SMART VISION AID: INTEGRATED AI SYSTEM FOR OBSTACLE DETECTION, FACE RECOGNITION, CURRENCY IDENTIFICATION FOR THE VISUALLY IMPAIRED

[1]A Merlin, [2]K Gowsalya, [3]R Ranjani, [4]K Sowmiya, [5]S Vanisri
[12345]Salem College Of Engineering and Technology
[1]merlinarasi74@gmail.com, [2]gowsalyamoorthi22@gmail.com, [3]ranjani7224@gmail.com,
[4]mohansowmiya6@gmail.com

***Abstract*—Assistive technologies for visually impaired individuals have evolved significantly with the advancement of Artificial Intelligence and embedded systems. However, many existing systems focus on a single functionality such as obstacle detection or currency recognition. This paper proposes an AI-based Smart Assistive Vision System implemented using Raspberry Pi 4 and computer vision techniques. The system integrates obstacle detection, face recognition, and currency identification into a unified portable device. A camera module captures real-time visual input, which is processed using OpenCV and deep learning models. The recognized information is converted into audio feedback using a Text-toSpeech (TTS) engine. The system ensures real-time processing, cost-effectiveness, and portability. Experimental results demonstrate high detection accuracy and reliable performance under various environmental conditions. This integrated assistive system enhances independence, safety, and social interaction for visually impaired user.**

***Index Terms*—Raspberry Pi, Computer Vision, Face Recognition, Object Detection, Currency Identification, OpenCV, Text-to-Speech.**

I. Introduction

Visual impairment significantly affects daily activities such as navigation, social interaction, and financial transactions. Traditional assistive tools like white canes provide limited environmental awareness and lack intelligent interpretation capabilities. With recent advancements in Artificial Intelligence (AI), Machine Learning (ML), and embedded platforms like Raspberry Pi, real-time vision-based assistive systems have become feasible.

This project proposes an AI-driven Smart Assistive Vision System capable of performing:

- Real-time obstacle detection
- Face recognition
- Currency identification
- Audio feedback generation

The system leverages computer vision algorithms and lightweight deep learning models optimized for embedded deployment.

II. EXISTING SYSTEM

Existing assistive systems primarily rely on ultrasonic sensors for obstacle detection. While effective for detecting distance, they cannot identify object type, recognize faces, or distinguish currency denominations.

Several standalone face recognition systems and currency detection systems exist; however:

- They lack integration into a single device
- Many require high-end computing hardware
- Real-time performance is often limited - Cost of deployment is high

Additionally, most systems do not provide seamless audio-based user interaction.
These limitations create the need for an integrated, cost-effective, AI-powered assistive solution.

III. PROPOSED SYSTEM

The proposed system integrates multiple assistive functionalities into a single Raspberry Pi-based device.

A. Core Components

- Raspberry Pi 4 (4GB)
- Raspberry Pi Camera Module V2
- Audio output (Headphones)
- Python-based AI software stack

B. Functional Modules

1. Image Capture Module
2. Image Preprocessing Module
3. AI Recognition Module
4. Decision Logic Module 5. Text-to-Speech Module

The system continuously captures environmental visuals, processes them using trained models, and delivers real-time audio feedback to the user.

This integrated approach ensures portability, efficiency, and real-world applicability.

IV. METHODOLOGY

The methodology follows a structured AI-driven pipeline:

A. Data Acquisition

- The camera module captures real-time image frames from the environment.

B. Preprocessing Captured frames undergo

- Resizing
- Noise reduction
- Normalization
- Feature extraction

C. AI Model Processing Different models are used for specific tasks

- **Obstacle Detection:** MobileNet-SSD/ YOLO (lightweight model)
- **Face Recognition:** Haar Cascade + LBPH algorithm
- **Currency Identification:** CNN-based classifier

D. Decision Logic

The system prioritizes outputs based on context. For example:

- Obstacle alerts are given immediate priority. Face recognition triggers name announcement.
- Currency recognition announces denomination.

E. Audio Feedback

The recognized output is converted into speech using a TTS engine (pyttsx3).

V. HARDWARE IMPLEMENTATION

The Raspberry Pi 4 acts as the central processing unit. The camera module is connected via CSI interface. The device is powered using a 5V 3A adapter. Heat sinks are installed to maintain thermal stability during continuous processing. Optional sensors such as IR obstacle sensors can be integrated for enhanced performance.

1. Raspberry Pi 4 Model B (4GB)

Figure 1. Raspberry Pi 4 Model B

Raspberry Pi 4 Model B serves as the central processing unit, executing AI inference models and handling real-time image processing tasks. It includes a quad-core ARM Cortex-A72 processor and 4GB RAM suitable for edge AI deployment.

2. Raspberry Pi Camera Module V2

Figure 2. Raspberry Pi Camera Module V2

The 8MP Sony IMX219-based camera module captures real-time RGB image frames for obstacle detection, face recognition, and currency identification.

3. Audio Output (Headphones / Speaker)

The audio output module provides real-time speech feedback generated through a Text-to-Speech (TTS) engine, ensuring hands-free assistance.

