Coffee Machine Assistance for the Visually Impaired: A Computer Vision-Based Mobile Application

by

Murad Huseynov

Bedir Esen

Hazem Moustafa

Alperen Dalgıç

Submitted to Computer Engineering

TED University

Table of Contents

Abstract

This thesis addresses the challenge of independent coffee machine operation for visually impaired individuals by developing a computer vision-based mobile application. The research problem lies in the inaccessibility of modern coffee machines.

The objective of this project is to design, implement, and evaluate a mobile application that enables visually impaired users to operate a specific coffee machine model independently.

To achieve this, a mobile application will be developed using JavaScript and React Native, integrating the EfficientDet object detection model and platform-specific Text-to-Speech engines (Android TTS and IOS VoiceOver). Data will be manually collected and augmented to train the EfficientDet model for real-time coffee machine recognition.

Key results will demonstrate the feasibility of accurate coffee machine recognition and effective voice guidance. Usability testing with visually impaired individuals will be conducted to indicate that the application provides a user-friendly and accessible method for operating the target coffee machine.

In conclusion, this research shows that a computer vision-based mobile application can offer a practical solution for coffee machine assistance, enhancing independence for visually impaired users.

Limitations of this project include the current focus on a single coffee machine model and the need for further optimization for diverse machine types and user needs. Future work should focus on expanding model compatibility and incorporating user feedback for continuous improvement.

List of Abbreviations

AI	Artificial Intelligence
API	Application Programming Interface
AT	Assistive Technology
BiFPN	Bidirectional Feature Pyramid Network
CNN	Convolutional Neural Network
COCO	Common Objects in Context (dataset)
CPU	Central Processing Unit
CV	Computer Vision
GPU	Graphics Processing Unit
IoU	Intersection over Union
mAP	Mean Average Precision
NNAPI	Neural Networks API
POUR	Perceivable, Operable, Understandable, Robust (principles)
R-CNN	Region-based Convolutional Neural Network
SIFT	Scale-Invariant Feature Transform
SSD	Single Shot Detector
TTS	Text-to-Speech
UCD	User-Centered Design
UI	User Interface
UX	User Experience
WCAG	Web Content Accessibility Guidelines
YOLO	You Only Look Once (object detection algorithm)

Glossary

Albumentations: A Python library for fast and flexible image augmentations, used to enhance training datasets for computer vision models.

Bidirectional Feature Pyramid Network (BiFPN): A feature network architecture used in EfficientDet that allows multi-directional information flow for better feature fusion.

Convolutional Neural Network (CNN): A class of deep neural networks most commonly applied to analyzing visual imagery in computer vision applications.

Depthwise Separable Convolutions: A type of convolution operation that factorizes a standard convolution into a depthwise convolution and a pointwise convolution, reducing computation and model size.

Detox: An end-to-end testing and automation framework for mobile apps, used for testing React Native applications.

EfficientDet: An object detection model architecture that uses compound scaling to achieve both high accuracy and efficiency, making it suitable for resource-constrained environments.

EfficientNet: A convolutional neural network architecture that uniformly scales all dimensions of depth, width, and resolution using a compound coefficient, serving as the backbone for EfficientDet.

Intersection over Union (IoU): A metric used to evaluate the overlap between predicted and ground truth bounding boxes in object detection.

Mean Average Precision (mAP): A metric used to evaluate the accuracy of object detectors, calculated by taking the mean average precision across all classes and/or IoU thresholds.

MobileNet: A lightweight deep neural network architecture designed for mobile and embedded vision applications, characterized by depthwise separable convolutions.

Neural Networks API (NNAPI): Android's API for hardware-accelerated machine learning operations, allowing apps to run computationally intensive neural networks on mobile devices.

Post-training Quantization: A technique that reduces model size and improves CPU and hardware accelerator latency by converting floating-point representations to more efficient formats after model training.

Transfer Learning: A machine learning technique where a model developed for one task is reused as the starting point for a model on a second, related task, reducing training time and improving performance.

Weight Pruning: A technique to reduce model size and computational requirements by removing less important connections (weights) in a neural network.

Chapter 1: Introduction

The daily routine of brewing coffee, a simple pleasure for many, presents a significant obstacle for individuals with visual impairments. Modern coffee machines, with their visually dependent interfaces and controls, create barriers to independent operation. This thesis addresses the need for assistive technologies that can bridge this accessibility gap, focusing on empowering visually impaired individuals to confidently use everyday appliances like coffee machines and enhance their daily independence and quality of life.

1.1 Leveraging Computer Vision for Mobile Assistance

Existing assistive technologies for appliance operation often rely on generic solutions or costly modifications. This thesis explores a more innovative and accessible approach: utilizing computer vision (CV) and smartphone technology. We focus on developing a mobile application that uses the smartphone camera to "see" and interpret coffee machine interfaces. This project aims to create a specific, real-time guidance system for a chosen coffee machine model, providing precise voice instructions for its operation. The scope of this research is the design, development, and evaluation of a CV-based mobile application to assist visually impaired individuals in independently operating a defined coffee machine model through voice guidance.

