Biosystems Engineering 259 (2025) 104288

Contents lists available at ScienceDirect

Biosystems Engineering

journal homepage: www.elsevier.com/locate/issn/15375110

Research Paper

AI-driven real-time weed detection and robotic smart spraying for optimised performance and operational speed in vegetable production

Vinay Vijayakumar [a], Yiannis Ampatzidis [a,*], Christian Lacerda [a], Tom Burks [a], Won Suk Lee [a], John Schueller [b]

[a] Department of Agricultural and Biological Engineering, University of Florida, IFAS, 2685 SR, 29 North, Immokalee, FL, 34142, USA
[b] Department of Mechanical & Aerospace Engineering, University of Florida, Gainesville, FL, 32611, USA

ARTICLE INFO

Keywords:
Field robotics
Machine vision
Precision agriculture
Precision weed management
Robotic sprayer

ABSTRACT

For effective weed control in vegetable farms, enhancing precision spraying through improved real-time detection is crucial. Over the years, weed detection studies have evolved from traditional feature-based methods to deep learning approaches, particularly convolutional neural networks (CNNs). While numerous studies have focused on improving detection accuracy by experimenting with different backbones, architectures, and hyperparameter tuning, fewer have addressed the real-time implementation of these models in field conditions. Existing research primarily benchmarks model inference speed but often neglects the broader algorithmic efficiency, which includes sensor data integration, processing pipelines, and microcontroller output handling. Furthermore, real-world deployment challenges, such as camera performance at different robot speeds, the optimal operational range for high detection accuracy, and the end-to-end latency of the machine vision system, remain underexplored. This study addresses these gaps by training a custom YOLOv8 nano model to detect three weed types (broadleaf, nutsedge, and grass) and two crop types (pepper and tomato) in plasticulture beds. The system runs on a robotic smart sprayer in real time, integrating GPS and camera data while transmitting control signals to the microcontroller. Beyond detection performance, we evaluate the entire processing pipeline by measuring the total loop time and its variation with the number of detections per frame. Additionally, the optimal robot operational speed was determined, finding that 0.45–0.89 m s^{-1} provides the best balance between detection accuracy and system responsiveness. By focusing on end-to-end real-time performance on vegetable beds, this study provides insights into the practical deployment of smart spraying, often been overlooked in prior research.

1. Introduction

Preventing and controlling weed growth among crops through site-specific weed management using precision sprayers dates back many decades (Gerhards et al., 1997; Oriade et al., 1996; Brown & Noble, 2005). Precision sprayers that target weeds in agricultural fields have evolved over time, adopting techniques to improve targeted application while reducing crop injury. Researchers have employed spectral sensors to differentiate between weeds and crops (Shapira et al., 2013; Wang et al., 2019; Zhang et al., 2006). Such studies focus on using an optical sensor to detect the location, colour, texture, and shape of weeds and crops. Some of these spectral methods can extract information under controlled conditions, but their performance suffers under conditions with varying lighting and background noise. Additionally, while spectrometric sensors differentiate vegetation from soil, they struggle to differentiate between species (Wang et al., 2019). A spectral method that uses a limited band from a hyperspectral camera and combines it with techniques, such as wavelet transform and dimensionality reduction, can identify weeds from crops, achieving 85 % classification accuracy (Liu et al., 2019). However, processing hyperspectral data is computationally intensive and cannot be used for real-time applications.

Robotic precision weed sprayers in the past have used a combination of plant shape features, such as compactness, leaf elongation, and curvature of the leaf boundaries, with a Bayesian classifier to distinguish tomatoes from weeds (Lee et al., 1999). Additionally, to reduce the processing speed in real-time operation, plant and weed area differentiation was done based on leaf area. Although the performance in terms of speed was good, the challenges associated with the variability of the

* Corresponding author.
E-mail address: i.ampatzidis@ufl.edu (Y. Ampatzidis).

https://doi.org/10.1016/j.biosystemseng.2025.104288
Received 18 February 2025; Received in revised form 2 September 2025; Accepted 10 September 2025
Available online 16 September 2025

