A Multispectral YOLOv8-Based System for Real-Time Object Detection and Distance Estimation in Blind Navigation

Ema Utami[1*], Erwin Syahrudin[2], Anggit Dwi Hartanto[3], Suwanto Raharjo[4]

[1,2,3] *Magister of Informatics, Faculty of Computer Science, Universitas Amikom Yogyakarta, Indonesia*

[4]*Informatics, Universitas AKPRIND Indonesia*

*corr-author: ema.u@amikom.ac.id

Abstract - Developing reliable real-time navigation systems for visually impaired individuals remains challenging, particularly in dynamic and low-light environments. This study proposes an integrated framework combining YOLOv8, OpenCV-based monocular distance estimation, and RGB–NIR multispectral imaging to enhance detection robustness and distance awareness. A dataset of 1,700 annotated images collected from diverse indoor and outdoor environments was used for training and evaluation using preprocessing techniques such as resizing, normalization, and data augmentation. System performance was evaluated using Precision, Recall, F1-Score, mean Average Precision (mAP), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Frames Per Second (FPS). Experimental results show that YOLOv8x achieved the best performance with an F1-Score of 0.91, mAP@50 of 0.74, MAE of 0.15 m, RMSE of 0.20 m, and a processing speed of 22 FPS. Multispectral RGB–NIR integration further improved low-light performance, increasing the F1-Score from 0.83 to 0.89 and reducing MAE from 0.28 m to 0.19 m with only a minor reduction in speed. These findings demonstrate that the proposed system provides an effective balance between accuracy and real-time performance for assistive navigation applications.

Keywords: YOLOv8; multispectral imaging; blind navigation; object detection; distance estimation.

I. INTRODUCTION

Assistive navigation systems for visually impaired individuals have attracted significant research attention due to their potential to improve independent mobility and situational awareness. Recent advances in computer vision and deep learning have enabled real-time object detection systems that can identify obstacles and environmental elements using monocular cameras [1-3]. Assistive navigation systems for visually impaired individuals have attracted significant research attention due to their potential to improve independent mobility and situational awareness. Recent advances in computer vision and deep learning have enabled real-time object detection systems that can identify obstacles and environmental elements using monocular cameras [4-6].

Despite these advancements, most existing assistive navigation systems primarily focus on object detection accuracy under normal lighting conditions, while distance awareness and robustness in low-light environments remain insufficiently addressed [7, 8]. Reliable distance estimation is a critical requirement for safe navigation, as visually impaired users rely not only on object presence but also on spatial proximity to make timely decisions [9, 10]. However, many studies either neglect distance estimation entirely or rely on additional depth sensors that increase system cost, complexity, and power consumption [11, 12].

Monocular vision–based distance estimation offers a cost-effective alternative and has been widely explored using geometric relationships between object size, camera parameters, and assumed physical dimensions [13-15].

Prior research on YOLOv3 and YOLOv4 tackled small-object detection and noisy environments but lacked focus. Nevertheless, the accuracy of monocular distance estimation is highly sensitive to detection stability and image quality, particularly under low-light conditions where RGB cameras suffer from noise, low contrast, and missed detections [16]. As a result, distance estimation errors tend to increase significantly in poorly illuminated environments, limiting the practical usability of monocular assistive systems [17].

Multispectral imaging, particularly the integration of Near-Infrared (NIR) and RGB modalities, has been shown to improve visual perception under challenging illumination by capturing complementary spectral information [18, 19]. Prior studies have demonstrated that NIR data can enhance object visibility and edge

definition in low-light scenarios [20]. However, most multispectral approaches focus on surveillance, remote sensing, or scene classification tasks, while their application to real-time assistive navigation systems remains limited [21]. Moreover, existing works often employ complex fusion architectures or computationally expensive models that are unsuitable for real-time deployment on assistive platforms, which limits their effectiveness in practical applications for navigation assistance in low-light environments [22].

Another important limitation in current research is the lack of integrated evaluation between object detection performance and distance estimation accuracy. Many studies report detection metrics such as precision or mean Average Precision (mAP) without analyzing how detection stability affects downstream distance estimation [23]. Conversely, works that emphasize distance estimation often rely on idealized detection assumptions or controlled environments, reducing their relevance to real-world navigation scenarios.

To address these gaps, this study proposes a real-time assistive navigation framework that integrates YOLOv8 with OpenCV-based monocular distance estimation and RGB–NIR multispectral imaging. Unlike prior approaches, this work emphasizes distance-aware navigation performance, particularly under low-light conditions, while maintaining real-time capability. Feature-level fusion is adopted to balance detection robustness and computational efficiency, and the system is evaluated using both detection and distance estimation metrics. By focusing on the interaction between multispectral detection stability and distance estimation reliability, this study contributes practical insights toward the development of affordable and effective assistive navigation systems for visually impaired users.

Different from existing assistive navigation systems based on YOLO and OpenCV, this study introduces several novel contributions. First, this work integrates RGB–NIR multispectral imaging with YOLOv8 at the feature-fusion level to enhance object detection stability under low-light conditions. Second, the proposed framework jointly evaluates object detection performance and monocular distance estimation accuracy, highlighting their interdependency in real-world navigation scenarios. Third, a comprehensive experimental analysis is conducted across multiple YOLOv8 variants (n, s, m, l, x) using a custom dataset collected in diverse indoor and outdoor environments. Finally, this study demonstrates that multispectral fusion significantly reduces distance estimation errors while maintaining real-time performance, providing a practical and cost-effective solution for visually impaired navigation. These contributions distinguish the proposed system from conventional YOLOv8 + OpenCV-based approaches and existing blind navigation systems.

II. METHOD

This study experimentally develops and evaluates an object detection and distance estimation system using YOLOv8 and OpenCV to assist visually impaired users with real-time object and distance information. The process includes data collection, preprocessing, model training, integration with OpenCV for enhanced detection and distance estimation, and real-time implementation. System performance is assessed through accuracy, error rate, and processing speed to ensure effectiveness in real-world conditions.

A. You Only Look Once (YOLO)

As shown in Fig. 1, YOLO divides an image into grids, where each cell predicts bounding boxes, confidence scores, and class probabilities. By combining these predictions in a single pass, YOLO efficiently detects multiple objects in real time.

B. Flowchart Explanation

This research follows a structured workflow to enhance object detection and distance estimation using YOLOv8. YOLOv8, a state-of-the-art deep learning model, simultaneously predicts bounding boxes, confidence scores, and class probabilities in a single forward pass.. Its real-time efficiency makes it ideal for assistive navigation systems for the visually impaired, where timely feedback is critical for safety. The scalable architecture also allows deployment across various devices, from embedded systems to smartphones and high-performance platforms.

