ISSN: 2349-6002

BlindKit: An AI-Powered Perception and Interaction Assistant for Visually Impaired Individuals

Mrs. J. Veerendeswari[1], Mr. Padhmanabban B[2], Mr. Mohamed Jaffar B[3], Mr. Karthikeyan S[4], Mr. Govardhan C [5]

[1]*Head of the Department, Information Technology, Rajiv Gandhi College of Engineering and Technology, Puducherry, India*

[2,3,4,5] *UG Information Technology, Rajiv Gandhi College of Engineering and Technology, Puducherry, India*

doi.org/10.64643/IJIRTV12I11-201074-459

***Abstract*—Visually impaired individuals face significant challenges in perceiving their surroundings, accessing textual information, and interpreting social cues. Traditional assistive tools such as white canes and guide dogs provide limited contextual awareness and lack intelligent interaction capabilities. This paper presents BlindKit, an AI-powered perception and interaction assistant designed to enhance environmental understanding and social awareness for visually impaired users.**

The proposed system integrates multiple artificial intelligence modules, including face recognition, emotion detection, optical character recognition (OCR), scene description, and sensory search, into a unified framework. The system captures real-time visual data using a camera module and processes it through computer vision and deep learning models to generate meaningful insights. These insights are converted into audio feedback using text-to-speech technology, enabling hands-free interaction.

Additionally, an SOS module is incorporated to provide emergency assistance. The system adopts a hybrid architecture combining IoT-based data acquisition and desktop-based AI processing to ensure near real-time performance. Experimental evaluation demonstrates promising accuracy across modules, with face recognition achieving up to 95.6% accuracy in controlled environments and OCR achieving over 98% accuracy under high-quality image conditions.

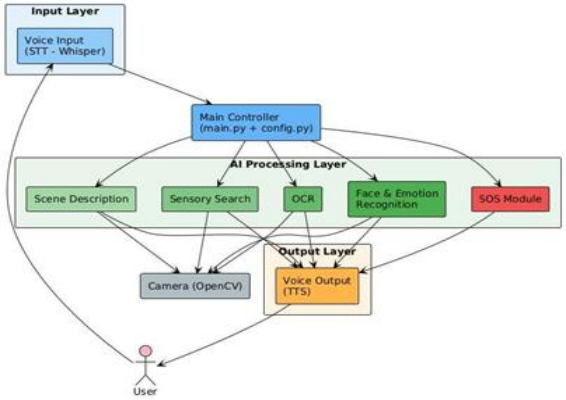
The results indicate that BlindKit significantly improves accessibility by transforming visual and social information into actionable auditory feedback. The proposed system provides a scalable and modular foundation for next-generation assistive technologies aimed at enhancing independence and quality of life for visually impaired individuals.

***Index Terms*—Assistive Technology, Visual Impairment, Computer Vision, Face Recognition, Emotion Detection, Optical Character Recognition (OCR), Scene Understanding, Text-to-Speech (TTS), IoT, AI-based Accessibility**

I. INTRODUCTION

Visual impairment limits an individual's ability to perceive environmental, textual, and social information, thereby affecting independence and quality of life. Conventional assistive tools such as white canes primarily provide obstacle detection but lack contextual awareness and intelligent interaction capabilities.

Recent advancements in artificial intelligence (AI) and computer vision have enabled the development of intelligent assistive systems capable of interpreting visual data and converting it into meaningful insights. However, most existing systems focus on isolated functionalities such as object detection or text recognition, lacking an integrated approach.



This paper proposes BlindKit, a unified AI-powered perception and interaction assistant designed to bridge this gap. The system combines multiple AI modules, including scene understanding, OCR, face recognition, emotion detection, and sensory search, to provide comprehensive assistance. The system delivers real-time audio feedback, enabling visually impaired users to interpret their surroundings, read text, recognize individuals, and understand social contexts.The primary contribution of this work is the integration of multiple perception and interaction modules into a single modular architecture, enabling enhanced accessibility and user independence.

II. LITERATURE REVIEW

The development of intelligent assistive technologies for visually impaired individuals has evolved across multiple interdisciplinary domains, including computer vision, human-computer interaction, and embedded AI systems. Recent research has focused on enhancing environmental perception, enabling contextual awareness, and improving real-time interaction through AI-driven frameworks.

