www.ijarp.com ISSN 2456-9992 Page: 01-16

Research Article

International Journal Advanced Research Publications

Volume: 02
Issue: 10

LOW-COST ESP32-CAM SMART GLASSES FOR BLIND ASSISTANCE

[1]K. B. Logeshwaran, [1]S. Mohammed Mushraf, [1]S. Balamuthu Kumar, *[2]R. Pavithra Devi

[1]Department of Information Technology, K.L.N. College of Engineering, Pottapalayam, Sivaganga, Tamil Nadu, India.

[2]Assistant Professor, Department of Information Technology, K.L.N. College of Engineering, Pottapalayam, Sivaganga, Tamil Nadu, India.

Article Received: 17 September 2026, Article Revised: 07 October 2026, Published on: 27 October 2026

***Corresponding Author: R. Pavithra Devi**

Assistant Professor, Department of Information Technology, K.L.N. College of Engineering, Pottapalayam, Sivaganga, Tamil Nadu, India.

Doi: https://doi-doi.org/101555/ijarp.6064

ABSTRACT

Visually impaired people face daily difficulty in handling money, avoiding obstacles and asking for help in an emergency. This project presents a low-cost smart glasses system that supports these needs through an ESP32-CAM module mounted on a spectacle frame and a companion Android application. The ESP32-CAM captures images and streams them over Wi-Fi to the smartphone, where the processing is done. A TensorFlow Lite image-classification model, trained with Google Teachable Machine, recognises Indian currency notes of ₹10 to ₹500 and announces the value through text-to-speech. An ultrasonic sensor detects nearby obstacles and warns the user with a buzzer or vibration that gets faster as the distance decreases. A single push button controls the system: a press starts the currency reader, and a long press triggers an SOS alert. When SOS is triggered, the app gives a short cancel window, obtains the GPS location, and sends an SMS with a Google Maps link to a registered caregiver. The prototype is built on a budget of about ₹2000 using commonly available components, and it is intended as an affordable assistive aid that improves safety and independence for visually impaired users.

INDEX TERMS: Smart Glasses, Assistive Technology, ESP32-CAM, Currency Recognition, TensorFlow Lite, Obstacle Detection, SOS Alert, Visual Impairment.

The proposed smart glasses support Sustainable Development Goal 3 (Good Health and Well-being) by providing an automatic SOS message with the user's location, which helps a

caregiver respond quickly after a fall or an unsafe situation. They support SDG 9 (Industry, Innovation and Infrastructure) by combining a low-cost camera microcontroller, on-device machine learning and a smartphone into a working assistive prototype. They also contribute to SDG 10 (Reduced Inequalities) by offering visually impaired people an affordable alternative to costly commercial assistive devices, so that independent money handling and safer mobility are within reach of more families.

I. INTRODUCTION

Vision is the sense on which most everyday activities depend. The World Health Organization reports that at least 2.2 billion people worldwide live with some form of near or distance vision impairment [1]. For a person who is blind or has severe low vision, three situations in particular reduce independence and safety: identifying money while shopping or travelling, detecting obstacles on the path that a white cane may not reach, and calling for help quickly when something goes wrong.

Handling cash is a particular difficulty in India, where paper notes of several denominations are in daily use. Notes of different values can be told apart by sighted people at a glance, but a visually impaired person must either trust the shopkeeper or depend on a companion, which exposes the person to errors and sometimes to deliberate cheating. Obstacles such as parked vehicles, hanging boards, open manholes, steps and other pedestrians add to the risk, and a white cane detects only what is within its sweep close to the ground. When a fall, a collision or an unsafe situation does occur, the person may not be able to find a phone, unlock it and describe the location to a relative.

Commercial wearable assistive devices can read text, recognise objects and identify banknotes, but their price is far beyond the reach of many users. At the same time, low-cost hardware has become very capable. The ESP32-CAM board combines a dual-core microcontroller with Wi-Fi and a camera on a module that costs only a few hundred rupees [2], [3]. Modern smartphones offer fast on-device inference through TensorFlow Lite [8], text-to-speech, GPS and SMS, which means that the heavy computation does not need to be placed on the glasses at all.

In this context, this paper presents a low-cost smart glasses system that brings together three functions on one platform: currency recognition with voice output, obstacle warning with a proximity-dependent buzzer or vibration, and a one-button SOS alert with a cancel window and an SMS carrying a Google Maps link. A single push button controls the whole system, so that the user needs to remember only two gestures. The glasses stream images over Wi-Fi to a

companion Android application, which runs a TensorFlow Lite classifier trained with Google Teachable Machine [7], [8] and speaks the result. The prototype is built for about ₹2000 from commonly available components and is intended as an affordable aid that complements, and does not replace, the white cane or guide dog.