1) Step 1 Data Collection and Preprocessing: The dataset includes common indoor and outdoor obstacles encountered by visually impaired users. Images are annotated, resized, normalized, and filtered to standardize input and improve clarity. These preprocessing steps ensure better model generalization and robustness in real-world applications.

2) Step 2 Model Configuration: YOLOv8 is enhanced with two key components:

- OpenCV, for preprocessing, edge detection, and stereo-based depth estimation [24].
- CAW (Coordinate Attention Weighting), which improves feature representation by embedding positional information into channel attention.
- The Coordinate Attention Weighting (CAW) module was incorporated to enhance spatial-channel feature

representation by encoding positional information as channel attention. Compared to conventional channel attention mechanisms, CAW preserves spatial context while maintaining computational efficiency, making it suitable for real-time applications, such as in autonomous driving or video surveillance systems where quick and accurate feature representation is crucial. Previous studies have demonstrated that CAW improves feature discrimination under complex backgrounds and low-light conditions [25].

3) Step 3 S × S Grid Division: The input image is divided into an S × S grid (e.g., 7×7 for S=7), where each cell detects objects within its region. This grid-based division enables YOLOv8 to localize multiple objects simultaneously [26].

4) Step 4 Bounding Boxes + Confidence: Each grid cell predicts multiple bounding boxes with a confidence score that represents both object presence probability (P_{obj}) and positional accuracy through Intersection over Union (*IoU*) as in (1) [24].

$$Confidence = P_{obj} \times IoU \quad (1)$$

5) Step 5 Class Probability Map: Each grid cell outputs class probabilities (e.g., chair, table) to identify object types. For example, 20 possible classes yield 20 probability values per cell [27].

6) Step 6 Final Detections: Bounding boxes, confidence scores, and class probabilities are merged using Non-Maximum Suppression (NMS) to select the most accurate bounding box for each object [28]. The box with the highest confidence and class probability is defined as the final detection [29].

C. Process Flow Stages

The process flow in Fig. 2 illustrates the development stages of an object detection and distance estimation system for the visually impaired. It begins with data acquisition from various environments, followed by preprocessing—labeling, resizing, normalization, and augmentation—to prepare the dataset. The processed data is then used to train the YOLOv8 model with optimized parameters to achieve high detection accuracy.

D. Data Collection

The YOLOv8 models were initialized using COCO-pretrained weights to leverage general object representations learned from large-scale datasets [30]. Model training and evaluation were conducted exclusively on a custom dataset consisting of 1,700 manually collected and annotated images, captured from real-world indoor and outdoor environments relevant to visually impaired navigation [31]. The dataset includes typical obstacles such as pedestrians, vehicles, doors, chairs, walls, and street infrastructure, reflecting realistic navigation scenarios. Images were acquired under diverse conditions, including crowded streets, enclosed indoor spaces, obstacle-rich areas, and varying illumination levels, to ensure robustness against dynamic environments and lighting variations.

Fig. 1 Detection process YOLO

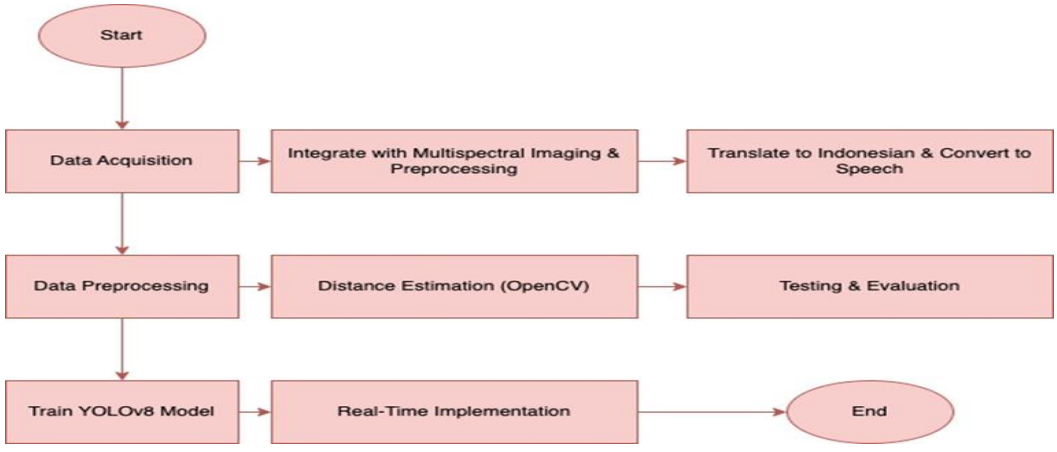

Fig. 2 Flow of stages of the research process

Although the dataset size consists of 1,700 images, which can be considered moderate for deep learning-based object detection, it was carefully collected from diverse real-world environments, including crowded streets, indoor spaces, obstacle-dense areas, and low-light conditions. This diversity helps improve data representativeness and reduces potential bias. Furthermore, data augmentation techniques such as rotation, flipping, and brightness adjustment were applied to increase data variability and mitigate the limitation of dataset size. Transfer learning using COCO-pretrained weights and early stopping were also employed to prevent overfitting and enhance model generalization.

Data collection, summarized in Table I, involves gathering images and videos from various real-world settings—both indoor and outdoor—to train and test the YOLOv8-based object detection and distance estimation models. These environments include crowded streets and complex indoor spaces, allowing the model to handle dynamic and static conditions under diverse lighting. This comprehensive dataset supports robust training for accurate detection and reliable distance estimation in practical navigation scenarios [32]. The dataset was divided into training, validation, and testing sets using a 70% / 20% / 10% split to ensure unbiased performance evaluation.

As summarized in Table I, the custom dataset covers diverse real-world environments encountered by visually impaired users, including crowded streets, indoor spaces, obstacle-dense areas, and low-light conditions [33]. Each environment presents distinct challenges, such as moving objects in outdoor scenes, reflections in enclosed spaces, and increased image noise under low illumination. This diversity enables the trained model to generalize effectively across realistic navigation scenarios [34].

TABLE I
TYPES OF DATA COLLECTED

No	Types of Environments	Number of Images
1	Crowded Streets	500
2	Closed Room	400
3	Areas with Obstacles	300
4	Low Light Conditions	200
5	Normal Light Conditions	300
Total	All Environments	1700

TABLE II
DATA PROCESSING METHODS

No	Preprocessing Stages	Description
1	Labeling	Each object in the image will be labeled.
2	Resize	Resizes the image to 640x640 pixels.
3	Normalization	Normalize the image pixels to the range [0, 1].
4	Data Augmentation	Addition of variation to the data with augmentation techniques.

E. Data Pre-processing

Table II outlines the preprocessing steps applied before training. This phase is essential to ensure consistent, high-quality input data, enabling the model to focus on relevant features rather than noise. Preprocessing improves dataset uniformity, diversity, and robustness—supporting faster training convergence and stronger real-world generalization.