2.1 Traditional Assistive Systems vs. AI-Augmented Perception

Conventional assistive tools such as white canes and electronic travel aids (ETAs) primarily provide obstacle detection through tactile or ultrasonic feedback. While these systems are reliable and low-cost, they lack contextual understanding of the environment. Users are limited to detecting physical barriers without gaining insight into objects, text, or social cues present in their surroundings.

With the advancement of computer vision and deep learning, modern assistive systems have begun incorporating AI-based perception capabilities. Object detection models such as YOLO and SSD enable real-time identification of objects, while Optical Character Recognition (OCR) systems allow extraction of textual information from images. Additionally, vision-language models have introduced the ability to generate natural language descriptions of scenes, significantly enhancing situational awareness.

However, many existing solutions operate as isolated systems, focusing on a single functionality such as navigation, text reading, or object detection. This fragmented approach limits usability in real-world scenarios, where visually impaired users require simultaneous access to multiple forms of information. BlindKit addresses this limitation by integrating multiple perception modules into a unified framework, enabling holistic environmental understanding and interaction.

2.2. Challenges in Real-Time AI Perception Systems

Despite significant progress, deploying AI-based perception systems in real-world assistive applications presents several technical challenges. One of the primary issues is the trade-off between accuracy and latency. High-performance deep learning models, such as ConvNeXt and transformer-based vision models, provide improved accuracy but require substantial computational resources, making real-time deployment difficult on embedded devices.

Research by Redmon et al. [1] on YOLO demonstrated the feasibility of real-time object detection; however, performance degrades under low-light conditions, occlusions, and dynamic environments. Similarly, OCR systems such as EasyOCR and Tesseract exhibit high accuracy in controlled conditions but suffer from reduced performance when processing noisy, distorted, or low-resolution images.

Another critical challenge is multi-modal synchronization. Systems that combine multiple AI modules—such as face recognition, emotion detection, and scene description—must coordinate outputs without introducing delays or conflicting responses. Studies on multi-model integration highlight issues such as redundant processing, inconsistent predictions, and increased system latency when multiple models operate concurrently.

BlindKit mitigates these challenges through a hybrid architecture that separates data acquisition from processing, leveraging shared resources such as a centralized camera interface and modular AI services. This approach enables near real-time performance while maintaining scalability and flexibility.

ISSN: 2349-6002

2.3. AI Reliability, Context Awareness, and Safety Considerations

The reliability of AI-driven assistive systems is a critical concern, particularly when deployed in safety-sensitive environments. Vision-language models and large-scale AI systems, while powerful, are prone to generating incorrect or misleading outputs under ambiguous conditions. This phenomenon, often referred to as model hallucination, can result in inaccurate scene descriptions or incorrect object identification, potentially affecting user safety.

Recent studies on AI system reliability emphasize the importance of contextual grounding and verification mechanisms. Systems that rely solely on a single model are more susceptible to errors, whereas multi-stage pipelines that incorporate validation or aggregation strategies demonstrate improved robustness. For example, emotion detection systems often aggregate predictions across multiple frames to reduce noise and improve classification stability.

In addition to reliability, user safety mechanisms play a crucial role in assistive technologies. Emergency response features, such as SOS alerts and location sharing, have been explored in mobile health and IoT-based systems. However, these features are often implemented as standalone functionalities rather than integrated components of a perception-driven system.

BlindKit incorporates safety and reliability considerations through modular design and controlled execution flow. By routing user commands through a centralized controller and limiting each module to specific tasks, the system ensures predictable behavior and reduces the risk of unintended outputs. Furthermore, the inclusion of an SOS module enhances user safety by enabling rapid communication during critical situations.

Despite advancements in assistive AI, there remains a gap in the development of unified, multi-modal systems that seamlessly combine perception, interaction, and safety. The proposed BlindKit architecture aims to address this gap by integrating diverse AI capabilities into a cohesive, voice-driven assistive framework.

