YOLO Object Detection And Voice Alert For Visually Impaired

B.Surya[1], S.U.Vino[2], R.Mogana Krishnan[3], K.M.Sai Kiruthika[4]
[1, 2, 3] Dept of computer science and engineering
[4]HOD, Dept of computer science and engineering
[1, 2, 3, 4] GKM College of Engineering and Technology

Abstract- *In recent years, with the development of object recognition technology, various industries have benefited from its implementation in areas such as autonomous vehicles, robots, and industrial facilities. However, individuals who are visually impaired, and who require this technology the most, are not able to fully utilize its benefits. To address this issue, this paper proposes an object detection system for the blind that utilizes deep learning technologies. The system incorporates voice recognition technology to identify the objects that a blind person needs, and then employs object recognition to locate the requested items. Additionally, a voice guidance technique is used to inform the visually impaired of the object's location. The deep learning model utilized in the object recognition system is based on a deep neural network architecture. Furthermore, speech-to-text (STT) technology is used to design the voice recognition component, while text-to-speech (TTS) technology is employed to synthesize the voice announcement for the visually impaired. The system is implemented using the python OpenCV tool. Through experimentation, the system is shown to be an efficient object-detection system that enables the blind to locate specific objects without assistance from others. Overall, the proposed system provides a practical solution to the problem of limited accessibility to object recognition technology for the visually impaired.*

Keywords- OpenCV, Text-ToSpeech, YOLOV3, Non-Max Suppression

I. INTRODUCTION

Object detection is a critical process within the field of computer vision that allows for the identification and recognition of real-world objects in images or videos. It is a technique that enables the recognition, localization, and detection of multiple objects within an image or video, and it has several practical applications in various areas such as image retrieval, and advanced driver assistance systems. There are several ways to achieve object detection, including Feature-Based Object Detection, Viola-Jones Object Detection, SVM Classifications with HOG Features, and Deep Learning Object Detection. In particular, object detection is especially important in video surveillance applications, where it involves the identification of specific objects in video sequences and the clustering of pixels of these objects. TensorFlow is an open-source software library that is widely used for building, training, and deploying object detection models. It provides a wide range of detection models pre-trained on various datasets, including COCO, Kitti, and Open Images. One of the popular detection models is the combination of Single Shot Detector (SSDs) and Mobile Nets architecture, which is fast, efficient, and requires only limited computational resources to achieve object detection. Another popular object detection technique is YOLO (You Only Look Once), which applies a single neural network to the entire image, dividing it into regions and predicting bounding boxes and probabilities for each region. YOLO's end-to-end optimization approach enables it to achieve high detection performance with a single network.

1.1 Problem Statement

This project aims to address the problem of limited access to object recognition technology for visually impaired individuals. Despite the development of object recognition technology for use in autonomous vehicles, robots, and industrial facilities, visually impaired individuals have not benefited from these advancements. This lack of access to such technology can create significant challenges for individuals with visual impairments, limiting their ability to live independently and complete everyday tasks. In light of this problem, the proposed solution is the development of an object detection system that uses deep learning and voice recognition technology to aid visually impaired individuals in locating objects in a specific space without the need for assistance from others. The implementation of such a system would help enhance the independence of visually impaired individuals and improve their quality of life by enabling them to identify and locate objects with greater ease and efficiency.

1.2 Existing System

Region-based Convolutional Neural Networks (RCNN): RCNN is a deep learning-based object detection