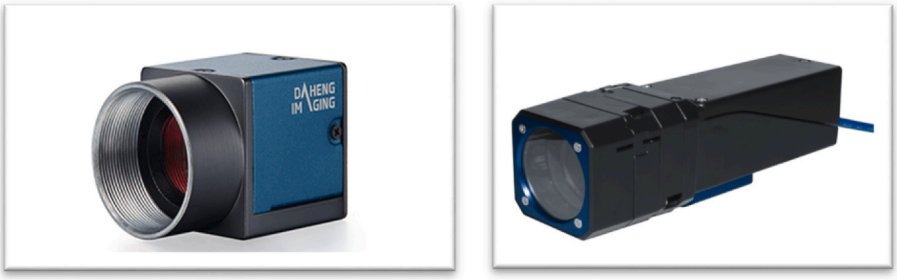


Fig. 1. a) Daheng GetCamera (without lens). b) IP67 housing for the cameras (Image source: https://www.get-cameras.com).

condition of plant leaves in an agricultural setting put a lot of limitations on the use of shape features for the detection of plants and weeds.

Researchers have also compared weed and crop discrimination techniques based on morphological characteristics, such as leaf shape factor, to techniques based on neural networks trained from input image pixels optimised using a genetic algorithm. Both methods gave less than 75 % recognition of carrot seedlings from two types of weeds (Aitkenhead et al., 2003). Researchers have also used a machine vision monochrome camera with image processing techniques such as a Gabon filter to identify the crop rows and a blob-colouring method for weed identification (Bossu et al., 2007). The system faced challenges due to variations in environmental conditions, which made it difficult to perform consistent weed identification in real-time conditions.

The use of convolutional neural networks (CNNs) for precision spraying applications has become popular in the past decade (Zhang & Fu, 2013; Vijayakumar et al., 2023; Wang et al., 2019). Partel et al. (2019) used YOLOv3 (You Only Look Once version 3) and Tiny YOLOv3 models on GTX 1070Ti GPU and on JetsonTX2, respectively, to identify and differentiate pepper plants and two weed types while spraying only on the weeds. It achieved an overall precision of 77 % with TX2 GPU, and 85 % with GTX 1070 Ti GPU evaluated over ten repeated trials with real plants. Modifications made to single stage detectors, such as YOLOv3, using feature maps at three different resolutions to improve predictions at different scales (large, medium, and small scales) have also been implemented in real-time weed and crop (potato) detection (Ruigrok et al., 2020). The model performed poorly (low recall and accuracy) in detecting potatoes under fluctuations in sunlight intensity, and on peaty soil with small sized weeds, where random objects were misidentified as potatoes. Three deep CNNs, AlexNet, VGG-16, and GoogleNet, were used for weed classification in strawberry plants (Liu et al., 2021). Based on the performance of real-time images captured in the field experiments, the VGG-16 model performed best. Dang et al. (2023) developed a large-scale dataset, CottonWeedDet12, comprising 5648 images of 12 weed species found in cotton fields. This allowed a broader evaluation of 25 YOLO-based detection models, yet challenges remained in detecting weeds under complex backgrounds and varied growth stages, highlighting the need for better generalisation. More recently, Diao et al. (2023) applied an improved YOLOv8s model to extract navigation lines in corn fields for autonomous spraying. While effective in real-time (mAP of 90.2 %), the system's performance declined under dense weed cover and variable lighting, indicating the need for improved robustness in real-world field deployment. Studies have also compared multiple CNNs to evaluate their performance in detection, weed classification, and efficiency (Ahmad et al., 2021). These networks were evaluated on F1 score, training and validation loss, and computational efficiency in terms of training time and resources. While the classification model achieved a high accuracy (98.9 %) and F1-score (99 %) in controlled settings, the object detection model had only moderate success (mAP of 54.3 %), largely due to the limited dataset size (452 images), which restricted its generalisability to real-world conditions. While many such studies have focused on improving detection models, they do not delve deep into the speed of the model and the algorithm to perform real-time detection. Addressing the challenges faced by such a system in real-time settings is crucial for developing an autonomous precision sprayer.