1.2 Building Upon Existing Assistive Technology Research

Assistive technology research has explored various methods for appliance accessibility, from tactile aids to auditory feedback and voice control. However, many solutions are limited in adaptability, integration ease, or real-world usability. While voice and gesture interfaces offer potential, challenges in robustness and accuracy remain. This thesis builds upon this foundation by investigating CV – a rapidly advancing field – to address coffee machine operation specifically. By using visual understanding and real-time guidance, this project aims to contribute a novel, potentially more universally applicable, and context-aware approach to appliance accessibility.

1.3 Research Questions and Project Objectives

This dissertation investigates the feasibility and effectiveness of a CV-based mobile application for coffee machine assistance, guided by the following questions:

1. Can a mobile CV application accurately recognize a coffee machine model in real-time?
2. Can voice instructions effectively guide visually impaired users to make coffee on the recognized machine?
3. Is the application usable, accessible, and helpful for independent coffee machine operation by visually impaired individuals?
4. What are the key design challenges and effective mitigation strategies for such an assistive application?

To answer these questions, this project aims to achieve the following objectives:

1. Design and implement an EfficientDet-based CV system for coffee machine recognition.
2. Develop a voice guidance system using Android TTS and iOS VoiceOver for clear coffee machine operation instructions.
3. Integrate these systems into an accessible mobile application.
4. Thoroughly test and evaluate the application's performance and usability with visually impaired users.

1.4 Technical Approach

To address the challenges outlined above, this thesis employs a technical approach centred around efficient deep learning for CV and accessible mobile application development. At its core, the application utilizes the **EfficientDet** [1] object detection model, chosen for its state-of-the-art balance between accuracy and computational efficiency, making it well-suited for real-time performance on mobile devices. The **MobileNet** [2] architecture serves as the backbone for EfficientDet, further enhancing its efficiency and reducing computational overhead. The model is trained using **TensorFlow/Keras** [3], leveraging its powerful deep learning capabilities, and subsequently optimized and deployed on mobile devices using TensorFlow Lite for efficient on-device inference.

For image processing and data augmentation, the project integrates **OpenCV** [4], a widely used CV library. Critically, the application's user interface and voice guidance system are developed using **React Native and JavaScript**, enabling cross-platform deployment to both Android and iOS platforms from a single codebase. This choice of React Native facilitates wider accessibility and future reach of the application. Voice guidance is implemented using the platform-specific Text-to-Speech engines (**Android TTS** [5] and **iOS VoiceOver** [6]), ensuring native accessibility support on each platform.

Furthermore, the application utilizes **SQLite** for local, on-device storage of structured data such as user settings and application preferences, and local file storage for efficient management and retrieval of the trained **TensorFlow Lite** models [7], ensuring offline functionality and data privacy. This technical stack is strategically selected to achieve high accuracy in coffee machine recognition, provide real-time and responsive voice guidance across platforms, and maximize accessibility and user reach while balancing development efficiency.

Chapter 2: Literature Review

2.1 Assistive Technology for Visually Impaired Individuals

Assistive technology (AT) has revolutionized the lives of individuals with visual impairments, enhancing their independence and improving their ability to perform daily tasks. These technologies range from simple tactile aids to advanced artificial intelligence-powered applications. The development of accessible solutions is particularly significant in the context of household management, where tasks such as appliance operation and kitchen activities can pose significant challenges.

In the case of kitchen appliances, for example, the Be My Eyes [8] app connects visually impaired users with sighted volunteers through video calls, providing real-time assistance for various tasks. While applications like Be My Eyes offer valuable assistance, they rely on human volunteers and may not always provide immediate or consistent guidance. Furthermore, such solutions, while helpful, do not provide the same level of independent skill development as systems that directly empower users to operate appliances themselves.

This thesis explores a more direct and autonomous approach through CV, aiming to provide immediate and consistent guidance without reliance on external human assistance, and promoting greater self-sufficiency in daily living.

2.2 Computer Vision for Accessibility

CV has emerged as a revolutionary technology in tackling accessibility challenges, especially for individuals with visual impairments. By allowing machines to interpret and comprehend visual information, CV enhances capabilities such as object recognition, scene understanding, and human-computer interaction. This progress significantly improves the independence and quality of life for visually impaired users.

One of the most impactful applications of CV in enhancing accessibility is object recognition. Mobile applications like Seeing AI [9] and Envision AI [10] leverage deep learning models to identify everyday objects, read text and even recognize faces. While they are groundbreaking for general environmental awareness, these applications often have limited functionality without an internet connection and can struggle when it comes to interacting with complex objects like kitchen appliances. Recognizing a coffee machine is different from identifying and understanding the function of its specific buttons, lights and interaction points needed for operation - a critical limitation this thesis aims to address.

2.3 Object Detection Algorithms

The core technical challenge in enabling CV-based appliance interaction lies in accurate and efficient object detection deployable on mobile devices.

Traditional methods like Haar cascades [11] and feature descriptors like SIFT [12] have limitations in robustness to variations in lighting, viewpoint, and object complexity, making them less suitable for diverse real-world appliance interfaces. Deep learning-based approaches, especially Convolutional Neural Networks (CNNs), have revolutionized the field.

These models can be broadly categorized into two-stage detectors like Faster R-CNN [13], known for high accuracy but slower speeds, and one-stage detectors like SSD [14] and YOLO [15], prioritizing speed, and efficiency. Two-stage detectors offer higher accuracy but are computationally expensive for mobile devices. One-stage detectors prioritize speed and efficiency, making them popular for mobile applications, but sometimes sacrifice accuracy, particularly for detecting small or overlapping objects - a key concern when identifying small buttons on a coffee machine.

