Blind Assist Door Exit Detector Android App

Project Report

Radwa Badran
School of Computational Science and Artificial Intelligence
Zewail City for Technology and Science

May 16, 2025

Contents

1 Introduction

This project presents an Android application designed to assist visually impaired users in safely navigating indoor environments by detecting doors and obstacles. The application leverages real-time camera input, machine learning object detection, and audio guidance to provide a navigational aid.

The app integrates the CameraX API for live video capture, Google ML Kit with a custom TensorFlow Lite object detection model for recognizing doors and obstacles, and Android's TextToSpeech engine to deliver spoken navigation instructions.

2 System Architecture

2.1 CameraX for Video Capture

CameraX API is used to access the device's rear camera and provide a continuous stream of video frames. It supports lifecycle-aware camera usage to optimize resource management.

2.2 Object Detection with ML Kit

The app uses Google ML Kit's object detection framework configured with a custom TensorFlow Lite model ('object$_detection.tflite$'). $The detector is set to single-image mode, enabling multipleo$

2.3 Text-to-Speech Navigation Guidance

Android's TextToSpeech engine is initialized to provide auditory navigation prompts, warning users of detected obstacles and guiding them towards doors. The app manages speech cooldowns to avoid overlapping messages.

2.4 Overlay Visualization

For debugging and demonstration purposes, a custom view overlays bounding boxes and labels for detected objects on the camera preview, scaling detection results appropriately.

3 Implementation Details

3.1 MainActivity Overview

The core logic resides in the `MainActivity` class, which performs:

- Initialization of CameraX components and permissions.
- Setup of the custom ML Kit object detector.
- Frame analysis throttling to optimize performance.
- Real-time object detection and processing.
- Management of TextToSpeech audio feedback.
- Updating the overlay view with detected objects.

3.2 Key Code Snippets

3.2.1 Camera Permission Handling

```
private fun checkCameraPermission(): Boolean =
    ContextCompat.checkSelfPermission(this, Manifest.permission.CAMERA)
        == PackageManager.PERMISSION_GRANTED

private fun requestCameraPermission() {
    ActivityCompat.requestPermissions(this, arrayOf(Manifest.permission
       .CAMERA), CAMERA_PERMISSION_REQUEST)
}
```

Listing 1: Camera Permission Request

3.2.2 Custom Object Detector Setup

```
private fun setupCustomObjectDetector() {
    val localModel = LocalModel.Builder()
        .setAssetFilePath("object_detection.tflite")
        .build()

    val options = CustomObjectDetectorOptions.Builder(localModel)
        .setDetectorMode(CustomObjectDetectorOptions.SINGLE_IMAGE_MODE)
        .enableMultipleObjects()
        .enableClassification()
        .setClassificationConfidenceThreshold(0.7f)
        .setMaxPerObjectLabelCount(5)
        .build()

    objectDetector = ObjectDetection.getClient(options)
}
```

Listing 2: Setting up ML Kit Object Detector

3.2.3 Image Frame Analysis

```
private fun analyzeCameraFrame(imageProxy: ImageProxy) {
    val mediaImage = imageProxy.image ?: run {
        imageProxy.close()
        return
    }

    val inputImage = InputImage.fromMediaImage(mediaImage, imageProxy.
       imageInfo.rotationDegrees)
    objectDetector.process(inputImage)
        .addOnSuccessListener { detectedObjects ->
            handleDetectedObjects(detectedObjects, imageProxy.width,
               imageProxy.height)
        }
        .addOnFailureListener { e -> Log.e(TAG, "Detection failed", e)
           }
        .addOnCompleteListener { imageProxy.close() }
}
```

Listing 3: Analyzing camera frames with object detection

3.2.4 Navigation Guidance Logic

```
private fun generateNavigationGuidance(doors: List<DetectedObject>,
    obstacles: List<DetectedObject>) {
    if (doors.isEmpty()) {
        speakGuidance("Scanning for doors. Please move slowly.")
        return
    }

    val door = doors.first()
    val doorLabel = door.labels.firstOrNull()?.text ?: "door"
    val doorConfidencePercent = ((door.labels.firstOrNull()?.confidence
        ?: 0f) * 100).toInt()

    if (obstacles.isEmpty()) {
        speakGuidance("Walk straight. $doorLabel is ahead with
            $doorConfidencePercent% confidence.")
        return
    }

    val obstacle = obstacles.first()
    val obstacleLabel = obstacle.labels.firstOrNull()?.text ?: "object"
    val obstacleConfidencePercent = ((obstacle.labels.firstOrNull()?.
       confidence ?: 0f) * 100).toInt()

    val doorCenterX = door.boundingBox.centerX()
    val obstacleCenterX = obstacle.boundingBox.centerX()

    val guidanceMessage = if (obstacleCenterX < doorCenterX) {
        "Caution: $obstacleLabel detected on your left with
            $obstacleConfidencePercent% confidence. Move right to avoid
            and reach the $doorLabel."
    } else {
        "Caution: $obstacleLabel detected on your right with
            $obstacleConfidencePercent% confidence. Move left to avoid
            and reach the $doorLabel."
    }

    speakGuidance(guidanceMessage)
}
```

Listing 4: Generating spoken guidance based on detection

3.3 OverlayView Class

The `OverlayView` class extends Android's `View` to draw bounding boxes and labels over detected objects. It uses precomputed scaling factors to correctly map detection coordinates to the preview display size.

4 Conclusion

The Blind Assist Door Exit Detector app demonstrates a practical application of computer vision and AI technologies to aid visually impaired users in real-world navigation.

By integrating CameraX, ML Kit object detection, and TextToSpeech, the app provides an accessible and responsive tool to enhance mobility and safety indoors.

Future improvements could include expanding detection categories, refining obstacle avoidance logic, and enhancing audio guidance personalization.

5 References

- Google ML Kit: `https://developers.google.com/ml-kit/vision/object-detection`
- Android CameraX API: `https://developer.android.com/training/camerax`
- TensorFlow Lite: `https://www.tensorflow.org/lite`
- Android TextToSpeech API: `https://developer.android.com/reference/android/speech/tts/TextToSpeech`