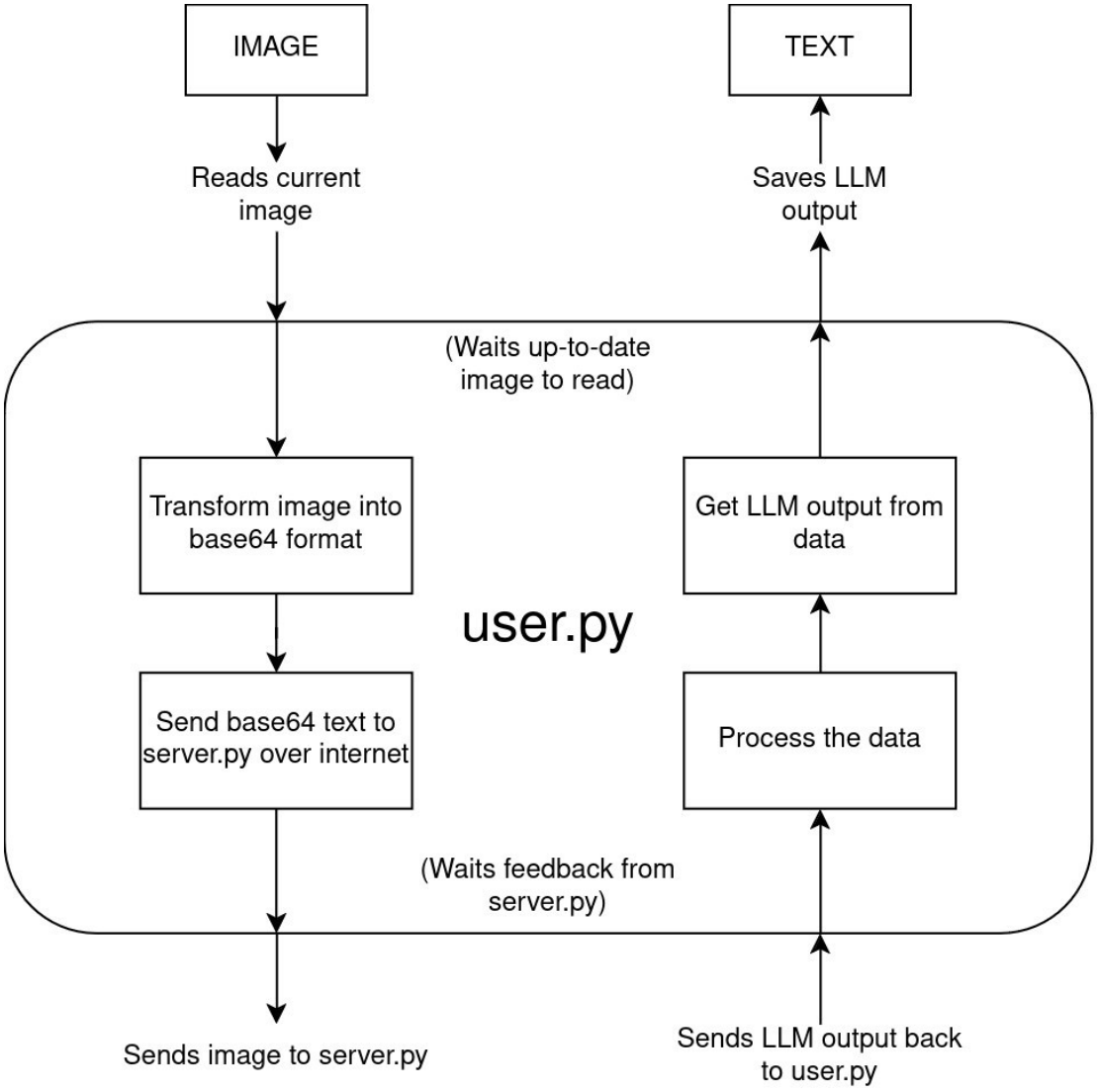


Fig. 6. Backend relations (user.py)

If we examine the user.py part in more detail, the following steps are followed when reading/sending images and receiving/saving text using the modules and packages mentioned above:

1. The last modification date of the image file on the shared disk is checked to see if an updated image has been saved by app.py.
2. The image read from the disk is converted to base64 format in order to be transmitted more efficiently over the internet.
3. An internet packet containing the image in base64 format is sent over the internet connection established with server.py.

