WWW.IJRPR.COM

International Journal of Research Publication and Reviews

Journal homepage: www.ijrpr.com ISSN 2582-7421

Yolo-Based Real-Time Object Detection with Voice Assistance for Visually Impaired Navigation

Fazila Shariff[1], Gandeti Dilleeswari [2], B.Siddartha Gowtham[3], Bonela Mounika[4], A.Sai Sharmila[5], Dr. S.Shanmathi[6]

(22341A0456),
(22341A0457),
(22341A0434),
(22341A0432),
(22341A0405),
B.tech ,Rajam 532127 , India

ABSTRACT :

The Real-Time Object Detection System with Voice Alerts is designed to assist visually impaired individuals in navigating their surroundings safely and independently. This system leverages YOLOv8, a state-of-the-art object detection model based on convolutional neural networks (CNNs), to recognize and classify objects in real time. The system continuously captures images from a webcam, balancing speed and accuracy to provide instant recognition. The integration of Google Text-to-Speech (gTTS) ensures that detected objects are announced through voice feedback, enhancing user awareness. This technology promotes autonomy, safety, and confidence, enabling individuals to move through both familiar and unfamiliar environments with ease.Object detection is a crucial field of computer vision, enabling machines to analyze images and videos, identify objects, and provide meaningful interpretations. This project employs OpenCV for video capture, PyTorch as the deep learning framework, and threading to optimize real-time processing. By integrating computer vision and AI-based voice assistance, this system provides a practical and cost-effective solution for accessibility, security, and automation.

Keywords: Real-time object detection, Voice alerts, Environmental awareness, Computer vision, YOLO V8.

1.Introduction

Computer Vision:

Computer vision has become a transformative field in artificial intelligence, enabling machines to interpret and understand visual information from the world. Among its many applications, object detection plays a vital role in areas such as surveillance, autonomous driving, medical imaging, and smart city systems. This project focuses on object detection using YOLOv8 (You Only Look Once, version 8), the latest iteration of the YOLO family known for its speed and accuracy. YOLOv8 offers a unified framework that balances high performance with real-time processing capabilities, making it suitable for a wide range of practical applications. The objective of this project is to implement and evaluate an object detection system using YOLOv8. By leveraging its advanced deep learning architecture, the project aims to identify and locate multiple objects within images or video streams efficiently. Throughout this project, we will explore the dataset preparation, model training, evaluation, and potential deployment scenarios. The goal is to demonstrate how state-of-the-art object detection techniques can be applied to solve real-world challenges using a robust and efficient model like YOLOv8.

1.1 Background And Motivation:

Vision plays a crucial role in our everyday life. From identifying vehicles on the road and locating objects to avoiding obstacles while walking, our ability to see allows us to interact effectively and safely with the world around us. However, for millions of people who are blind or visually impaired, even simple tasks like crossing the street or recognizing objects in their path can become challenging and potentially dangerous. According to the World Health Organization (WHO), approximately 285 million people worldwide live with some form of visual impairment, with 39 million classified as blind and 246 million having low vision. A significant number of them rely heavily on others for assistance with navigation and basic daily tasks. This dependency not only limits their independence but also affects their confidence, social participation, and quality of life.

With advancements in Artificial Intelligence (AI), particularly in the fields of computer vision and machine learning, it is now possible to build systems

that can "see" and "understand" visual environments. These technologies can be leveraged to assist visually impaired individuals by identifying objects in real-time and conveying that information through audio feedback. This project is designed with that exact goal in mind — to empower the visually impaired with a Real-Time Object Detection System with Voice Alerts that acts as an intelligent assistant or "digital guide.

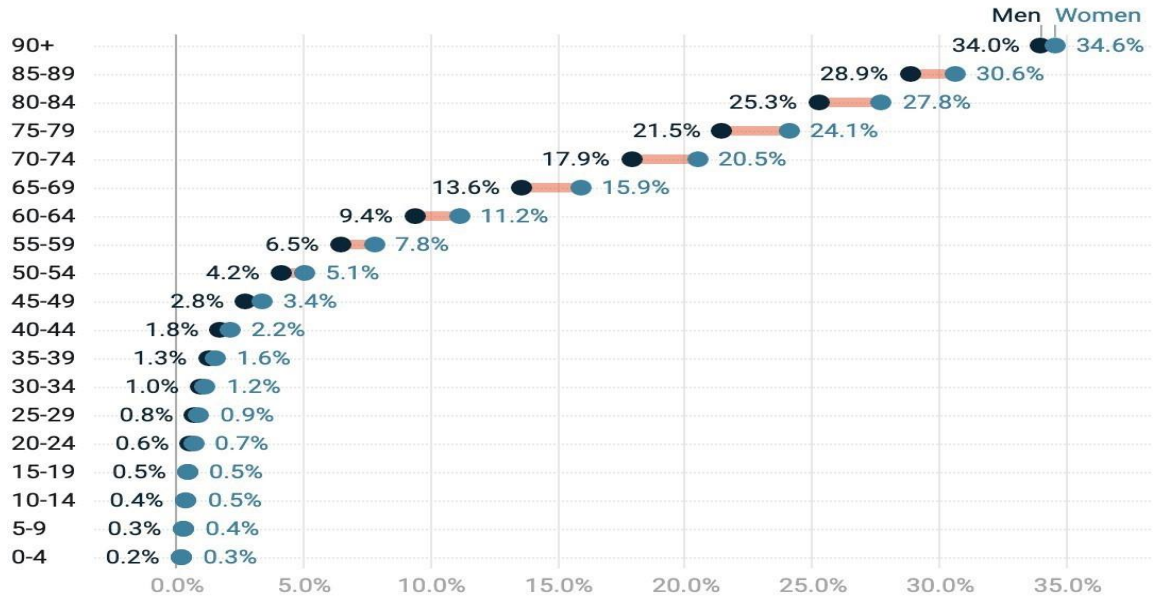

Fig. 1.1: Crude Prevalence (%) of Moderate to Severe Vision Impairments by Age and Gender

1.2 TYPES OF YOLOV8:

YOLOv8 (You Only Look Once version 8), developed by Ultralytics, comes in several model sizes. Each one offers a different trade-off between speed, accuracy, and resource usage, so you can pick the one that best fits your project or hardware setup.

