Autonomous Navigation System for the Visually Impaired

Project Report

Team HVBR : Vijay Aravynthan (2023111007), Bhaskar Itikela (2023111032), Harshil Singh (2023111003), Monosij Roy (2023111016)

Introduction

Our team has designed a working prototype for a device which can help visually impaired people navigate in real time in environments specially configured to work with the device. The device uses YOLO segmentation AI, a camera and an ultrasonic sensor and the code can be deployed even on a smartphone.

Problem Statement

The goal is to create an assistive system that can guide a visually impaired person safely through different environments using visual markers and cues. The system detects obstacles and indicates directions using color detection and object recognition.

Hardware/Software Used

- YOLOv11 Segmentation

  YOLOv11 is a high-performance deep learning model for object detection and segmentation with high accuracy and speed.
  We use YOLOv11 Segmentation to detect obstacles in the path of the person, and also use it to detect whether the person and we use color detection along with visual cues to determine whether the person can or should turn the right or left.

- Camera

  YOLOv8 gets its input from the main camera.

- HC-SR04 Ultrasonic Sensor

  We use this ultrasonic sensor alongside the camera for depth perception, i.e. to determine how far an object is from the person. We use the sensor to determine if there is a wall or some other obstacle in front of the person and to check whether the person can still move forward.

- ESP32

  An ESP32 is used to get readings from the Ultrasonic sensor and send these readings to the main program. Arduino C++ is used to program the ESP32.

- QIDK

  While it is not used for the working prototype, we have also looked into deploying the app on QIDK, which has a dedicated neural processing unit for such tasks. However, we weren't able to develop a new android app for this solution (and none of the given examples could be easily modified to work as our solution).

The main program is written in **Python**.