The main contributions of this work are: (i) a smartphone-assisted architecture in which the ESP32-CAM only captures and streams, so that the glasses stay light and inexpensive; (ii) an Indian currency recogniser for ₹10 to ₹500 notes built with a no-code training tool; (iii) a distance-to-interval mapping that makes the warning faster as an obstacle comes closer; and (iv) a single-button interface with a cancel window that reduces false SOS messages.

II. LITERATURE REVIEW

Assistive technology for blind users has developed along several separate lines. The white cane remains the most widely used mobility aid because it is cheap and reliable, but it gives no information about obstacles above the waist or at a distance. Electronic travel aids based on ultrasonic ranging extend this reach by measuring the time of flight of a sound pulse; low-cost modules such as the HC-SR04 report distances from a few centimetres up to about four metres [5]. Such devices are simple and respond quickly, but they usually give only an audible or tactile warning and do not recognise what the obstacle is.

Currency recognition has been addressed mainly through smartphone applications and dedicated hand-held readers. Image-classification networks such as MobileNets are designed for mobile devices and reach useful accuracy with small models [6], and tools like Google Teachable Machine allow such a classifier to be trained from webcam or uploaded images without writing training code [7]. The exported model can be converted for TensorFlow Lite and run directly on an Android phone [8]. The recognition quality, however, depends strongly on the images used for training, on lighting, and on the way the note is held, which is why a background class and a multi-frame confidence check are useful.

Camera-based wearables put the sensing close to the user's eyes and hands, but many designs either run the model on an expensive embedded computer or depend on a cloud service that needs a continuous internet connection. Microcontroller camera modules such as the ESP32-CAM are inexpensive and include Wi-Fi [2], [3], but their processing power and memory are limited, so inference is better done on the phone that the user already carries. Emergency alerting is the third line: many systems send an SMS with GPS coordinates, and a cancel option is widely regarded as necessary because accidental triggers are common.

Existing solutions therefore tend to address one need at a time. A cane-mounted ultrasonic

device warns about obstacles, a phone application reads notes, and a separate personal-safety application sends alerts. Using three separate tools is difficult for a blind user, who must find the device, open the correct application and navigate its screen. The contribution of the present work is the integration of the three functions behind one hands-free, button-operated wearable built from low-cost parts. Table I compares the main categories of approach with the proposed system.

Table I. Comparison of Assistive Approaches for Visually Impaired Users.

Approach	Primary technique	Output	Remark
White cane	Mechanical tactile probing	Touch / sound	Cheap and reliable; short reach; no money or SOS support
Ultrasonic electronic travel aid	Ultrasonic time-of-flight [5]	Buzzer / vibration	Obstacle warning only; no recognition of objects or notes
Smartphone currency application	On-device image classifier [6], [8]	Voice announcement	Needs the user to operate the phone screen and camera
Commercial smart wearable	Embedded vision computer, cloud services	Voice output	Multi-function but costly and often needs internet
Proposed smart glasses	ESP32-CAM stream + phone TFLite classifier, ultrasonic sensor, one button, GPS + SMS	Voice, buzzer / vibration, SMS with map link	About ₹2000; needs Wi-Fi link to phone and GSM signal for SOS

III. METHODOLOGY

This section describes the method used to develop the smart glasses. The methodology combines embedded programming on the ESP32-CAM, machine-learning model training, Android application development and a small power-supply design. The system is divided into six key phases:

1. Image acquisition and Wi-Fi streaming
2. Currency recognition and voice announcement
3. Obstacle detection and proximity warning
4. Single-button control
5. SOS alert with cancel window, GPS and SMS
6. Power supply management

A. *System Architecture and Overall Methodology*

The system has two parts: a spectacle-mounted unit and an Android smartphone. The ESP32-

CAM is the controller of the spectacle unit. It captures images through its OV2640 camera and serves them as an MJPEG stream over Wi-Fi, reads the HC-SR04 ultrasonic sensor, drives the buzzer or vibration motor, and monitors the push button. The smartphone receives the stream, runs the TensorFlow Lite currency classifier, speaks the result through text-to-speech, and handles the SOS sequence with the phone's GPS and SMS functions. Because the processing is done on the phone, the glasses need no extra computing hardware. Fig. 1 shows the block diagram, Fig. 2 shows the process flow and Fig. 3 shows the currency-recognition pipeline.