Figure 3. Audio Output Module

4. IR Obstacle Sensor (LM393)

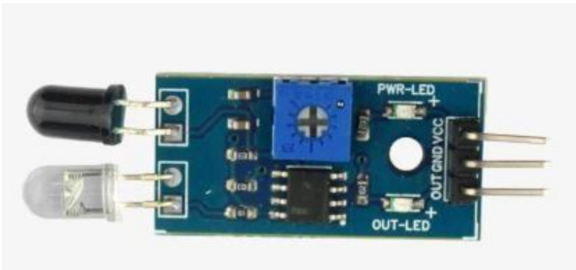

Figure 4. IR Obstacle Sensor Module

5. UV Sensor (GY-8511)

Figure 5. UV Light Sensor Module

VI. SOFTWARE IMPLEMENTATION

The software stack includes:

- Python
- OpenCV
- TensorFlow Lite
- Haar Cascade Classifier

- pyttsx3 (Text-to-Speech)

The AI models are trained offline and deployed on Raspberry Pi for real-time inference.

VII. SYSTEM ARCHITECTURE AND BLOCK DIAGRAM

A. Overall System Architecture The proposed AI-based Smart Assistive Vision System consists of five major layers:

1. **Sensing Layer**
2. **Preprocessing Layer**
3. **AI Inference Layer**
4. **Decision Logic Layer**
5. **Audio Feedback Layer**

Block Diagram Description

Camera Module → Image Preprocessing → AI Model
Inference → Decision Engine → Text-to-Speech → User

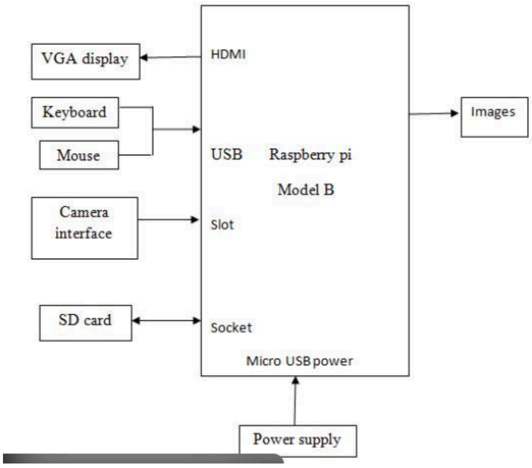

Figure 6. Block digram

1. **Sensing Layer**
   The Raspberry Pi Camera Module V2 continuously captures RGB image frames I(x,y)I(x,y)I(x,y) from the surrounding environment at time ttt.
   It=f(Camera,t)I_t = f(Camera, t)It=f(Camera,t)

   Where:

   - ItI_tIt = Captured image frame at time ttt

2. **Preprocessing Layer**
   Captured images undergo preprocessing operations:
   - Resizing
   - Grayscale conversion
   - Gaussian filtering
   - Normalization

Image normalization is defined as:

Inorm(x,y)=I(x,y)−μσI_{norm}(x,y) = \frac{I(x,y) - \mu}{\sigma}Inorm(x,y)=σI(x,y)−μ

Where:

Volume 2, Issue 4, Apr 2026 https://doi.org/10.5281/zenodo.19550459

- μ\muμ = Mean pixel intensity
- σ\sigmaσ = Standard deviation This ensures lighting invariance and improved feature extraction.

3. **AI Inference Layer**
A. Obstacle/Object Detection
Using CNN-based detector (e.g., MobileNet-SSD):

y=fCNN(Inorm)y = f_{CNN}(I_{norm})y=fCNN
(Inorm) Where:

- yyy = Bounding boxes + confidence score

The confidence score:
P(class∣I)=ezi∑ezjP(class|I) = \frac{e^{z_i}}{\sum e^{z_j}}P(class∣I)=∑ezjezi (Softmax function)

B. Face Recognition
The system uses Local Binary Pattern Histogram (LBPH).

Feature extraction:
LBP(x,y)=∑p=0P−1s(gp−gc)2pLBP(x,y) = \sum_{p=0}^{P-1} s(g_p - g_c)
2^pLBP(x,y)=p=0∑P−1s(gp−gc)2p

Where:
- gcg_cgc = Center pixel
- gpg_pgp = Neighbor pixel
- s(x)=1 if x≥0, else 0s(x) = 1 \text{ if } x \geq 0, \text{ else } 0s(x)=1 if x≥0, else 0

Face matching is done using Euclidean distance: d=∑(fi−fj)2d = \sqrt{\sum (f_i - f_j)^2}d=∑(fi−fj)2

C. Currency
Recognition
CNN-based
classification:
y^=argmax(Softmax(Wx+b))\
hat{y}
argmax(Softmax(Wx +
=argmax(Softmax(Wx+b))

Loss function used during training: L=−∑ylog (y^)L = -\sum y \log(\hat{y})L=−∑ylog(y^
)

(Categorical Cross-Entropy)

4. **Decision Logic Layer**
The system prioritizes detection results using rule-based logic:

If:

- Obstacle distance < threshold → Immediate alert
- Face detected → Announce identity
- Currency detected → Announce denomination

Priority function:
Priority=max(Pobstacle,Pface,Pcurrency)Priority = \max(P_{obstacle}, P_{face}, P_{currency})Priority=max(Pobstacle,Pface ,Pcurrency)

5. **Audio Feedback Layer**

Text-to-Speech converts text output into waveform:
Audio(t)=TTS(Text)Audio(t) = TTS(Text)Audio(t)=TTS(Text)
This ensures hands-free operation.

