Chapter 8
Let the Blind See: An AIIoT-Based Device for Real-Time Object Recognition with the Voice Conversion

Puja Gupta, Mukul Shukla, Neeraj Arya, Upendra Singh, and Krishnanand Mishra

8.1 Introduction

India is unfortunate to have the largest number of blind people in the world. Poverty is a disability that a person can overcome with proper training, guidance, and support. It is estimated that out of 285 million visually impaired people worldwide [1] 39 million are blind, and 246 million have poor blindness (moderate disability). Around 90% of the world's blind people live in growing countries (like India). Of the 39 million blind people worldwide, more than 15 million are Indians. Globally, faulty errors are a major cause of visual impairment. Most of the people in developing countries cannot afford expensive solutions. Sixty-five percent of the visually challenged and 81.9% of the blind are above 50, even though this age group is around 20% of the people worldwide.

8.1.1 Vision Impairment

Visual acuity (VA) usually refers to the sharp sight of vision. The strength of visual acuity depends on the physical and internal factors, namely, (i) the intensity of the retinal focus within the eye, (ii) the fitness and function of the retina, and (iii) the tact of the brain's interpretive intelligence. Visual impairment is divided into two groups, the distance and the closest to introducing visual impairment.

P. Gupta · M. Shukla · N. Arya (✉) · U. Singh · K. Mishra
Department of Information and Technology Shri Govindram, Seksaria Institute of Technology & Science, Indore, India

F. Al-Turjman, A. Nayyar (eds.), *Machine Learning for Critical Internet of Medical Things*, https://doi.org/10.1007/978-3-030-80928-7_8

Visual Impairment

- Mildness –introduces visual clarity poor than 6/12.
- Medium – introduces visual acuity poor than 6/18.
- Powerful – introduces visual clarity poor than 6/60.
- Blindness – introduces visual acuity poor than 3/60.

Nearby Sight

Presenting near-visible weight is much poorer than the N6 or N8 at 40 cm by the ongoing adjustment.

Blind people face many challenges in their daily lives, most important of them is identifying the surrounding objects in nature while moving. Majority of the time, they accept travel restrictions because often they get lost when they are moving alone. Visual impairment or snow blindness cannot be completely cured. But with the support of modern science, their suffering can be decreased.

Computer and IoT surveillance [15] technology, especially distinct artificial neural networks, have grown swiftly in recent years. It is promising to use cutting-edge computer viewing techniques to help people who have lost sight. In this study, we proposed a device that can see objects around us.

There are many tools available to use computer vision systems to help blind people. The proposed function includes a Pi camera connected to a Raspberry Pi device. Raspberry Pi device and sensor are then attached to a lightweight non-heavy rod that one can carry while walking. Ultrasonic sensors are used for pre-scanning. The area around the blind is determined by blind signals.

The camera takes pictures of the surroundings, and the object recognition algorithm detects nearby objects in the image, and the sensor works by calculating its distance to feel an object closer, and when certain risk objects move near the person, then the sensor gives the signal and through the audio jack gives audio/vocal information about the object to the person through which we can make them aware of that object. Audio guidance will help the person navigate alone with safety and avoid any obstacle encountered, whether fixed or mobile obstruction. This sensor-based obstacle detection device will improve the mobility of both blind and blind people in a specific area by detecting obstacles, and it even recognizes pits and manholes on the ground to make them free to walk. We are detecting objects using this entire setup and giving voice instructions about those objects. It tells people about obstacles and also provides information about the appropriate barrier-free route.

8.2 Related Works

The literature survey will critique the discovery of object detection in many prospects, which includes historical instruments. The discovery has gone through two historical periods: “traditional discovery time” and “discovery time based on in-depth learning in the deep learning period.”

The history of real-time detection of anthropomorphic faces without obstacles (e.g., skin color separation) originates from P. Viola and Mr. Jones who followed a more forward-looking approach, i.e., moving windows: bypassing all the areas and criterion available in the picture to realize if there is a window containing a human face.

N. Dalal et al. [3] stated HOG (Histogram of Oriented Gradients) algorithm [2] can be regarded as an essential developmental variable factor and the standing conditions of its duration, evaluating the variability of a feature (including translation, scale, brightness) and nonlinearity (in classifying various categories of objects).

D. T. Nguyen et al. [4] analyze the parameters used in the SSD detection and state that there is a need for adjustment to that. Introducing the fusion SSD model for the limited-access factor, this algorithm improved the acquisition of small objects in a traditional SSD.

Love Lin et al. [5] found the reasons behind it and proposed RetinaNet in 2017. They said the excessive inequality of the front-back section experienced during the preparation of thick locators is a significant reason. Until this point in time, another deficit work called "fixed shortfall" has been presented on RetinaNet by adjusting the ordinary cross-entropy deficit with the goal that the identifier can zero in additional on erroneous models during training.

R. Girshick [6] introduced the Fast RCNN detector, an approach based on the continuous development of RCNN and SPPNet [16, 17]. The fast RCNN empowers us to concurrently train the detector and the controller of the connecting box with similar network parameters applied to both. In the VOC 07 database, Fast RCNN improved the mAP (mean average precision) from 58.5% (RCNN) to 70.0% with 200 times recovery speed than that achieved through RCNN.

H. Zhao et al. [7] stated that face detection is a crucial task, and its proper detection is a must. The Fast RCNN algorithms work well as they use two stages of detection. The disadvantages of YOLO model were eliminated by the use of the provided technique, which is YOLOv3 face based on YOLO v3 model. The result was more accurate for face detection in the presented method.

W. Liu et al. [8] proposed SSD with the main contribution toward the proposition of the multi-reference and multi-goal identification methods which significantly improves the exactness of location of a uni-stage indicator, solely for some minor items.

Limit W. [9] has improved the acquisition of an object-oriented approach that combines visualization above and the separation of the image below and above. The two foremost steps in this process are the production of the hypothesis step and the verification step. If we talk of the top-down hypothesis generation step, it is designed with an advanced Shape Context feature, which shows robust behavior toward counter flexibility and contextual tension.