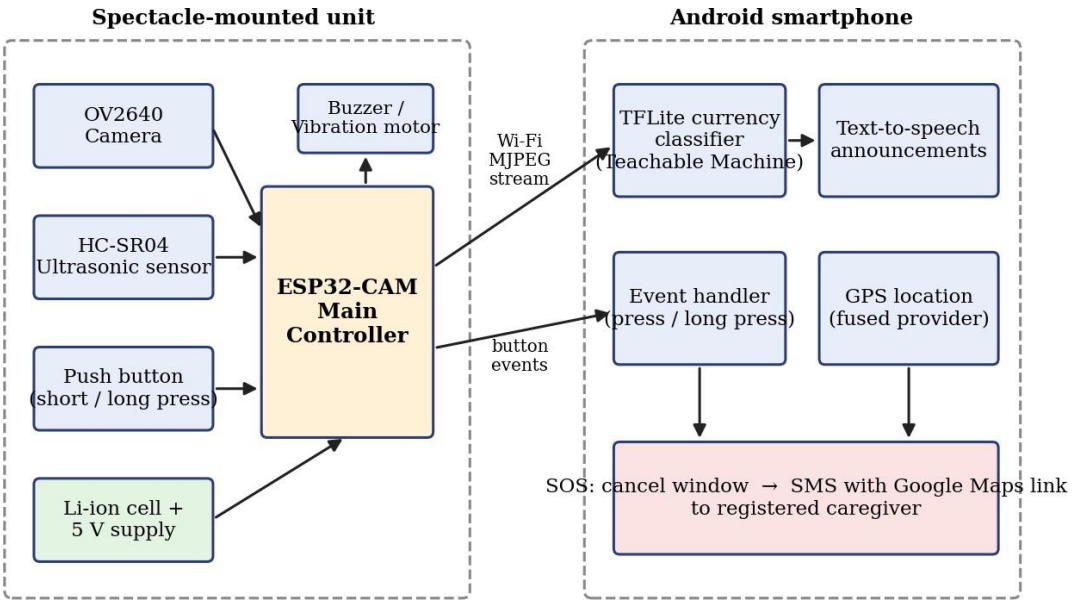


Fig. 1. System block diagram of the ESP32-CAM smart glasses.

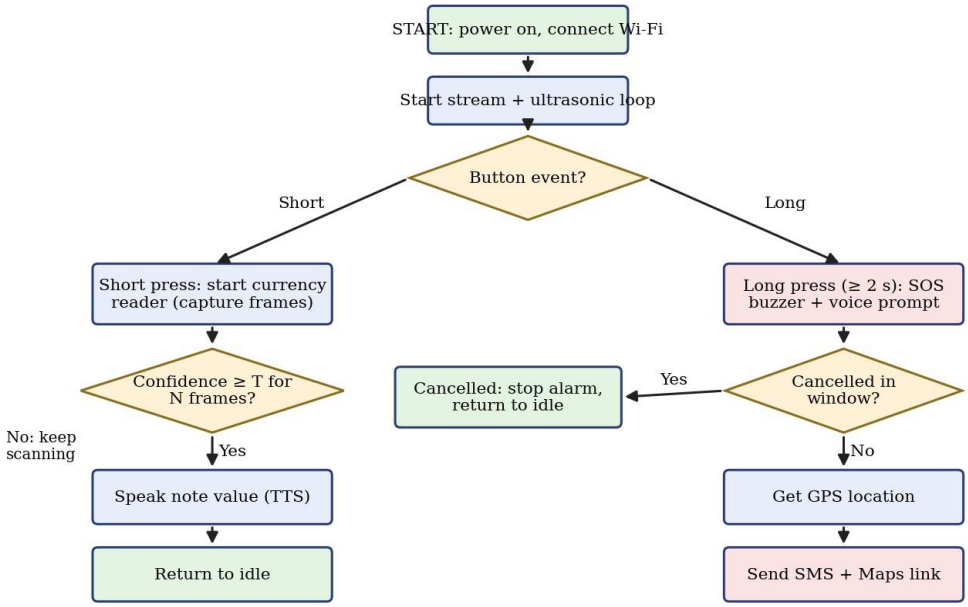


Fig. 2. Process flow of the smart glasses system.



Fig. 3. Currency recognition pipeline.

B. *Hardware Components and Power Supply*

The ESP32-CAM is an ESP32 module with a built-in OV2640 camera and Wi-Fi [2]–[4]. It has no USB interface, so an FTDI USB-to-serial adapter is used for programming. The HC-SR04 ultrasonic module measures distance from the echo time [5]. A push button gives the user input, and a buzzer and a small vibration motor give the warning. A single-cell Li-ion battery with a charging and boost module supplies 5 V to the board. Table II lists the hardware used in each module.

Table II. Module-Wise Hardware, Interface and Supply.