EfficientDet represents a significant advancement by optimizing the accuracy-efficiency trade-off. Leveraging the EfficientNet backbone, Bi-directional Feature Pyramid Network (BiFPN), and compound scaling, EfficientDet models achieve competitive accuracy while maintaining high efficiency suitable for mobile platforms. Compared to models like YOLO which might prioritize sheer speed, EfficientDet often demonstrates superior accuracy, particularly for reliable detection of small control elements required for safe and effective appliance operation guidance in this project. While optimization (using TensorFlow Lite) is still required, EfficientDet provides a solid foundation for achieving the necessary balance between reliable detection and real-time mobile performance.

2.4 Voice Guidance and Text-to-Speech Systems

TTS technology is fundamental, converting recognized object information and instructional steps into spoken output. TTS is used in various applications, such as screen readers, navigation assistance, object and text recognition, and voice-controlled devices. For our project, voice guidance will assist users in operating a coffee machine by providing step-by-step spoken instructions, helping them navigate the process without needing visual input.

Cloud-based TTS (like Google Cloud TTS [16]) services offer highly natural voices but require constant internet connectivity and raise potential privacy concerns. On-device TTS engines, such as the native Android TTS Engine or iOS VoiceOver, guarantee offline functionality, crucial for an assistive application that must be dependable regardless of network status. They also offer lower latency and enhanced privacy. While potentially less natural-sounding than cloud alternatives, the paramount need for reliability, consistency, and offline access in an assistive context makes on-device TTS the preferred choice for this project, ensuring the user can always access guidance.

2.5 User-Centred Design and Accessibility Principles

User-centered design (UCD) [17] prioritizes understanding and involving the intended users (in this case, visually impaired individuals) throughout the design process. Key UCD principles include early and continuous user involvement, iterative design driven by user feedback, and a primary focus on usability and user experience. This approach ensures that assistive technologies are not only technically sound but also genuinely meet user needs and are practical for real-world use. For our application, UCD will guide the development of voice guidance and interaction to be intuitive and effective for visually impaired users in a coffee-making context.

The Web Content Accessibility Guidelines (WCAG) [18] provide a robust framework, particularly the POUR principles:

> **Perceivable:** Information must be presentable to users in ways they can perceive. For this project, this means ensuring all visual information is conveyed effectively through clear auditory output.
>
> **Operable:** Interface components and navigation must be operable. The application prioritizes simple interaction models, relying primarily on auditory output and minimal, accessible input methods compatible with screen readers.
>
> **Understandable:** Information and the operation of the user interface must be understandable. This translates to clear, simple language in voice instructions and a logical, predictable interaction flow.
>
> **Robust:** Content must be robust enough that it can be interpreted reliably by a wide variety of user agents, including assistive technologies. This involves using standard development practices and ensuring compatibility with platform screen readers.

2.6 Identifying the Gap and Justifying the Research

This review reveals significant progress in AT and CV, yet a critical gap persists: the lack of robust, accessible, and autonomous mobile solutions specifically designed to guide visually impaired individuals through the interaction steps required for operating complex household appliances like modern coffee machines. Existing general-purpose CV apps lack the necessary granularity and contextual understanding for safe operation, while human-in-the-loop solutions don't foster true independence. Traditional AT methods struggle with the diversity of modern appliance interfaces.

This thesis directly addresses this gap by proposing and developing a mobile application that integrates state-of-the-art, efficient object detection (EfficientDet) tailored for recognizing specific coffee machine controls with clear, sequential, on-device voice guidance. Grounded in UCD and accessibility principles, this project aims to provide a practical, reliable, and user-validated solution. Its contribution lies in demonstrating the feasibility of using targeted CV and mobile technology to create a context-aware assistive tool that enhances independence in a specific, challenging daily living task, moving beyond generic object recognition towards interactive task assistance.

Chapter 3: Methodology

3.1 System Design and Architecture

Overall Architecture: The system consists of four primary modules:

1. **Input Module:** Handles camera input with interval-based image capture, implemented using React Native's camera capabilities with platform-specific optimizations for Android and iOS.
2. **Computer Vision Module:** Processes the captured images to detect and identify coffee machine components using the EfficientDet model with MobileNet backbone.
3. **Voice Guidance Module:** Converts detection results into contextual voice instructions using native TTS engines.
4. **User Interface Module:** Provides an accessible interface optimized for screen readers.

Data Flow: The application's data flow follows a streamlined path from camera capture through preprocessing, CV processing, contextual analysis, to voice output and user interaction.

1. The camera maintains a continuous preview feed for user positioning while capturing discrete high-quality images at strategic intervals.
2. Captured images undergo preprocessing including resizing, normalization, and format conversion appropriate for the neural network input.
3. The CV module processes these images through the TensorFlow Lite-optimized EfficientDet model, identifying coffee machine components and their spatial relationships.
4. The detection results feed into the contextual processing unit, which maintains state awareness of the coffee-making process and determines the appropriate next instruction.
5. The voice guidance module converts the contextual instruction into spoken output using the device's native TTS capabilities.
6. User interactions (through accessible touch gestures or voice commands) are processed by the UI module and may trigger state changes or adjust image capture intervals.