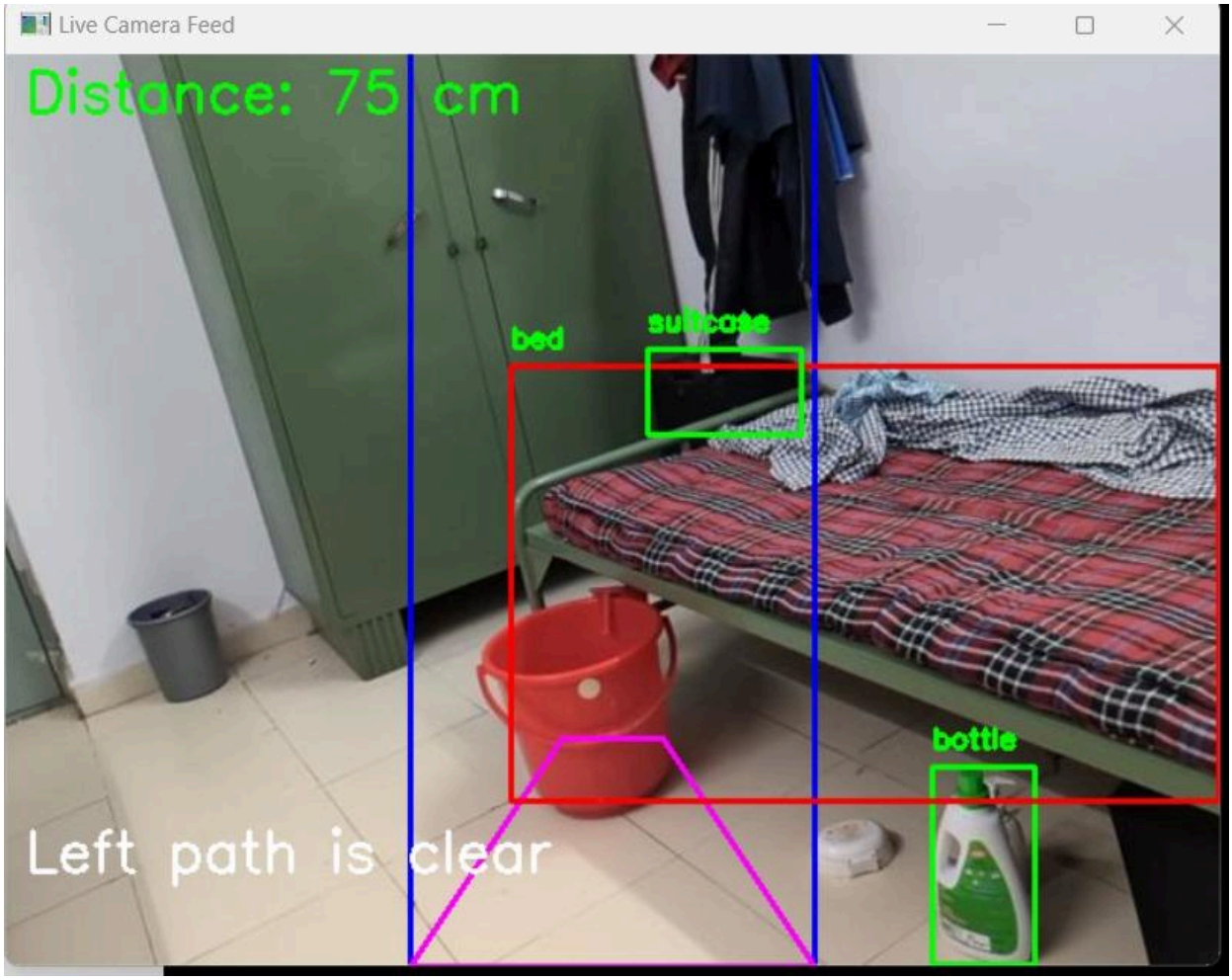
Working

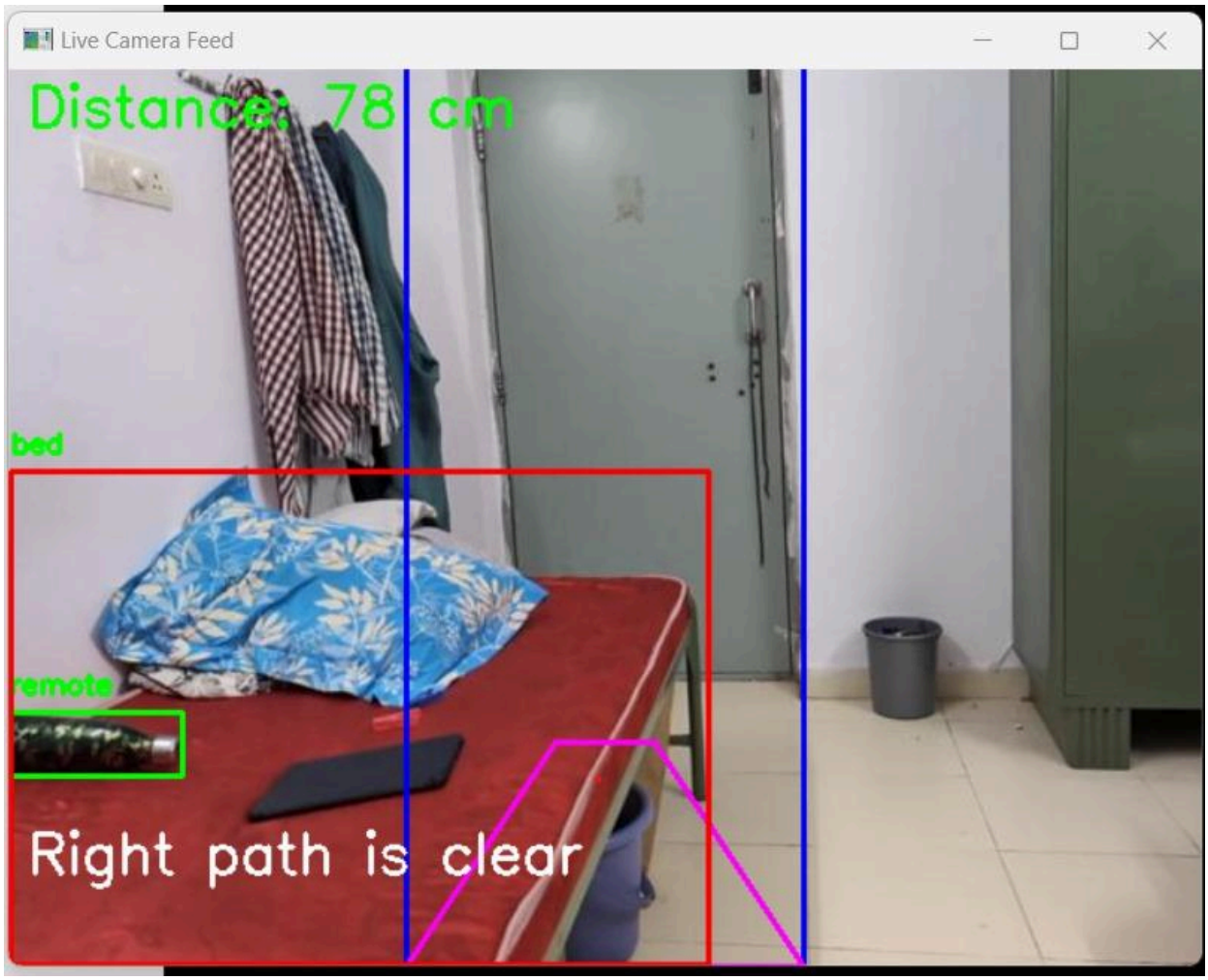
- Obstacle Detection
  - YOLOv11 detects any obstacles in front of the person, and also detects obstacles to the left or right to inform the person that left or right turns aren't possible.
  - If YOLOv11 doesn't detect any obstacle, (for e.g. if there is a wall), then the ultrasonic sensor readings are taken into account to detect if the person can move forward or not.
- Visual cues
  - YOLOv11 can also detect color-based visual cues to inform the person of the direction they can/should go.
  - Blue cues on both sides: The person can turn any way.
  - Blue cue on one side, red on other: The person should turn towards the blue cue.
  - Red cues on both sides: The person should not turn left or right.
  - We are trying arrow based cues as well.
- Audiovisual feedback
  - The program relays the necessary information about any obstacles and directions to the person by audio, and additionally also displays the information on the screen.