algorithm that uses region proposals to detect objects. The algorithm first generates region proposals using selective search or other similar algorithms, and then extracts features from each proposal using a CNN. The extracted features are then fed into a set of support vector machines (SVMs) to classify the region and determine the object's location. RCNN has high accuracy but is relatively slow, making it unsuitable for real-time applications. Faster RCNN (F-RCNN): F-RCNN is an improved version of RCNN that uses a single deep neural network to generate region proposals and classify objects. It uses a region proposal network (RPN) to generate proposals and feeds the proposals and the input image through a CNN to classify and refine the proposals. F-RCNN is faster and more accurate than RCNN and is suitable for real-time applications. Convolutional Neural Networks (CNN): CNNs are a type of deep neural network that can be used for object detection. CNNs consist of multiple layers of interconnected neurons that can learn features from images through a process called convolution. CNNs are trained using large datasets of labeled images and can detect objects in real-time. Histogram of Oriented Gradients (HOG): HOG is a classic algorithm used in computer vision for object detection. It works by computing histograms of image gradients and using them to detect object boundaries. HOG is a relatively simple algorithm and is faster than deep learning-based algorithms. However, it is less accurate and requires manual feature engineering.

1.3 Proposed System

The proposed work involves building a system that uses YOLO v3, a deep learning based object detection algorithm, to detect objects in real-time and provide voice alerts to visually impaired individuals. The system will be designed to work with both image and video stream inputs. In YOLO v3, the input image is divided into a grid, and each cell in the grid is responsible for detecting objects within that cell. Each cell predicts a fixed number of bounding boxes, each with its associated objectness score and class probabilities. YOLO v3 uses a combination of different techniques, including anchor boxes and multi-scale feature maps, to improve object detection accuracy. Once an object is detected, the system will generate a voice alert to inform the visually impaired user about the object. The system will use text-to-speech conversion to convert the object label into speech and provide real-time feedback to the user. For example, if the system detects a chair in the image, it will generate a voice alert saying "chair detected." The proposed system will also include a graphical user interface (GUI) to display the detected objects in the image or video stream. The GUI will draw bounding boxes around the detected objects and display their labels. This will be helpful for sighted individuals who are assisting the visually impaired user. One of the major advantages of YOLO v3 is its speed, making it suitable for real-time applications. The proposed system will be designed to take advantage of this speed to provide real-time feedback to visually impaired individuals. The system will also be trained on a large dataset to improve object detection accuracy and reduce false positives. In summary, the proposed work involves building a real-time object detection system using YOLO v3 to assist visually impaired individuals. The system will use text-to-speech conversion to provide voice alerts about the detected objects and include a GUI to display the detected objects. The system will be designed to be fast and accurate and trained on a large dataset to improve object detection accuracy.

II. RELATED WORKS

In many fields, the detection and tracking of target objects while handling occlusions and other complexities is essential. To address this, numerous researchers, such as Almeida and Guting (2004), Hsiao-Ping Tsai (2011), and Nicolas Papadakis and Aurelie Bugeau (2010), have attempted various approaches to object tracking, with techniques varying depending on the application domain. Object detection is an important yet challenging task, critical to many applications, such as image search, image auto-annotation and scene understanding, object tracking, and smart video surveillance (Arun Hampapur 2005), artificial intelligence, military guidance, safety detection, robot navigation, medical and biological applications. Despite the number of successful single-object tracking systems developed in recent years, detecting objects becomes difficult when several objects are present, and objects are partially or fully occluded. The proposed MLP-based object tracking system utilizes an optimal selection of unique features and the Adaboost strong classification method, making it robust. Horprasert et al.'s (1999) background subtraction method, which modeled the background statistically on each pixel and classified each pixel into four categories: original background, shaded background or shadow, highlighted background, and moving foreground objects, coped with local illumination changes such as shadows and highlights. Liyuan Li et al. (2003) developed an approach to detect foreground objects in non-stationary complex environments containing moving background objects, and Álvaro Bayona et al. (2010) developed an algorithm focused on obtaining stationary foreground regions, useful for applications such as the detection of abandoned/stolen objects and parked vehicles. Template matching is a technique for finding small parts of an image that match a template image, and Schweitzer et al. (2011) developed an algorithm that uses upper and lower bounds to detect the 'k' best matches, with Euclidean distance and Walsh transform kernels used to calculate the match measure. There

are two categories of visual tracking methods: feature-based and region-based, as proposed by Ken Ito and Shigeyuki Sakane (2001). The feature-based approach estimates the 3D pose of a target object to fit the image features (edges), given a 3D geometry.