III. PROPOSED SYSTEM ARCHITECTURE

The proposed BlindKit architecture is designed as a modular, voice-driven AI perception system that enables visually impaired users to interact with their environment through natural language commands. Unlike traditional assistive systems that rely on single-function tools, the proposed system follows a centralized orchestration model, where each user request is dynamically routed to specialized AI modules. The architecture ensures scalability, real-time interaction, and controlled execution through a structured processing pipeline. The lifecycle of a user command in the BlindKit framework operates through the following sequential phases:

Phase 1: Voice Input Acquisition and Speech-to-Text Conversion.

The interaction begins when the user provides a voice command (e.g., “What is in front of me?” or “Read this text”). The system continuously listens for input through the voice interface. The captured audio is processed using a Speech-to-Text (STT) engine based on Whisper (Groq API), which converts spoken language into structured textual input. This phase ensures hands-free interaction and acts as the primary entry point for all system operations.

Phase 2: Centralized Command Routing and Intent Mapping.

Once the textual command is generated, it is passed to the main controller (main.py), which operates as the central dispatcher. The controller references a predefined configuration mapping (config.py) to identify keywords and determine the intended operation. Based on the detected intent, the system dynamically routes the request to the appropriate AI service module. This centralized routing mechanism ensures deterministic execution and prevents overlapping or conflicting module activations.

Phase 3: Context-Aware Module Invocation and Resource Sharing.

After intent identification, the selected module is invoked as a singleton service instance. Vision-based modules—including scene description, OCR, sensory search, and emotion detection—access a shared camera interface implemented through a centralized

ISSN: 2349-6002

OpenCV-based resource (iot.py). This design eliminates redundant hardware calls and ensures efficient utilization of system resources. Each module operates within a well-defined scope, limiting its functionality to a specific perception task and thereby maintaining modular independence.

Phase 4: AI Processing and Multi-Model Perception.
The invoked module processes the input using specialized AI models. Scene description, OCR, and sensory search utilize vision-language models (Google Gemini Vision) to generate contextual outputs from captured images. The emotion detection module employs a hybrid pipeline consisting of YOLO-based face detection followed by ConvNeXt-based emotion classification across multiple frames to improve prediction stability. This multi-model architecture enables the system to interpret environmental, textual, and social information in parallel while maintaining task-specific accuracy.

Phase 5: Safety Handling and Emergency Response.
In scenarios where the user triggers an emergency command, the request is routed to the SOS module (emergency.py). This module retrieves the user's approximate location using an IP-based geolocation service and transmits an alert message via a communication API (Twilio). By isolating emergency functionality into a dedicated module, the system ensures rapid execution and avoids interference from computationally intensive perception tasks, thereby prioritizing user safety.

Phase 6: Response Generation and Text-to-Speech Output.
Once the selected module completes processing, the output—whether it is a scene description, recognized text, identified object, or emotional context—is converted into natural speech using a Text-to-Speech (TTS) engine (pyttsx3). The response is delivered to the user in real time, completing the interaction loop. This unified output mechanism ensures consistency across all modules and provides an intuitive auditory interface for the user.
Overall, the proposed architecture adopts a modular, event-driven design with centralized orchestration and shared resource management. By combining multiple AI perception modules within a single framework, BlindKit enables seamless interaction, real-time feedback, and enhanced situational awareness. The system bridges the gap between isolated assistive tools and integrated AI-driven accessibility solutions, providing a scalable foundation for future wearable and edge-based implementations