Redmon J. [10] introduced YOLO, a new acquisition method. Pre-discovery function restores classifiers to perform detection. Instead, the detection of an enclosed object is a big problem of retreating into geographically separated boxes and corresponding phase opportunities. One neural network predicts binding boxes and class opportunities unswervingly from full-scale images in a single test. As the

entire acquisition pipe is a solitary network, it can be possibly configured to end after the acquisition operation.

Ren S. [11] introduced the region proposal network (RPN) which transfers complete picture signals to the acquisition network and consequently enables less expensive regional proposals. RPN is a complete network of solutions that concurrently predicts parameters of object and opposition total in each area. RPN is instructed endwise to produce excellent regional proposals, which Fast RCNN uses to find.

Srinivasan L. [12] proposed a hybrid system that uses the multilayer convolutional neural network (CNN) to produce graphical lexicons and long short-term memory (LSTM) to organize sound sentences using generated keywords. The convolutional neural network compares the image to a large database of training images and creates an accurate description using trained captions.

Mettah et al.'s [13] proposed model is the last to the end and uses both letter and voice presentations. Character presentations are read during model training through the convolutional neural network (CNN). For word-level representation, it includes several trained embeddings (Word2Vec, FastText, and GloVe). To address the issue of poor access to unspecified communication data, the transfer learning (TL) approach has been implemented.

Shahira et al. [14] developed a help program that detects a blockage, classifies it, and alerts the handler with a voice-over. Sensitivity and measurement of the barrier's distance from the surface using an ultrasonic sensor, were detected. Obstruction of the barrier was performed using the YOLOv2 algorithm in a picture obtained through a laptop webcam.

8.3 Proposed Model Component

Usually, a blind person uses a white cane to get a guide through difficult situations. Most of the surrounding area can be covered by this cane. But things far from reed are not easily accessible by the person. Proposed items provide information about user constraints as well as proper or constraint-free paths. This proposed system consists of hardware as well as software combination which can sense short as well as far distance and update person about it. The use of object detection techniques can open up new possibilities in assisting indoor navigation for blind and blind people.

8.3.1 Hardware Components

The system includes the following hardware components: Raspberry Pi Model 3B+, Pi camera, microSD card, sound sensor, and lightweight rod.

8.3.1.1 Raspberry Pi (Model 3B+)

Researchers used the Raspberry Pi as a small computer unit, which means that the microprocessor, memory, and input-output unit are all on one circuit board. Pi is a Linux computer, such that it can perform everything as a Linux computer can do like programs, libraries, and providers.

TensorFlow 1.4 officially supports Raspberry Pi when we are running Raspbian, since then we need to install Raspbian (Pi's OS). The quad-core Raspberry Pi Model 3B+ is both faster and more capable than its predecessor.

Its features are system on chip (SoC): Broadcom (BCM2837); CPU: quad-core ARM Cortex-A53, 1.2 GHz; GPU: Core IV Broadcom Video; RAM: 1 GB (frequency: 900 MHz); networking: 10/100 Ethernet, 2.4 GHz 802.11n wireless; Bluetooth: Bluetooth 4.1 Classic, Bluetooth Low Energy (BLE); storage: MicroSD card; GPIO: 40-pin header, populated; and port: HDMI, 3.5 mm analog audio-video jack, four USB 2.0, Ethernet, camera serial interface (CSI), and display serial interface (DSI).

Figure 8.1 shows the Raspberry Pi Model 3B+ used in the proposed system and its basic features and input-output unit supported by the development board.

8.3.1.2 Camera

We have used a 5 MP Raspberry Pi camera for capturing frames.

Figure 8.2 shows the Raspberry Pi camera used in the proposed system and its connection with the mainboard.

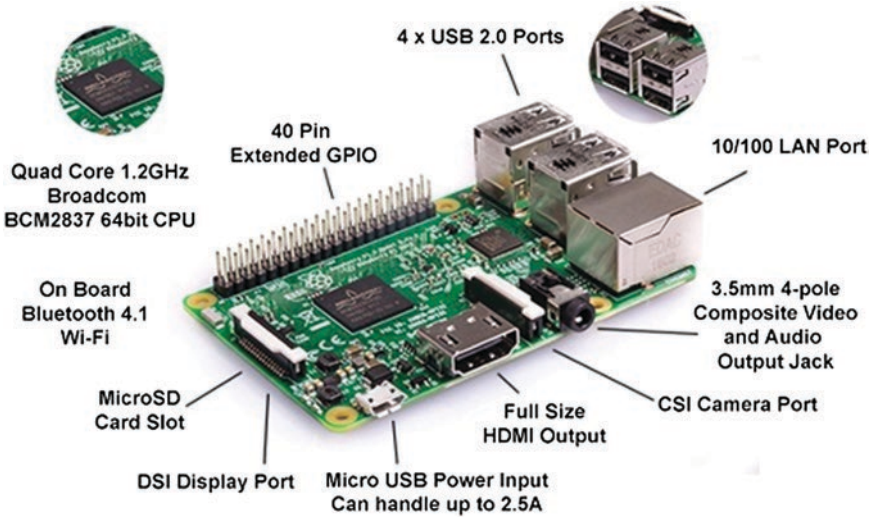

Fig. 8.1 Raspberry Pi Model 3B+ [17]

Fig. 8.2 Raspberry Pi with camera module [17]

Fig. 8.3 Ultrasonic range sensor (Hc-Sr04) [18]

8.3.1.3 MicroSD

We also need a microSD card, with 16 GB of storage memory which is required for building a workable OpenCV procedure. We have put all our required libraries, object recognition files, and other necessary dependencies since the board does not have any built-in storage.

8.3.1.4 Ultrasonic Range Sensor (HC-SR04)

Figure 8.3 shows the ultrasonic sensor model Hc-Sro4 used in the proposed system for detecting an object. This sensor can detect objects by emitting ultrasonic sound. The range up to which they can detect the object is 5–6 m. The ultrasonic sensor can be connected to Raspberry Pi with the help of the breadboard.