Understanding the limitations of a smart sprayer's machine vision system through real-time field evaluation is critical for assessing both detection accuracy and processing speed. Moreover, previous studies have not examined the structural design of the code that enables real-time sensor fusion and translates it into meaningful, actionable actuator commands. This study bridges these gaps by training a custom YOLOv8 nano model for detecting three weed types, broadleaf (*Portulaca oleracea* and *Richardia brasilensis)*, nutsedge (*Cyperus esculentus)*, and grass (*Panicum*), and two crop types, pepper (*Capsicum annuum*) and tomato (*Solanum lycopersicum*) in plasticulture beds. Unlike previous studies that evaluate models on curated datasets without considering real-world variations, the model's performance in this study was assessed under real-time conditions. A key contribution of this work is the evaluation of the entire processing pipeline, beyond just inference speed, across multiple trials. Additionally, the optimal robot operational speed was determined, finding a range of operational speeds that balances detection accuracy and processing speed per frame. Performance tests were conducted on plasticulture vegetable beds to demonstrate the effectiveness of this approach, highlighting both detection accuracy and the computational efficiency of the full implementation.

2. Materials and methods

2.1. Camera

2.1.1. Camera selection criteria

For a real-time detection system, it is essential to choose the right camera that can perform well for real-time detection tasks and operate in the intended environment. The selection criteria for the camera depend upon the targeted use, software restrictions, hardware compatibility, speed restrictions, environmental factors, machine vision implementation prowess, flexibility, and price. For this study, the camera must meet certain criteria (in terms of fps, resolution, and compatibility) and operate in an environment with high heat (>30 °C), humidity, and dust. The following criteria are used in this study to select the right camera:

a. ***Colour camera***: The camera will capture crops, weeds, and other details from vegetable beds. A red, green, and blue (RGB) camera is preferred to a monochrome camera to avoid issues with sedge and grass weeds with similar leaf structures, which would be difficult to distinguish in monochrome images (Meraj et al., 2023; Bossu et al., 2007).

a) b)

Fig. 2. Robotic smart sprayer platform with the camera mounted in front, facing downwards: a) back view and b) side view.

Table 1
Image sources and capture details of the images used for the dataset.

Image sources	Image resolution	Acquisition Date	Lighting condition	Number of images
Daheng GetCamera	1,440x1,080	2023 (Dec), 2024 (Jan, Feb, Jun, Aug, Sep)	Sunny (without and with shadows), overcast	1200
Canon EOS 5D	6,720x4,480	2021(April, May, Jun), 2022 (Apr, Oct), 2023 (Apr, Sep)	Sunny (without and with shadows), overcast	800
Microsoft modern webcam	640x360	April 2023	Sunny (evening time with shadows)	200

b. ***Horizontal Field of View (FOV)***: The FOV in the horizontal direction depends on the width of the vegetable beds. The selected camera should provide a good resolution when capturing a horizontal FOV of 0.96–1.02 m. This covers the vegetable beds (~0.84 m) and an additional small region (0.08 m) on either side of the beds.
c. ***Working Distance***: This working distance is calculated from the robotic smart sprayer, which has options for height adjustment of the camera's mounting point. The estimated height of the current setup is between 1.22 and 1.27 m.
d. ***Frames per second (fps) requirement***: A higher fps camera receives more frames, and the increased frequency helps reduce the latency between getting a new image for information and processing the image to send the information. It also reduces motion blur in real-time operations (Dinh et al., 2023; Inoue et al., 2017). A camera with a minimum of 30 fps is required.
e. ***Compatibility:*** A camera that is easily compatible with Linux systems and embedded systems, like the microcontrollers from NVIDIA, is preferred.
f. ***Environmental conditions:*** The camera must be protected from environmental conditions such as high heat, humidity, rain, and dust. Hence, a camera that has an option for a IP67 rated protective casing is required.
g. ***Low cost:*** The budget was set to be less than US$1000 to keep the whole system relatively low cost.

Based on these criteria, the 1.6-MP DAHENG GetCamera (Daheng Group, Inc., Beijing, China) with a 5 MP (F2.0, 6 mm) non-distortion lens is chosen. It uses a USB 3.0 cable for data transfer. An IP67-rated camera housing is used to protect the camera setup and mount it to the robotic smart sprayer. The camera and the housing are shown in Fig. 1. The camera also provides manual and automatic exposure control options, frame rate, gain, and white balance. Both these options are explored in real-time testing conditions, and the method most appropriate for the varying sunlight intensity is chosen. The 5 MP lens attached to the camera in this setup is rated as non-distortion (<0.01 %), and hence, no further distortion correction is necessary.