1.YOLOv8n (Nano): This is the smallest and fastest model in the YOLOv8 family. It sacrifices some accuracy in exchange for extremely fast inference and minimal resource use.Best for: edge devices, mobile applications, or real-time tasks where speed and efficiency are more important than precision.

2. YOLOv8s (Small): A lightweight model that strikes a better balance between speed and accuracy than Nano. It's still fast, but slightly more accurate. Best for: systems with limited but decent hardware where you want solid performance without high resource demand.

3.YOLOv8m (Medium): A middle-ground option that balances model size, speed, and accuracy. Best for: general-purpose use cases where you have average hardware and need good, reliable detection performance.

4. YOLOv8l (Large): This model leans more toward accuracy but is larger and slower than the previous ones. Best for powerful machines or cloud setups where you can afford the extra compute in return for better results.

1.3 YOLO EVOLUTION:

Evolution of YOLO (You Only Look Once) Models Up to YOLOv8.The YOLO (You Only Look Once) object detection algorithm has undergone significant evolution since its introduction in 2015. Here's a breakdown of its development through version 8:

YOLOv1 (2015): YOLOv1 introduced the concept of real-time object detection by treating object detection as a regression problem rather than a classification problem. A single convolutional neural network was used to predict bounding boxes and class probabilities directly from full images in one evaluation, eliminating the need for complex pipelines. This novel approach enabled it to achieve speeds of up to 45 frames per second (FPS), with a faster variant reaching up to 150 FPS. However, despite its impressive speed, YOLOv1 struggled with accurately detecting small objects and exhibited issues with localization, making it less suitable for applications requiring fine-grained detection.

YOLOv2 (YOLO9000, 2016): YOLOv2, also known as YOLO9000, addressed many of the limitations present in the first version. It introduced batch normalization across all convolutional layers, which significantly improved convergence during training. Anchor boxes were added to enable more precise bounding box predictions, while dimension clustering was used to determine better anchor box priors. YOLOv2 also implemented direct location prediction, which enhanced the stability of training. A standout feature of this 4 version was its ability to detect over 9,000 object categories through a joint training approach using both detection and classification datasets, greatly expanding its versatility and utility.

YOLOv3 (2018): YOLOv3 brought major improvements in both architecture and performance. It introduced feature pyramid networks (FPN) to enable multi-scale detection, which greatly enhanced the detection of objects at varying sizes, especially smaller ones. The backbone network was upgraded to Darknet-53, a more powerful and efficient architecture with residual connections and more convolutional layers. YOLOv3 made use of three different detection scales and adopted binary cross-entropy loss for class predictions instead of softmax, making it better suited for multi-label classification tasks. These improvements led to a more balanced model that delivered strong accuracy while maintaining high inference speed.

YOLOv4 (2020): YOLOv4, developed by Alexey Bochkovskiy, incorporated a wide range of advancements to optimize both detection accuracy and speed. It adopted CSPDarknet53 as the backbone network to enhance gradient flow and reduce computational load. Feature aggregation was improved using PANet, while the Spatial Attention Module (SAM) helped the model focus on important spatial features in the input image. The inclusion of the

Mish activation function added further non-linearity benefits, contributing to performance gains. YOLOv4 struck a strong balance between accuracy and speed, making it one of the most practical versions for real-world use cases at the time of its release.

YOLOv5 (2020): YOLOv5 was released by Ultralytics and stood out as the first version of YOLO written entirely in PyTorch, unlike previous versions which relied on the Darknet framework. This made YOLOv5 much more accessible and easier to integrate into modern deep learning workflows. It introduced automated anchor box generation to streamline model configuration and used extensive data augmentation techniques such as mosaic and mixup to improve generalization. The architecture was also simplified for ease of deployment, and its modular design allowed for faster experimentation. Despite not being an official version from the original authors, YOLOv5 gained immense popularity due to its performance and user friendliness.

YOLOv6 (2022): Developed by Meituan in 2022, YOLOv6 introduced several architectural and efficiency-oriented innovations. It featured a RepVGG-style backbone, which balanced computational efficiency with high accuracy. YOLOv6 adopted an anchor-free detection 5 head, simplifying the detection mechanism and reducing overhead. The use of SimOTA for dynamic label assignment during training improved the accuracy of object matching. Additionally, YOLOv6 was optimized for hardware efficiency, making it suitable for deployment on edge devices where computational resources are limited. This version emphasized practical implementation while maintaining strong detection capabilities.

YOLOv7 (2022): YOLOv7 continued to push the boundaries of object detection by introducing the concept of a "trainable bag-of-freebies," a set of enhancements that could be applied during training without increasing inference time. This version expanded on efficient layer aggregation networks and offered flexible model scaling to accommodate different hardware constraints and use cases. YOLOv7 aimed to improve accuracy without compromising on speed, making it an ideal choice for scenarios where both real-time performance and precision are critical. The improvements in model structure and training strategies made YOLOv7 a strong contender in the object detection landscape.

YOLOv8 (2023): YOLOv8, the latest release from Ultralytics in 2023, represents the most advanced and versatile version of the YOLO family. It introduced a f a fully anchor-free detection head, simplifying the design while improving detection accuracy. A new and more efficient backbone network enhanced both speed and precision. YOLOv8 added native support for not just object detection and classification but also instance segmentation, making it a comprehensive tool for various computer vision tasks. The architecture is highly flexible and modular, allowing for custom configurations based on specific requirements. Advanced

data augmentation techniques and robust training strategies further boosted performance. Moreover, its user-friendly Python API made YOLOv8 more accessible to developers, researchers, and industry practitioners alike.