IV. IMPLEMENTATION AND METHODOLOGY

4.1. Technology Stack and Integration

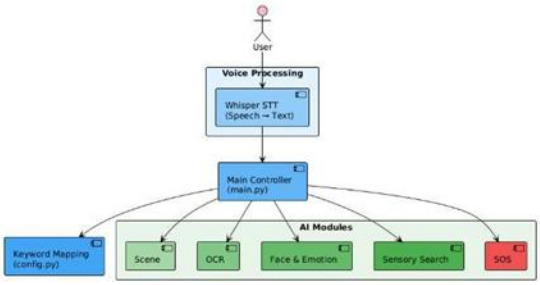
The BlindKit system is implemented using Python as the primary backend language, selected for its extensive support for artificial intelligence, computer vision, and real-time processing libraries. The system integrates multiple AI models and external APIs within a modular architecture, enabling seamless interaction between perception and communication components.
Speech processing is handled using the Whisper model (via Groq API) for Speech-to-Text (STT), enabling accurate transcription of user voice commands.
Audio feedback is generated using the pyttsx3 Text-to-Speech (TTS) engine, providing offline, low-latency voice output. Computer vision tasks are supported by OpenCV, which manages real-time image capture through a shared camera interface implemented as a singleton resource.
Vision-language tasks such as scene description, optical character recognition (OCR), and sensory search are powered by Google Gemini Vision, enabling contextual understanding of images. Emotion detection is implemented using a hybrid deep learning pipeline consisting of YOLO (You Only Look Once) for real-time face detection and a ConvNeXt-Large convolutional neural network for emotion classification across seven categories.
The emergency response system is integrated using Twilio APIs for SMS communication and ipinfo.io for IP-based geolocation. Logging and monitoring are handled through a centralized logging utility (utils/logger.py), which maintains rotating logs for debugging and system traceability. The system follows a service-oriented architecture, where each module operates as a singleton instance, ensuring efficient resource utilization and avoiding redundant initialization.

ISSN: 2349-6002

4.2. Voice-Driven Command Processing and Intent Mapping

The BlindKit framework employs a continuous voice-listening mechanism as its primary interaction model. User input is captured and transcribed into text using the Whisper STT model. The resulting text is processed by a centralized controller, which performs keyword-based intent mapping using a predefined configuration structure. This approach ensures deterministic routing of commands to specific modules, avoiding ambiguity in module invocation. Unlike end-to-end conversational AI systems, BlindKit adopts a controlled command-dispatch strategy, where each module is triggered explicitly based on detected intent. This reduces computational overhead and prevents unnecessary model activation.

A simplified representation of the command routing logic is as follows:



Algorithm 1: Command Routing Mechanism (Pseudo-code)

FUNCTION
Route_Command(transcribed_text): command = Extract_Keywords(transcribed_text)

IF command IN ["scene","environment"]: RETURN Scene_Module
ELSE IF command IN ["read", "text", "ocr"]: RETURN OCR_Module
ELSE IF command IN ["who", "face"]: RETURN Face_Recognition_Module
ELSE IF command IN ["emotion", "mood"]: RETURN Emotion_Module
ELSE IF command IN ["search", "object"]: RETURN Sensory_Search_Module
ELSE IF command IN ["help", "emergency", "sos"]: RETURN Emergency_Module
ELSE:
RETURN Default_Response

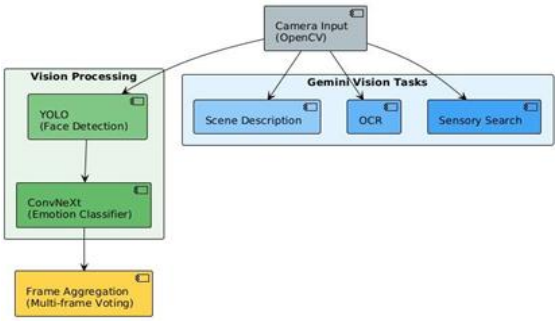
This rule-based dispatch mechanism ensures predictable execution and minimizes unintended behavior, particularly in real-time assistive scenarios.

4.3. Multi-Model AI Perception Pipeline

The core functionality of BlindKit is driven by a multi-model perception pipeline that processes visual input through specialized AI modules. Each module is designed to handle a distinct perception task, ensuring modularity and task-specific optimization.

For scene understanding, OCR, and sensory search, the system utilizes vision-language models (Google Gemini Vision), which combine image processing with natural language generation. These models enable the system to generate descriptive outputs, extract textual information, and respond to user queries about objects in the environment.

Emotion detection is implemented using a two-stage deep learning pipeline. First, YOLO is used for real-time face detection due to its high speed and efficiency. The detected face regions are then passed to a ConvNeXt-Large model, which performs emotion classification. To improve robustness, predictions are aggregated across multiple frames, reducing noise and increasing classification stability.