III. YOLO OBJECT DETECTION

The proposed work aims to develop a system that employs YOLO v3, a deep learning-based object detection algorithm, to identify objects in real-time and provide voice alerts to individuals with visual impairments. This system will function with both image and video stream inputs. YOLO v3 splits the input image into a grid, with each cell responsible for detecting objects within its bounds. Each cell will predict a fixed number of bounding boxes, each with its associated objectness score and class probabilities. YOLO v3 employs several techniques, including anchor boxes and multi-scale feature maps, to improve object detection accuracy. Once an object is identified, the system will generate a voice alert conveying the object's label to the user. Text-to-speech conversion will convert the object label into speech and provide real-time feedback to the user. Furthermore, the system will come equipped with a graphical user interface (GUI) to display detected objects in the image or video stream. The GUI will draw bounding boxes around the detected objects and display their labels, allowing sighted individuals to assist the visually impaired user. YOLO v3's speed is a significant advantage, making it suitable for real-time applications. The proposed system will be designed to take advantage of this speed to provide fast and accurate feedback to visually impaired individuals. Additionally, the system will be trained on a vast dataset to improve object detection accuracy and minimize false positives. Overall, the proposed work involves constructing a real-time object detection system utilizing YOLO v3 to assist individuals with visual impairments. The system will be designed to be fast and accurate, use text-to-speech conversion to provide voice alerts about detected objects, and incorporate a GUI to display detected objects.

IV. METHODOLOGIES

The field of Computer Vision that focuses on detecting instances of semantic objects in images/videos by creating bounding boxes around them is known as Object Detection. Afterwards, we can convert the annotated text into voice responses and provide the basic positions of the objects within the view of the person or camera.

1. The Common Objects In Context (COCO) dataset is used to train the model for Object Detection, which detects semantic objects in images/videos by creating bounding boxes around them. You can check out the annotated images in the link provided.
2. The You Only Look Once (YOLO) algorithm is used as the model, which runs on a complex Convolutional Neural Network architecture called Darknet. Although we are using the advanced YOLO v3 model, the original YOLO algorithm will also be explained. The python cv2 package can set up Darknet from the configurations in the yolov3.cfg file.
3. To expedite the process, a pre-trained model will be used, as others have already trained COCO on YOLO v3, and the weights are stored in a file of over 200MB.
4. Weights are analogous to finding the best fit line in Linear Regression. In our complex prediction task, there are millions of Xs when we feed images into the network, and each X has an m value. These m values are the predicted weights, which have been continuously adjusted to minimize a loss function and are stored in the yolov3.weights file.
5. For input data, the webcam will feed images at 30 frames per second, and the model will process every other frame to speed things up.
6. The objects detected by the model will be classified as a string and include the object's coordinates in the image. The position of the object, such as "top," "mid," "bottom," "left," "center," or "right," will be appended to the object class prediction. The text description will be sent to the Google Text-to-Speech API using the gTTS package.
7. The coordinates of the bounding box of each detected object will be obtained and overlaid on the object in the frame. The frames with overlaid bounding boxes will be returned as a video playback. Additionally, voice feedback will be scheduled for the first frame of each second, such as "bottom left cat," indicating a cat was detected on the bottom-left of the camera view.

Flow Chart

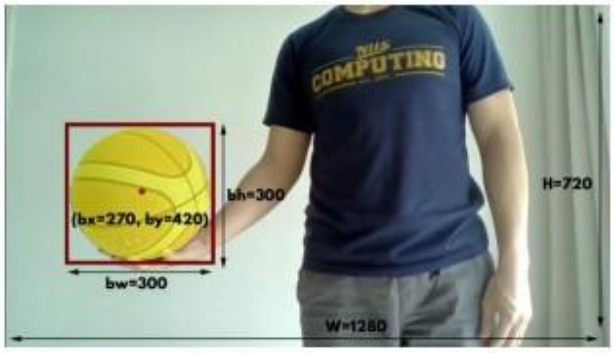
4.1 Training Time

The coco.names file indexes object classes. C is the class index for the object being labeled, and in the case of a sports ball, C=33. The COCO dataset has already been used