Fig 1.2: Timeline of YOLO

YOLOv8 represents the current state-of-the-art in the YOLO series, offering improved performance, flexibility, and ease of use compared to previous versions while maintaining the real-time capabilities that made YOLO famous. YOLO (You Only Look Once) has significantly advanced the field of computer vision by introducing a fast, efficient, and accurate approach to real-time object detection. Unlike traditional two-stage detectors (like R-CNN and Faster R-CNN), which first propose regions and then classify them, YOLO treats object detection as a single regression problem, predicting bounding boxes and class probabilities directly from the image in one pass. This makes it exceptionally fast, capable of running at 30-60+ FPS on standard GPUs, and even on edge devices like NVIDIA Jetson or mobile phones. In computer vision applications, YOLO processes an image by dividing it into a grid system, where each grid cell predicts multiple bounding boxes along with confidence scores and class probabilities. The model uses a convolutional neural network (CNN) backbone (such as Darknet or CSPDarknet) to extract features, followed by a neck architecture (like FPN or PANet) to combine multi-scale features for better detection of objects of varying sizes. The detection then refines these predictions, outputting precise bounding box coordinates, objectness scores, and class labels. YOLO has evolved to support not just object detection but also instance segmentation (YOLOv8), pose estimation (YOLOv8-Pose), and multi-object tracking (MOT when combined with algorithms like DeepSORT or ByteTrack. Its efficiency makes it ideal for autonomous vehicles, surveillance, robotics, medical imaging, and industrial automation. However, challenges remain, such as detecting small or heavily occluded objects and maintaining accuracy in high-resolution scenes. Future advancements may integrate Transformer-based architectures (like YOLO TR) for better contextual understanding, 3D object detection for LiDAR data, and further optimizations for edge AI deployment. YOLO's balance of speed and accuracy ensures its continued dominance in real-time computer vision applications, pushing the boundaries of what's possible in AI-driven visual perception.

Literature Survey

[1] Tiny YOLO Optimization Oriented Bus Passenger Object Detection.
"ZHANG Shuo, WU Yanxia, MEN Chaoguang, and LI Xiaosong"

The real-time collection of bus passenger object detection is an essential part of developing a smart bus system. The difficulty of object detection mainly lies in the objective factors, such as clothing, hairstyle and accessories, light, etc. Traditional object detection methods with artificial feature extraction suffer from insufficient strength in expression, generalization, and recognition rate. The object detection method based on deep learning mainly uses the convolutional neural network in deep learning to learn features from a large set of data. The learned features can describe the rich information inherent in the data, and improve the expression ability of the features as well as the recognition accuracy. Due to too many parameters of the Convolutional neural network (CNN) model, the amount of calculation is too large to be operated on the vehicle terminal. To reduce the calculation burden and improve the operation speed, we employ the depth wise separable convolution method to optimize the convolutional layer of the tiny YOLO network model. It decomposes a complete convolution operation into depth wise convolution and point wise convolution, thus reducing the parameter amount of the CNN and improving the operation speed. The experiment results reveal that the speed of bus passenger object detection detected by our improved model is 4 times faster than the previous one but with the nearly same detection accuracy.

Fig.2.1: The above is the result showing the small objects getting detected

Methodology

YOLOv8 (You Only Look Once, Version 8) is a state-of-the-art deep learning model designed for real-time object detection, classification, and segmentation. Developed by Ultralytics, it brings significant improvements in speed, accuracy, and efficiency over earlier versions. YOLOv8 is capable of analysing entire images in a single forward pass, making it suitable for applications like autonomous driving, surveillance, robotics, and assistive devices. The architecture of YOLOv8 consists of three main components: Backbone, Neck, and Head. The Backbone, based on CSPDarkNet, extracts rich features from the input image. It uses convolutional layers with residual connections and CSP modules to enhance learning while keeping the model lightweight. These features help the network understand shapes, textures, and object boundaries. The Neck, built on a variation of the Path Aggregation Network (PANet), combines features from different scales to improve the detection of objects, especially smaller or overlapping ones. Attention mechanisms embedded in this layer allow the model to focus on the most relevant parts of the image. he Head is where the final predictions are made. YOLOv8 introduces an anchor free detection mechanism, which allows the model to predict object locations and class labels directly, without relying on predefined anchor boxes. This makes the detection process more flexible and reduces complexity. Optimized for deployment on a wide range of hardware, YOLOv8 supports FP16 and INT8 quantization, allowing it to run efficiently even on resourceconstrained devices like NVIDIA Jetson. Implemented in Python, it integrates smoothly with popular libraries like PyTorch and TensorFlow, making it accessible for developers. Overall, YOLOv8 offers a powerful combination of speed, accuracy, and simplicity, making it one of the most effective models for real-time object detection in practical scenarios.

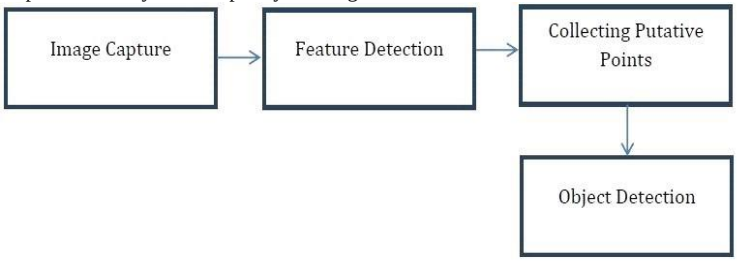

Fig. 3.1: Object detection pipeline

The object detection process in YOLOv8 begins with the input image and progresses through a series of stages, as illustrated in Fig. 3.3. This pipeline provides a high-level overview of the system's architecture and flow.