The sensor can detect distance with an object and send it to Raspberry Pi in the GPIO pin (5V).The ultrasonic sensor has four pins:

8.3.1.5 Ground Pin (GND – Pin 6)

- Echo pulse output pin (ECHO – Pin 23).
- Trigger pulse input pin (TRIG – Pin 12).
- 5 V supply pin (VCC – Pin 2).

The proposed system empowers the module utilizing VCC (5V) and granulates it utilizing GND and our Raspberry Pi that impart the info sign to TRIG, which makes the sensor communicate ultrasonic heartbeat/sound. Blowing waves detonate on any close item, and some are visible sensors. The sensor identifies these return waves and measures the time between the ammo and the recuperated beat, which at that point imparts a 5V sign to the ECHO pin.

The 5V output coming from the ECHO pin is then converted into a 3V signal using a voltage divider. Then, the 3V signal is then sent into the GPIO pin of Raspberry Pi.

8.3.1.6 Jumper Wires

Figure 8.4 shows the jumper wire used in the proposed system for connection between different components. Jumper strings are used on breadboards to "jump" from one connection to another. It has different connectors at each end. The end of "pin" will go into breadboard.

8.3.1.7 The Breadboard

Breadboard provides a way to connect the ultrasonic range sensor and Raspberry Pi without mixing them. Holes in the breadboard are connected by a pattern. The jump wires are connected to the breadboard.

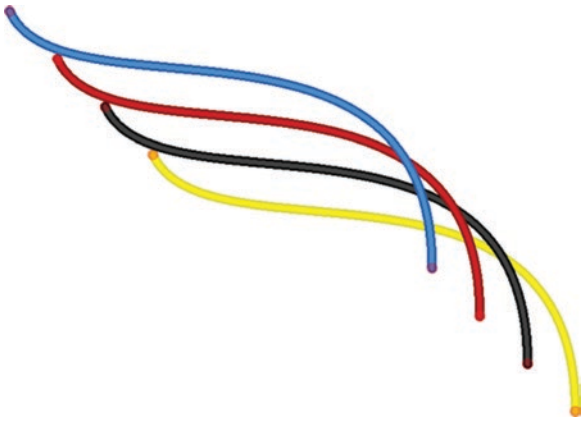

Fig. 8.4 Jumper wires for device connection

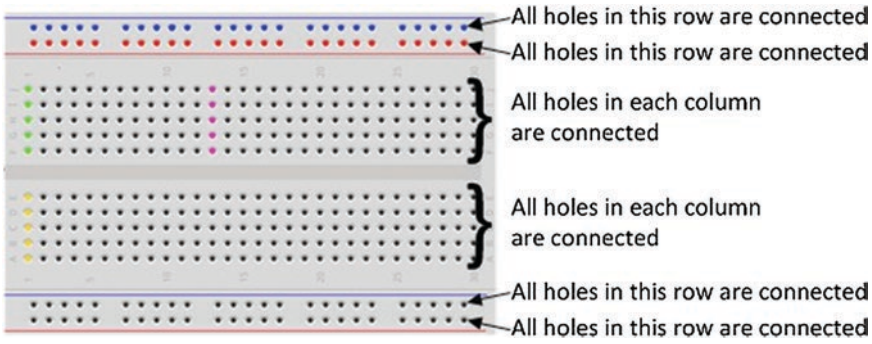

Fig. 8.5 Breadboard [19]

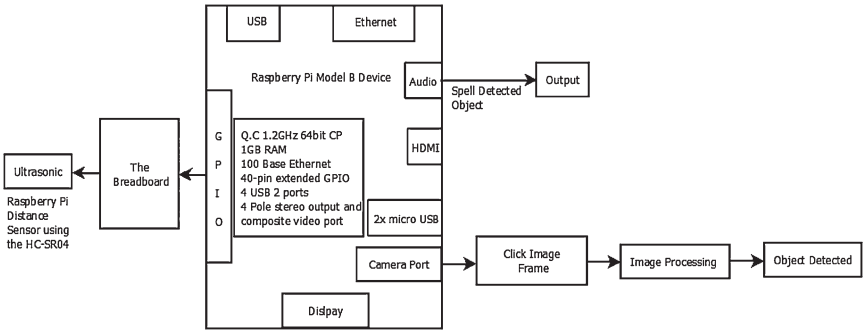

Fig. 8.6 Architectural of proposed real-time object detection

Figure 8.5 shows the breadboard used in the proposed system for adding different components required in the system.

8.4 Proposed System Architecture

The proposed system of this paper consists of a Raspberry Pi camera, a voice recognition device, an image processing unit, the Raspberry Pi Model 3B+, and an ultrasonic sensor.

The first proposed system detects the object as it comes in front of the system with the help of an ultrasonic sensor; after that, Raspberry Pi camera captures the image of the object and processes it using SSD technique to detect and recognize the object and after that convert its voice from the text.

Figure 8.6 shows the architectural design of the system and included different component and their connectivity to perform a dedicated operation.

8.5 Methodology

8.5.1 Object Detection Algorithm

Various algorithms are available such as RCNN, Fast RCNN, SSD, and YOLO, to observe objects from nearby. RCNN uses a field motion method to generate possible bounding boxes in an image. It applies different converts to classify each box and then output the result with a bounding box and label it with the classified object. But the RCNN model is difficult to train.

Fast RCNN uses maximum pooling to find regions and then combines the computation of ConvNet to locate features from each region and produces facilities from all regions at once.

Fast RCNN is based on RCNN with some improvements. After the last layer of the caster, RCNN includes a region-based network. Both RCNN and Fast RCNN methods increase computational time with precision. But pipelines of these methods are still relatively complex and difficult to adapt. All these methods were inappropriate for locating objects in real time.

Therefore, considering the need to find the real-time objective in our project, we use the SSD algorithm. SSDs can efficiently detect relatively good objectives with high speed.