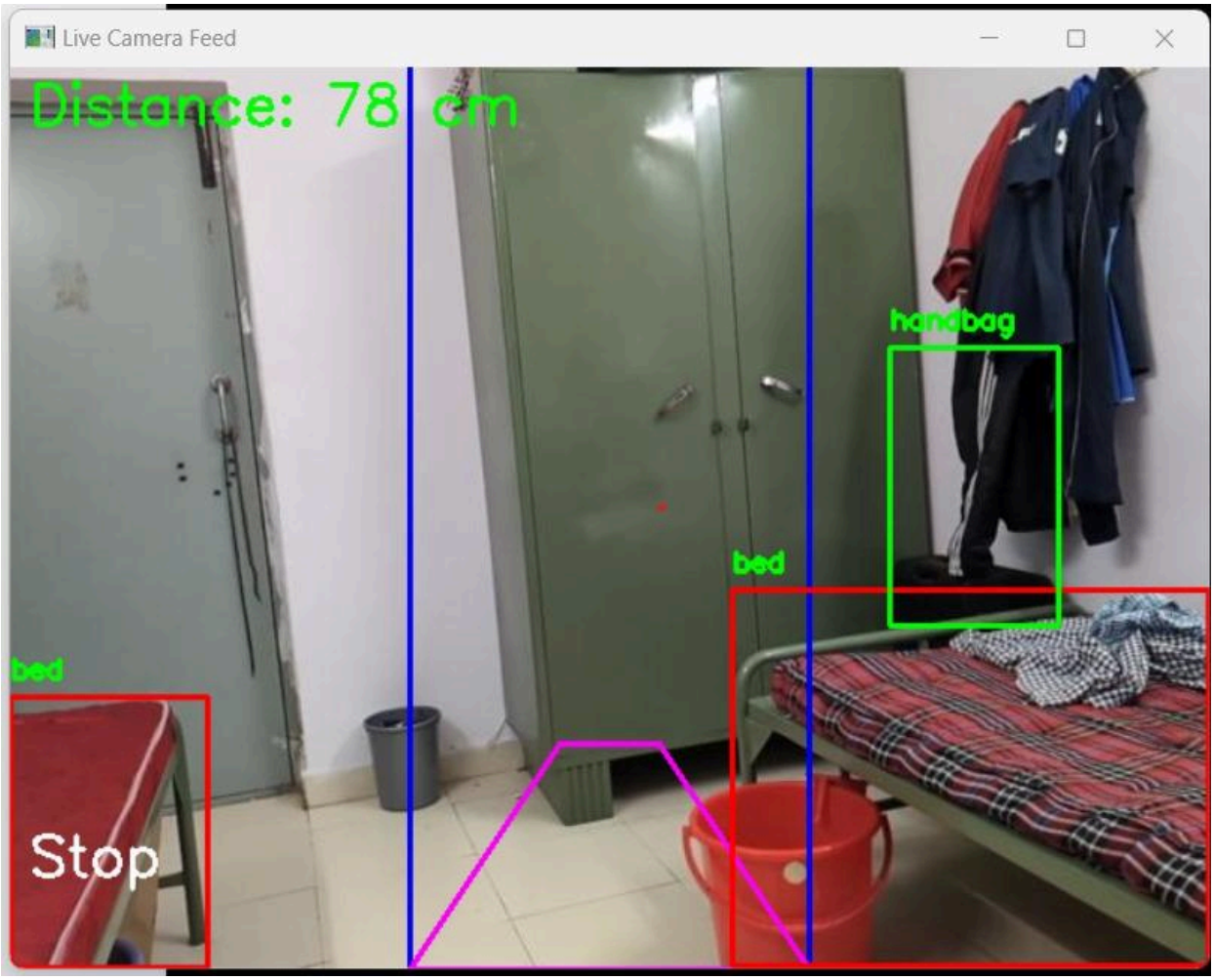
Examples

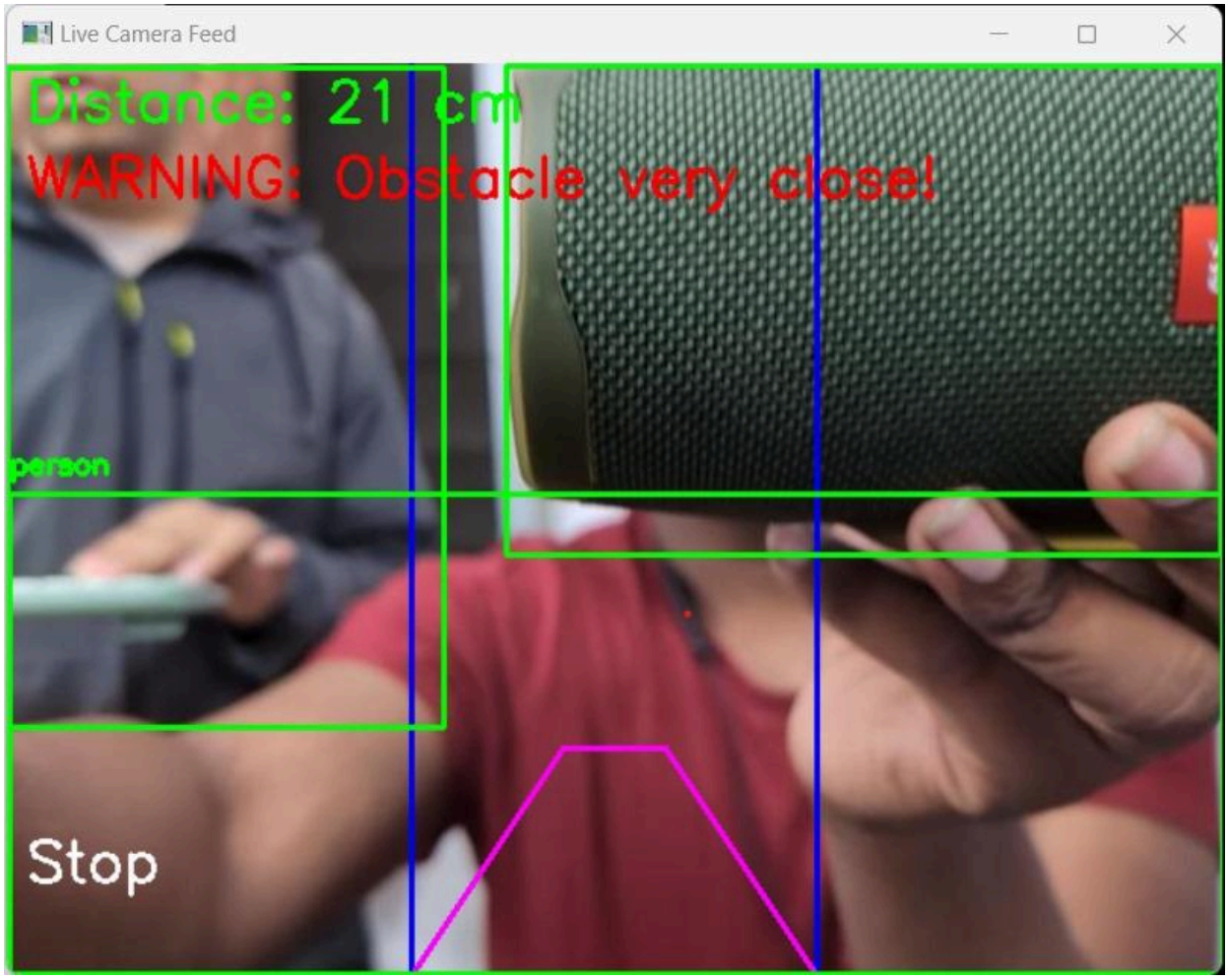
Note that inputs were modified specifically for demonstration purposes. Ultrasonic sensor warnings are displayed in red.