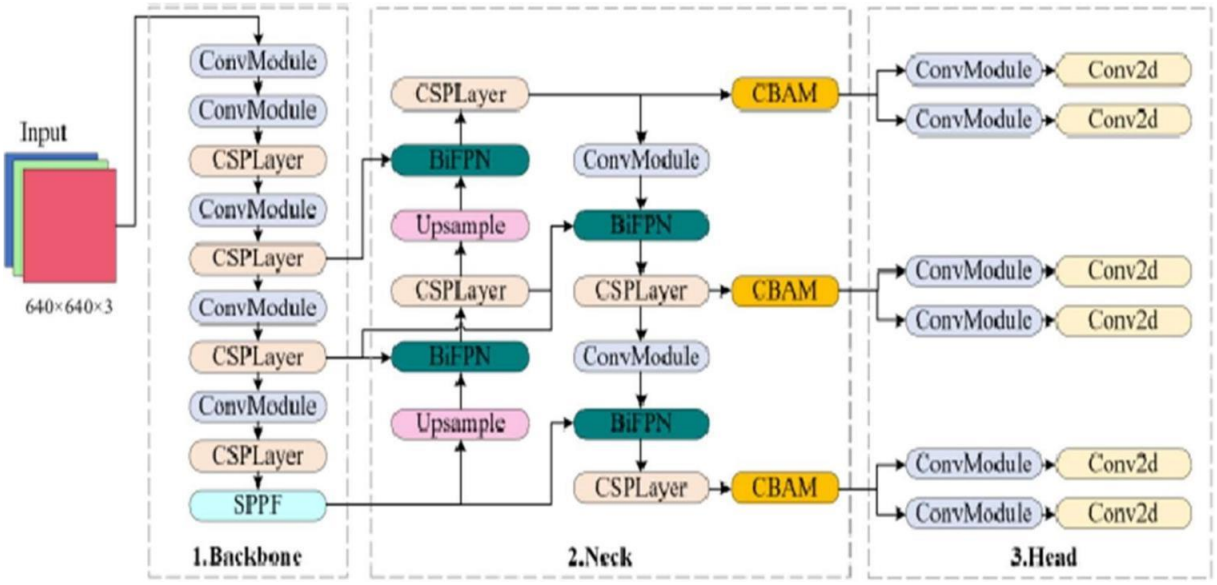

Fig. 3.2: Architecture of YOLOv8

3.1 YOLOV8 ARCHITECTURE OVERVIEW:

The architecture of YOLOv8 is structured into three key components: the Backbone, the Neck, and the Head. Each of these components plays a distinct and critical role in the object detection process, working together to deliver high accuracy and real-time performance. This streamlined yet powerful structure enables YOLOv8 to excel in a variety of real-world applications, from surveillance to assistive technologies.

3.1.1 Backbone (Feature Extraction):

The Backbone of YOLOv8 is primarily responsible for feature extraction from the input image. It processes the image through multiple convolutional layers to learn and identify meaningful spatial patterns such as edges, textures, and shapes. The input to the 16 network is an RGB image of resolution 640×640×3, where 640x640 represents the width and height, and the '3' corresponds to the three color channels—Red, Green, and Blue. Within the Backbone, convolutional modules (ConvModules) consisting of convolutional layers, batch normalization, and activation functions (such as the SiLU activation) are used to transform raw pixel data into deep feature maps. These modules are supported by specialized components such as the Cross Stage Partial Layer (CSPLayer), which promotes gradient flow and efficient feature reuse, and the Spatial Pyramid Pooling Fast (SPPF) module that captures multi-scale spatial features. Together, these components help the model to effectively analyze both fine and coarse details in the image.

3.1.2 Neck (Feature Fusion):

The Neck functions as a bridge between the Backbone and the Head, enhancing the features extracted in the previous stage and preparing them for the final detection task. One of the central elements in the Neck is the Bidirectional Feature Pyramid Network (BiFPN), which facilitates the fusion of features across different scales in both top-down and bottom up directions. This fusion enables the model to better detect objects of varying sizes, including small and overlapping ones that are often difficult to localize. Additionally, the Neck includes upsampling layers that increase the spatial resolution of the feature maps, further aiding in the detection of small objects. Another vital component is the Convolutional Block Attention Module (CBAM), which applies an attention mechanism to guide the model's focus towards the most informative and relevant regions of the image. This attention based refinement improves detection precision by reducing the influence of less significant background details.

3.1.3 Head (Final Predictions):

The Head is the final component of the YOLOv8 architecture and is tasked with generating predictions based on the processed and refined feature maps. It consists of a final ConvModule, which performs additional convolutional operations to sharpen the features, followed by a Conv2D layer that outputs the actual detection results. This output includes the bounding box coordinates (indicating the location and size of each detected object), the object class labels (such as "person," "car," or "dog"), and the corresponding confidence scores (indicating the model's certainty about each prediction). Notably, YOLOv8 employs an anchorfree detection mechanism in its Head, which eliminates the need for manually defined anchor boxes. This approach simplifies the architecture and improves flexibility, especially 17 when dealing with complex or cluttered scenes. The Head also incorporates Non-Maximum Suppression (NMS) to filter out redundant bounding boxes, retaining only the most confident detections.