Table II preprocessing method in this study was carried out in several stages, including labeling, resizing, normalization, and augmentation [35].

Normalization was applied to standardize pixel values, ensuring better model learning, while augmentation introduced additional variations to the data in order to reduce the risk of overfitting. The augmentation techniques included rotation, flipping, and brightness adjustment, whereas normalization was performed by dividing pixel values by 255 to scale them within the range [0,1]. The expected result of this process was to improve the model's generalization capability and accelerate training convergence.

F. YOLOv8 Model Training

After completing the data preprocessing stage, the YOLOv8 model was trained using the prepared dataset consisting of normalized, resized, and augmented images to ensure input consistency and variability. The training process employed empirically selected hyperparameters aimed at balancing detection accuracy and computational efficiency. The main training parameters were as follows: the Adam optimizer, a learning rate of 0.01, a batch size of 32, and training for 10 epochs with early stopping to mitigate overfitting given the limited dataset size. Data collection, summarized in Table I, involves gathering images and videos from various real-world settings—both indoor and outdoor—to train and test the YOLOv8-based object detection and distance estimation models. These environments include crowded streets and complex indoor spaces, allowing the model to handle dynamic and static conditions under diverse lighting. This comprehensive dataset supports robust

training for accurate detection and reliable distance estimation in practical navigation scenarios [32]. Due to the limited dataset size, training was intentionally restricted to 10 epochs with early stopping to reduce overfitting, as extended training showed diminishing performance gains.

Model training was conducted on a workstation equipped with an Intel Core i7-14700K CPU, 32 GB RAM, and an NVIDIA RTX 3090 GPU with 24 GB VRAM, enabling efficient GPU acceleration for deep learning workloads. GPU parallelism supported batch processing and reduced training time, facilitating rapid experimentation and model comparison across different YOLOv8 variants.

The YOLOv8 training process is guided by a multi-component loss function that measures the discrepancy between predicted outputs and ground truth labels. This loss function combines bounding box regression loss, objectness confidence loss, and classification loss, each weighted by a corresponding coefficient to balance localization accuracy and detection confidence, as expressed in (2):

$$\text{Loss} = \lambda_{bbox} \cdot L_b + \lambda_{cls} \cdot L_c + \lambda_{obj} \cdot L_o \quad (2)$$

where λ_bbox λ_cls λ_obj, and is the coefficients that control the contribution of each loss component to the total loss. The setting of these values is important to achieve a good balance between location accuracy, classification, and detection confidence [36].

G. Integration with OpenCV for Distance Estimation

After training, the YOLOv8 model was integrated with OpenCV to perform monocular, geometry-based distance estimation in real time, as illustrated in Fig. 3. Object distance was estimated using the relationship between the detected object's apparent size in the image, the calibrated camera focal length, and an assumed real-world object dimension [37]. Prior to distance computation, intrinsic camera calibration and lens distortion correction were applied to improve geometric accuracy. Representative physical dimensions were assigned to each object class based on common real-world measurements. This approach provides approximate distance awareness suitable for assistive navigation, where relative proximity is more critical than exact metric depth. Ground truth distances were measured manually during data collection and used to compute Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) for performance evaluation.

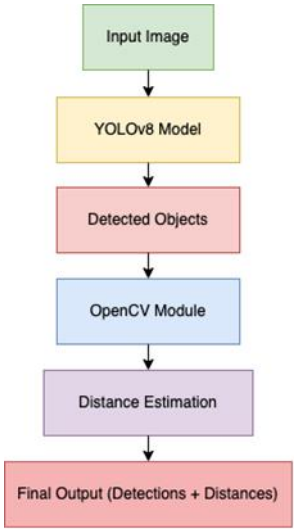

Fig. 3 YOLOv8 integration with OpenCV for distance estimation

The camera captures live images that are processed by YOLOv8 for object detection. Detected bounding boxes are then used by OpenCV to estimate object distance using a monocular geometric model based on camera calibration and assumed object dimensions.

The distance estimation process follows these steps: the camera captures an image, YOLOv8 detects objects and determines bounding boxes, and OpenCV computes object distance using (3) [38] :

$$\text{Distance} = \frac{Real\ Height \times Focal\ Lenght}{Object\ Height\ in\ Image} \quad (3)$$

where $Real\ Height$ denotes the assumed real-world object height, $Focal\ Lenght$ is the calibrated camera focal length, and $Object\ Height\ in\ Image$ represents the object height in the image plane.

It should be noted that the proposed monocular distance estimation method relies on assumed average physical dimensions for each object class. In real-world scenarios, object sizes may vary significantly due to individual differences, camera perspective, and partial occlusion. Consequently, inaccuracies in bounding box detection or deviations from assumed object dimensions may lead to distance estimation errors. This limitation is inherent to monocular geometry-based approaches and represents a trade-off between system simplicity and measurement precision. Future endeavors will rectify this limitation by integrating adaptive size estimation or depth-sensitive sensors.

H. Coordinate Attention Weighting (CAW)

Coordinate Attention Weighting (CAW) is applied to enhance feature representation in YOLOv8 by

embedding spatial coordinate information into channel attention. Given an input feature map as in (4).

$$X \epsilon R^{C \times H \times W}, \quad (4)$$

CAW first performs directional global pooling along horizontal and vertical directions to preserve positional information. The pooled features are computed as in (5).

$$Z_h(c,x) = \frac{1}{W}\sum_{y=1}^{W} X(c,x,y), \quad (5)$$

$$Z_h(c,y) = \frac{1}{H}\sum_{x=1}^{H} X(c,x,y) \quad (6)$$

The horizontal and vertical features are then concatenated and transformed using a 1×1 convolution followed by ReLU activation as in (7).

$$\mathrm{f} = \delta(\mathrm{Conv}_{1\times1}([\mathrm{z}_h, \mathrm{z}_w])), \quad (7)$$

where $\delta(\cdot)$ denotes the ReLU function. The encoded feature is split and processed using sigmoid activation to generate attention weights as in (8) and (9).

$$\mathrm{g}_h = \sigma(\mathrm{Conv}_{1\times1}(\mathrm{f}_w)), \quad (8)$$

$$\mathrm{g}_w = \sigma(\mathrm{Conv}_{1\times1}(\mathrm{f}_h)), \quad (9)$$

with $\sigma(\cdot)$ representing the sigmoid function. Finally, the original feature map is reweighted using the obtained attention maps as in (10).

$$\mathrm{Y(c,x,y)} = \mathrm{X(c,x,y)} \times \mathrm{g}_h\mathrm{(c,x)} \times \mathrm{g}_w\mathrm{(c,y)}, \quad (10)$$

resulting in the output feature map *Y*.