Modular Design Benefits: The modular architecture was selected for several compelling reasons:

1. **Development Efficiency:** The separation of concerns allows specialized team members to work independently on their areas of expertise (CV, TTS, mobile development).

2. **Maintenance and Updates:** Individual modules can be updated, optimized, or replaced without significant impact on other components.
3. **Cross-Platform Compatibility:** React Native facilitates development for both Android and iOS platforms, while the modular approach allows for platform-specific optimizations where necessary.
4. **Scalability:** The architecture supports future expansion to additional coffee machine models by swapping or extending the CV model without restructuring the entire application.
5. **Offline Functionality:** The complete on-device processing ensures the application works reliably without internet connectivity, critical for assistive technology.
6. **Performance Optimization:** The modular approach enables targeted optimization of computation-intensive components (particularly the CV module) without compromising the entire application's responsiveness.

3.2 Computer Vision Implementation

Model Selection and Justification: EfficientDet was selected as the primary model for this application due to its optimal balance of accuracy and computational efficiency.

1. **Accuracy-Efficiency Trade-off**: EfficientDet models achieve comparable or superior accuracy to alternatives like Faster R-CNN while requiring significantly fewer computational resources, critical for real-time mobile performance.
2. **Scalable Architecture**: The compound scaling method of EfficientDet allows for selecting an appropriate model size based on the device's computational capacity.
3. **Small Object Detection**: Coffee machines feature numerous small control elements (buttons, dials, indicators) that must be reliably detected. EfficientDet's BiFPN (Bidirectional Feature Pyramid Network) excels at detecting small objects compared to single-stage detectors like SSD or YOLO.
4. **TensorFlow Lite Compatibility**: EfficientDet models can be effectively optimized using TensorFlow Lite for mobile deployment without significant accuracy degradation.
5. **Single-Image Processing Effectiveness**: EfficientDet performs well on individual frames without requiring temporal information, making it well-suited for our interval-based capture approach.

Data Collection Strategy: The model training relies on a comprehensive dataset specific to the target coffee machine. To obtain the dataset, our strategy includes:

1. **Controlled Image Capture**: Systematic photography of the target coffee machine from various angles (0°, 45°, 90°, etc.), distances (close-up, medium, full view), and under different lighting conditions (natural light, indoor lighting, low light).

2. **Real-world Usage Scenarios**: Video recordings of actual operation sequences, from which key frames are extracted to ensure the model learns relevant operational states.
3. **Environmental Variation**: Images capturing the coffee machine in different realistic environments (kitchen countertops, office break rooms) with varying backgrounds and potential obstructions.
4. **User Perspective Emphasis**: Special attention to capturing images from angles and distances that represent realistic usage scenarios by visually impaired users, including slightly off-center perspectives and variable distances.
5. **Interval-Simulation Data**: Collection includes sequence sets that mimic the expected interval-based capture pattern to ensure the model can handle sequential state recognition from discrete images rather than continuous video.

Data Augmentation: To enhance model robustness without requiring excessive manual data collection, several augmentation techniques are applied:

1. **Geometric Transformations**: Random rotations (±15°), horizontal flips, slight perspective transformations, and crops to simulate different viewing angles and positions.
2. **Photometric Augmentations**: Adjustments to brightness (±25%), contrast (±20%), saturation, and hue to simulate different lighting conditions and camera variations.
3. **Noise and Blur Addition**: Gaussian noise, motion blur, and compression artifacts to enhance robustness against lower-quality camera inputs.
4. **Background Augmentation**: Synthetic background replacements to increase environmental diversity.
5. **Simulated Motion Effects**: Slight motion blur and position shifts to simulate handheld device movement between interval captures.
6. **Implementation Details**: Implemented using Albumentations library [20], Configuration for preserving bounding box validity post-transformation, metadata tracking for experimental analysis

Preprocessing:

- Resizing images to the input dimensions required by EfficientDet
- Normalization to the [0,1] range
- Color space adjustments for consistency
- JPEG artifact removal for cleaner input

Model Training Procedure: The EfficientDet model training follows a systematic approach:

1. **Transfer Learning**: Initializing from pre-trained weights (trained on the COCO [19] dataset) to leverage generalized feature detection capabilities.
2. **Custom Class Definition**: Defining specific classes for coffee machine components (e.g., power button, water reservoir, coffee dispenser, control panel, etc.).

3. **Training Regime**:
    - Initial feature extraction phase: Training only the classification and box regression heads while freezing the backbone
    - Fine-tuning phase: Gradually unfreezing and training deeper layers of the network
4. **Validation Strategy**: Implementing k-fold cross-validation to ensure model robustness, with 80% of data used for training and 20% for validation in each fold.
5. **Hyperparameter Optimization**: Grid search for optimal batch size, learning rate, and regularization parameters appropriate for the dataset size.
6. **Sequence Recognition Training**: Additional training on sequential image sets captured at intervals to enhance the model's ability to recognize state transitions from discrete captures rather than continuous video.