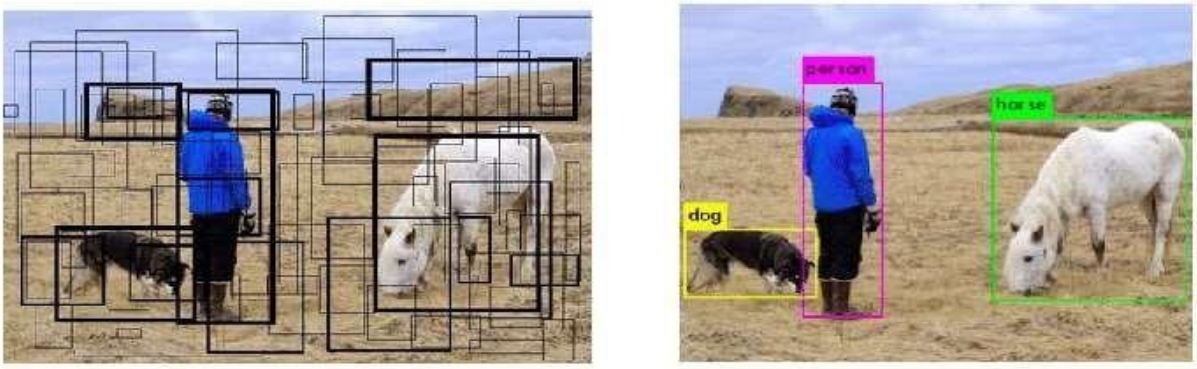

Fig. 3.3: YOLOv8 bounding boxes

3.2 CO3.2 COCO DATASET (COMMON OBJECTS IN CONTEXT):

The COCO dataset, also known as Common Objects in Context, is a large-scale image dataset developed by Microsoft for various computer vision tasks such as object detection, instance segmentation, key point detection, and image captioning. It contains more than 200,000 images with over 1.5 million object instances across 80 diverse categories. These categories include people, animals, vehicles, household items, and commonly found everyday objects, offering a comprehensive and realistic representation of the real world. Each image is annotated with bounding boxes that indicate object locations, segmentation masks that enable pixel-level object delineation, and key point annotations that specify important landmarks on the human body. This extensive labeling makes COCO an ideal benchmark for training and evaluating deep learning models, including YOLOv8, Faster R CNN, and Mask R-CNN.

3.2.1 Characteristics and Diversity:

A key strength of the COCO dataset lies in its diversity and complexity. The dataset includes images taken in natural, real-world settings with various challenges such as occlusion, cluttered backgrounds, overlapping objects, and differing lighting conditions. This makes the dataset highly valuable for developing models that generalize well to practical environments. It also supports multiple tasks in a single framework, enabling researchers to test and compare performance across detection,segmentation, and pose estimation.

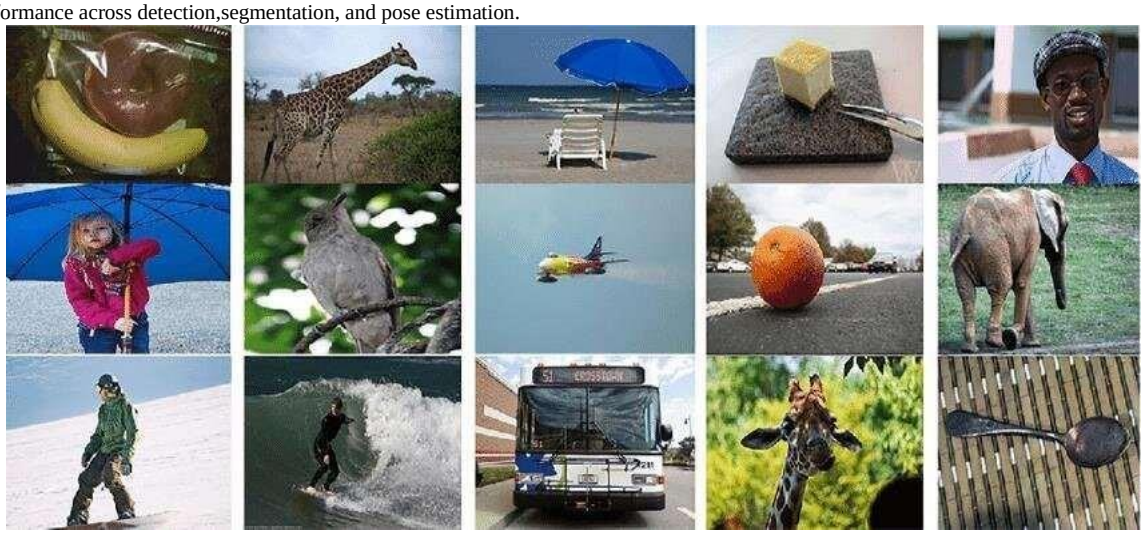

Fig 3.4: Some of the sampler images of the COCO dataset

3.2.2 Structure and Annotations:

The dataset is divided into multiple subsets, most notably train2017 and val2017, which are primarily used for training and validation, respectively. The training subset contains approximately 118,000 images, while the validation set consists of around 5,000 images. Each image in these subsets is annotated with detailed object information such as class labels and bounding box coordinates. While the COCO dataset also provides segmentation masks and human pose keypoints, models like YOLOv8 typically use only the bounding box coordinates and class labels for standard object detection tasks. Before training a YOLOv8 model on the COCO dataset, it is necessary to convert the original JSON-format annotations into a format compatible with YOLO. In YOLO format, each image has an accompanying text file containing the annotations. These annotations consist of the class ID followed by the normalized x and y coordinates of the bounding box center, along with the width and height of the bounding box, all scaled relative to the dimensions of the image. Additionally, the dataset must be organized into separate directories for training and validation images, each with corresponding label files to ensure smooth integration with the training pipeline.

3.2.3 Training with YOLOv8:

Training YOLOv8 on the COCO dataset involves the use of a configuration file in YAML format. This file specifies important parameters such as the number of object classes, the names of the classes, and the paths to the training and validation data folders. YOLOv8 supports training from scratch or using pretrained weights, although training from scratch on COCO is often used to benchmark and compare new object detection architectures. YOLOv8 introduces several architectural improvements over its predecessors, including an anchor-free 19 detection head, a redesigned neck for better feature aggregation, and support for various model scales ranging from lightweight Nano models to high-performance XLarge models. Data augmentation plays a crucial role in enhancing the model's ability to generalize. During training, YOLOv8 applies several advanced augmentation techniques. Mosaic augmentation combines four images into one, helping the model learn to detect objects at various scales and arrangements. MixUp blends two images and their labels, promoting robustness against label noise. HSV augmentation adjusts the hue, saturation, and value of the images to simulate different lighting conditions. These techniques collectively improve the model's ability to detect objects accurately in varied and complex scenarios.