By applying coordinate-aware attention, CAW enables the network to emphasize spatially relevant regions while maintaining positional consistency. In assistive navigation, accurate localization of nearby obstacles is critical. By incorporating CAW into YOLOv8, spatially important regions such as pedestrians, doors, and vehicles are emphasized, leading to more stable detections and improved distance estimation reliability.

I. Real-Time Implementation with Panic Button Feature

The system is designed for real-time use by visually impaired users, with an emergency button that activates object detection and distance estimation. When pressed, the system provides verbal information about detected objects, helping users respond quickly to their surroundings.

Fig. 4 shows a real-time system architecture designed to assist visually impaired users in navigation with the panic button feature. This diagram illustrates the from data acquisition to the delivery of information to users.

As shown in Fig. 5, the real-time navigation system assists visually impaired users through an integrated detection and feedback pipeline. The camera captures live input processed by the YOLOv8 model for object detection, while OpenCV calculates the distance to detected objects. A panic button feature enables emergency activation, and all detection outputs are converted into speech feedback, providing immediate auditory guidance. This setup enhances both safety and independence for users in critical environments.

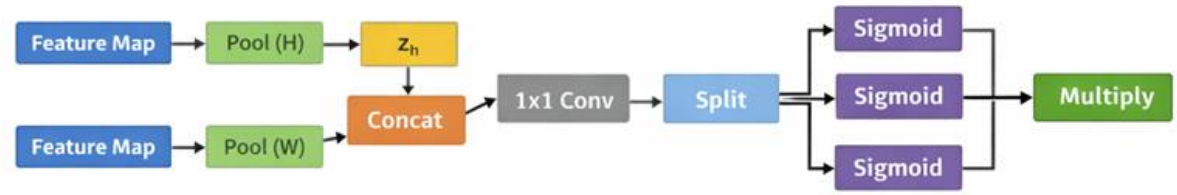

Fig. 4 Architecture of Coordinate Attention Weighting (CAW) module

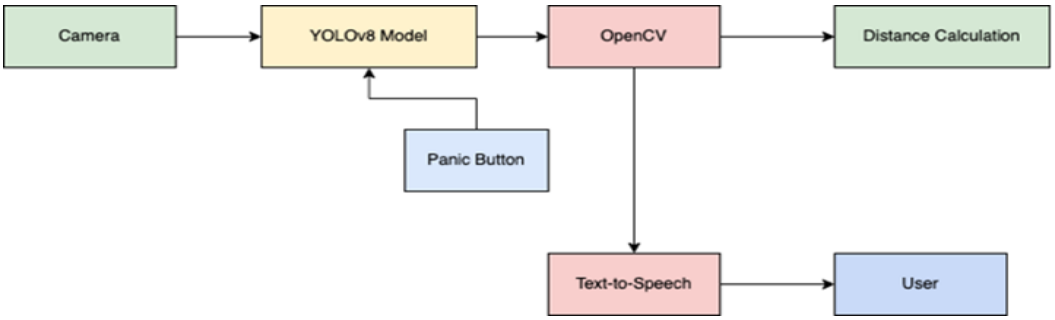

Fig. 5 Panic Button Stages for object detection

J. Multispectral Imaging Integration

In the workflow Fig. 5, this study integrates visible RGB imagery and Near Infrared (NIR) data to improve robustness of object detection under various lighting conditions, particularly in low-light environments. RGB images are captured using a monocular RGB camera, while NIR images are acquired using an infrared-sensitive camera operating at approximately 850 nm. Prior to fusion, RGB and NIR images are spatially aligned using feature-based image registration to ensure pixel-level correspondence [39]. Both modalities are normalized to the [0,1] range, with histogram normalization applied to the NIR channel to reduce luminance noise. Multispectral fusion is performed at the feature level within the YOLOv8 framework by concatenating RGB and NIR feature maps, followed by a 1×1 convolutional layer to enable effective spectral-spatial feature interaction with minimal computational overhead. This integration improves detection accuracy and distance estimation reliability in low-light conditions, with only a slight reduction in processing speed.

RGB and NIR images in Fig. 6 are normalized and spatially aligned through image registration before being fused at the feature level using channel concatenation and a 1×1 convolution layer within the YOLOv8 backbone. The fused features are used for object detection, while OpenCV performs distance estimation to provide real-time navigation feedback under normal and low-light conditions.

K. System Evaluation

System performance was tested under real-world conditions typical for visually impaired users. Evaluation covered object detection accuracy, distance estimation, processing speed, and usability. Accuracy was measured using precision, recall, and F1-score, while distance estimation used MAE and RMSE. FPS was used to assess real-time performance, and usability testing ensured outputs were easily understandable. Tests across various lighting and environments—such as low light, crowded streets, and obstacle-rich areas—validated the system's robustness, adaptability, and reliability [40].

Table III summarizes the evaluation parameters and their measurement methods.

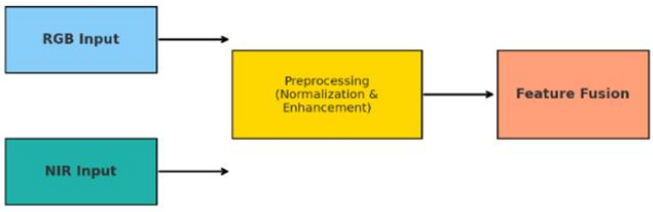

Fig. 6 Multispectral imaging integration flow

TABLE III
SYSTEM EVALUATION PARAMETERS

No	Parameter	Description	Measurement Method
1	Object Detection Accuracy	The percentage of objects that were detected correctly.	Precision, Recall, F1-Score
2	Distance Estimation Accuracy	Deviation between the estimated and actual distance.	Mean Absolute Error (MAE), RMSE
3	Processing Speed	The time required to detect and calculate distances.	FPS (Frames Per Second)

The proposed method outlines all key stages in developing a YOLOv8 and OpenCV-based system for object detection and distance estimation to assist visually impaired navigation. It includes data collection, preprocessing, model training with optimized parameters, and OpenCV integration for distance measurement. This framework ensures a balance between accuracy and efficiency across varied real-world conditions such as crowded, low-light, and obstacle-rich environments.

By applying augmentation and normalization, the model achieves strong generalization, while evaluation using precision, recall, F1-score, MAE, RMSE, and FPS ensures comprehensive performance assessment. Designed for real-time implementation, the system provides immediate audio feedback, emphasizing usability, reliability, and adaptability for visually impaired users.

III. RESULT AND DISCUSSION

This section discusses the experimental results of the proposed YOLOv8–OpenCV framework with multispectral integration, focusing on object detection accuracy, distance estimation performance, processing speed, and robustness under low-light conditions. The analysis emphasizes the relationship between detection stability and distance estimation reliability for assistive navigation.