SSD The SSD algorithm is designed for real-time object detection. SSD runs a convolutional neural network (CNN) image. CNN scans the image only once and calculates the related feature map. The 3×3-sized CNN kernel is working on the feature map to force bounding boxes and also calculate the possibility of a hierarchical object.

Figure 8.7 describes the architecture of SSD with different boundary box; these border boxes are hand-selected. SSD characterizes the scale as an incentive in each layer of the element map. Beginning in the left, Conv4_3 gets things with at least 0.2 (or 0.1 here and there) and afterward rises individually in the correct layer on a size of 0.9. Consolidating the size of the scale with the components of the objective components, we figure out the width and height of the default boxes. For layers that make six forecasts, the SSD begins with five objective sizes: 1, 2, 3, 1/2, and 1/3.

Figure 8.8 shows the architecture block of a convolutional neural network (CNN). The SSD takes a single shot at the image to detect multiple objects. In the case of high amplitude, the SSD faster works better than the RCNN. SSD has two components:

1. **Spinal model:** The backbone model is a pre-trained image classification network with a profound neural organization that can pull out from the picture. It does not usefully join layers. Two hundred fifty-six 7×7 feature maps have been created from the backbone model.

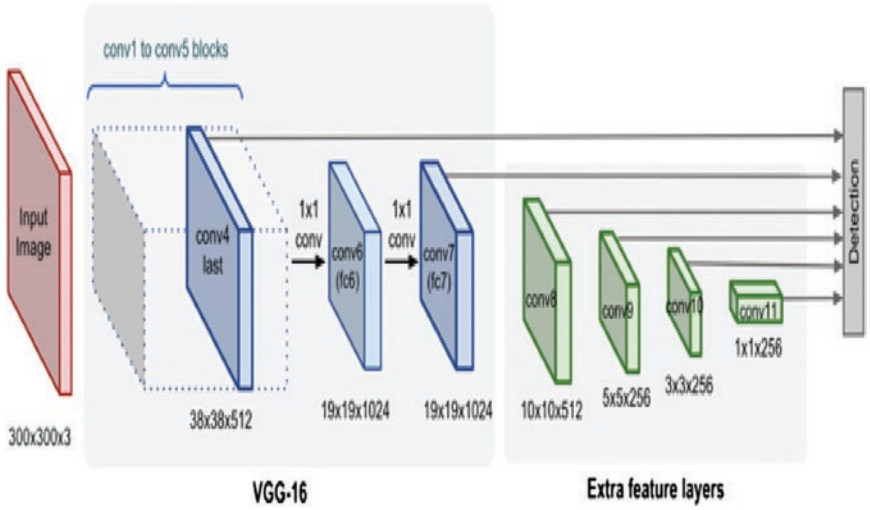

Fig. 8.7 Architecture of SSD [21]

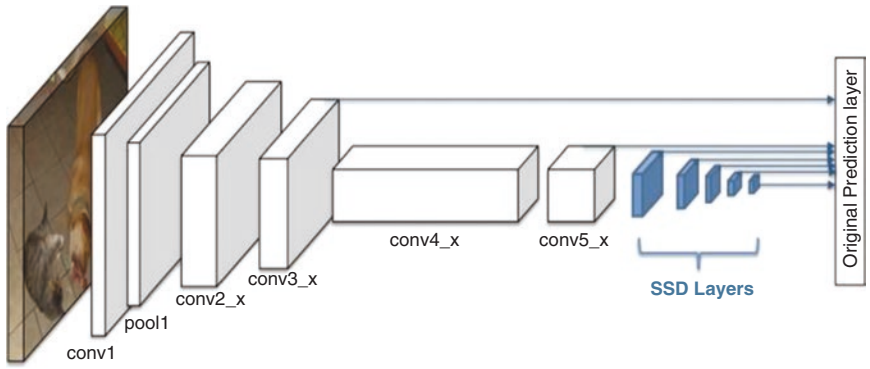

Fig. 8.8 Architecture of a convolutional neural network with an SSD detector [20]

2. **SSD head:** The head of an SSD is at least one concentric layer that is added to this spine. The head is responsible for the bounding of boxes at the spatial location of the object and identifies its orbit in the last layer activation.

Grid Cell

The location and size of that object are output in each network cell. Rather than utilizing a sliding window, the input picture (feature map) is divided into several grids, where the network cell is answerable for finding objects here of the cell picture.

This implies foreseeing the square and area of an article inside that territory. This is done in the major part of SSD. Ignore space if there is no object in the cell.

Figure 8.9 shows how an object can follow a grid cell. If a single grid cell contains multiple objects or contains various items for identifying numerous objects of various shapes, an anchor box and responsive field are used.

Anchor Box: Initially, every network cell is appointed with various covered anchor/earlier boxes/default boxes. Such anchor boxes are pre-characterized. SSD contains 8732 default boxes.Everyone is liable for the size and shape of a network cell. While training, SSD matches the fitting presenter box with the proper framing boundary boxes of each ground truth object inside a picture. The presenter box with the most extensive level of cover with an object is then anticipated with the item's class and position. This property is utilized for preparing the organization for foreseeing the distinguished objects and their areas.

Figure 8.10 shows two boxes that covered the object in horizontal and vertical length.

Aspect Ratio: All things are not square. Some are longer, and some are excessively broad. Aspect ratio parameter is used to demonstrate the assorted point of view extents of the presenter box accomplices with each grid cell at each zoom/scale level.

Figure 8.11 shows a bounding box that covered the complete bottle with proper object detection accuracy

Zoom Level The zoom boundary is utilized to indicate how much vertically up or down the anchor boxes should be about every grid cell.