for training.Pixel values are measured from the top-left corner at (0,0). An example of how images are labeled during model training is provided. During training, we manually label each object in the image with 5 values in the format C bx by bw bh. We also normalize the four b values to a range of 0-1, representing them as proportions of the image's width and height (1280 x 720 px).

```
29  suitcase
30  frisbee
31  skis
32  snowboard
33  sports ball
34  kite
35  baseball bat
36  baseball glove
37  skateboard
```

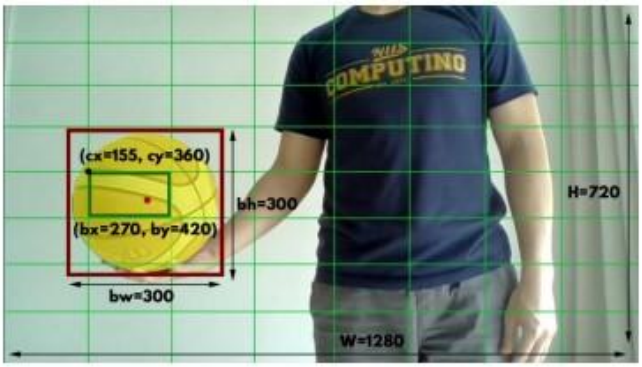
If there are two labeled objects in the frame, such as a sports ball and a person, they can be represented by tensors, which are a general form of a vector that will be fed into the model during training.



Object Detected

4.2 Prediction Time

At Prediction time, our camera feeds frames with dimensions of 1280 x 720 into YOLO. The YOLO algorithm automatically resizes the frame to 416 x 234 and pads any excess space with 0s, allowing it to fit into a standard-sized network of 416 x 416. The YOLO algorithm then divides the resized image into S x S cells, each with a size of 32 x 32, resulting in a total of 13 x 13 cells for our 416 x 416 image (where reduction factor=32). To illustrate this, we can use an 8x8 cell grid, with the dark-green box representing the cell that contains the center of the object.



Prediction Time

4.3 Non Max Suppression

The grid cell containing the object's center (dark green box) may have B duplicate detections, but NMS is used to eliminate detections with low box confidence scores (BC) to avoid predicting multiple sports balls when there is only one. A possible implementation of NMS involves the following steps:

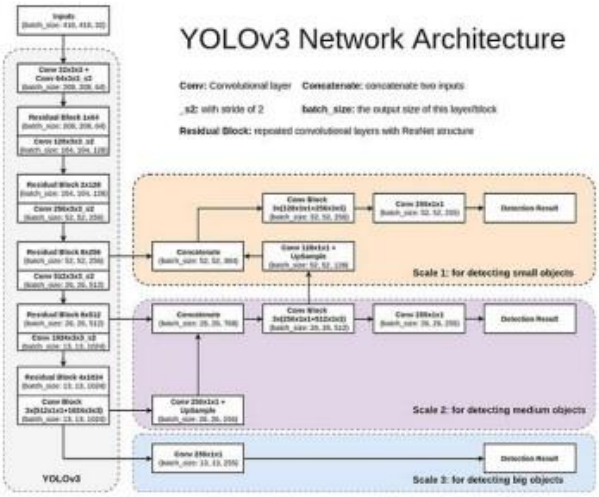
1. Select the bounding box with the highest BC as a starting point.
2. Eliminate any remaining bounding boxes that have an overlap greater than the specified threshold of 0.5.
3. Repeat steps 1 and 2 until there are no more bounding boxes left.