A. Effect of Data Processing on Model Performance

Data preprocessing was designed to minimize input variability and enhance model generalization. All images were resized to 640×640 pixels and normalized to the [0,1] range, while augmentation techniques such as rotation, flipping, and brightness adjustment were applied [41].

Table IV shows that standardizing image resolution to 640×640 and normalizing pixel values to the [0,1] range effectively reduces scale and illumination inconsistencies. Original image sizes varied from

800×600 to 2560×1440 pixels, which could cause unstable feature extraction. After preprocessing, training convergence became more stable, and data augmentation further reduced overfitting, particularly given the limited dataset size of 1,700 images [42, 43].

B. Object Detection Accuracy

Object detection performance was evaluated using precision, recall, and F1-score to assess localization accuracy and classification reliability as in Table V [44].

Based on Table V, precision improves from 0.85 (YOLOv8n) to 0.92 (YOLOv8x), while recall increases from 0.81 to 0.90, resulting in an F1-score improvement from 0.83 to 0.91. This indicates that larger models achieve more reliable localization and classification, reducing both false positives and missed detections. YOLOv8m and YOLOv8l provide intermediate performance (F1 = 0.87 and 0.89), offering a balanced trade-off between accuracy and computational efficiency.

C. Distance Estimation Accuracy

Data in Table VI evaluates distance estimation using MAE and RMSE metrics. Lower values indicate higher accuracy. Results show that as model complexity increases, both errors decrease, meaning larger YOLOv8 versions provide more precise and reliable distance predictions. [45].

Table VI shows that MAE decreases from 0.36 m for YOLOv8n to 0.15 m for YOLOv8x, while RMSE decreases from 0.45 m to 0.20 m, corresponding to relative reductions of approximately 58% and 56%, respectively. This trend indicates that increasing model capacity improves bounding box localization stability, which directly reduces geometric measurement errors in monocular distance estimation. Larger models extract richer spatial features, resulting in more consistent distance predictions that are critical for safe assistive navigation as in (11) and (12).

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i| \quad (11)$$

$$RMSE = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2} \quad (12)$$

Lower MAE and RMSE values reflect more accurate and consistent distance predictions, essential for reliable navigation assistance.

D. Processing Speed

Processing speed was evaluated using frames per second (FPS) to assess real-time system performance. High FPS is essential for assistive navigation applications, as it ensures timely obstacle detection and smooth user feedback in dynamic environments.

As shown in Table VII, YOLOv8n achieves the highest speed at 45 FPS, followed by YOLOv8s and YOLOv8m with 38 FPS and 32 FPS, respectively, while YOLOv8x operates at 22 FPS. Although processing speed decreases with model complexity, all configurations satisfy real-time requirements. This demonstrates that the proposed system provides flexible deployment options, allowing users to balance responsiveness and accuracy based on hardware capabilities.

TABLE IV
SYSTEM EVALUATION PARAMETERS

No	Object	Initial Dimensions	Resizing Dimension	Maximum Value of Normalization
1	Chair	1920x1080	640x640	1.0
2	Door	1280x720	640x640	1.0
3	Car	2560x1440	640x640	1.0
4	People	800x600	640x640	1.0

TABLE V
OBJECT DETECTION ACCURACY FOR DIFFERENT VERSIONS OF YOLOV8

YOLOv8 Verses	Precision	Recall	F1-Score
YOLOv8n	0.85	0.81	0.83
YOLOv8s	0.87	0.83	0.85
YOLOv8m	0.89	0.86	0.87
YOLOv8l	0.91	0.88	0.89
YOLOv8x	0.92	0.90	0.91

TABLE VI
OBJECT DETECTION ACCURACY FOR DIFFERENT VERSIONS OF YOLOV8

YOLOv8 Model	MAE (meters)	RMSE (meters)
YOLOv8n + OpenCV	0.36	0.45
YOLOv8s + OpenCV	0.28	0.38
YOLOv8m + OpenCV	0.22	0.30
YOLOv8l + OpenCV	0.18	0.25
YOLOv8x + OpenCV	0.15	0.20

TABLE VII
PROCESSING SPEED (FPS) FOR DIFFERENT VERSIONS OF YOLOV8

YOLOv8 Models	FPS
YOLOv8n	45
YOLOv8s	38
YOLOv8m	32
YOLOv8l	28
YOLOv8x	22

E. Performance with Multispectral Imaging

The impact of RGB–NIR multispectral integration was evaluated using F1-score, MAE, RMSE, and FPS. The integration of Near-Infrared (NIR) information enhances feature visibility under low-light conditions and improves detection stability.

Fig. 7 shows that multispectral fusion increases the F1-score from 0.83 to 0.89, while MAE and RMSE decrease from 0.28 m to 0.19 m and from 0.38 m to 0.26 m, respectively, with FPS only slightly decreasing from 23 to 21. The comparative bar-chart further confirms that RGB–NIR integration significantly reduces distance estimation errors while preserving practical inference speed. These improvements indicate that NIR features enhance object boundary clarity and feature consistency, resulting in more stable detections and more reliable distance estimation under low-light conditions. Therefore, this study emphasizes F1-score, MAE, RMSE, and FPS as primary evaluation metrics, as they directly reflect distance-aware navigation performance, while comprehensive mAP-based benchmarking is reserved for future work.