Module	Hardware	Interface / Supply
Control and camera	ESP32-CAM (OV2640)	Wi-Fi, GPIO; 5 V supply, 3.3 V logic
Obstacle sensing	HC-SR04 ultrasonic module	TRIG output, ECHO input (5 V echo level reduced by divider); 5 V
User input	Push button	Digital input to GND with internal pull-up
Alert indication	Buzzer, vibration motor	Digital outputs; motor through NPN transistor
Processing and voice	Android smartphone	Wi-Fi client, TFLite, text-to-speech, GPS, SMS
Power	Li-ion cell, charger and boost module	About 5 V, 0.5 A or more

The ESP32-CAM draws a much higher current than a plain microcontroller board because the camera and Wi-Fi radio operate together, and the current rises further when the on-board flash LED is lit. For this reason the board is powered from a supply that can deliver at least 500 mA, and a capacitor of a few hundred microfarads is placed near the 5 V input to absorb current peaks. Weak USB or battery supplies are a common cause of brown-out resets and unstable streaming. The flash LED is kept off during normal operation. Fig. 6 gives the approximate current of each module. These values are indicative figures taken from typical datasheet ratings [3], [5] and are not measured data.

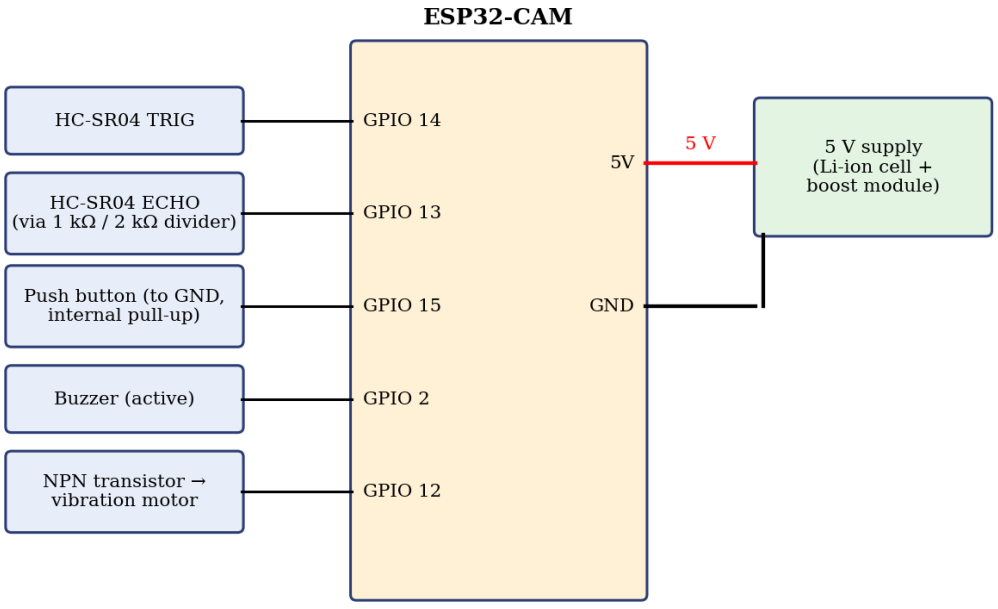
C. *Circuit Connections*

Fig. 4 shows the wiring between the ESP32-CAM and the other components, and Table III lists the pin-to-pin connections. Because the camera occupies many GPIO pins, only a few remain free; the pins listed below are those that are normally available when the SD card

is not used. GPIO 12 and GPIO 15 influence the boot configuration, so the transistor base and the button must not hold them in the wrong state at power-up. The HC-SR04 echo pin outputs 5 V, which is above the 3.3 V tolerance of the ESP32 input, so a resistor divider is placed on the echo line. All grounds are common. The pin assignment can be changed in the firmware if a different board variant is used.

Table III. Component Pin Connections.

Component / Pin	ESP32-CAM	Purpose
HC-SR04 TRIG	GPIO 14	Trigger pulse output
HC-SR04 ECHO	GPIO 13 (through 1 kfi / 2 kfi divider)	Echo pulse input
HC-SR04 VCC / GND	5 V / GND	Sensor power
Push button	GPIO 15 / GND	Short press and long press input
Buzzer (active)	GPIO 2	Proximity and SOS tones
Vibration motor (via NPN transistor)	GPIO 12	Silent proximity and SOS warning
Supply input	5 V / GND	From Li-ion cell and boost module
All GND	Common GND	Common reference



HC-SR04 VCC from 5 V; all grounds common; ESP32-CAM programmed through an FTDI adapter

Fig. 4. Wiring diagram of the prototype.

D. Currency Model Development

The currency classifier is an image-classification model trained with Google Teachable Machine [7]. One class is created for each denomination of ₹10, ₹20, ₹50, ₹100, ₹200 and ₹500, and one extra class named *Background* is added for frames that contain no note, so that

the model does not announce a value when nothing is held in front of the camera. The training images are captured with the ESP32-CAM itself, so that the colour, resolution and distortion of the training data match what the model sees in use. For each class, images are taken with both sides of the note, at different distances and angles, under indoor and daylight conditions, with the note flat and slightly folded, and with different hands and backgrounds. Teachable Machine uses transfer learning on a MobileNet backbone [6], which allows a useful model to be trained from a small number of images. After training, the model is exported in TensorFlow Lite format together with its label file and copied into the assets folder of the Android project [8].