Mobile Optimization and Deployment: To ensure efficient processing of captured images on mobile devices, several optimization techniques are applied:

1. **TensorFlow Lite Conversion**: Converting the trained model to TensorFlow Lite format with post-training quantization to reduce model size and accelerate inference.
2. **Weight Pruning**: Applying structured and unstructured pruning to remove redundant connections without significant accuracy loss.
3. **Hardware Acceleration**: Leveraging platform-specific hardware accelerators:
    - Android: Neural Networks API (NNAPI) and GPU delegates
    - iOS: Core ML and Metal Performance Shaders
4. **Inference Optimization**:
    - Parallel preprocessing of images while previous image is still being analyzed
    - Adaptive resolution based on device capabilities and detection confidence
    - Early-stopping for non-relevant images
    - Confidence thresholding to filter uncertain detections
5. **Memory Management**: Implementing efficient buffer reuse and minimizing unnecessary tensor copies during inference, with particular attention to proper cleanup of processed images.
6. **Interval Optimization**: Dynamic adjustment of image capture intervals based on:
    - Current processing speed
    - Detection confidence levels
    - Battery status
    - Stage of the coffee-making process (more frequent captures during critical operations)

3.3 Voice Guidance System Implementation

Text-to-Speech Integration: The application leverages platform-specific TTS engines to ensure optimal performance and accessibility:

1. **Platform-Specific Implementation:**
    - For Android devices: native Android TTS engine through the TextToSpeech API

- For iOS devices: VoiceOver TTS via the AVSpeechSynthesizer framework

2. **Cross-Platform Abstraction Layer:**
    - A React Native bridge provides a unified interface to both TTS implementations
    - The abstraction layer handles platform-specific initialization, configuration, and error management
3. **Speech Configuration Optimization:**
    - Voice selection prioritizes clarity and intelligibility over naturalness
    - Speech rate is configured at approximately 175-200 words per minute, identified as optimal for comprehension in assistive technology research
    - Pitch and volume parameters are exposed to user configuration to accommodate individual preferences and hearing needs
4. **Offline Functionality:**
    - All TTS processing occurs locally on the device without cloud dependencies
    - Pre-packaged voice data ensures immediate availability without downloads

Voice Instruction Design: The design of voice instructions follows established principles:

1. **Instruction Structure**:
    - Clear action verbs begin each instruction ("Press," "Turn," "Locate")
    - Spatial references use clock positions familiar to visually impaired users ("at 2 o'clock position")
    - Confirmation phrases follow actions ("You should hear a beep when pressed correctly")
    - Contextual awareness phrases maintain orientation ("You are now at the control panel")
2. **Language Optimization**:
    - Concise phrasing eliminates unnecessary words while maintaining clarity
    - Consistent terminology throughout the guidance process
    - Implementation of user-centered vocabulary testing to verify comprehension
3. **Auditory Hierarchy**:
    - Primary instructions delivered with standard voice parameters
    - Critical warnings or cautions indicated through subtle pitch adjustments
    - Progress confirmations differentiated with slight speed variations
4. **Continuous Feedback Loop**:
    - Acknowledgment of detected state changes ("Water tank has been removed")
    - Periodic reorientation cues during extended operations
    - Encouraging feedback upon successful completion of steps

Sequential Guidance Logic: The voice guidance system implements a state machine architecture to manage the coffee-making process:

1. **State Representation**:
    - Each operational state (e.g., "MachineOff," "WaterReservoirRemoved," "BrewingInProgress") is explicitly modeled

    - State transitions occur based on CV detection results and elapsed time parameters
    - The current state determines available instructions and expected next states
2. **Decision Tree Implementation**:
    - A hierarchical decision tree maps detected machine states to appropriate guidance
    - The tree includes fallback paths for handling uncertain or transitional states
    - Decision nodes incorporate confidence thresholds from the CV system
3. **Progress Tracking**:
    - Linear sequence tracking for standard operations
    - Branch handling for optional features or user choices
    - Loop management for repetitive tasks
    - Checkpoint verification to ensure critical steps are not missed
4. **Error Handling and Recovery**:
    - Detection of incorrect user actions through state inconsistencies
    - Contextual correction instructions based on expected vs. actual state
    - Progressive assistance escalation for persistent issues
    - Capability to reset to known states when sequence is broken
5. **Timing and Pacing Logic**:
    - Dynamic instruction timing based on task complexity
    - Pause insertion between sequential instructions
    - User-configurable pacing options
    - Automatic timing adjustments based on user interaction patterns

3.4 Mobile Application Development

Development Framework and Platform Support: The application is developed using React Native to enable cross-platform deployment while maintaining native performance:

1. **Technology Stack**:
    - React Native framework for cross-platform core functionality
    - JavaScript for application logic
    - Platform-specific modules for native integration:
        - Native Modules for Android integration
        - Native Extensions for iOS integration
2. **Development Environment**:
    - React Native CLI for project scaffolding and management
    - Jest and React Native Testing Library for unit and component testing
    - Detox [21] for end-to-end testing across platforms
3. **Cross-Platform Considerations**:
    - Shared codebase for core functionality
    - Platform-specific implementations for camera interfaces, TTS engines, and accessibility services
    - Conditional rendering and API abstraction to handle platform differences

4. **Deployment Targets**:
    - Android 7.0 and above
    - iOS 13 and above

Accessibility-First UI/UX Design:

1. **Interaction Model**:
    - Large, semantically meaningful touch targets (minimum 48×48dp)
    - Simple linear navigation flow optimized for screen readers
    - Minimal required user input with clear audio confirmation
    - Gesture-based shortcuts for experienced users (double-tap, swipe patterns)
2. **Screen Layout Principles**:
    - High contrast elements (minimum 4.5:1 contrast ratio)
    - Simple, consistent layout patterns across screens
    - Strategic use of haptic feedback to complement audio cues
    - Layout optimization for TalkBack and VoiceOver focus order
3. **Accessibility Service Integration**:
    - Comprehensive content descriptions for all UI elements
    - Custom TalkBack/VoiceOver actions for specialized functions
    - Announcements for dynamic content changes
    - Live regions for critical status updates
4. **User Customization Options**:
    - Adjustable speech rate for voice guidance
    - Font size and contrast settings for low-vision users
    - Vibration intensity for haptic feedback
    - Instruction detail level preference

Camera Integration:

1. **React Native Camera Integration**:
    - Implementation using react-native-camera with enhanced accessibility hooks
    - Camera permission handling with clear explanatory prompts
    - Automatic orientation detection and adjustment
    - Configuration for optimal CV input requirements
2. **Interval-Based Frame Capture System**:
    - Continuous camera preview for user positioning but with interval-based image capture
    - Strategic sampling of frames at configurable intervals
    - Dynamic interval adjustment based on:
        - Detection confidence (shorter intervals when machine is partially detected)
        - Device performance capabilities
        - Battery level considerations
    - Capture triggers for significant scene changes detected through lightweight preprocessing

3. **Image Processing Pipeline**:
    - Dedicated processing queue for captured frames
    - Priority-based processing for frames with higher likelihood of containing relevant features
    - Buffering system to maintain application responsiveness during intensive processing
    - Memory-efficient management of captured images with automatic cleanup
4. **User Guidance for Camera Positioning**:
    - Audio cues for optimal distance and angle
    - Vibration patterns to indicate when coffee machine is in/out of frame
    - Simple verbal directions for repositioning ("move slightly right," "move closer")
    - Confidence-based feedback to confirm stable positioning
5. **Accessibility Considerations**:
    - Audio confirmation of camera activation
    - Privacy indicators with audio announcement
    - Camera timeout with warning to conserve battery
    - Emergency shortcuts to exit camera mode

Integration of Computer Vision and Voice Guidance:

1. **Component Communication**:
    - Event-based communication system between modules
    - Redux state management for application-wide state coordination
    - Debounced detection events to prevent instruction flooding
    - Priority queue for voice instructions to handle concurrent detections
2. **Processing Pipeline**:
    - Background processing of captured frames to maintain UI responsiveness
    - Parallel execution of detection and speech synthesis where possible
    - Caching of recent detection results to reduce redundant processing
    - Adaptive processing based on device capability detection
3. **Performance Optimization**:
    - Selective GPU utilization based on available hardware
    - Dynamic resolution adjustment to maintain efficient processing
    - Background task scheduling for non-critical operations
    - Battery usage monitoring with power-saving mode options
4. **Offline Functionality**:
    - Local storage of machine recognition models
    - Cached instruction sets for continuous operation
    - No dependency on cloud services for core functionality

3.5 Data Collection

Dataset Requirements and Planning: Before beginning data collection, specific requirements were established to ensure dataset quality and coverage:

1. **Target Volume**:
    - Initial collection target of 500 base images
    - Post-augmentation target of 1,500-2,000 training samples
    - Validation set comprising 20% of the total dataset
2. **Diversity Requirements**:
    - Multiple coffee machine states (off, standby, brewing)
    - Comprehensive component coverage (buttons, reservoirs, dispensers, displays)
    - Environmental variations (different locations, backgrounds, lighting conditions)
    - Perspective variations (different angles and distances)
3. **Quality Parameters**:
    - Minimum resolution of 1080×720 pixels
    - Clear focus on machine components
    - Consistent labeling schema across all images
    - Detailed metadata for each capture (lighting conditions, distance, angle)

Image Collection Methodology: The data collection follows a structured approach to ensure comprehensive coverage:

1. **Controlled Environment Collection**:
    - Studio-like setup with controlled lighting conditions
    - Systematic capture of the coffee machine from predefined angles (0°, 45°, 90°, 135°, 180°, 225°, 270°, 315°)
    - Multiple distances (30cm, 60cm, 100cm) to simulate different usage scenarios
    - Controlled lighting variations (bright overhead, side lighting, dim lighting)
2. **Real-World Environment Collection**:
    - Original captures in kitchens, break rooms, and home environments
    - Natural lighting conditions across different times of day
    - Various backgrounds and surrounding objects to simulate real-world conditions
    - Different surface placements (countertops, tables of varying colors and materials)
3. **Operational State Documentation**:
    - Systematic capture of each operational state (power off, standby, warming up, brewing, error states)
    - Sequential documentation of complete coffee-making process
    - Detailed capture of UI indicators in various states (blinking lights, display messages)

    - Multiple examples of water reservoir, coffee container, and waste container at different fill levels
4. **Collection Equipment**:
    - Primary camera: Smartphone cameras representing mid-range Android and iOS devices
    - Secondary validation with different device models to ensure cross-device applicability
    - Consistent camera settings (auto-focus disabled when appropriate, fixed white balance)
    - Tripod usage for controlled shots with precise angle documentation

Data Augmentation Implementation:

To increase the diversity of training samples and reach the target dataset volume specified in Section 3.5.1 (Target Volume), data augmentation was methodically implemented after the initial collection of base images. The augmentation methods, transformations, and implementation specifics are covered in detail in Section 3.2 under "Data Augmentation." This procedure was essential for creating a solid dataset that could be used to train a model that would withstand a variety of real-world scenarios without requiring the laborious task of manually taking thousands of distinct photos.