Receptive Field This is the territory of the input space. Various layers of CNN have various sizes of shapes. As the CNN layers develop, the size is recognized by

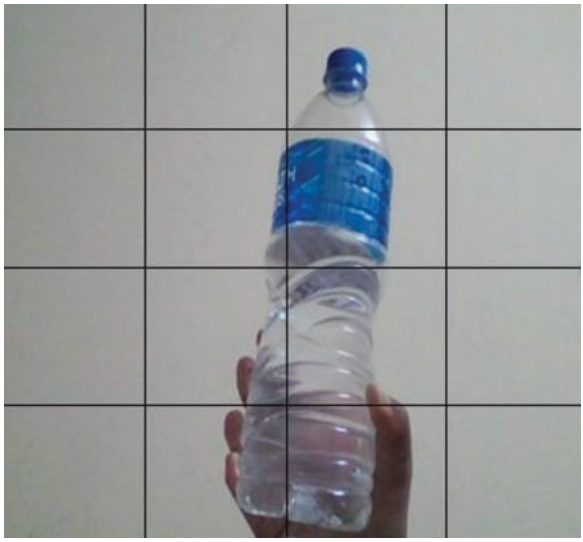

Fig. 8.9 Example of a 4×4 grid

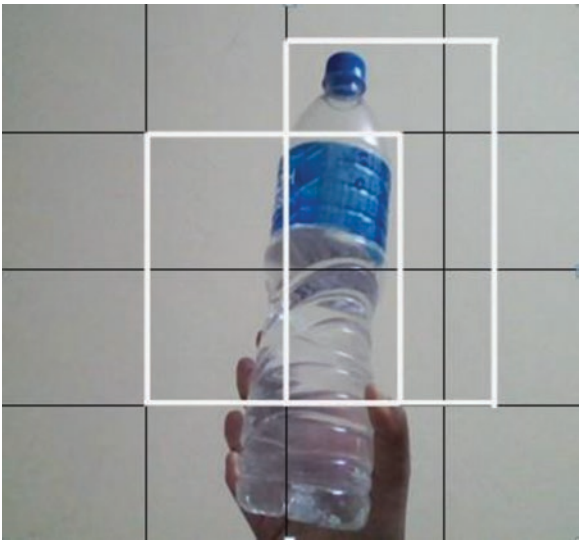

Fig. 8.10 Example of two anchor (presenter) boxes

Fig. 8.11 The bounding box of the bottle (category_44 is the alias of the bottle)

the expansion of the component. In the example underneath, the base layer is 5×5, and afterward, in the wake of applying a convolution, it changes into the center layer (3×3), where a feature (green pixel) can be recognized to the 3×3 region of the input layer (base layer). And afterward, again, applying the convolution to the center layer brings about the top layer (2×2), where each property relates to a 7×7 zone on the info picture. These feature maps (orange, green) apply a similar feature extractor to various areas of the information map in a sliding window way.

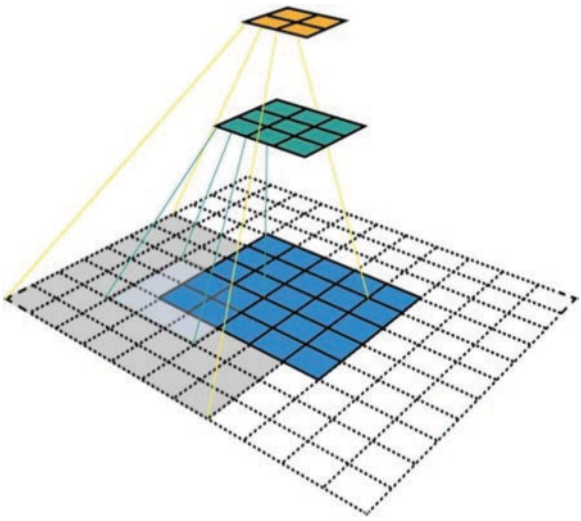

Fig. 8.12 Visualizing CNN feature maps and receptive field [21]

Figure 8.12 shows the complete visualization of the CNN feature map and receptive field with its different levels of layers from the top layer and a center layer.

8.5.2 Proposed System Flowchart

Figure 8.13 shows the proposed system flowchart for describing each step and its result for further processing.

The following steps explain the above-drawn flowchart:

1. The first ultrasonic sensor detects an object such that if it detects an object in a range less than 5 cm, then the vibration motor starts vibrating and the camera connected with the system gets enabled.
2. Now Pi camera captures the image and sends it to Raspberry Pi for recognition using the SSD algorithm.
3. After recognizing an object, its text value is passed to the text to speech module which converts it to voice for the speaker.
4. The user can get directions for a particular location by sending a voice signal to the mic attached with the device. Then, voice to text system converts it to text. Now, using the GPS location system, we get a minimum path, and the speaker gives step-by-step instruction for a particular location path.

Figure 8.14 shows the flowchart for single-shot multiple-object detection (SSD) for object detection from an image frame.