E. Android Application

The companion application is written for Android. It connects to the Wi-Fi network created by the phone hotspot, to which the ESP32-CAM also connects, and reads the MJPEG stream from the glasses. Each frame is decoded into a bitmap, resized to the input size of the model (224 × 224 pixels) and normalised as required by the exported model. The TensorFlow Lite interpreter returns a probability for each class. The Android TextToSpeech engine speaks the recognised value, the countdown and the status messages. Android location services supply the GPS position, and the SmsManager class sends the SOS message. A caregiver's phone number is entered once, ideally with the help of a sighted person, and is stored on the phone. The application requests the SMS and location permissions at installation, and all spoken prompts are kept short so that the user is not overloaded.

F. *Currency Recognition and Announcement*

A short press of the button starts the currency reader. The phone then classifies incoming frames. Let P_k be K the highest class probability of frame k and C the corresponding class. A note value is accepted and k announced only when the same class, other than Background, is returned with a probability above a confidence threshold T for N consecutive frames:

$$R = C_k \text{ if } C_k = C_{k-1} = \ldots = C_{k-N+1} \neq Background \text{ and } P_k \geq T_c \quad (1)$$

In the prototype T c is set to 0.80 and N to 3, and both values are adjusted after testing. This check removes single-frame errors caused by motion blur or a partly visible note. When a result is accepted, the application speaks the value, for example "one hundred rupees", and

the reader stops. If no note is accepted within about ten seconds, the application tells the user to hold the note closer and flat and then stops, so that the reader does not keep running and drain the battery.

G. *Obstacle Detection and Proximity Warning*

The ESP32-CAM sends a 10 µs trigger pulse to the HC-SR04 and measures the width of the echo pulse. The distance is obtained from the speed of sound v ≈ 343 m/s, and the factor of two accounts for the path to the obstacle and back:

$$d = (v \times t_{echo}) / 2 \approx t_{echo} / 58 \text{ (cm, with } t_{echo} \text{ in } \mu s) \quad (2)$$

To reduce spurious readings, the median of the last three measurements is used as the filtered distance. Readings outside the range of the sensor are discarded. Obstacle checking is performed locally on the ESP32-CAM, not on the phone, so that the warning does not depend on the Wi-Fi link or on the inference
load of the phone. The warning starts when the filtered distance is below a far limit d and becomes faster as the obstacle comes closer. The interval Tb between two beeps or vibration pulses is calculated as:

$$T_b = T_{min} + (d - d_n) \times (T_{max} - T_{min}) / (d_f - d_n), \text{ for } d_n < d < d_f \quad (3)$$

where dn is a near limit below which the tone is continuous. In the prototype d f = 150 cm, d = 20 cm, T min = 100 ms and T= 1000 ms. Fig. 5 shows the resulting mapping. The warning is given by the buzzer, max the vibration motor or both, which allows a user to choose a silent mode in a noisy or quiet environment.

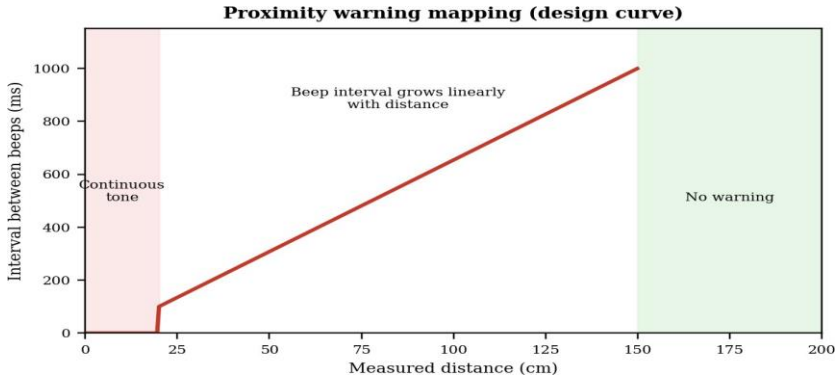


Fig. 5. Distance-to-interval mapping used for the proximity warning (design curve, not measured data).

H. *Single-Button Control*

One push button on the frame operates the whole system. The firmware measures how long the button stays pressed and reports the event to the phone: a press shorter than one second is a short press and starts the currency reader, and a press that lasts at least two seconds (T) is a long press and starts the SOSL sequence. Presses between these limits are ignored to avoid accidental triggers. The button is read with debouncing, and the phone checks the event state from the ESP32-CAM at a short interval of about 200 ms through a simple HTTP request. During the SOS cancel window, any further press cancels the alert.