3.6 Testing and Evaluation Methodology

Testing Framework and Strategy:

1. **Testing Levels**:
    - Unit testing of individual components and functions
    - Integration testing of module interfaces
    - System testing of complete application workflows
    - Acceptance testing with target users
2. **Testing Dimensions**:
    - Functional testing of core features
    - Performance testing under varying conditions
    - Accessibility compliance testing
    - Usability testing with visually impaired participants
3. **Testing Schedule**:
    - Continuous unit and integration testing throughout development
    - Weekly system testing incremental builds
    - Formal usability testing at 50% and 90% development milestones
    - Final acceptance testing before release
4. **Testing Environment**:
    - Device matrix covering representative Android and iOS devices
    - Physical environment variations (lighting, background noise)
    - Accessibility service configurations (TalkBack, VoiceOver with various settings)

Technical Performance Evaluation:

1. **Computer Vision Performance Metrics**:
    - Precision, recall, and accuracy for component detection
    - Mean Average Precision (mAP) at Intersection over Union (IoU) thresholds of 0.5 and 0.75
    - Confusion matrix analysis for component classification
    - Detection speed (frames per second) on target devices
    - False positive analysis across environmental conditions
2. **System Performance Metrics**:
    - Application startup time
    - Memory usage during operation
    - Battery consumption per hour of active use
    - CPU and GPU utilization
    - Heat generation during extended use
3. **Voice Guidance Evaluation**:
    - Instruction latency (time from state detection to speech output)
    - Speech clarity assessment through objective audio quality metrics
    - Interruption handling efficiency
    - Contextual accuracy of instructions
4. **Reliability Testing**:
    - Error rate during extended operation
    - Recovery success rate from intentionally induced errors
    - Performance degradation analysis over continuous usage
    - Edge case handling effectiveness

Accessibility Compliance Testing:

1. **Standard Compliance**:
    - Web Content Accessibility Guidelines (WCAG) 2.1 AA level testing
    - Mobile application accessibility best practices verification
    - Platform-specific accessibility guidelines (Android, iOS)
2. **Testing Tools**:
    - Automated accessibility scanners
    - Programmatic interface testing for screen reader compatibility
    - Color contrast analyzers
    - Touch target size verification tools
3. **Screen Reader Compatibility**:
    - Comprehensive TalkBack (Android) and VoiceOver (iOS) testing
    - Testing with different verbosity settings and speech rates
    - Verification of appropriate content descriptions and semantics

User-Centered Evaluation:

1. **Participant Recruitment**:
    - Target of 10-15 visually impaired participants
    - Diverse representation across:
        - Age groups (18-60+)
        - Vision impairment types and degrees
        - Prior technology experience levels
        - Previous coffee machine experience
    - Recruitment through partnerships with vision impairment organizations
2. **Testing Protocol**:
    - Task-based usability testing with common coffee-making scenarios
    - Think-aloud protocol during task completion
    - Post-task questionnaires for subjective feedback
    - Semi-structured interviews for qualitative insights
    - Video recording with consent for interaction analysis
3. **Evaluation Metrics**:
    - Task completion success rate
    - Time-on-task measurements
    - Error frequency and recovery
    - Custom satisfaction questionnaire for domain-specific feedback

Chapter 4: System Implementation

This chapter presents the implementation of the "Coffee Machine Assistant" mobile application, detailing how the system architecture, accessibility features, guidance logic, and camera interaction framework were developed. The implementation follows the methodology outlined in Chapter 3, with each subsystem constructed to support a modular, inference-ready application capable of integrating the EfficientDet-based computer vision model once training is complete.

4.1 Core Application Architecture

The application was developed using **React Native** with the **Expo** framework, enabling cross-platform deployment to both Android and iOS from a unified JavaScript codebase. A modular, decoupled architecture was a primary design goal, enabling maintainability and clear separation of concerns. This was achieved using React's Context API for global state management and React Navigation for screen flow control.

Two global providers structure the application’s data flow:

1. **SettingsProvider:** Manages user-configurable preferences such as voice speed, haptic feedback, and instruction detail level. Settings are persisted via local storage to ensure a consistent experience across sessions. Delayed rendering prevents default values from flashing before stored preferences load, supporting a seamless user experience.
2. **GuidanceProvider:** Implements the sequential state machine that governs the coffee-making guidance process. The provider maintains the current operational step, processes detection results, and outputs the appropriate voice instructions and transitions. It also accesses user settings from SettingsProvider to adapt guidance behavior.

Navigation is handled using React Navigation with a minimal three-screen structure:

- **HomeScreen**: The main entry point, providing access to the other two screens.
- **CameraScreen**: The core interactive component where the CV and guidance systems operate.
- **SettingsScreen**: A dedicated screen for user customization.

This linear and simple navigation flow was designed specifically to be intuitive for users relying on screen readers.

4.2 Accessibility-First User Interface (UI/UX)

The user interface was developed from the ground up using accessibility-first principles and WCAG/POUR guidelines. All interactive elements include semantically meaningful accessibility labels, hints, and roles to support both TalkBack (Android) and VoiceOver (iOS). Screen focus events trigger contextual announcements that orient the user immediately upon entering a screen, requiring no initial interaction.