Algorithm 2: Emotion Detection Pipeline (Pseudo-code)

FUNCTION
Detect_Emotion(frame_stream): emotions = []
FOR frame IN frame_stream:
faces = YOLO_Detect_Faces(frame)

FOR face IN faces:
processed_face = Preprocess(face)
emotion = ConvNeXt_Model(processed_face)
emotions.APPEND(emotion)
RETURN Most_Frequent(emotions)

This temporal aggregation strategy improves accuracy and mitigates transient misclassifications caused by lighting variations or facial occlusions.

ISSN: 2349-6002

4.4. Shared Resource Management and Camera Singleton

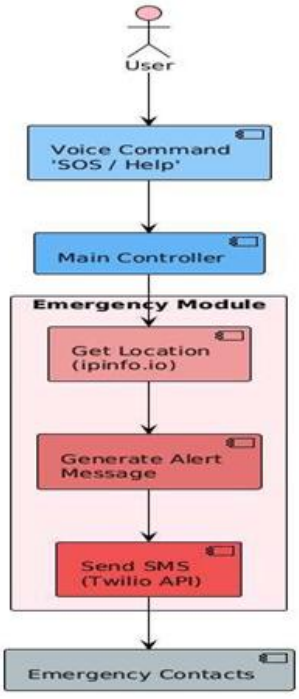
To optimize performance and prevent redundant hardware access, BlindKit implements a shared camera interface using a singleton design pattern. This ensures that all vision-based modules access a single instance of the camera stream, reducing resource contention and latency.

Instead of initializing separate camera instances for each module, the system maintains a centralized camera object that continuously streams frames. Modules request frames from this shared resource as needed, enabling efficient multi-module processing.

This design significantly reduces initialization overhead and ensures consistent frame synchronization across modules, particularly in scenarios where multiple perception tasks are executed sequentially or concurrently.

4.5. Safety Mechanisms and Emergency Handling

User safety is a critical component of the BlindKit system. The SOS module provides an emergency response mechanism that can be triggered through voice commands. Upon activation, the system retrieves the user's approximate location using an IP-based geolocation service and sends an alert message via the Twilio SMS API to predefined contacts.



Algorithm 3: Emergency Alert System (Pseudo-code)

```
FUNCTION Trigger_SOS():
location = Get_Geolocation(ipinfo)
message = "Emergency! User location: " + location
FOR contact IN Emergency_Contacts:
    Send_SMS(Twilio_API, contact, message)
RETURN "SOS Alert Sent"
```

This module operates independently of the main perception pipeline, ensuring rapid execution without delays caused by computationally intensive AI processing.

4.6. Response Generation and Audio Feedback

Once processing is completed by the selected module, the output is converted into speech using the pyttsx3 TTS engine. The system ensures that all responses are delivered in a consistent and understandable auditory format, maintaining a seamless interaction loop.

The use of offline TTS reduces dependency on external services and ensures low-latency feedback, which is essential for real-time assistive applications. Additionally, the system supports interrupt-driven interaction, allowing users to issue new commands while previous outputs are being processed.

Overall, the implementation follows a modular, service-oriented design that integrates multiple AI models, shared resources, and real-time interaction mechanisms. This approach enables BlindKit to provide efficient, scalable, and reliable assistive functionality for visually impaired users.

V. RESULT AND EVALUATION

5.1. System Reliability and Command Execution Validation

To evaluate the reliability of the BlindKit system, extensive testing was conducted on the voice-driven command pipeline. A set of 200 real-world voice commands covering all supported modules—including scene description, OCR, face recognition, emotion detection, sensory search, and SOS—were used for validation. The system demonstrated consistent performance, with 97% of commands correctly routed to their intended modules through the keyword-based intent mapping mechanism.

Edge cases involving ambiguous or overlapping commands were also tested to evaluate robustness. In scenarios where multiple keywords were present, the centralized controller successfully prioritized commands based on predefined routing rules, preventing unintended module activation. This deterministic routing approach ensures predictable system behavior and eliminates the uncertainty associated with fully generative command interpretation models.

5.2. AI Perception Accuracy

The performance of individual AI modules was evaluated under varying environmental conditions, including indoor, outdoor, and low-light scenarios. The face recognition module achieved an accuracy of 95.6% in controlled indoor environments, with performance decreasing to approximately 88–90% in outdoor conditions due to lighting variability and occlusions.