I. *SOS Alert, GPS Location and SMS*

When a long press is detected, the phone announces that SOS has started, the buzzer sounds, and a cancel window of T= 5 s begins with a spoken countdown. If the button is pressed within this window, the alert w is cancelled and the system returns to idle. Otherwise the SOS is confirmed, as expressed by:

$$S = 1,\ if\ (press \geq T_{_}) \ and\ (no\ press\ within\ T_{_});\ S = 0,\ otherwise \qquad (4)$$

When S = 1, the application requests the current location from the Android fused location provider with a time limit of about 15 seconds. If a fresh location cannot be obtained in time, the last known location is used, and if none is available the message says that the location is unavailable. The location is formatted into a Google Maps link of the form https://www.google.com/maps/search/?api=1&query=<lat>,<lon> [11], which opens directly in the Maps application on the caregiver's phone. The SMS, which contains a short help text and this link, is sent to the registered caregiver. The application then speaks a confirmation, or an error message if the SMS could not be sent, so that the user knows the result. Fig. 7 shows the sequence of events.

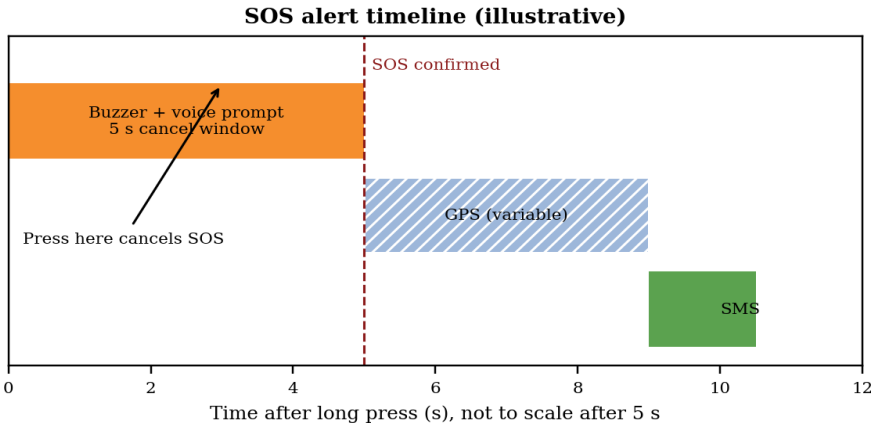


Fig. 7. SOS alert timeline (illustrative).

J. *Hardware Implementation and Calibration*

The prototype is assembled on a small perfboard and fixed to a spectacle frame, with the ESP32-CAM at the front, the ultrasonic sensor facing forward, and the battery and button on the temple arm or in a small pouch. The following steps are used for assembly and calibration:

- Connect all grounds together first, then wire the modules as in Table III, keeping the wires short and secured to the frame.
- Programme the ESP32-CAM through the FTDI adapter, then power it from the battery and confirm that it joins the phone hotspot and that the stream opens in a browser.
- Check the supply with a multimeter, and confirm that the board does not reset when Wi-Fi streaming starts.
- Verify the echo divider with a multimeter so that the ESP32 input never sees more than 3.3 V.
- Test the ultrasonic distance against a ruler at 20, 50, 100 and 150 cm, and adjust the near and far limits if needed.
- Capture the training images with the camera at its final mounting angle, then train, export and test the model on the phone.
- Tune T and N from tests with new notes that were not used for training.

K. *Firmware and Application Configuration*

The ESP32-CAM firmware is written in Embedded C/C++ in the Arduino IDE using the Arduino-ESP32 core and its camera driver. The main settings are as follows.

- Camera: OV2640 at QVGA or VGA resolution with a JPEG quality value that keeps each frame small enough for smooth streaming.
- Proximity: HC-SR04 read about every 60 ms; d=150 cm, d= 20 cm, T =100 ms, T= 1000 f n min max ms.
- Button: internal pull-up, debounce time of about 30 ms, short press below 1 s, long press at 2 s or more.
- Recognition: input 224 × 224, T = 0.80, N = 3, time-out of about 10 s.c
- SOS: cancel window of 5 s, location time-out of about 15 s, one registered caregiver number.

L. *Algorithm of the Proposed System*

Step 1: Power on the glasses, join the phone hotspot, start the camera stream and open the

application. Step 2: Continuously measure the distance with the ultrasonic sensor and drive the buzzer or vibration according to Eq. (3).

Step 3: If the button is pressed for less than 1 s, start the currency reader and classify frames.

Step 4: Accept a result according to Eq. (1), announce the note value by voice and stop the reader.

Step 5: If the button is pressed for 2 s or more, announce SOS, sound the buzzer and start the 5-second cancel window.

Step 6: If the button is pressed within the window, cancel the alert and return to Step 2. Step 7: If not cancelled, obtain the GPS location and generate the Google Maps link.

Step 8: Send the SMS to the registered caregiver, announce the result and return to Step 2.