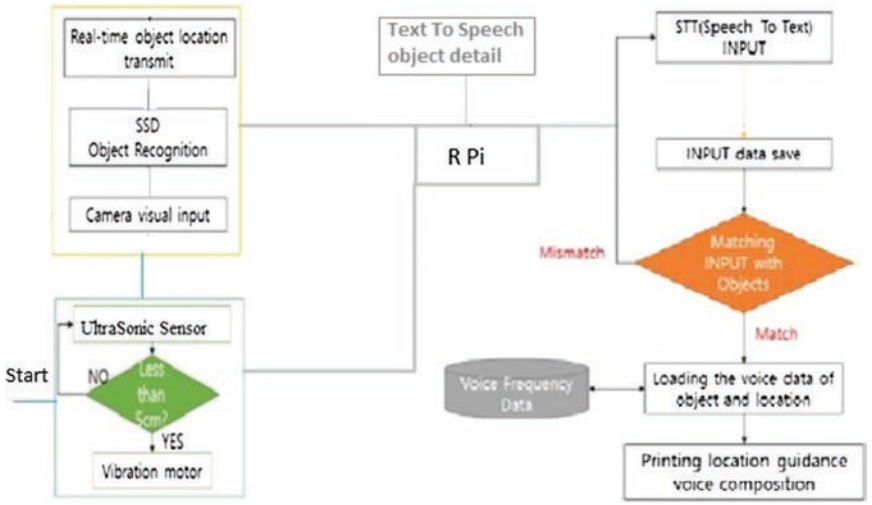

Fig. 8.13 Proposed system flowchart

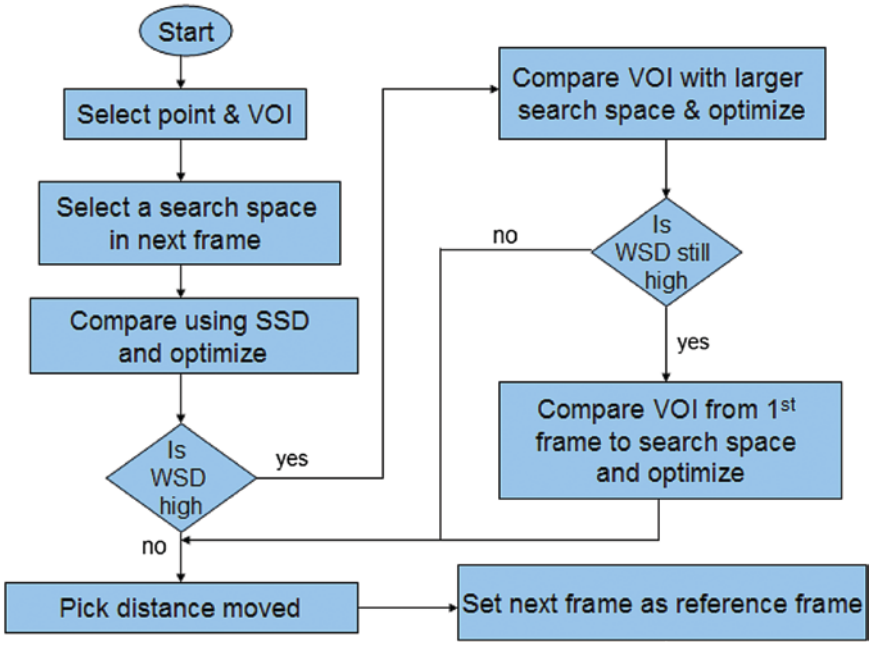

Fig. 8.14 Flowchart of SSD algorithm

8.5.3 Pseudocode with Description

8.5.3.1 SSD Algorithm Steps

Choose the height of the box b.

Input Data
P(x)←Input Picture
Col←Convolutional Layer
W(b)←Size of Box
M(f)←Feature Map
dim←dimension of boxes 4×4, 8×8, 16×16
P(i)←Change in Intensity of pixel

Output Data
B←2^{nd}no. of boxes in W(b)

Method

```
Introduce the size 1 till dim
  loop1 will be continued till W(b) recognize M(f)
  {
    a. M(f) ← { Min Col +Max P(i)}
  }
  loop2 continue till each P(i)
  {
    if P(i)==1
    {
    compute
    i. Breadth (b)= Col × P(i)
    ii. Peak  point (p)= Col ÷ P(i)
    }
    otherwise
    {
    alter the size of box with another conceivable measurement
    }
  }
```

8.5.3.2 Strategy in Recognizing Box Capacity

Calculation 2: Set of matchbox M(B).

Sources of Info
α←Boundary value
t←No. of actual box

b←Number of unused boxes
$B_e 2^b$←Set of boxes
$T_e 2^{t\times4}$←actual boxes set
c[l]←Types of labels set
N←Total no. of types of labels set
O←Final item

Technique

```
    Introduce all items with default esteems loop1 continue till
each B[i]
    {
      loop2 continued till actual box having c[l]
      {
       Similar box (B[i],class[l])=1- M(i)←{ Min Col +Max P(i)} on
condition that(B[i],c[l])≥α
        {
       c[l]=1
          i. Recognize the item (O)
          ii. Label the item (c[l])
       }
       Otherwise
       {
        c[l]=0
        Continue to step no. 2 until the class label recognized
       }
      }
    }
```

Output
Q←Total no. of class labels
I(p)←Indexed positive boxes

8.6 Implementation

8.6.1 System Configuration

The accompanying setup is utilized for the usage of the calculations built-in: Python programming language, IP-based HD camera, 15.6 in HD contact screen (1366 × 768), Intel Core i7-1065G7 1.3 GHz up to 3.9 GHz, 8GB DDR4 SDRAM 2666 MHz, 512GB SSD, HD audio with sound system speakers, Realtek

RTL8821CE 802.11b/g/n/ac, Bluetooth 4.2, one HDMI 1.4, one USB 3.1, and two USB 3.1. Python programming was running on a Windows 10 OS.

8.6.2 Dataset Description

We have used the dataset from the IoT-based system, the image dataset, and the video dataset. Image dataset consists of around 2000 images which are used for object detection like humans and many other objects. Image datasets are collected from the SGSITS (Shri Govindram Seksaria Institute of Technology and Science) college event, some family events. In our experiment, we have used a training and testing ratio of 80:20%.

8.6.3 Performance Analysis

8.6.3.1 Real-Time Object Recognition

The performance of the proposed object recognition system based on SSD technology was configured to operate on the Raspbian OS with a Raspberry Pi microcontroller. However, the deep learning technology required a lot of computational processing, which proved too much for Raspberry Pi's processing power. While the SSD model organized the TensorFlow object detection API code in the Python development language, the SSD model consumed 18.106504 seconds. Also, when real-time object recognition was performed, the SSD model performed at 17 FPS.

The proposed object recognition system as shown in Figures 8.15, 8.16, 8.17, and 8.18 found the object and provided the location information invoice format.