2.1.2. Camera performance under speeds

For a real-time application, such as detection and autonomous spraying in the field, the camera's performance is evaluated at different speeds to identify the number of continuous frames that would include a specific plant class (crops and weeds). This is crucial to ensure a high probability of the class being included in all the (continuous) images as the camera (mounted on a robot) moves over the field. Consequently, there would be a higher probability of the target class being detected in the region of interest (i.e., where the actuator, which in this case would be the nozzles, needs to be triggered).

The camera is mounted at a height of 1.27 m from the ground on a 1.78 m wide and 1.78 m long robotic sprayer platform. The platform (Fig. 2) is driven by four motorised wheels (UU motors; UU Motor Technology Co. Ltd, Changzhou, China) and is powered by a 48 V battery (OgrphyTech; Shenzhen, China). To evaluate the performance under varying speeds, the platform operates at three speeds, 0.45 m s^{-1}, 0.89 m s^{-1}, and 1.34 m s^{-1}, on a test bed with weeds. This is repeated twice for each speed. An RTK-GPS (Sparkfun GPS-RTK-SMA; SparkFun Electronics, Niwot, CO, USA) with a multi-band magnetic mount antenna is used to measure speed. The RTK-GPS provides high-accuracy location solutions (10 mm accuracy). The GPS's accuracy was evaluated by mounting it on a vehicle and comparing it with the speedometer readings. The vehicle was driven at speeds of 2.24 m s^{-1}, 4.47 m s^{-1}, 6.7 m s^{-1}, and 8.94 m s^{-1}, with GPS readings recorded in real time. The GPS readings closely matched the speedometer values, confirming its reliability as a suitable speed sensor for these experiments. With the GPS and antenna combination mounted on the robotic platform, frames are captured and saved in real time, and the GPS readings are noted for each run. Out of all the frames captured, a few continuous frames where the robot's speed is relatively constant are chosen for analysing the camera performance under varying speeds.

2.2. Data collection

Images for the custom object detection model for target weeds and crops are collected from three different sources: the Daheng Get Camera (MER2-160-227U3C), Canon EOS 5D, and Microsoft Modern camera. Images from multiple sources are included to add variety to the dataset and create a robust model that can work on images taken from multiple sources under varying conditions. The images are collected during Florida's growing season for tomato and pepper (November to January, March–May) for the years 2021–2024. The image capture conditions vary from morning time with sunny and clear skies, noon time with no

Table 2
Dataset distribution in terms of the number of images of each class and the total number of instances of the class.

Class name	Number of instances	Image per class
broadleaf	3252	646
sedge	701	369
grass	861	326
pepper	1140	415
tomato	913	640

shadows, and evening time with shadows and slightly overcast conditions. The specific capture time and environmental conditions are detailed in Table 1.

2.3. Dataset distribution

The distribution of the dataset for the five classes is presented in Table 2. The distribution accounts for the combined dataset acquired from all three cameras: Daheng GeTCamera, Canon EOS camera, and Microsoft Modern webcam, post the first step of data cleaning (section 2.3.1), resulting in a total of 1717 images. Fig. 3 presents the distribution through "image per class" and the "number of instances of each class" in terms of percentages. For each class, any image with at least one instance of the class is counted as an image that contributes to the "image per class" pie chart. The "number of instances of each class" counts the total instances of each class in every image of the dataset.

This dataset was used to train the object detection model (YOLO v8). The model's performance was subsequently evaluated.

2.3.1. Data cleaning

Data cleaning is an important part of pre-processing before training a model. The standard practice of data cleaning in object detection tasks includes removing noise, blurry images, and mislabels. For the current dataset, the first step of data cleaning involved removing 483 images owing to blurriness, large size of weeds with occlusions, and noise. The second step of data cleaning involved balancing out the need for greater generalisation with a more focused learning approach that enables the model to learn from images that are closer to the expected weed condition during the targeted spraying application. Data cleaning through the removal of unnecessary images improves dataset relevance and focus, and keeps the images that are closer to the detection goals. Out of the total dataset of images, specific images are removed due to multiple reasons:

a) Late-growth stage plants and weeds (Fig. 4a).
b) Difficulty in identifying the weed type on a soil background due to a combination of bad resolution and small-sized leaves of the weeds Fig. 4b.
c) Overgrown weeds in certain areas grow in very close proximity, causing heavy occlusion and difficulty labelling (Fig. 4c).