IV. RESULTS AND DISCUSSION

1. *Test Procedure and Evaluation Metrics*

Each part of the system is tested individually before integrated testing, as summarised in Table IV, which has a column for the observed result of each test. The currency classifier is evaluated with notes that were not used for training, in indoor and outdoor lighting, and with both sides of every denomination. For each denomination the overall recognition accuracy is calculated as:

$$Accuracy\ (\%) = (number\ of\ correct\ announcements\ /\ number\ of\ attempts) \times 100 \qquad (5)$$

In addition, the time from the start of the reader to the spoken result is recorded as the sum of the capture, transfer, inference and speech times, and the proximity warning is checked against measured distances. SOS tests are carried out with a spare caregiver phone, and the time from confirmation to message arrival is noted. The observed results are to be entered from the bench tests of the prototype.

Table IV. Test Cases and Expected Outcomes.

Test	Expected outcome	Observed result
Camera stream	ESP32-CAM joins the hotspot and the phone displays a steady stream	To be recorded
Ultrasonic distance	Readings match a ruler within the sensor tolerance at 20–150 cm	To be recorded
Proximity warning	Beep interval shortens as the obstacle approaches; continuous tone below d n	To be recorded
Currency recognition	Correct value announced for ₹10 to ₹500 notes; no announcement for background	To be recorded
Button gestures	Short press starts reader; long press starts SOS; mid-length press ignored	To be recorded
SOS cancel window	Press within 5 s cancels the alert and no SMS is sent	To be recorded
GPS and Maps link	Valid coordinates inserted into the Google Maps link	To be recorded
SOS SMS	Caregiver receives SMS with a working map link	To be recorded
Battery run time	Glasses operate for the target period on one charge	To be recorded

2. ***Currency Recognition Evaluation***

Table V is the record sheet used for the recognition tests. Each denomination is tested with both sides of the note in several lighting conditions, and the number of correct announcements is divided by the number of attempts. A confusion between ₹10 and ₹20 or between ₹100 and ₹500 notes is likely to be the most visible error, because these notes share dominant colours in some lighting conditions, and the tests should therefore pay particular attention to these pairs.

3. ***Illustrative System Behaviour***

To complement the test plan, Figs. 5–7 present three indicative visualisations: the proximity warning mapping of Eq. (3), the approximate typical current drawn by each module, and the SOS timeline. They are illustrative and are not measured data. The current chart shows that the ESP32-CAM, when streaming, is the dominant load, and the cell capacity and the boost converter must be chosen for this load.

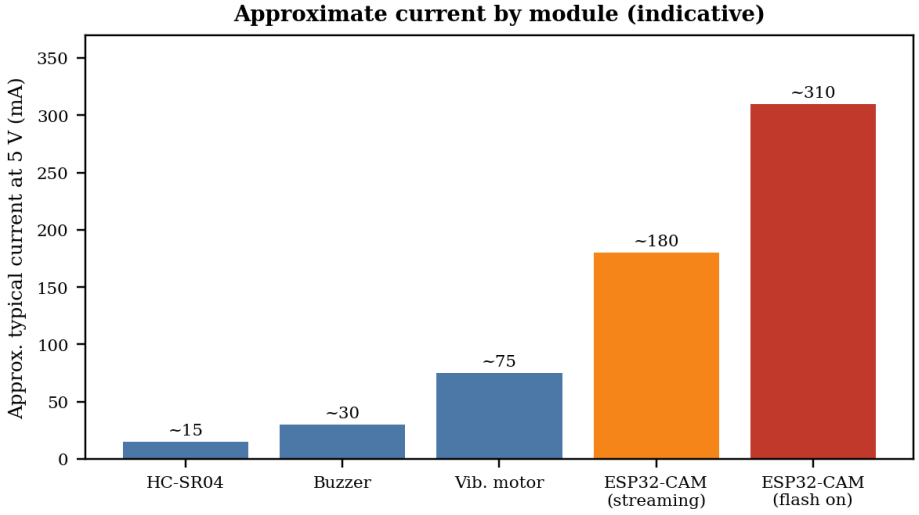


Fig. 6. Approximate typical current drawn by each module (indicative values).

A rough estimate shows why the supply matters. If the glasses draw about 250–300 mA at 5 V in use, a 2000 mAh Li-ion cell at 3.7 V (about 7.4 Wh) with a boost converter of around 85% efficiency gives a run time of the order of four hours. The actual figure must be confirmed by a discharge test, and a larger cell or a lower frame rate extends the time between charges.

4. ***Cost Analysis***

The prototype is built on a budget of about ₹2000 using commonly available components. Table VI gives an approximate bill of materials. The prices are typical retail figures that vary

between suppliers and over time, and they should be replaced by the actual bill of the prototype. The Android phone is assumed to be already owned by the user and is not included.