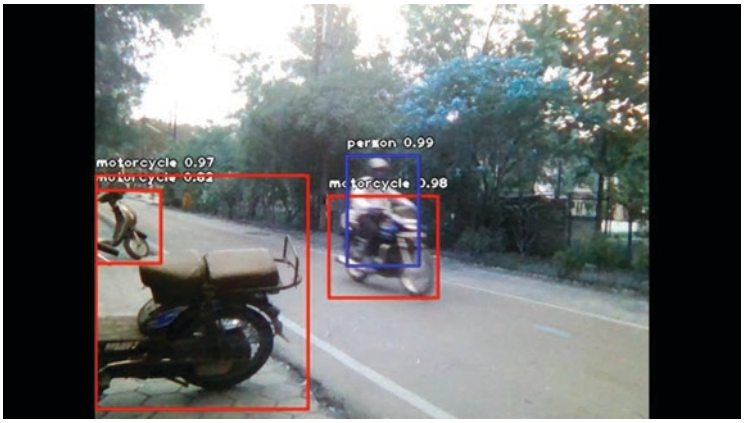

Fig. 8.15 This shows object detections in a vehicle mobile environment

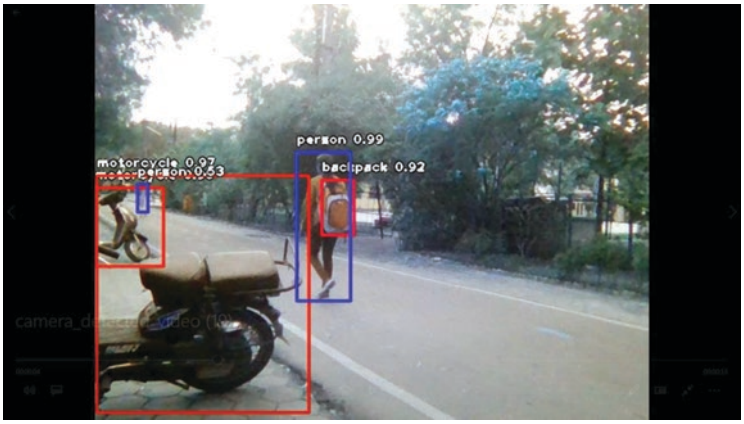

Fig. 8.16 This shows object detections in a vehicle and human mobile environment

Fig. 8.17 This shows object detections in a human mobile environment

After that, the test subject found objects with only voice-guided location information. The above experiment was conducted 50 times by changing the position of the object, and the test subject pinpointed objects 46 times. Thus, we pegged the accuracy of the proposed object recognition system at 94.16%.

Fig. 8.18 This shows object detection in a steady environment

Table 8.1 Results for hardware performance in our model

Evaluation list	Performance
Image analysis rate (on i7 CPU specs)	SSD: 18.106504 seconds
The real-time object recognition rate	17 frames per second
STT rate	2-second latency
Object detection system	The average accuracy of 94.16%

8.7 Result

The developed system has been using Raspberry Pi for detecting objects and GPS for particular location search. Object detection in the range of 5 cm is done by the ultrasonic sensor which causes us to distinguish the article close by, and it gives caution to alarm them about the nearing object. System performance is observed at the hardware level as well as a concerned individual algorithm level.

Table 8.1 gives the numerical values for hardware performance obtained for the image and video dataset that issued for the AIIoT-based real-time object detection.

8.7.1 Algorithm Performance

Develop system using single-shot multiple-object detection (SSD) for object detection and algorithm performance in terms of accuracy is 94.6% on 2000 images dataset. The algorithm is implemented on Raspberry Pi development. The following is the result of the algorithm for the training dataset.

Table 8.2 gives the numerical values obtained for the image and video datasets that are trained for the SSD model of the IoT-based system. The obtained values are satisfying. The results show that the model is well trained for any incoming inputs.

Table 8.2 Results for training dataset in SSD

Data	Precision	Recall	F1
Image 1	0.97	0.92	0.95
Image 2	0.98	0.90	0.94
Image 3	0.98	0.89	0.93
Image 4	0.97	0.88	0.92
Image 5	0.98	0.84	0.94
Image 6	0.99	0.89	0.96

Table 8.3 Results for testing dataset in SSD

Data	Precision	Recall	F1
Image 1	0.97	0.89	0.93
Image 2	0.99	0.90	0.95
Image 3	0.98	0.84	0.94

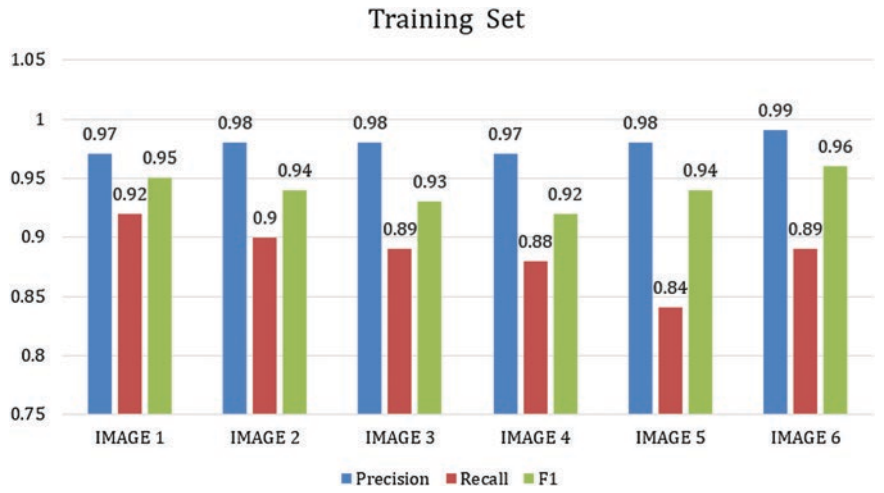

Fig. 8.19 The figure for training dataset in SSD

The training dataset observations in Figure 8.19 for the SSD model have been stated in the graph. The results obtained are more accurate. The average precision is 0.98, the average recall obtained is 0.87, and the average F1 measure is 0.94.

Table 8.3 shows results obtained for the testing dataset used for the SSD implementation from an IoT-based system have achieved greater heights. The image and the video datasets were used here for the evaluation. The average value obtained is 0.95, 0.84, and 0.90 for precision, recall, and F1 measures, respectively.

The graphical representation in Figure 8.20 shows the numerical values obtained for all three parameters, i.e., precision, recall, and F1 measure. The values obtained at an average is far better in SSD for the IoT-based system.