3.2.4 Evaluation Methodology:

After training, the performance of the model is evaluated using standard object detection metrics established by the COCO evaluation protocol. These include average precision, or AP, calculated at different Intersection over Union (IoU) thresholds. For example, AP50 measures precision at an IoU of 0.5, while AP75 measures it at a stricter 0.75 threshold. A more comprehensive metric, AP@[.5:.95], computes the mean precision across ten IoU thresholds from 0.5 to 0.95 in increments of 0.05. In addition to these general metrics, the COCO evaluation system also provides metrics for different object sizes, such as APs for small objects, APm for medium-sized objects, and APl for large objects. These metrics offer insights into how well a model performs across a range of object scales and scene complexities. The COCO dataset remains a cornerstone in the field of computer vision, particularly for object detection using deep learning models like YOLOv8. Its comprehensive annotations, challenging real-world imagery, and standardized evaluation framework provide a reliable benchmark for developing, training, and validating high-performance object detection systems. The integration of COCO with YOLOv8 facilitates a robust pipeline for real-time object detection and ensures that model performance is comparable with other leading-edge systems. This makes the COCO dataset not only essential for academic research but also highly applicable to real-world deployments in areas such as autonomous vehicles, surveillance, assistive technologies, and robotics.

3.3 ALGORITHMS USED IN YOLOV8 FOR OBJECT DETECTION:

YOLOv8 incorporates a combination of advanced algorithms and modern strategies to achieve real-time object detection with remarkable speed and accuracy. These methods are applied across various stages of the architecture, including the backbone, neck, head, and loss computation mechanisms. Each stage contributes uniquely to enhancing the model's performance and precision. The backbone of YOLOv8 is primarily responsible for extracting rich visual features from the input image. It makes use of the c2f module, which stands for cross-stage partial with fusion. This module is an evolution of the CSP (cross-stage partial) architecture and is designed to reduce computational overhead while preserving gradient flow. By enabling efficient reuse of features, it contributes to deep and expressive representations. The convolutional layers used in the backbone are both standard and depth-wise separable, ensuring compactness without compromising performance. For activation, YOLOv8 employs the SiLU (Sigmoid Linear Unit), also known as Swish, which introduces smooth gradients and helps in achieving better convergence and accuracy during training. The neck of the model is responsible for aggregating and refining features from different stages of the backbone to enhance object detection at various scales. YOLOv8 integrates a combination of Feature Pyramid Networks (FPN) and Path Aggregation Networks (PAN) in the neck design. The FPN helps to combine semantically strong features from lower-resolution layers with spatially rich features from higher-resolution layers. On the other hand, the PAN structure introduces both bottom-up and top-down paths, facilitating better feature propagation and improving localization, especially for detecting small objects. This robust aggregation of multi-scale features ensures that the model can detect objects of varying sizes effectively. Additionally, neck outputs are carefully prepared for further processing by the head, with structures that support multi-level predictions. The head of YOLOv8 is responsible for making final predictions, including class labels, bounding box coordinates, and confidence scores. Unlike earlier versions, YOLOv8 uses an anchor-free detection approach, directly predicting the canter, width, and height of bounding boxes from feature maps. This reduces training complexity and improves inference speed without sacrificing accuracy. It features separate branches for classification and box regression, enhancing detection precision, and includes an objectness score to better identify true objects and reduce false positives.

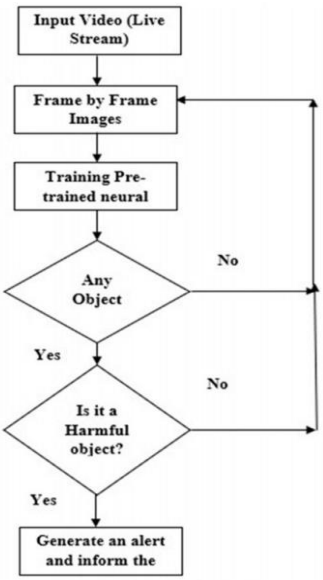

Fig. 3.5: Object detection decision flow

YOLOv8 also employs advanced loss functions tailored for anchor-free detection. These include improved bounding box regression and classification losses, optimized for accurate localization and clearer decision boundaries. Together, these design choices ensure efficient training, fast inference, and high accuracy, making YOLOv8 ideal for real-time object detection in dynamic environments. The flowchart outlines the decision-making process behind the real-time object detection and alert system, particularly tailored for visually impaired users. The flowchart illustrates the core logic behind the real-time object detection and alert system, designed to assist visually impaired individuals. The process begins with continuous video capture through a webcam. Each video frame is processed using the YOLOv8 model, which identifies objects in real-time by returning their class labels, bounding box coordinates, and confidence scores. Once objects are detected in a frame, the system evaluates each one by first classifying it (e.g., "person," "vehicle") and then checking whether it falls under a predefined list of harmful objects. These could include sharp tools, moving vehicles, or other 22 threats depending on the application. If an object is deemed harmful, the system triggers an alert usually through audio feedback using Google Text-to-Speech (gTTS), though other forms like visual warnings or caregiver notifications can also be integrated. If the object is considered safe, the system resumes monitoring without issuing alerts, avoiding unnecessary repetition. In cases where no objects are detected, the system skips evaluation and proceeds to the next frame. To maintain real-time performance and responsiveness, the system uses threading and asynchronous processing to handle detection and alerting tasks concurrently.