Table VI. Approximate Bill of Materials.

Item	Approx. cost (₹)
ESP32-CAM module	700
FTDI USB-to-serial adapter	180
HC-SR04 ultrasonic module	80
Buzzer, vibration motor and push button	70
Li-ion cell with charging and boost module	350
Spectacle frame	150
Resistors, transistor, capacitor, wires, perfboard	120
Mounting, enclosure and contingency	250
Total (approximate)	**1900**

5. DISCUSSION

The threshold values used for recognition and warning are a trade-off. A low confidence threshold announces notes faster but raises the risk of a wrong value, which is worse than no answer for a user who cannot verify it. The three-frame rule and the Background class are therefore important. In the same way, a far limit of 150 cm gives time to react at walking speed but may warn about objects that are not in the path, so the limits should be tuned with real users. The one-button design keeps the interface simple, but it needs clear audio feedback after each gesture, otherwise the user cannot tell whether the press was recognised.

The main advantages of the proposed system are a very low cost, hands-free operation, three functions in one wearable, local obstacle warning that does not depend on the Wi-Fi link, an SOS message that carries a map link which is easy to open, and a modular design in which each part can be tested or replaced independently. Its main limitations are:

- The glasses depend on a Wi-Fi link to the phone; if the link is lost, currency reading stops, although the local obstacle warning continues.
- Recognition accuracy depends on lighting, note condition (worn, folded or dirty) and the training images, and the model recognises only ₹10 to ₹500 notes; coins and newer notes are not covered, and counterfeit notes are not detected.
- The ultrasonic sensor has a narrow field of view, can miss thin or angled objects, and gives poor readings on soft surfaces that absorb sound.
- The SOS message needs a working SIM with mobile coverage, and the GPS position may be slow or inaccurate indoors; Android also restricts SMS permissions for applications

distributed through public stores, so the prototype is installed directly.

- The ESP32-CAM runs hot and uses a high current when streaming, which limits battery time and comfort on a spectacle frame.
- The system is an assistive aid and must be used together with, and not instead of, the white cane or guide dog.

V. CONCLUSION AND FUTURE SCOPE

In this study, a low-cost smart glasses prototype has been designed to help visually impaired users with three everyday needs: identifying Indian currency notes, avoiding nearby obstacles, and asking for help in an emergency. An ESP32-CAM mounted on a spectacle frame streams images to a companion Android application, where a TensorFlow Lite classifier trained with Google Teachable Machine recognises ₹10 to
₹500 notes and announces the value by voice. An ultrasonic sensor gives a buzzer or vibration warning that becomes faster as the distance decreases, and a single push button starts the currency reader or, with a long press, triggers an SOS alert with a cancel window and an SMS carrying a Google Maps link to a registered caregiver. The prototype is built for about ₹2000 from commonly available components, which makes it a practical and affordable assistive aid.

Future work includes training the model with a larger and more varied set of notes and adding coins and the newer note designs, extending the recogniser to general object, text and face recognition, and supporting voice output in regional languages such as Tamil. Obstacle sensing can be improved with a wider-angle or multiple ultrasonic sensors, or with time-of-flight sensors, and fall detection through an accelerometer could trigger SOS automatically. Multiple caregiver contacts, automatic SMS retry, live location sharing, a caregiver application, voice commands and a rechargeable battery with level announcements would make the system more dependable. On the hardware side, a custom compact PCB, a lighter enclosure and a lower-power camera module would prepare the design for long daily use.

REFERENCES

1. World Health Organization, *World Report on Vision*. Geneva, Switzerland: WHO, 2019.
2. Espressif Systems, *ESP32 Series Datasheet*. Shanghai, China: Espressif Systems.
3. Ai-Thinker Technology, *ESP32-CAM Product Specification*. Shenzhen, China: Ai-Thinker.
4. OmniVision Technologies, *OV2640 Color CMOS UXGA (2.0 Megapixel) CameraChip*

Datasheet. Santa Clara, CA, USA: OmniVision.

5. Elec Freaks, *HC-SR04 Ultrasonic Ranging Module User Guide*.
6. Howard *et al.*, "MobileNets: Efficient convolutional neural networks for mobile vision applications," arXiv preprint arXiv:1704.04861, 2017.
7. Carney *et al.*, "Teachable Machine: Approachable web-based tool for exploring machine learning classification," in *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*, ACM, 2020.
6. Google, "TensorFlow Lite documentation." [Online]. Available: https://www.tensorflow.org/lite
7. Android Developers, "Android API reference: TextToSpeech, SmsManager and FusedLocationProviderClient." [Online]. Available: https://developer.android.com/reference
8. Reserve Bank of India, "Banknotes." [Online]. Available: https://www.rbi.org.in
9. Google, "Google Maps URLs." [Online]. Available: https://developers.google.com/maps/documentation/urls