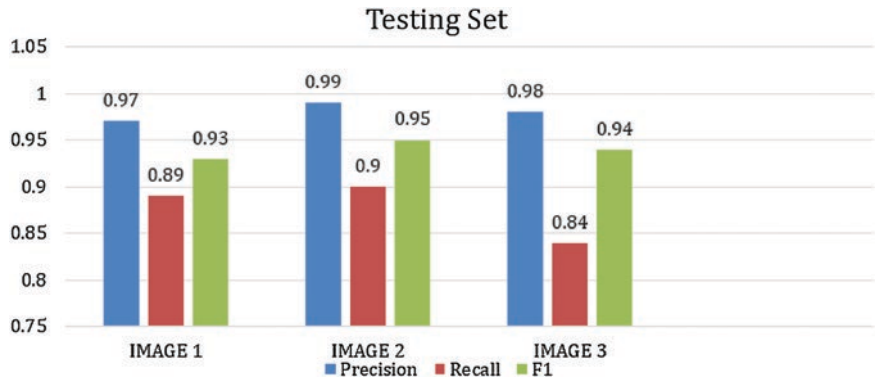

Fig. 8.20 The figure for testing set in SSD

8.8 Conclusion

In this paper, we develop an object observation system using a deep learning object recognition technique and voice recognition technology. This system's voice synthesis provides convenient features for the visually impaired. As one of the areas where deep learning technology can be applied, our study was conducted by focusing on how to effectively aid the blind. As a result, voice recognition and voice guidance technologies were added to the system, and its performance was tested. For security reasons, wired serial communications were used instead of a wireless server. If the information is linked to a server, it could be leaked onto the Internet. Since the information in question contains a lot of privacy- and camera-based observations, such leaks could create critical security issues for users. However, a wired connection can secure the information by keeping it offline. Continuous research is expected to solve server security problems, eliminate blind spots in observations by connecting the Internet of Things (IoT) cameras to a secure network, and increase precision in object recognition. This study can be used widely to provide the blind with privacy and convenience in everyday life. Also, it is expected to be applied to industrial areas where diminished visibility occurs, such as coal mines and sea beds, to greatly help production and industrial development in extreme environments.

References

1. Viola, P., & Jones, M. (2001). *Rapid object detection using a boosted cascade of simple features*. In Computer Vision and Pattern Recognition. Proceedings of the 2001 IEEE Computer Society Conference on CVPR 2001. vol. 1. IEEE, pp. I–I.
2. Dalal, N., & Triggs, B. (2005). *Histograms of oriented gradients for human detection*. In Computer Vision and Pattern Recognition, 2005. IEEE Computer Society Conference on CVPR 2005. vol. 1. IEEE, pp. 886–893.

3. Nguyen, D. T., Nguyen, T. N., Kim, H., & Lee, H. J. (2019). A high-throughput and power-efficient FPGA implementation of Yolo CNN for object detection. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 27*(8), 1861–1873.
4. Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Doll'ar, P. (2018). Focal loss for dense object detection. In *IEEE transactions on pattern analysis and machine intelligence*.
5. Girshick, R. (2015). Fast r-CNN. In *Proceedings of the IEEE international conference on computer vision* (pp. 1440–1448).
6. Zhao, H., Li, Z., Fang, L., & Zhang, T. (2020). A balanced feature fusion SSD for object detection. *Neural Processing Letters*.
7. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., & Berg, A. C. (2016). SSD: Single shot multibox detector. In *European conference on computer vision* (pp. 21–37). Springer.
8. Wang, L., Shi, J., Song, G., & Shen, I. F. (2007, November). Object detection combining recognition and segmentation. In *The Asian conference on computer vision* (pp. 189–199). Springer.
9. Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 779–788).
10. Ren, S., He, K., Girshick, R., & Sun, J. (2015). Faster r-CNN: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems* (pp. 91–99).
11. Lakshminarasimhan Srinivasan, D. S., & Amutha, A. L. (2018). Image Captioning-A Deep Learning Approach. *International Journal of Applied Engineering Research, 13*(9), 7239–7242.
12. Meftah, S., & Seminar, N. (2018, May). *A neural network model for part-of-speech tagging of social media texts*. In Proceedings of the eleventh international Conference on Language Resources and Evaluation (LREC 2018).
13. Shahira, K. C., Tripathy, S., & Lijiya, A. (2019). Obstacle detection, depth estimation, and warning system for visually impaired people. In *TENCON 2019—2019 IEEE region 10 conference (TENCON), Kochi, India* (pp. 863–868).
14. Chen, L. B., Su, J. P., Chen, M. C., Chang, W. J., Yang, C. H. & Sie, C. Y.. (2019). *An Implementation of an Intelligent Assistance System for Visually Impaired/Blind People*.
15. Gupta, P., Kulkarni, A., & Sarda, A. (2013). An embedded health care supervisory systems. *International Journal of Latest Trends in Engineering and Technology (IJLTET), 3*, 379–386.
16. Ariyanto, M., Haryanto, I., Setiawan, J. D., Munadi, M., & Radityo, M. S. (2019, November). Real-time image processing method using raspberry Pi for a car model. In *2019 6th International Conference on Electric Vehicular Technology (ICEVT)* (pp. 46–51). IEEE.
17. Kole, D., Chakraborty, S. & Bhunia, S., *Ultrasonic Peripatetic Scanner For Autonomous Test Bench Using Raspberry Pi*.
18. Run, R. S., Chang, J. H., & Yen, M. C. (2016). *The wire-free breadboard – A feasibility study on digital circuit*.
19. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C. Y., & Berg, A. C. (2016, October). SSD: Single shot multibox detector. In *European conference on computer vision* (pp. 21–37). Springer.
20. Younis, A., Shixin, L., Jn, S., & Hai, Z. (2020, January). Real-time object detection using pre-trained deep learning models MobileNet-SSD. In *Proceedings of 2020 the 6th international conference on computing and data engineering* (pp. 44–48).
21. Gupta, P., Sharma, V., & Varma, S. (2021). People detection and counting using YOLOv3 and SSD models. *Materials Today: Proceedings*