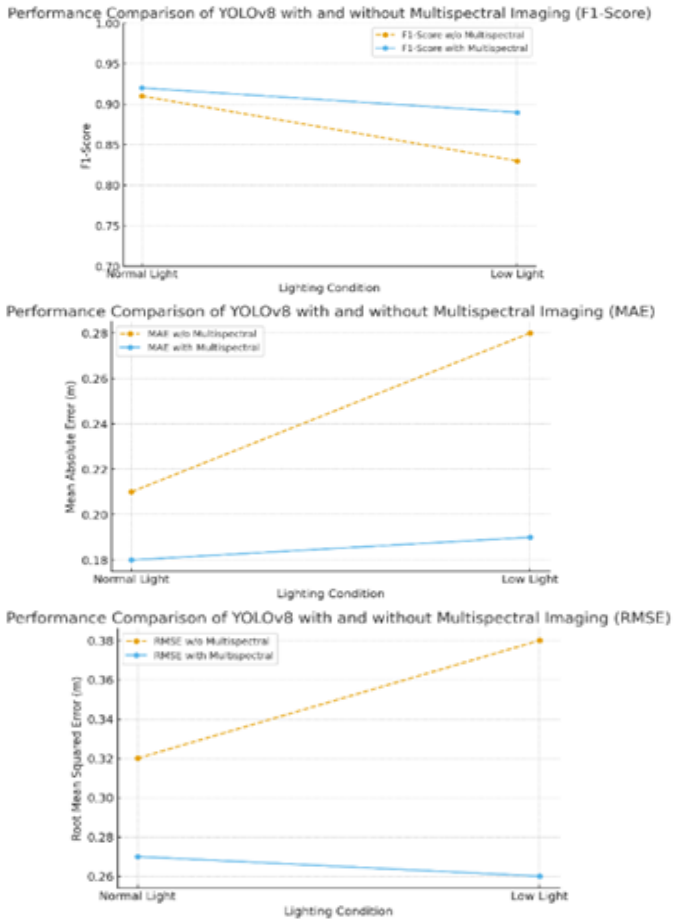

Fig. 7 Performance overall F1-score, MAE, and RMSE

Fig. 8 shows that integrating NIR data consistently improves distance estimation accuracy across all YOLOv8 variants. Both MAE and RMSE decrease with multispectral input, while FPS slightly drops. For example, YOLOv8x decreases from 23 FPS to 21 FPS, while achieving significant accuracy gains. The model ranking remains unchanged, with YOLOv8n being the fastest and YOLOv8x the most accurate. However, the performance gap between models becomes narrower, indicating that multispectral fusion enhances efficiency across different model scales.

F. Detection Performance Evaluation Using mAP and Precision–Recall Analysis

In addition to distance-based metrics, detection performance was evaluated using mean Average Precision (mAP) and Precision–Recall (PR) analysis to provide auxiliary validation of detection robustness.

The RGB–NIR model achieves an mAP@50 of 0.74 and mAP@50–95 of 0.46, outperforming the RGB-only model, which achieves 0.69 and 0.41, respectively. These results show that combining features from different wavelengths makes detection more reliable at different IoU thresholds, especially when the lighting is difficult.

Precision–Recall curves in Fig. 9 further show that the RGB–NIR model maintains higher recall at comparable precision levels, indicating reduced missed detections without sacrificing prediction confidence. These improvements are consistent with the observed reductions in MAE and RMSE, confirming that enhanced detection stability directly contributes to more reliable distance estimation for assistive navigation.

G. Ablation Study on CAW Module

An ablation study was conducted to evaluate the contribution of the Coordinate Attention Weighting (CAW) module by comparing baseline YOLOv8 models with and without CAW under identical experimental conditions.

The results show that incorporating CAW increases the average F1-score from 0.83 to 0.89, corresponding to an improvement of approximately 7.2%. Distance estimation accuracy also improves, with MAE decreasing from 0.28 m to 0.19 m, representing a reduction of about 32%. These gains indicate that CAW strengthens spatial–channel feature interactions and enhances feature representation in low-light environments. Importantly, these improvements are achieved without significant computational overhead, demonstrating that CAW plays a crucial role in improving system robustness for real-time assistive navigation.

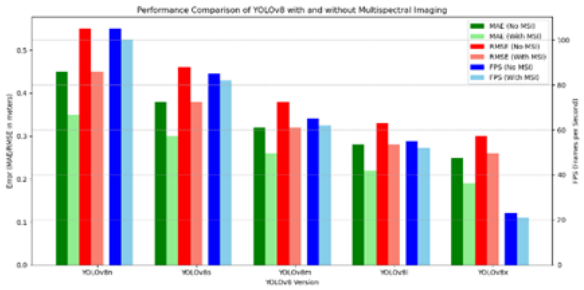

Fig. 8 Performance comparison overall MAE, RMSE, and FPS

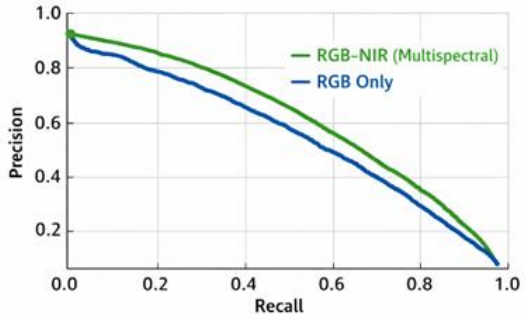

Fig. 9 precision–recall curve comparison between RGB and RGB–NIR models

IV. CONCLUSION

This study concludes that integrating YOLOv8 with OpenCV significantly enhances object detection and distance estimation, with performance varying by model size. The YOLOv8x variant achieved the highest accuracy and lowest distance errors (MAE and RMSE), while lighter models such as YOLOv8n offered faster processing suitable for real-time operation. The integration of multispectral (RGB–NIR) imaging further improved system robustness, particularly under low-light conditions, where the F1-Score increased from 0.83 to 0.89, MAE decreased from 0.28 m to 0.19 m, and RMSE from 0.38 m to 0.26 m. Although the processing speed slightly decreased from 23 FPS to 21 FPS, the system maintained real-time capability, demonstrating a favorable balance between accuracy and efficiency for assistive navigation applications. Despite these promising results, this study is subject to several inherent limitations. First, the dataset size of 1,700 images, although collected from diverse real-world environments, remains relatively limited for deep learning-based object detection. While data augmentation, transfer learning using COCO-pretrained weights, and early stopping were employed to mitigate overfitting and enhance generalization, a larger and more diverse dataset is expected to further improve model robustness. Second, the experimental evaluation was primarily conducted under controlled and semi-realistic conditions, and broader validation across more complex and dynamic real-world scenarios is still required. In addition, distance estimation relies on assumed object dimensions and monocular geometry, which may introduce inaccuracies under extreme perspective variations or occlusion. Future work will therefore focus on expanding the dataset scale and diversity, conducting large-scale field testing with visually impaired users, incorporating additional sensors such as depth cameras or LiDAR, and applying advanced model optimization techniques—including compression, quantization, and pruning—to enable efficient deployment on mobile and embedded platforms for real-world assistive navigation.

ACKNOWLEDGEMENT

This research was funded by the Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi (Kemendikbudristek) through the Hibah Penelitian Pascasarjana – Penelitian Tesis Magister for the 2024 fiscal year. The authors express their gratitude to the Ministry for its support and contribution to advancing research and innovation.

REFERENCES

[1] H. Wang, C. Liu, Y. Cai, L. Chen, and Y. Li, "YOLOv8-QSD: An Improved Small Object Detection Algorithm for Autonomous Vehicles Based on YOLOv8," *IEEE Trans. Instrum. Meas.*, vol. 73, pp. 1–16, 2024, doi: 10.1109/TIM.2024.3379090.

[2] M. Swathi Y. and Challa, "YOLOv8: Advancements and Innovations in Object Detection," in *Smart Trends in Computing and Communications*, C. and J. A. Senjyu Tomonobu and So–In, Ed., Singapore: Springer Nature Singapore, 2024, pp. 1–13.