3.4 COMPARISON BETWEEN YOLOV8 AND YOLOV5:

Table 3.1: Comparison between YOLOv5 and YOLOv8

Model Size	YOLOv5	YOLOv8	Difference
Nano	28	37.3	+33.21%
Small	37.4	44.9	+20.05%
Medium	45.4	50.2	+10.57%
Large	49	52.9	+7.96%
Xtra Large	50.7	53.9	+6.31%

The performance comparison between YOLOv8 and YOLOv5 highlights the advancements introduced in the YOLOv8 architecture across a range of model sizes, all evaluated at a standard image resolution of 640×640 pixels. This comparison is based on average precision (AP) scores, a key metric for evaluating object detection models. In the nano model category, which targets ultra-lightweight and highly efficient deployments such as those on edge devices, YOLOv5 achieves an average precision of 28. YOLOv8, on the other hand, significantly outperforms it with a score of 37.3, reflecting an improvement of approximately 33.21 percent. This substantial leap demonstrates YOLOv8's ability to bring robust detection capabilities even to resource-constrained environments. For the small model variant, YOLOv5 records an AP of 37.4, while YOLOv8 achieves 44.9. This improvement of 20.05 percent indicates strong gains in detection accuracy for small-scale applications, such as mobile devices or embedded systems, where balancing performance and efficiency is crucial. 23 In the medium model category, often considered a balanced option between speed and accuracy, YOLOv5 posts a score of 45.4 compared to YOLOv8's 50.2. The increase of 10.57 percent represents a meaningful improvement in mid-tier models, suitable for general purpose object detection tasks that require both performance and responsiveness. Moving to the large models, which are typically used in high-performance environments where computational resources are less restricted, YOLOv5 achieves an AP of 49. YOLOv8 improves upon this with a score of 52.9, showing a performance gain of 7.96 percent. While the gain is more modest compared to the smaller variants, it still reflects an important step forward for demanding real-time applications. Finally, in the extra-large (Xtra Large) model category, YOLOv5 reaches a score of 50.7, while YOLOv8 pushes that further to 53.9, resulting in an improvement of 6.31 percent. This relatively smaller gain is expected in the largest model variants, where enhancements become harder to achieve due to the saturation of performance benefits. In conclusion, YOLOv8 delivers measurable improvements in average precision over YOLOv5 across all model sizes. The most notable gains appear in smaller models, making YOLOv8 ideal for real-time applications on lowresource devices. While larger models see smaller gains, YOLOv8 still offers meaningful advantages in detection accuracy and efficiency

RESULTS AND DISCUSSION:

In this project, we used image recognition, and voice generation modules for the development of the project. As of now, accuracy is good but in case if we want to increase the accuracy we have to train the model with more objects/images in the dataset. This project is a small experiment that is useful for blind persons, who can be able to find the objects that are surrounded by them. The blind person can stand alone and carry out tasks independently making this blind assistance device useful for object detection by voice warnings. The device's camera serves as the blind person's virtual eye, capturing every detail of their environment. The voice alerts keep the person informed about the surroundings so that accidents are decreased. There are so many people present in the world who are visually impaired and illiterate in different parts of the world. Combining YOLOv8's real-time detection with voice alerts enhances safety, accessibility, and automation, making it a powerful solution across industries.

4.1 EXPERIMENTAL ENVIRONMENT AND PARAMETER CONFIGURATION:

This experiment was conducted on a Windows 11 System with 16 GB of RAM (15.9 GB available), Intel Core i5-12600 processor, and NVIDIA GTX4060Ti graphics card. The software environment Includes Python 3.x, PyTorch, open CV, and YOLOv8(cloned from Git Hub) The Parameter configurations were defined as: all images were re-size to 640X640 Pixels.

Fig 4.1: A Sample of images from the data set.

4.2 DATA SETS:

The algorithm was experimentally improved on the COCO dataset for pre-trained models. It comprised 10,204 images, divided into training, Validation, and test sets. These images were Captured and Photographed under diverse environmental conditions us, including objects vehicles, and some categories. A selection of images from the dataset is shown in fig.1 In our experiments, we did not perform data augmentation on the dataset and directly used the original COCO Dataset for training and evaluation. the dataset was divided into training, validation, and test sets in a 7:1:2 ratio. All training and testing processes in the experiments of this paper were completed under the same hardware, software, and parameter configuration conditions. Due to the randomness in the training process of deep Learning models, we conducted multiple training for the same model. After improved the algorithm on the COCO dataset, to verify the generalization ability of the improved algorithm.

TABLE 4: YOLOv8 ALGORITHM FUNCTIONALITY TESTING

YOLOv8s	MFE	IDDH	SPD	SAC	SDA	mAP@0.5	mAP@0.5:0.95	Precision	Recall	FLOPs
✓						38.6%	23.1%	50.5%	37.9%	28.5G
✓	✓					39.3%	23.5%	50.1%	38.3%	34.4G
✓	✓	✓				45.5%	27.8%	56.1%	43.2%	45.9G
✓	✓	✓	✓			46.8%	29%	55.5%	45.5%	53.2G
✓	✓	✓		✓		45.6%	27.9%	54.7%	43.6%	26.1G
✓	✓	✓			✓	47.6%	29.3%	55.6%	45.8%	34.3G

4.3 FUNCTIONALITY TESTING:

To Validate the effectiveness of an even improved module, the ablation experiments were conducted and the results were presented in TABLE 1. From TABLE 1, we can clearly demonstrate the effectiveness of each Module we proposed. Building upon the Yolov8 s model. The addition of the MFE module enhanced future extraction capabilities, resulting in a 0.7% Improvement in mA@0.5 compared to Yolov8 s. By incorporating the IDDH, Small targets can be more accurately located, thus improving their detection capabilities.

Fig 4.2 The plot of visualization results on a dataset with YOLOv8s

It can be seen that the improved YOLOv8 algorithm obtained a good performance on the COCO dataset, we also Validated the robustness of the algorithm and validated the performance of the algorithm on other datasets performance testing of the algorithm in other datasets. which consists of images collected on the internet. It encompasses Complex scenes primarily categorized into mobile, balanced objects specifically designed for objects for detection in Long-range and large-background ratings.