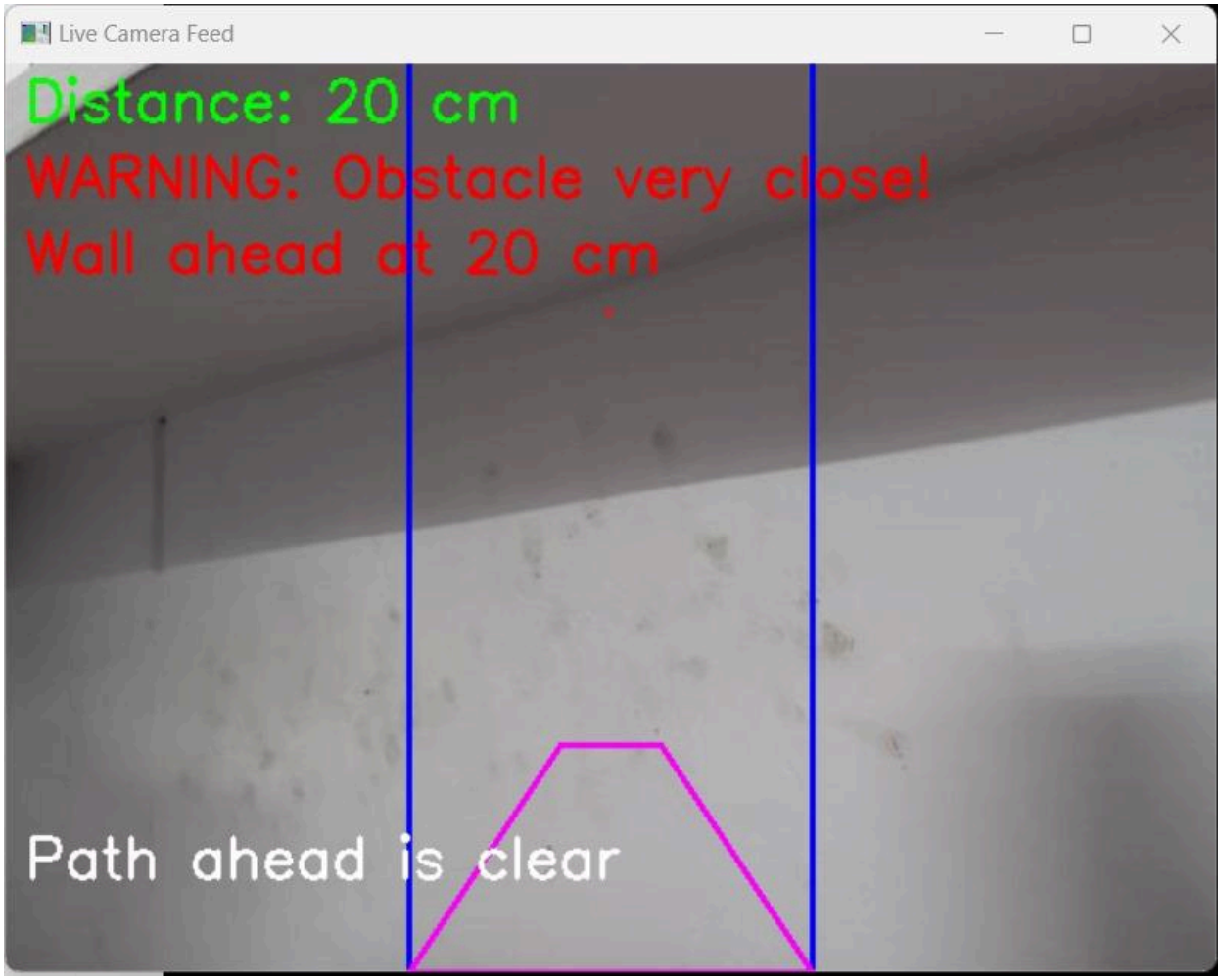

Live Camera Feed
Distance: 73 cm
bed
bed
cup
Path ahead is clear

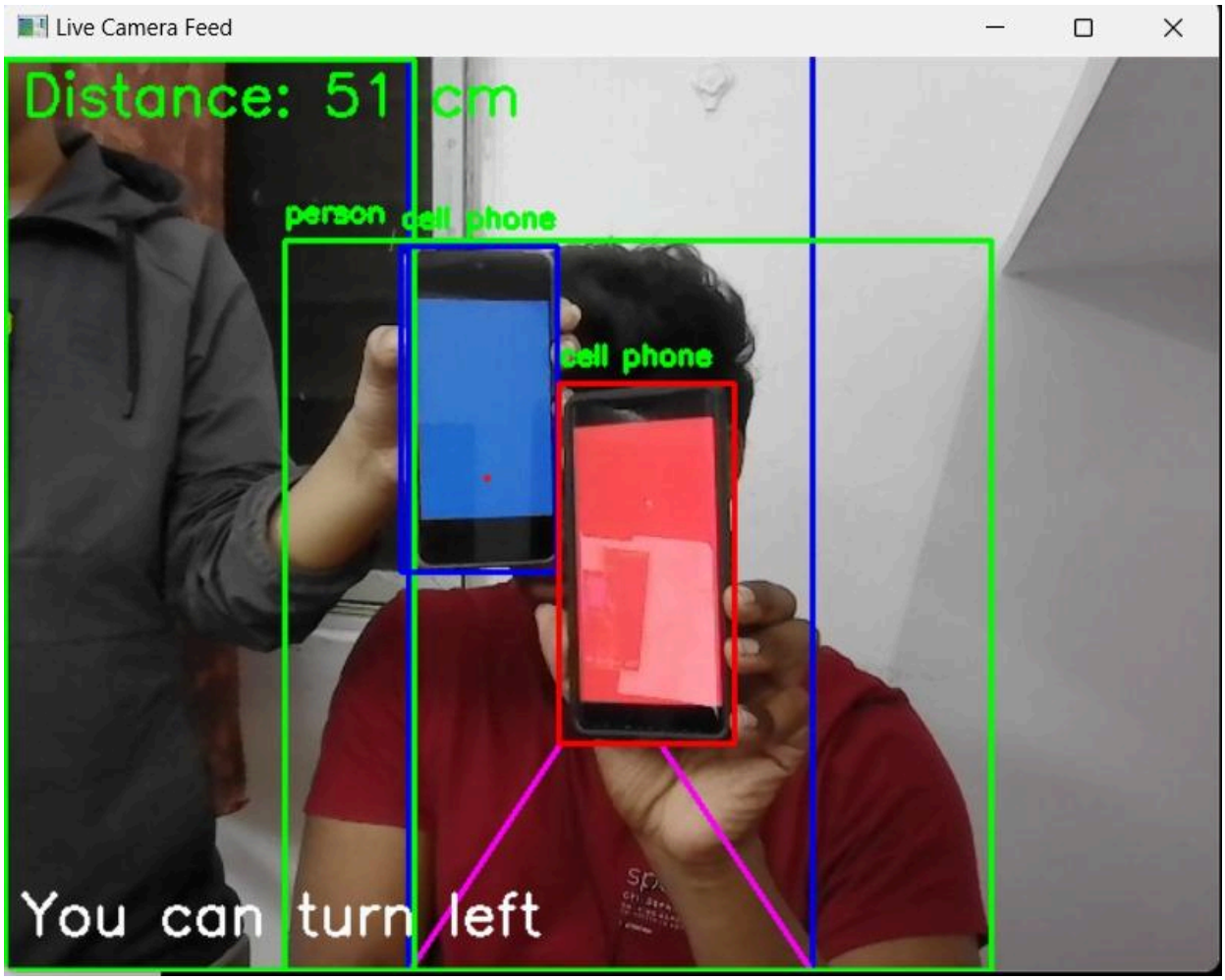

Live Camera Feed
Distance: 75 cm
bed
suitcase
bottle
Left path is clear

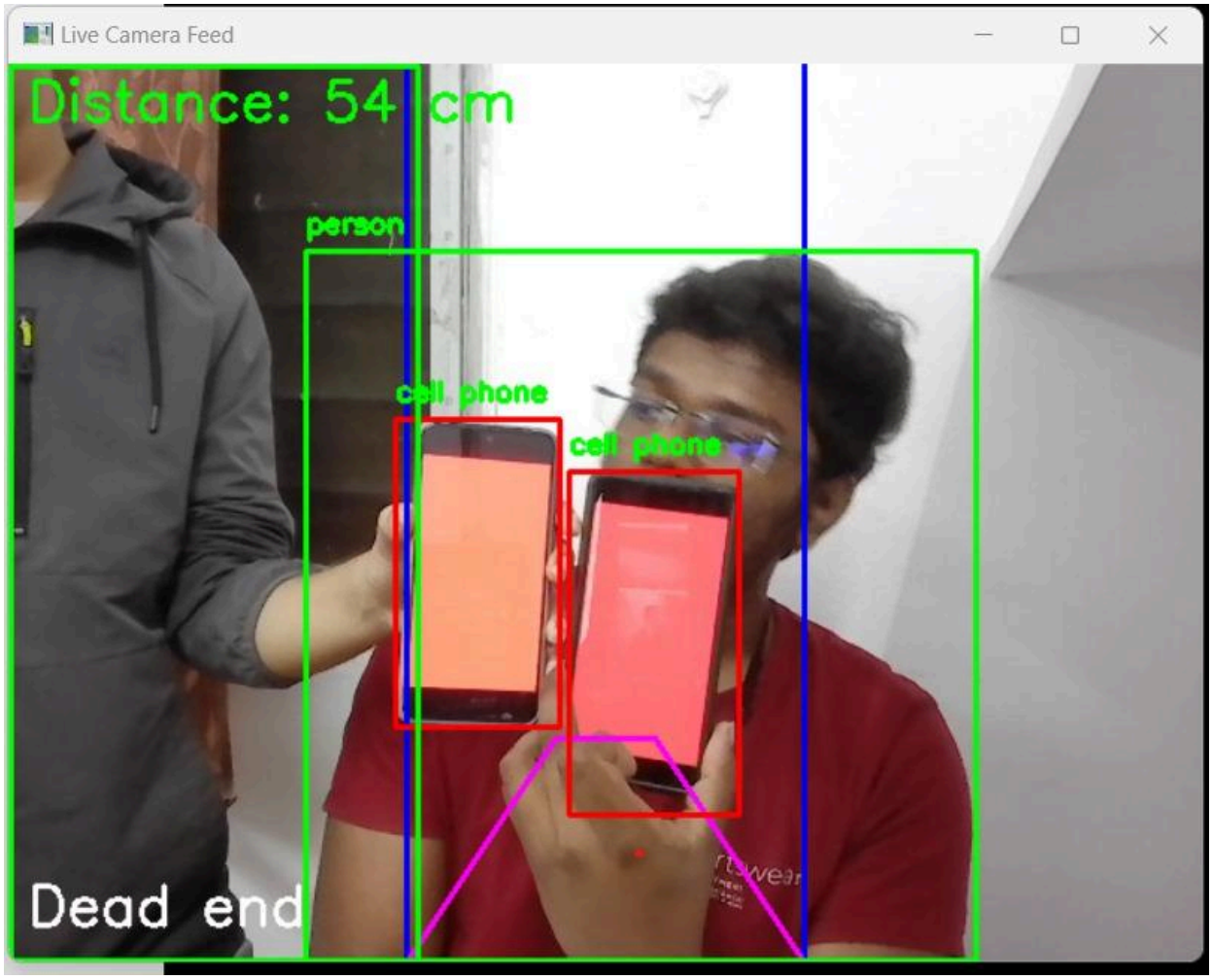

Live Camera Feed
Distance: 78 cm
bed
remote
Right path is clear

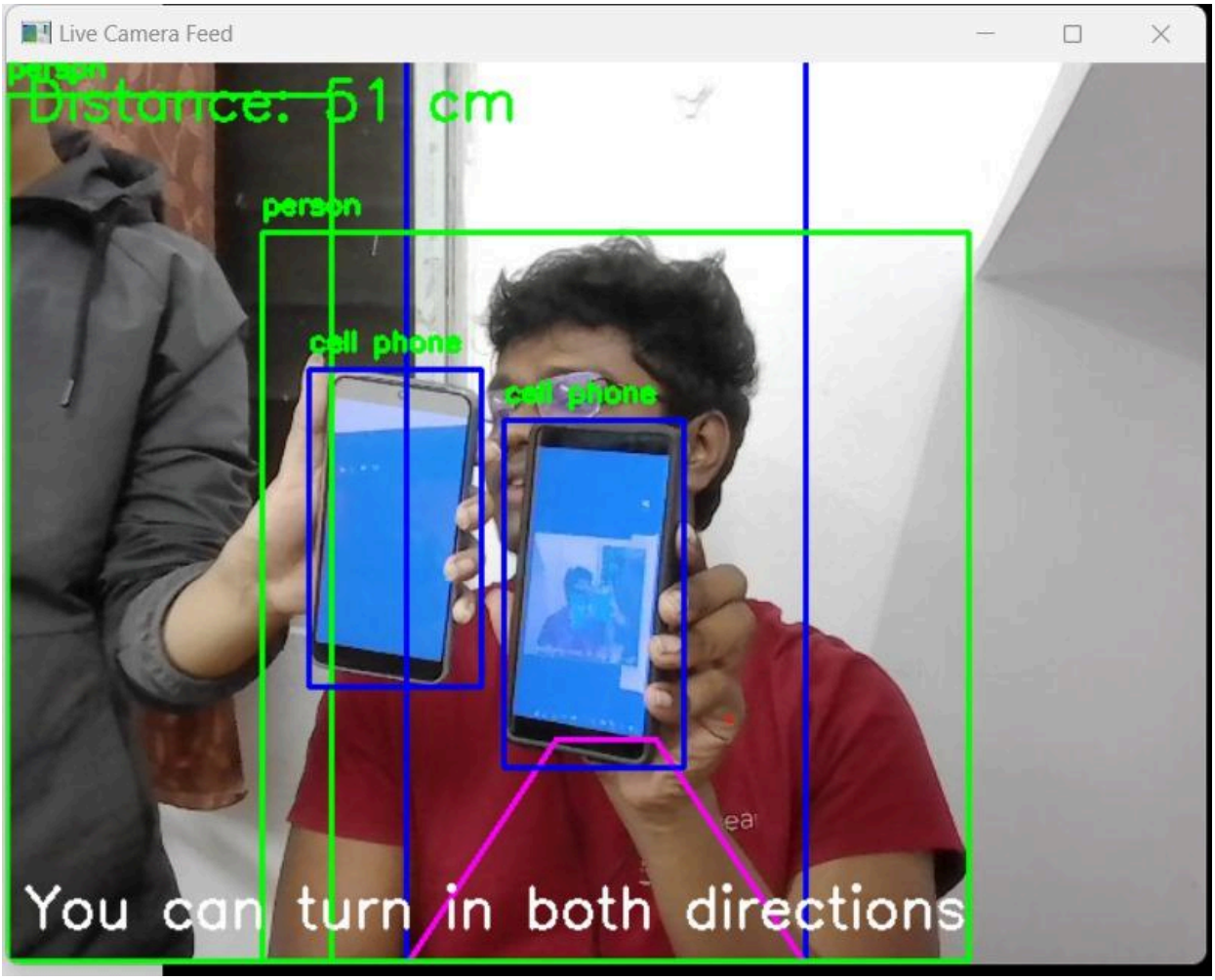

Live Camera Feed
Distance: 78 cm
handbag
bed
bed
Stop


Live Camera Feed
Distance: 21 cm
WARNING: Obstacle very close!
person
Stop


Live Camera Feed
Distance: 20 cm
WARNING: Obstacle very close!
Wall ahead at 20 cm
Path ahead is clear


Live Camera Feed
Distance: 51 cm
person
cell phone
cell phone
You can turn left


Live Camera Feed
Distance: 54 cm
person
cell phone
cell phone
Dead end



Use Cases

- Environments with proper visual cues
    - Buildings with required cues at intersections and dead ends to support accurate guidance.
    - Additionally, if the map of the building is known beforehand, then the program can be modified to guide the user to any destination within the building.
- Environments without any cues
    - In such environments, the program still works properly as an obstacle/wall detector.

Challenges/Assumptions