[3] A. S. Romadhon and A. K. Husein, "Smart Stick for the Blind Using Arduino," in *Journal of Physics: Conference Series*, Institute of Physics Publishing, Jul. 2020. doi: 10.1088/1742-6596/1569/3/032088.

[4] A. L. R. Siriani, V. Kodaira, S. A. Mehdizadeh, I. de Alencar Nääs, D. J. de Moura, and D. F. Pereira, "Detection and tracking of chickens in low-light images using YOLO network and Kalman filter," *Neural Comput. Appl.*, vol. 34, no. 24, pp. 21987–21997, 2022, doi: 10.1007/s00521-022-07664-w.

[5] M. Safaldin, N. Zaghden, and M. Mejdoub, "An Improved YOLOv8 to Detect Moving Objects," *IEEE Access*, vol. 12, pp. 59782–59806, 2024, doi: 10.1109/ACCESS.2024.3393835.

[6] H. V Chakri Shadakshri, V. M. B, and K. V Rudra Gana Dev, "OpenCV Implementation of Grid-based Vertical Safe Landing for UAV using YOLOv5," *IJACSA) International Journal of Advanced Computer Science and Applications*, vol. 13, no. 9, 2022, doi: 10.14569/IJACSA.2022.0130957.

[7] F. Wahab, I. Ullah, A. Shah, R. A. Khan, A. Choi, and M. S. Anwar, "Design and implementation of real-time object detection system based on single-shoot detector and OpenCV," *Front. Psychol.*, vol. 13, Nov. 2022, doi: 10.3389/fpsyg.2022.1039645.

[8] Y. Li, Y. Huang, and Q. Tao, "Improving real-time object detection in Internet-of-Things smart city traffic with YOLOv8-DSAF method," *Sci. Rep.*, vol. 14, no. 1, p. 17235, 2024, doi: 10.1038/s41598-024-68115-1.

[9] K. Yu, G. Tang, W. Chen, S. Hu, Y. Li, and H. Gong, "MobileNet-YOLO v5s: An Improved Lightweight Method for Real-Time Detection of Sugarcane Stem Nodes in Complex Natural Environments," *IEEE Access*, vol. 11, pp. 104070–104083, 2023, doi: 10.1109/ACCESS.2023.3317951.

[10] M. A. Rahman, S. Siddika, M. A. Al-Baky, and M. J. Mia, "An automated navigation system for blind people," *Bulletin of Electrical Engineering and Informatics*, vol. 11, no. 1, pp. 201–212, Feb. 2022, doi: 10.11591/eei.v11i1.3452.

[11] E. Syahrudin, E. Utami, and A. D. Hartanto, "Enhanced Yolov8 with OpenCV for Blind-Friendly Object Detection and Distance Estimation," *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 8, no. 2, pp. 199–207, Mar. 2024, doi: 10.29207/resti.v8i2.5529.

[12] S. Khan, S. Nazir, and H. U. Khan, "Analysis of Navigation Assistants for Blind and Visually Impaired People: A Systematic Review," *IEEE Access*, vol. 9, pp. 26712–26734, 2021, doi: 10.1109/ACCESS.2021.3052415.

[13] P. Pawaskar, D. Chougule, and A. Mali, "Smart Cane for Blind Person Assisted with Android Application and Save Our Souls Transmission," *International Journal of Engineering and Management Research*, vol. 8, Mar. 2018, doi: 10.31033/ijemr.8.3.31.

[14] D. S. Bacea and F. Oniga, "Single stage architecture for improved accuracy real-time object detection on mobile devices," *Image Vis. Comput.*, vol. 130, Feb. 2023, doi: 10.1016/j.imavis.2022.104613.

[15] V. D. Nguyen Ngoc Dung and Thang, "Real-Time Object Detection Based on Yolov8," in *Smart Objects and Technologies for Social Good*, C. and D.-T. N. Huynh Hiep Xuan and Pham, Ed., Cham: Springer Nature Switzerland, 2025, pp. 150–160.

[16] T. J. Alahmadi, A. U. Rahman, H. K. Alkahtani, and H. Kholidy, "Enhancing Object Detection for VIPs Using YOLOv4_Resnet101 and Text-to-Speech Conversion Model," *Multimodal Technologies and Interaction*, vol. 7, no. 8, Aug. 2023, doi: 10.3390/mti7080077.

[17] E. Syahrudin, E. Utami, and A. D. Hartanto, "Augmentation for Accuracy Improvement of YOLOv8 in Blind Navigation System," *J. RESTI (Rekayasa Sist. Teknol. Inf.)*, vol. 10, no. 4, pp. 579–588, 2024, doi: 10.29207/resti.v8i4.5931.

[18] Y. Cao, Z. Liu, F. Wang, S. Su, Y. Sun, and W. Wang, "An improved YOLOv7 for the state identification of sliding chairs in railway turnout," *High-speed Railway*, Apr. 2024, doi: 10.1016/j.hspr.2024.04.002.

[19] E. Utami, E. Syahrudin, and A. D. Hartanto, "Comparative analysis of YOLOv8 techniques: OpenCV and coordinate attention weighting for distance perception in blind navigation systems," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 15, no. 3, p. 3267, Jun. 2025, doi: 10.11591/ijece.v15i3.pp3267-3278.

[20] E. Syahrudin, E. Utami, and A. D. Hartanto, "YOLOv8-Based Distance Estimation for Blind Navigation: Performance Comparison of OpenCV and Coordinate Attention Techniques," *CommIT Journal*, vol. 19, no. 1, pp. 45–57, Apr. 2025, doi: https://doi.org/10.21512/commit.v19i1.11820.

[21] D. Rocha, L. Pinto, J. Machado, F. Soares, and V. Carvalho, "Using Object Detection Technology to Identify Defects in Clothing for Blind People," *Sensors*, vol. 23, no. 9, May 2023, doi: 10.3390/s23094381.

[22] D. Huang, Y. Tu, Z. Zhang, and Z. Ye, "A Lightweight Vehicle Detection Method Fusing GSConv and Coordinate Attention Mechanism," *Sensors*, vol. 24, no. 8, Apr. 2024, doi: 10.3390/s24082394.

[23] E. Syahrudin, E. Utami, A. D. Hartanto, and S. Raharjo, "Evaluating YOLOv8-Based Distance Estimation: A Comparison of OpenCV and Coordinate Attention Weighting in Blind Navigation Systems," *INTENSIF*, vol. 9, no. 2, pp. 2549–6824, 2025, doi: 10.29407/intensif.v9i2.24395.

[24] S. Sun, B. Mo, J. Xu, D. Li, J. Zhao, and S. Han, "Multi-YOLOv8: An infrared moving small object detection model based on YOLOv8 for air vehicle," *Neurocomputing*, vol. 588, Jul. 2024, doi: 10.1016/j.neucom.2024.127685.