1. Packet listening starts via the internet connection established with server.py.
2. When the internet package containing the LLM printout is received, the package is processed
3. The unpackaged text data is saved to the shared disk for app.py to read.

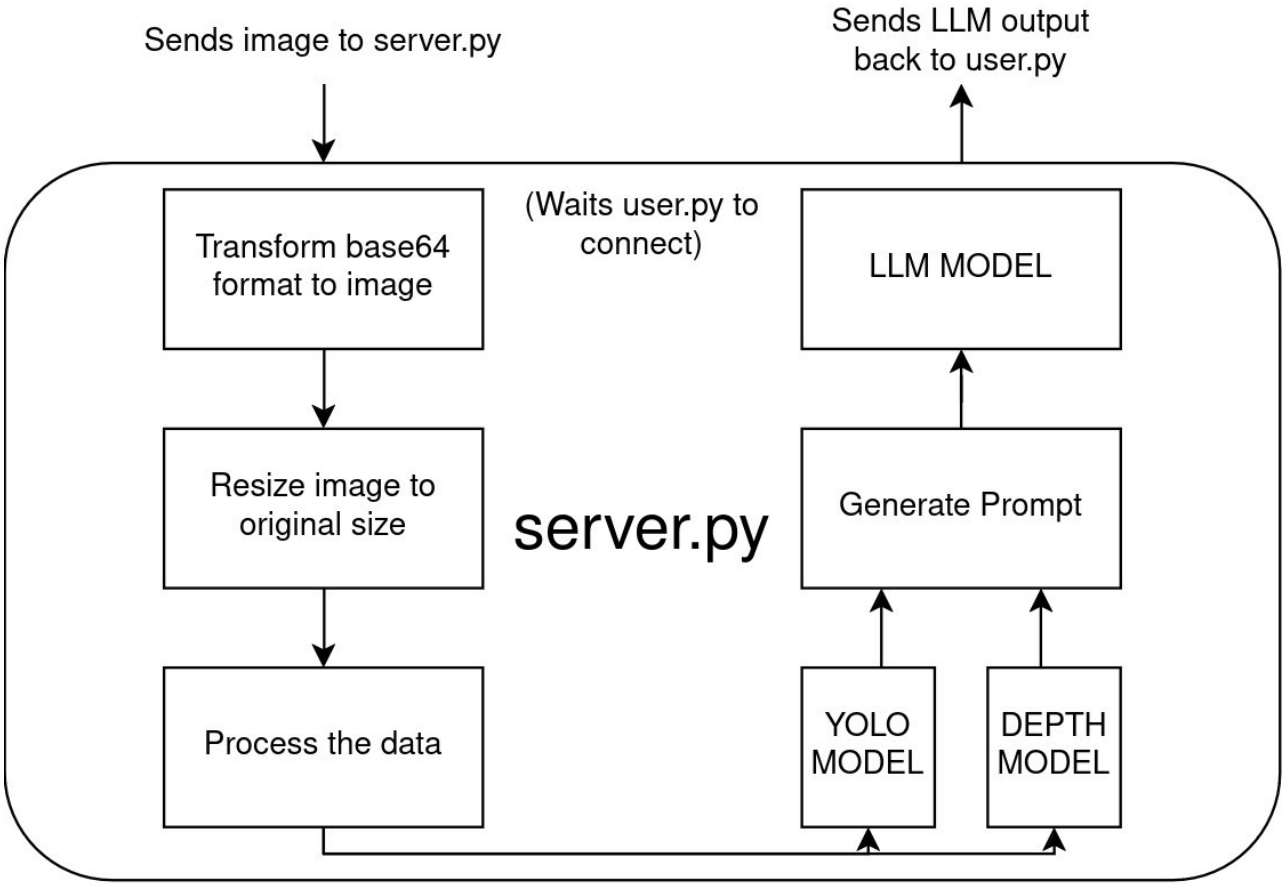


Fig. 7. Backend relations (server.py)