- Visual cues are necessary for proper guidance.
- Accuracy of the program depends upon the processing power of the hardware.
- The program has high power consumption since it records live camera feed and processes it in real time, therefore it will drain the battery quicker.
- ESP32 requires WiFi/Hotspot to send ultrasonic sensor readings to the main program.
- Lighting conditions can affect the accuracy of obstacle and colour detection. The program works best in well lit environments.

Future Direction

- The prototype can be turned into an APK and deployed on QIDK and even Android phones with sufficient processing power.
- The program can be modified to detect doors and convey this information to the user.
- Instead of an ultrasonic sensor, multiple cameras may be used for depth perception.

Progress Report

- Week 1 - Discussion of viable project ideas and selecting what AI models we would use
  We discussed problem statements such as People Density Monitoring and Management, Classroom HVAC,Light and Fan Automation, PPE Monitoring for IIIT Construction Sites, Intruder Detection System , Parking Management System, Smart Traffic Management System and Waste Segregation using Smart Bins. We discussed using various AI Models such as YOLOv8 , SAM2 etc.
- Week 2 - Finalizing project proposals and submitting
  We finalised the idea of using the YOLOv8 model for color detection and various utilities associated
  with color detection. We decided on our current problem statement after further discussion
- Week 3 - Learning to use YOLOv8 model
  We spent the week learning how to run the model locally on our devices and understanding the working
- Week 4 - Integrate YOLOv8 into program and make bounding boxes around colours and objects
  We learnt how to integrate the model into our code and ran a few test cases.
- Week 5 - Modify initial project proposal to use colour cues to aid visually impaired people in navigation
  After discussion with the professor, we decided to color cues as a viable solution since we had already been working on it.
- Week 6 - Define logic for navigation and make prototype using colour detection for navigation
  Came up with the logic for navigation and wrote code to implement the same
- Week 7 - Attempted porting the python solution into an Android app on QIDK
  It was now time to develop an android app, so we tried to build an android app locally and integrate the python solution
- Week 8 - Added audio feedback in addition to visual feedback and also implemented obstacle detection
  After further discussion with the professor , we add audio feedback as an additional feature in order to make the solution more feasible in real-life situations
- Week 9 - Integrated ultrasonic sensor to the program for better depth sensing
  We found that the android app was infeasible with our project. Since distance was still a parameter for consideration , we decided to use the ultrasonic sensor to measure distance in appropriate situations. Wrote code to integrate ultrasonic sensor working with our existing code
- Week 10 - Tested the program in various real life conditions
  Stress tested our code on various use cases

Contribution

- Vijay Aravynthan
    - YOLOv8 Integration
    - Navigation Logic
    - Report and PPT
    - Testing
- Bhaskar Itikela
    - YOLOv8 Integration
    - Navigation Logic
    - Look into Android deployment
    - Main script
    - Testing
- Harshil Singh
    - Integrate ultrasonic sensor
    - Report and PPT
    - Look into Android deployment
    - Testing
- Monosij Roy
    - Integrate ultrasonic sensor
    - Report and PPT
    - Look into Android deployment
    - Testing