4.4Multi Scale-Prediction

What was the DarkNet output? It was a 13 x 13 x 1024 grid, where each cell can serve as the center of a bounding box. However, since each cell is composed of 32 x 32 pixels, only large objects can be detected using this representation. In the case of small objects like a 16 x 16 pixel one, it can't be recognized because the box's center is 32 x 32 pixels. To address this limitation, the idea of multi-scale prediction suggests predicting using different grid sizes, not just a 13 x 13 grid. Prior to applying the last stride-2 convolution, the size would have been 26 x 26 x 512, and before the second-last stride-2 convolution, the size would have been 52 x 52 x 256. This approach reduces the number of pixels contained within each cell, allowing for the detection of smaller objects. Therefore, to detect smaller objects, predicting bounding boxes with grid sizes of 13 x 13, 26 x 26, and 52 x 52 is necessary. This concept led to the development of Feature Pyramid Networks (FPNs), a network designed to address the issue of detecting objects at different scales.

The concept of Image Pyramid has been around in image processing for 30-40 years. If an image is subsampled and multiple stride-2 convolutions are applied, the image size is halved each time, resulting in a pyramid-like structure. This concept is utilized in FPNs. YOLOv3's authors proposed the idea of having a convolutional network for each grid size to generate bounding boxes: NetworkNo1 for 13 x 13, NetworkNo2 for 26 x 26, and NetworkNo3 for 52 x 52. However, there is an issue as the information learned at one level is not being reused by the level below it to predict bounding boxes. To address this, FPNs were introduced to leverage real-time data analytics information at multiple scales. In the DarkNet architecture, stride-2 convolutions are used for downsampling, and a bunch of convolutions are applied to obtain a grid size of 13 x 13 x 255, from which bounding boxes are predicted. To predict boxes, information from the same level and the level above it is used. Borrowing information from smaller grid sizes to predict boxes at larger grid sizes helps retain more information instead of discarding it. This architectural approach is sometimes called FPNs with lateral connections.

Combining Boxes from Various Scales:



YOLOv3 Architecture

Let's perform a quick calculation. Suppose my image is 416 x 416 in size, and I'm computing three bounding boxes at three different grid sizes: 52 x 52 cells, 26 x 26 cells, and 13 x 13 cells. Each of these cells can potentially be the center of a bounding box, and at most three bounding boxes can be predicted. This results in a total of 10,647 bounding boxes. However, this is the maximum number of bounding boxes possible at a multi-scale level, and many of these bounding boxes may be very small, nearly zero in size.

The top value of 32 refers to the batch size used for sending input batches. The DarkNet 53 architecture spans from the initial convolutional layer up to the final residual block, which results in an output of 13 x 13 x 1024. As previously mentioned, a 255-1 x 1 convolution can be used to convert the output from 13 x 13 x 1024 to 13 x 13 x 255, as seen in YOLO v1. However, the creators of YOLO v3 decided to add more convolutions to increase the model's power. In particular, they used three sets of convolutions consisting of 512 1 x 1 kernels followed by 1024 3 x 3 kernels and another 255 1 x 1 kernels to convert the output to 13 x 13 x 255. This approach improves performance by using 3 times 512 kernels with 1 x 1 convolution, followed by 1024 kernels of 3 x 3, rather than using all 1 x 1 convolutions. This is because the 13 x 13 cell grid needs information from surrounding cells to predict the bounding box accurately.

Using the FPN concept, the same approach is applied to the 26 x 26 and 52 x 52 cell grids to detect objects of different sizes. To predict bounding boxes for the 26 x 26 grid, data is obtained from the 26 x 26 x 512 residual block previously used for the 13 x 13 grid. Information from the 13 x 13 grid is also utilized through a 256 1 x 1 convolution that converts the 13 x 13 x 1024 output from the DarkNet 53 block to 13 x 13 x 256, which is then upsampled to 26 x 26 x 256. By concatenating this output with the 26 x 26 x 512 data from the previous residual block, the final output becomes 26 x 26 x 768.
Finally, a similar process is applied to predict bounding boxes for the 52 x 52 cell grid.