If we examine the server.py part in more detail, the following steps are followed using the above-mentioned modules and packages during image receiving/processing and text generation/sending:

1. Packet listening starts via the internet connection established with user.py.
2. The packet containing image data in base64 format is received and the packet is processed.
3. The unpackaged image is converted from base64 format to RGB format and processed back to its original size.
4. The resulting image is fed to the YOLO and DEPTH models.
5. By combining the data about the objects detected by the YOLO model with the depth information estimated by the DEPTH model, the user is obtained information about if there is an object nearby: how close and what it is to the right/left, or if there is an unidentified obstacle nearby: how close and to the right/left there is an obstacle. .
6. The scene information obtained with the previously tested prompts that give the desired type of output in the LLM model is combined and fed to the LLM.
7. An internet package containing the text received from LLM is sent over the internet connection established with user.py.

5. DIVIDE OF WORK WITHIN THE TEAM

Meriç DEMİRÖRS

- Researching pre-trained YOLO models and writing object detection, classification and localization codes.
- RTSP connection with TCP protocol to transfer images and text for communication between the user and server side processes on the server computer
- Writing codes by researching the LLM model that will run locally on the server computer and developing prompts in accordance with the project format,
- Writing codes that will run user and server-side operations on the server computer and embedding the project pipeline in these codes,
- Researching depth detection models suitable for the project and creating an environment suitable for testing,
- Temporarily adding a Text-to-Speech module

Emre BELİKIRIK

- Researching pre-trained Depth Estimation models
- Creating a depth map via the frame and returning the estimated distance of the detected object
- Researching pre-trained YOLO models and combining them with the depth model and writing the combined prediction code of Object Detection + Depth Estimation
- Conducting project suitability testing for different Depth and YOLO models
- Developing a structure that detects obstacles through the Depth model in a scenario where YOLO cannot detect objects

Zeynep Meriç AŞIK

- Researching pre-trained depth models
- Interface design of the mobile application
- Establishing a connection from the mobile application to the backend (sending frames from the phone camera)
- Connecting to the mobile application from the backend (getting LLM output)
- Mobile application voiceover (intro, startup and LLM output sound as text-to-speech)

REFERENCES

SOURCE FOR VISUALLY IMPAIRED PEOPLE STATISTICS:
[1] https://www.blindlook.com/tr/blog/detay/gorme-engelli-nufusu
[2] https://aile.gov.tr/media/130921/eyhgm_istatistik_bulteni_ocak_23.pdf

VISION MODELS FOR VISUALLY IMPAIRED PERSONS PAPERS:
MAGICEYE: AN INTELLIGENT WEARABLE TOWARDS INDEPENDENT LIVING OF VISUALLY IMPAIRED
[1] Obstacle Detection, Depth Estimation And Warning System For Visually Impaired People
[2] Embedded implementation of an obstacle detection system for blind and visually impaired persons
[3] Real-Time Object Detection for the Visually Impaired
[4] Assisting Blind People with Object Detection and Localization Using Deep Learning
[5] AssistiveGlasses: A Wearable Object Detection System for the Visually Impaired
[5] Machine Learning-Based Obstacle Detection and Navigation Assistance for Visually Impaired People

RGB TO DEPTHMAP PAPERS:
[1] LiteDepth: Digging into Fast and Accurate Depth Estimation on Mobile Devices
[2] AdaBins: Depth Estimation using Adaptive Bins
[3] BinsFormer: Revisiting Adaptive Bins for Monocular Depth Estimation
[4] DepthFormer: Exploiting Long-Range Correlation and Local Information for Accurate
[5] Monocular Depth Estimation
[6] Transformer-Based Attention Networks for Continuous Pixel-Wise Prediction