[25] R. Wang, F. Liang, B. Wang, and X. Mou, "ODCA-YOLO: An Omni-Dynamic Convolution Coordinate Attention-Based YOLO for Wood Defect Detection," *Forests*, vol. 14, no. 9, Sep. 2023, doi: 10.3390/f14091885.

[26] R. Arifando, S. Eto, and C. Wada, "Improved YOLOv5-Based Lightweight Object Detection Algorithm for People with Visual Impairment to Detect Buses," *Applied Sciences (Switzerland)*, vol. 13, no. 9, May 2023, doi: 10.3390/app13095802.

[27] Y. Li, Q. Fan, H. Huang, Z. Han, and Q. Gu, "A Modified YOLOv8 Detection Network for UAV Aerial Image Recognition," *Drones*, vol. 7, no. 5, May 2023, doi: 10.3390/drones7050304.

[28] S. Wang, "Research towards Yolo-Series Algorithms: Comparison and Analysis of Object Detection Models for Real-Time UAV Applications," in *Journal of Physics: Conference Series*, IOP Publishing Ltd, Jun. 2021. doi: 10.1088/1742-6596/1948/1/012021.

[29] E. Casas, L. Ramos, C. Romero, and F. Rivas-Echeverría, "A comparative study of YOLOv5 and YOLOv8 for corrosion segmentation tasks in metal surfaces," *Array*, p. 100351, Jun. 2024, doi: 10.1016/j.array.2024.100351.

[30] Z. Cheng, C. Chen, Z. Zhao, P. Qian, X. Li, and X. Yang, "COCO-TEACH: A Contrastive Co-Teaching Network For Incremental 3D Object Detection," in *2023 IEEE International Conference on Image Processing (ICIP)*, 2023, pp. 1990–1994. doi: 10.1109/ICIP49359.2023.10222538.

[31] M. Talib, A. H. Y. Al-Noori, and J. Suad, "YOLOv8-CAB: Improved YOLOv8 for Real-time Object Detection," *Karbala International Journal of Modern Science*, vol. 10, no. 1, pp. 56–68, 2024, doi: 10.33640/2405-609X.3339.

[32] A. and B. A. K. and X. T. and G. Y. and S. Y.-Z. Chowdhury Pinaki Nath and Sain, "FS-COCO: Towards Understanding of Freehand Sketches of Common Objects in Context," in *Computer Vision – ECCV 2022*, G. and C. M. and F. G. M. and H. T. Avidan Shai and Brostow, Ed., Cham: Springer Nature Switzerland, 2022, pp. 253–270.

[33] Iqra and K. J. Giri, "SO-YOLOv8: A novel deep learning-based approach for small object detection with YOLO beyond COCO," *Expert Syst. Appl.*, vol. 280, p. 127447, 2025, doi: https://doi.org/10.1016/j.eswa.2025.127447.

[34] P. Kumar and V. Kumar, "Exploring the Frontier of Object Detection: A Deep Dive into YOLOv8 and the COCO Dataset," in *2023 IEEE International Conference on Computer Vision and Machine Intelligence (CVMI)*, 2023, pp. 1–6. doi: 10.1109/CVMI59935.2023.10464837.

[35] A. Narmilan, F. Gonzalez, A. S. A. Salgadoe, U. W. L. M. Kumarasiri, H. A. S. Weerasinghe, and B. R. Kulasekara, "Predicting Canopy Chlorophyll Content in Sugarcane Crops Using Machine Learning Algorithms and Spectral Vegetation Indices Derived from UAV Multispectral Imagery," *Remote Sens. (Basel).*, vol. 14, no. 5, 2022, doi: 10.3390/rs14051140.

[36] S. Muhamad Itikap, M. Syahid Abdurrahman, E. B. Soewono, and T. Gelar, "Geometry and Color Transformation Data Augmentation for YOLOV8 in Beverage Waste Detection," *Journal of Software Engineering, Information and Communication Technology (SEICT)*, vol. 4, no. 2, pp. 123–138, 2023, doi: 10.17509/seict.

[37] A. B. Abadi and S. Tahcfulloh, "Digital Image Processing for Height Measurement Application Based on Python OpenCV and Regression Analysis," *International Journal on Informatics Visualization*, pp. 763–769, Dec. 2022, doi: 10.30630/joiv.6.4.1013.

[38] J. Sigut, M. Castro, R. Arnay, and M. Sigut, "OpenCV Basics: A Mobile Application to Support the Teaching of Computer Vision Concepts," *IEEE Transactions on Education*, vol. 63, no. 4, pp. 328–335, 2020, doi: 10.1109/TE.2020.2993013.

[39] P. P. Ruwanpathirana *et al.*, "Evaluation of Sugarcane Crop Growth Monitoring Using Vegetation Indices Derived from RGB-Based UAV Images and Machine Learning Models," *Agronomy*, vol. 14, no. 9, 2024, doi: 10.3390/agronomy14092059.

[40] M. Hussain, "YOLO-v1 to YOLO-v8, the Rise of YOLO and Its Complementary Nature toward Digital Manufacturing and Industrial Defect Detection," Jul. 01, 2023, *Multidisciplinary Digital Publishing Institute (MDPI)*. doi: 10.3390/machines11070677.

[41] W. Zhao, M. Syafrudin, and N. L. Fitriyani, "CRAS-YOLO: A Novel Multi-Category Vessel Detection and Classification Model Based on YOLOv5s Algorithm," *IEEE Access*, vol. 11, pp. 11463–11478, 2023, doi: 10.1109/ACCESS.2023.3241630.

[42] Y. Hui, J. Wang, and B. Li, "DSAA-YOLO: UAV remote sensing small target recognition algorithm for YOLOV7 based on dense residual super-resolution and anchor frame adaptive regression strategy," *Journal of King Saud University - Computer and Information Sciences*, vol. 36, no. 1, Jan. 2024, doi: 10.1016/j.jksuci.2023.101863.

[43] D. Kumar and N. Muhammad, "Object Detection in Adverse Weather for Autonomous Driving through Data Merging and YOLOv8," *Sensors (Basel).*, vol. 23, no. 20, Oct. 2023, doi: 10.3390/s23208471.

[44] Y. Dai, D. Kim, and K. Lee, "An Advanced Approach to Object Detection and Tracking in Robotics and Autonomous Vehicles Using YOLOv8 and LiDAR Data Fusion," *Electronics (Switzerland)*, vol. 13, no. 12, Jun. 2024, doi: 10.3390/electronics13122250.

[45] S. M. Robeson and C. J. Willmott, "Decomposition of the mean absolute error (MAE) into systematic and unsystematic components," *PLoS One*, vol. 18, no. 2 February, Feb. 2023, doi: 10.1371/journal.pone.0279774.