These factors are used as the basis for removing certain images from the dataset, as some of these cases are deemed non-critical for the intended application of this study. For example, occlusion due to overgrown sedge and broadleaf weeds might be seen in the later stages of plant growth. The intended use of a model trained on the current dataset is to be deployed on a robotic smart sprayer that would detect and kill

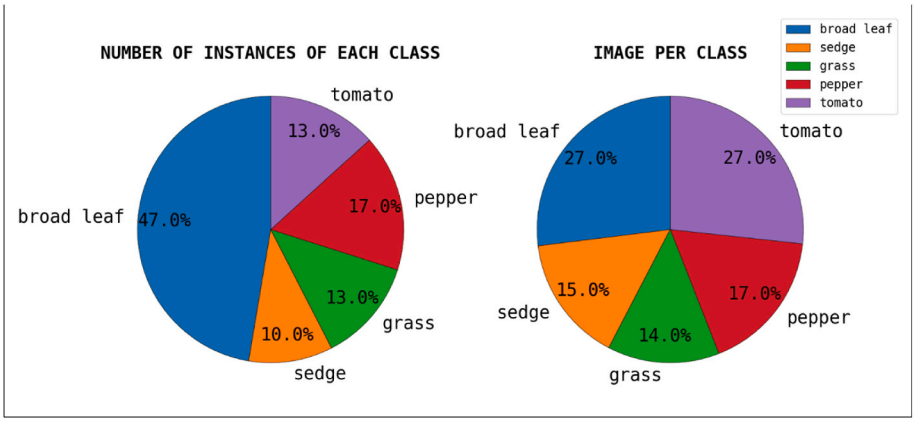


Fig. 3. Dataset distribution for images captured using all three cameras. The pie chart on the left shows the distribution of the classes in terms of the total number of instances, while the pie chart on the right shows the distribution of the classes in terms of the number of images with at least one instance of the class.

a) b) c)

Fig. 4. Examples of images removed from the dataset to aid focused learning and improve model performance. Reasons for images being removed from the dataset: a) Late-growth stage of the weeds, b) Issues with the soil background and small-sized leaves, and c) Occlusion.

Table 3
Grass dataset augmentation methods used.

Name	Augmentation type	Parameter values	Number of images
GA1	Temporal augmentation 1 + Gaussian noise	shift limit (x) = 0.05 shift limit (y) = 0.05; variance limit = (10, 50)	55
GA2	Temporal augmentation 2 + Gaussian noise	shift limit (x) = 0.10 shift limit (y) = 0.05; variance limit = (10, 50)	55
GA3	Horizontal flip + random brightness and contrast modification	brightness limit = 0.25, contrast limit = 0.25	55
GA4	Vertical flip + random brightness and contrast modification	brightness limit = 0.15, contrast limit = 0.15	55
GA5	Hue saturation value (HSV) shift	hue shift limit = 20, sat shift limit = 30, val shift limit = 20	55
GA6	Random size image crop with bounding box	height = 460, width = 460	55

weeds in the early to mid-stage of weed and plant growth. Moreover, extremely small-sized broadleaf weeds growing in the row-middle would not be targeted due to the target area being too small. Therefore, it is necessary to remove these images from the dataset even though they captured the target classes.

2.3.2. Data augmentation

Selective data augmentation was applied to images containing sedge and grass classes to address the class imbalance present in the dataset. Based on initial training results, the model performance is deemed to be very good (>98 % mean average precision (mAP)) on the plant classes. Hence, the images that predominantly comprise pepper and tomato plants are not chosen for augmentation. The broadleaf class is not augmented, as it is already overrepresented in the initial dataset, both in terms of the number of instances and the total number of images.

To prepare for class-specific data augmentation, all the images with at least one instance of grass are moved from the main dataset to a subdirectory. From these images, a subset of images chosen that either have only had grass or have more instances of grass as compared to other classes. Images with a few instances of grass among many instances of broadleaf are not used for augmentation to avoid increasing the number of images and instances of the broadleaf. The Albumentation library (Buslaev et al., 2020) is used for augmentation.