The UI design applies the following key principles:

- **Clear spoken orientation:** Each screen provides an automatic introductory message explaining its function and available actions.
- **Accessible controls:** Buttons, switches, and sliders provide full accessibility metadata and use consistent terminology and phrasing.
- **User customization:** Speech rate, instruction verbosity, and haptic feedback can be adjusted in the SettingsScreen, with an accessible “Test Voice” feature enabling immediate auditory confirmation.
- **Minimal visual complexity:** Layouts use simple structures optimized for screen-reader navigation order and large touch targets, as recommended in accessibility best practices.

Together, these features provide a user experience tailored to visually impaired individuals while maintaining platform consistency.

4.3 Sequential Voice Guidance System

The voice guidance subsystem uses a finite state machine to guide users through the coffee-making process in a clear, safe, and structured manner. States represent discrete steps (e.g., locating the machine, finding the power button), and transitions occur when the correct component is detected.

Each state defines:

- a target object,
- a simple instruction,
- an optional detailed instruction,
- a next step, and
- correction prompts for incorrect detections.

This design ensures safe task progression, for example, preventing guidance from advancing until the machine is powered on. The system also adapts to user preferences by selecting detailed or concise instructions based on the stored settings.

4.4 Camera Screen Integration

The CameraScreen combines camera interaction, detection handling, and guidance communication. The final system is designed around an interval-based detection loop, as described in Chapter 3. In the current implementation, this loop serves as the interface point for simulated detections, but the structure is fully prepared for real TFLite inference.

The camera subsystem includes:

- Lifecycle-aware speech control, stopping TTS and clearing timers when leaving the screen
- An interval-driven detection loop, which periodically processes detection results
- Haptic feedback patterns distinguishing correct detections, incorrect detections, and neutral states

To optimize performance and maintain battery efficiency, the detection interval was set to three seconds—long enough to avoid overloading the device, yet frequent enough to maintain responsiveness.

During the current stage, this loop receives simulated detection identifiers, enabling full end-to-end validation of the guidance system. The detectionResult variable serves as the standardized interchange point for future TensorFlow Lite output, ensuring that integrating the actual inference results will require minimal structural changes.

References

[1] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and Efficient Object Detection," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020. arXiv:1911.09070.

[2] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications," arXiv:1704.04861, 2017.

[3] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al., "TensorFlow: A System for Large-Scale Machine Learning," in 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI), 2016, pp. 265-283.

[4] OpenCV team, "OpenCV (Open Source Computer Vision Library)". [Online]. Available: https://opencv.org/. [Accessed: March 28, 2025].

[5] Android Developers, "TextToSpeech | Android Developers". [Online]. Available: https://developer.android.com/reference/android/speech/tts/TextToSpeech. [Accessed: March 29, 2025].

[6] Apple Inc., "VoiceOver | Apple Developer Documentation," [Online]. Available: https://developer.apple.com/documentation/accessibility/voiceover. [Accessed: March 29, 2025].

[7] TensorFlow, "TensorFlow Lite Guide". [Online]. Available: https://www.tensorflow.org/lite/guide. [Accessed: March 30, 2025].

[8] Be My Eyes, "Be My Eyes - Bringing sight to blind and low-vision people," [Online]. Available: https://www.bemyeyes.com. [Accessed: March 30, 2025].

[9] Microsoft, "Seeing AI: An app for visually impaired people that narrates the world around you," [Online]. Available: https://www.microsoft.com/en-us/ai/seeing-ai. [Accessed: March 31, 2025].

[10] Envision Technologies B.V., "Envision AI: Vision technologies for the blind and low-vision community," [Online]. Available: https://www.letsenvision.com. [Accessed: March 31, 2025].

[11] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2001.

[12] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," International Journal of Computer Vision, vol. 60, no. 2, pp. 91-110, 2004.

[13] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in Advances in Neural Information Processing Systems, 2015. arXiv:1506.01497.

[14] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single shot multibox detector," in European Conference on Computer Vision, 2016. arXiv:1512.02325.

[15] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016. arXiv:1506.02640.

[16] Google Cloud, 'Text-to-Speech AI', [Online].
Available: https://cloud.google.com/text-to-speech. [Accessed: April 1, 2025].

[17] J. Gulliksen, B. Göransson, I. Boivie, S. Blomkvist, J. Persson, and Å. Cajander, "Key principles for user-centred systems design," Behaviour & Information Technology, vol. 22, no. 6, pp. 397-409, 2003.

[18] World Wide Web Consortium, "Web Content Accessibility Guidelines (WCAG) 2.1," W3C Recommendation, 2018. [Online]. Available: https://www.w3.org/TR/WCAG21/. [Accessed: April 1, 2025].

[19] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common Objects in Context," in European Conference on Computer Vision (ECCV), 2014, pp. 740-755. arXiv:1405.0312.

[20] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, and A. A. Kalinin, "Albumentations: Fast and Flexible Image Augmentations," Information, vol. 11, no. 2, p. 125, 2020. arXiv:1809.06839.

[21] Wix, "Detox: Gray Box End-to-End Testing and Automation Framework for Mobile Apps". [Online]. Available: https://github.com/wix/Detox. [Accessed: April 2, 2025].