Project Report: Real-Time Object Detection for Door and Obstacle Identification

1 Introduction

This project develops an Android application designed to assist users in identifying doors and obstacles using real-time object detection. Leveraging the YOLOv8 model, the application processes live camera feeds and provides audio feedback, making it particularly useful for visually impaired individuals or hands-free navigation scenarios.

2 Model Description

The application employs the YOLOv8 model, a state-of-the-art object detection algorithm known for its efficiency and accuracy. YOLO (You Only Look Once) processes images in a single pass, making it ideal for real-time applications. The specific variant used here, YOLOv8-TFLite, is optimized for mobile devices using TensorFlow Lite, ensuring efficient performance on Android hardware.

The model is trained to detect various objects listed in the `labels.txt` file, with a focus in this project on identifying doors and obstacles that may impede the user's path.

3 Code Description

The application is structured into several key components, each serving a distinct purpose. Below is an overview of the major files, including code snippets with comments.

3.1 MainActivity.kt

This file serves as the application's entry point, managing camera setup, object detection, and audio feedback.

```
// Initializing camera and text-to-speech in onCreate
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    binding = ActivityMainBinding.inflate(layoutInflater)
    setContentView(binding.root)
    cameraExecutor = Executors.newSingleThreadExecutor()
    tts = TextToSpeech(this, this) // Initialize TextToSpeech
    // ... (Permission checks and camera start)
```

```
}

// Processing detections and providing audio feedback
override fun onDetect(boundingBoxes: List<BoundingBox>, inferenceTime: Long) {
    runOnUiThread {
        binding.overlay.apply {
            setResults(boundingBoxes) // Draw boxes on screen
            invalidate()
        }
        var guidanceText = ""
        // Logic to determine door position and obstacles
        if (doorDetected) {
            guidanceText = when {
                doorCenterX < 0.35f -> "Door is to your left."
                doorCenterX > 0.65f -> "Door is to your right."
                else -> "Door is ahead."
            }
        }
        speak(guidanceText) // Speak the guidance
    }
}
```

3.2 Detector.kt

Handles the YOLOv8 model inference and post-processing of detections.

```
// Running the model on a frame
fun detect(frame: Bitmap) {
    val resizedBitmap = Bitmap.createScaledBitmap(frame, tensorWidth,
        tensorHeight, false)
    val tensorImage = TensorImage(INPUT_IMAGE_TYPE)
    tensorImage.load(resizedBitmap)
    val processedImage = imageProcessor.process(tensorImage)
    interpreter.run(processedImage.buffer, output.buffer)
    val bestBoxes = bestBox(output.floatArray)
    // Notify listener with results
}
```

3.3 Other Components

- **BoundingBox.kt**: Data class for detected object properties.
- **OverlayView.kt**: Custom view to draw bounding boxes on the screen.
- **Constants.kt**: Defines model and label file paths.
- **activity_main.xml**: UI layout (XML).
- **AndroidManifest.xml**: Declares permissions and components.

4 Results

The application is expected to:

- Detect doors and obstacles in real-time using the device's camera.
- Draw bounding boxes around detected objects on the screen.
- Provide audio feedback, e.g., "Door is to your left" or "Obstacle detected, it's a chair in front."

Without actual execution logs, these outcomes are inferred from the code logic and model capabilities.

5 Discussion

5.1 Effectiveness

The application should perform well in controlled environments with clear visibility, leveraging YOLOv8's robust detection capabilities.

5.2 Improvements

- Fine-tune the model on a dataset tailored to specific environments.
- Add depth estimation for distance information.
- Support multiple languages for audio feedback.

5.3 Challenges

- Handling varying lighting conditions.
- Ensuring performance on low-end devices.
- Managing battery consumption during prolonged use.

6 Conclusion

This project integrates advanced object detection into a mobile application, offering visual and audio assistance for identifying doors and obstacles. Future enhancements could expand its functionality and performance across diverse devices.