1. Filtering

We have 'n' and 'm' values for every box. By multiplying each 'n' and 'm' with all the 's', we obtain 80 probabilities for each box. We then select the maximum probability among all of them. If this maximum probability is less than 0.5, we can disregard it as it is not useful. Although we previously obtained over 10,000 boxes, we only require the relevant ones and can discard the rest.

Filtering

2. Voice Feedback

Using the values of bx and by, which are relative to the width and height of the image, we can determine the location of the detected objects and convert it into a text string, which can then be passed on to gTTS using the following command:
tts = gTTS("The sports ball is located towards the middle left of the image", lang='en')

Once the audio file is generated, I utilized pydub and ffmpeg to perform various manipulations on the audio file.

3.Demo

Real-time frame return is not possible as it would result in jerky video playback due to processing every 30th frame. I attempted to use multi-threading to process every 30th frame in a separate process while keeping another process for video playback. However, the verbal description of objects detected in real-time on my webcam is more crucial since blind individuals cannot see bounding boxes. As for the video below, I recorded myself before generating the bounding boxes and verbal responses.

V. CONCLUSION AND FURTHER WORK

To summarize, creating a real-time object detection system with voice output for the visually impaired is a substantial contribution to enhancing accessibility and independence for this group. The system leverages advanced computer vision techniques to recognize objects and text-to-speech technology to communicate the object's name or description to the user. This real-time feedback can considerably enhance the quality of life for visually impaired individuals by helping them avoid obstacles and navigate their surroundings confidently.

As for future improvements, expanding the system's vocabulary, integrating machine learning algorithms, and adding navigation and direction capabilities are areas that can enhance the system's performance. Additional sensors such as ultrasonic and infrared sensors can further improve the system's obstacle and hazard detection capabilities. Overall, the proposed system has the potential to improve the accessibility and independence of visually impaired individuals, and with continued development and improvement, it can become an invaluable tool for navigating their daily lives more efficiently and with greater ease.

REFERENCES

[1] TensorFlow Object Detection API tutorial Tensorflow Object Detection API https://github.com/tensorflow/models/tree/master/research/object_detection
[2] Mansi Mahendru, Sanjay Kumar Dube, Real Time Object Detection with Audio Feedback using Yolo, Published in: 2021 11th International Conference on Cloud Computing, Data Science & Engineering (Confluence)
[3] Juan Du, Understanding of Object, Detection Based on CNN Family and YOLO, 2018
[4] N. Najva, K. Bijoy, SIFT and Tensor Based Object Detection and Classification in Videos Using Deep Neural Networks, 2016.
[5] Md. Moshiur Rahman; Shajeeb Chakma; Dewan Mamun Raza; Sadia Akter; Abdus Sattar, Real-Time Object Detection using Machine Learning, 2021
[6] Krizhevsky A, Sutskever I, Hinton G E "ImageNet classification with deep convolutional neural networks" International Conference on Neural Information Processing Systems, 2012.
[7] Yang-Lang Chang, Amare Anagaw, Lena Chang, Yi Chun Wang, Chih-Yu Hsiao, and Wei-Hong Lee"Ship Detection Based on YOLOv2 for SAR Imagery" Remote Sensing 2019
[8] Caglar Gulcehre, Deep Learning, 2015, http://deeplearning.net/
[9] Daniyal Rajput, Faheem Ahmed, Habib Ahmed, Engr Zakir Ahmed Shaikh, Aamir Shamshad, 2019, "Smart Obstacle Detector for Blind Person," Jo
[10] Prof. Seema Udgirkar, Shivaji Sarokar, Sujit Gore, Dinesh Kakuste, Suraj Chaskar, 2019, "Object Detection System for Blind People," International Journal of Innovative Research in Computer and Communication Engineering
[11] Payal Panchal, Gaurav Prajapati, Savan Patel, Hinal Shah and Jitendra Nasriwala, "A Review on Object Detection and Tracking Methods,"2018.
[12] L. A. Johnson and C. M. Higgins. A navigation aid for the blind using tactile-visual sensory substitution. Conf Proc IEEE Eng Med Biol Soc, 1:6289–6292, 2016.