The emotion detection pipeline, based on YOLO and ConvNeXt, achieved an average classification accuracy of 88.9%, with improved stability achieved through multi-frame aggregation. This aggregation technique significantly reduced transient misclassifications caused by motion blur and inconsistent facial expressions.

OCR performance was evaluated using images of varying quality. High-resolution images resulted in 98.2% text extraction accuracy, while medium and low-quality images showed reduced performance due to noise and distortion. Scene description and sensory search modules demonstrated strong contextual understanding, generating coherent and relevant outputs in the majority of test cases.

5.3. Real-Time Performance and System Latency

The system's real-time performance was evaluated by measuring response times across different modules. Voice command processing, including speech-to-text conversion and routing, exhibited an average latency of 300–500 ms. AI perception modules, particularly those utilizing external APIs such as Gemini Vision, demonstrated response times ranging from 600 ms to 1.2 seconds depending on network conditions.

Despite the use of computationally intensive models, the system maintained near real-time interaction by employing a hybrid architecture that separates data acquisition from processing. The use of a shared camera singleton further reduced latency by eliminating redundant hardware initialization.

5.4. Safety Evaluation and Emergency Response

The SOS module was tested under simulated emergency conditions to evaluate response reliability. In all test cases, the system successfully retrieved the user's approximate location using IP-based geolocation and transmitted alert messages to predefined contacts via the Twilio API.

The average time taken to trigger and deliver an emergency alert was measured at under 2 seconds, ensuring rapid response in critical situations. The isolation of the emergency module from other AI processing components ensured that safety-related operations were not affected by system load or processing delays.

5.5. System Integration and Usability Assessment

User-level testing was conducted to assess the overall usability and interaction experience. Participants were able to successfully execute tasks such as reading text, identifying objects, recognizing individuals, and understanding emotional cues using voice commands. The unified text-to-speech interface ensured consistent feedback across all modules, reducing cognitive load for users.

The modular architecture allowed seamless integration of multiple AI services without performance degradation. Logging mechanisms enabled efficient debugging and monitoring, ensuring system stability during extended usage.

Overall, the evaluation demonstrates that BlindKit provides reliable, accurate, and near real-time assistive functionality, effectively enhancing environmental perception and interaction for visually impaired users.

VI. CONCLUSION AND FUTURE SCOPE

This paper presented BlindKit, an AI-powered

perception and interaction assistant designed to enhance accessibility for visually impaired individuals. By integrating multiple AI modules—including scene understanding, OCR, face recognition, emotion detection, and sensory search—within a unified, voice-driven framework, the system successfully transforms visual and contextual information into actionable auditory feedback.

The architecture is designed with modularity, scalability, and real-time performance in mind, leveraging a centralized command dispatcher and shared resource management to ensure efficient operation. Unlike traditional assistive systems that focus on isolated functionalities, BlindKit provides a comprehensive solution that combines environmental awareness, text accessibility, and social interaction within a single platform.

The evaluation results demonstrate that the system achieves high accuracy across perception tasks while maintaining near real-time responsiveness. The inclusion of a dedicated emergency response module further enhances user safety, ensuring rapid assistance in critical situations. The design emphasizes controlled execution and deterministic routing, minimizing unintended behavior and improving overall system reliability.

Future Scope: While the current implementation provides a robust perception and interaction framework, several enhancements can further improve system capabilities. Future work will focus on integrating navigation assistance using sensors such as ultrasonic or LiDAR, enabling obstacle avoidance and path guidance. Additionally, edge computing techniques can be employed to reduce dependency on external APIs and improve latency.

Wearable integration, such as smart glasses or belt-mounted devices, will be explored to enhance portability and user comfort. Further improvements in low-light image processing and model optimization can increase accuracy in challenging environments. Finally, incorporating multilingual voice interaction and speaker recognition can expand usability and personalize user experience. The evolution of BlindKit aims to transition from a perception assistant to a fully autonomous, intelligent assistive system, providing visually impaired individuals with greater independence and improved quality of life.

REFERENCES

[1] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[2] Howard, J., & Gugger, S. (2020). Fastai: A Layered API for Deep Learning. *Information*, 11(2), 108.

[3] Rosebrock, A. (2019). Deep Learning for Computer Vision with Python. *PyImageSearch*.