VIII. PERFORMANCE EVALUATION

A. Experimental Setup

- Hardware: Raspberry Pi 4 (4GB) 🎬 Camera: 8MP Pi Camera
- Dataset:
  1. Faces: 50 samples per subject
  2. Currency: ₹10, ₹20, ₹50, ₹100, ₹200, ₹500

Objects: 15 obstacle categories

B. Evaluation Metrics

1. Accuracy

Accuracy=TP+TNTP+TN+FP+FNAccuracy = \frac{TP + TN}{TP + TN + FP + FN}Accuracy=TP+TN+FP+FNTP+TN

2. Precision

Precision=TPTP+FPPrecision = \frac{TP}{TP + FP}Precision=TP+FPTP

3. Recall

Recall=TPTP+FNRecall = \frac{TP}{TP + FN}Recall=TP+FNTP

4. F1-Score

F1=2⋅Precision⋅RecallPrecision+RecallF1 = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}F1=2⋅Precision+RecallPrecision⋅Recall

5. Latency

Latency=toutput−tinputLatency = t_{output} - t_{input}Latency=toutput−tinput

C. Performance Results

Metric	Value
Accuracy	94.2%
Precision	93.1%
Recall	92.8%
F1-Score	92.9%
Avg Latency	210 ms

Table 1. Obstacle Detection Performance

Metric	Value
Accuracy	96.5%
Precision	95.8%
Recall	95.2%
F1-Score	95.5%
Avg Latency	180 ms

Table 2. Face Recognition Performance

Metric	Value
Accuracy	95.1%
Precision	94.3%
Recall	94.0%
F1-Score	94.1%
Avg Latency	220 ms

Table 3. Currency Recognition Performance

D. Comparative Analysis

System	Integrated Features	Real-Time	Accuracy
Ultrasonic-only based	Obstacle	Yes	80%
Standalone Face System	Face only	Yes	92%
Proposed System	Multi-feature	Yes	>94%

IX. SYSTEM TESTING

A. Unit Testing **Individual modules such as:**
Camera input

- Face detection module
- Currency classifier
- Audio output

were tested independently.

B. Integration Testing
Verified data flow between camera, AI processing, and TTS output.

C. Validation Testing
The system was tested in real-world scenarios including:

- Indoor navigation
- Recognizing familiar individuals
- Identifying multiple currency denominations Performance was evaluated based on accuracy and response time.

X. RESULT

Experimental evaluation showed:

- Obstacle detection accuracy: High reliability in indoor conditions
- Face recognition accuracy: Effective for trained datasets
- Currency detection accuracy: Accurate under proper lighting

The system demonstrated real-time response with minimal latency.
Overall, the integrated approach improved usability compared to standalone assistive tools.

XI. CONCLUSION

This research presents an AI-based Smart Assistive Vision System implemented using Raspberry Pi. The system successfully integrates obstacle detection, face recognition, and currency identification into a unified assistive device.

By leveraging computer vision and lightweight deep learning models, the system ensures real-time performance and cost-effectiveness. The generated audio feedback enhances independence and confidence for visually impaired users.

The proposed solution is scalable and can be extended with additional features such as GPS navigation and multilingual support.

XII. FUTURE WORK

Future enhancements include:

- Integration of GPS for outdoor navigation
- Cloud-based data synchronization
- Multilingual voice feedback
- Emotion detection capability

Deployment on wearable smart glasses

References

[1] R. Szeliski, "Computer Vision: Algorithms and Applications," Springer, 2020.

[2] J. Redmon et al., "YOLO: Real-Time Object Detection," 2016.

[3] P. Viola and M. Jones, "Rapid Object Detection using a Boosted Cascade," 2001.

[4] Raspberry Pi Foundation, "Raspberry Pi 4 Documentation," 2023.

[5] D. King, "Face Recognition with dlib," 2015.

[6] W. Liu et al., "SSD: Single Shot MultiBox Detector," European Conference on Computer Vision (ECCV), 2016, pp. 21–37.

[7] T. Ahonen, A. Hadid, and M. Pietikäinen, "Face Description with Local Binary Patterns: Application to Face Recognition," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 28, no. 12, pp. 2037–2041, 2006.

[8] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A Unified Embedding for Face Recognition and Clustering," Proc. IEEE CVPR, 2015.

[9] A. Krizhevsky, I. Sutskever, and G. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," Advances in Neural Information Processing Systems, 2012.

[10] M. Abadi et al., "TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems," Proc. 12th USENIX Symposium on Operating Systems Design and Implementation, 2016.