CONCLUSION

In conclusion real time object detection with voice alert is a powerful solution for enhancing environmental awareness, particularly for individuals with disabilities. By using advanced technologies like YOLOV8 for object detection and GTTS for voice alerts it assist individuals specially with disabilities in navigation and all-time monitoring. Its applications are used in diverse healthcare, focusing on situational awareness, safety and interaction with surroundings in a user-friendly way. Ultimately this system aim is to create a more and safer environment for users to respond effectively with their surroundings. With its focus on real time monitoring, situational awareness, and user-friendliness, this technology paves the way for safer and smarter interactions with the environment, fostering inclusion and improving quality of life. Its adaptability and efficiency make it a cornerstone for future advancements in accessibility and safety systems. This innovative system uses AI, computer vision, and voice technology to help people with disabilities navigate their surroundings safely. By recognizing objects in realtime and providing voice alerts, it enhances environmental awareness and promotes independence. With applications in healthcare, transportation, and more, this system improves safety, accessibility, and user experience. object detection system with voice alerts is a cutting-edge innovation aimed at enhancing environmental awareness and ensuring safety. Its applications extend far beyond assisting visually impaired users, finding relevance in autonomous vehicles, healthcare, industrial settings, gaming, and security enhancement. By fostering situational awareness, timely reactions, and seamless user interaction, this integrated system aims to create safer and more inclusive environments, ultimately enriching lives and interactions with the surrounding world. By leveraging the power of modern machine learning and computer vision, the system provides a crucial layer of real-time information about the surroundings, transforming visual data into actions. This capability is vital for navigation assistance for the visually impaired, enabling them to perceive and react to obstacles and points of interest in their environment. This multimodel approach bypasses the need for constant visual attention, making it invaluable not only for visually impaired individuals but also for enhancing safety in various other contexts, including gaming, security systems, autonomous vehicles, healthcare, and industrial settings. Ultimately, this user-friendly system aims to improve situational awareness, promote safer interactions with the environment, and empower individuals with greater

independence. By focusing on accessibility and realtime monitoring, this technology holds immense potential to significantly enhance the quality of life and safety for a diverse range of user.

REFERENCES

[1] J. An, D. Hee Lee, M. Dwisnanto Putro and B. W. Kim, "DCE-YOLOv8: Lightweight and Accurate Object Detection for Drone Vision," in IEEE Access, vol. 12, pp. 170898- 170912, 2024, doi: 10.1109/ACCESS.2024.3481410.

[2] M. Yue, L. Zhang, Y. Zhang and H. Zhang, "An Improved YOLOv8 Detector for Multi Scale Target Detection in Remote Sensing Images," in IEEE Access, vol. 12, pp. 114123- 114136, 2024, doi: 10.1109/ACCESS.2024.3444606.

[3] F. Najihah Muhamad Zamri, T. S. Gunawan, S. Hajar Yusoff, A. A. Alzahrani, A. Bramantoro and M. Kartiwi, "Enhanced Small Drone Detection Using n Optimized YOLOv8 With Attention Mechanisms," in IEEE Access, vol. 12, pp. 90629-90643, 2024, doi: 10.1109/ACCESS.2024.3420730.

[4] Z. Zhang, L. Tao, L. Yao, J. Li, C. Li and H. Wang, "LDSI-YOLOv8: Real-Time Detection Method for Multiple Targets in Coal Mine Excavation Scenes," in IEEE Access, vol. 12, pp. 132592-132604, 2024, doi: 10.1109/ACCESS.2024.3450582.

[5] Y. Gong, Z. Chen, W. Deng, J. Tan, and Y. Li, "Real-Time Long-Distance Ship Detection Architecture Based on YOLOv8," in IEEE Access, vol. 12, pp. 116086-116104, 2024, doi: 10.1109/ACCESS.2024.3445154.

[6] Z. F. Khan et al., "Real-Time Polyp Detection From Endoscopic Images Using YOLOv8 With YOLO-Score Metrics for Enhanced Suitability Assessment," in IEEE Access, vol. 12, pp. 176346-176362, 2024, doi: 10.1109/ACCESS.2024.3505619.

[7] H. Yi, B. Liu, B. Zhao, and E. Liu, "Small Object Detection Algorithm Based on Improved YOLOv8 for Remote Sensing," in IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 17, pp. 1734-1747, 2024, doi: 10.1109/JSTARS.2023.3339235.

[8] H. Sodhro, A. Kannam, and M. Jensen, "Near Real-Time Efficiency of YOLOv8 in Human Intrusion Detection Across Diverse Environments and Recommendation," SSRN, 2024. Available: https://ssrn.com/abstract=5139372. doi: 10.2139/ssrn.5139372.

[9] Y. Lin, K. Wang, W. Yi and S. Lian, "Deep Learning Based Wearable Assistive System for Visually Impaired People," 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), Seoul, Korea (South), 2019, pp. 2549-2557,doi: 10.1109. 29

[10] Lavanya, G., & Pande, S. D. (2023). Enhancing Real-time Object Detection with YOLO Algorithm. EAI Endorsed Transactions on Internet of Things, 10. https://doi.org/10.4108/eetiot.4541

[11] Lee, J., & Hwang, K. (2021). YOLO with adaptive frame control for real-time object detection applications. Multimedia Tools and Applications, 81(25), 36375–36396. https://doi.org/10.1007/s11042-021-11480-0

[12] Fang, W., Wang, L., & Ren, P. (2019). Tinier-YOLO: A Real-Time Object Detection Method for Constrained Environments. IEEE Access, 8, 1935–1944. https://doi.org/10.1109/access.2019.2961959

[13] Alahmadi, T. J., Rahman, A. U., Alkahtani, H. K., & Kholidy, H. (2023). Enhancing object detection for VIPs using YOLOV4_Resnet101 and Text-to-Speech conversion model. Multimodal Technologies and Interaction, 7(8), 77. https://doi.org/10.3390/mti7080077

[14] Murthy, J. S., Siddesh, G. M., Lai, W., Parameshachari, B. D., Patil, S. N., & Hemalatha, K. L. (2022). ObjectDetect: a Real-Time object detection framework for advanced driver assistant systems using YOLOV5. Wireless Communications and Mobile Computing, 2022, 1–10. https://doi.org/10.1155/2022/9444360