Six different augmentation methods are used for the grass subset. Table 3 details the augmentations with the parameters used and the number of images augmented. A combination of temporal shifts in x and y directions with Gaussian noise is included in the augmentation methods. This is because images produced due to temporal shifts are similar to the standard sequence of images the model would encounter in real-time implementation as the robot moves forward in the field. Horizontal and vertical flips, combined with random brightness and contrast enhancements, are also included. Additionally, a random HSV shift is applied to the images. The final augmentation applied is a bounding box specific augmentation from the Albumentations library (RandomSizedBBoxSafeCrop). This augmentation crops the image to a random size while preserving the bounding box information in the crop. A total of 330 images of grass are obtained from the augmentations. An example of all the augmentations applied to an image from the grass subset is shown in Fig. 5.

There was a higher number of better-quality images of sedge in the dataset (compared to grass images). Performing a higher number of augmentations of sedge might cause a class imbalance in favour of sedge, and it would necessitate further augmentation requirements for

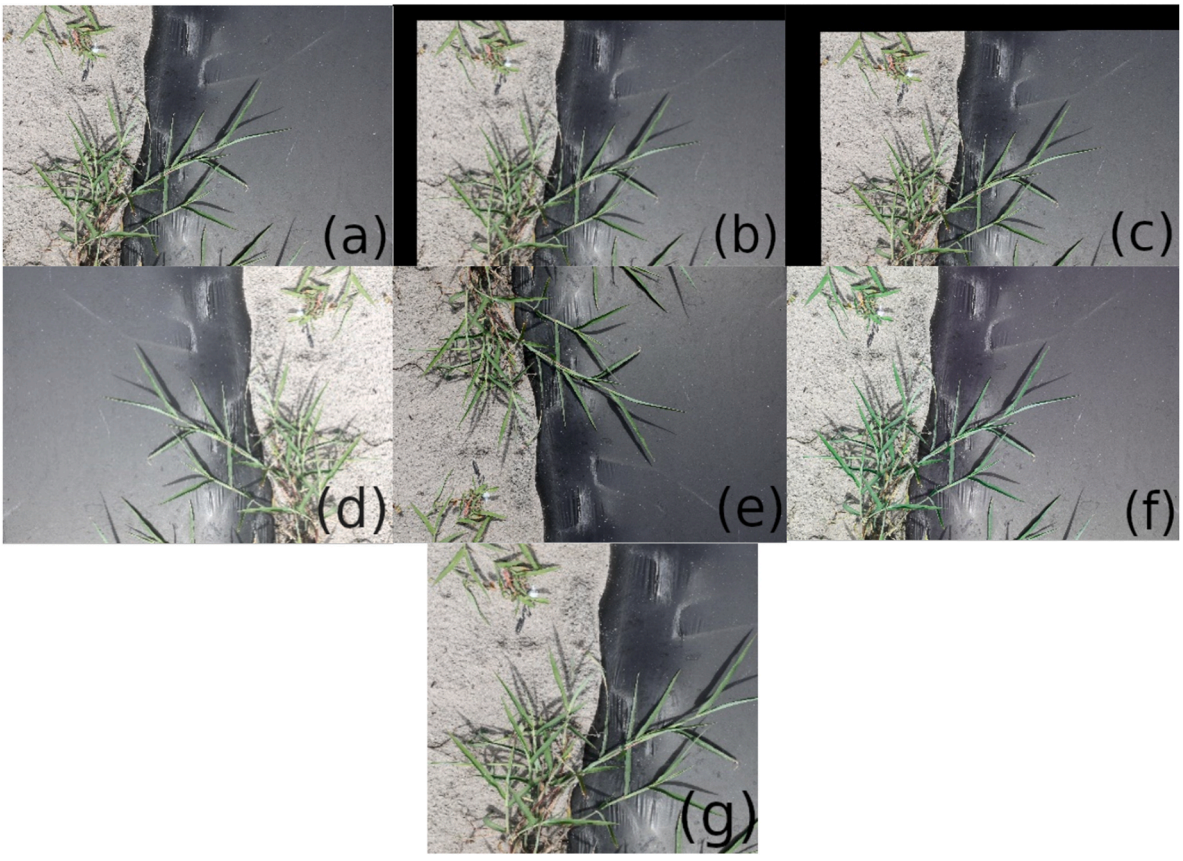


Fig. 5. An example of six augmentation methods applied to an image of grass. (a) Original image (b) GA1 (c) GA2 (d) GA3 (e) GA4 (f) GA5 (g) GA6 (details in Table 3).

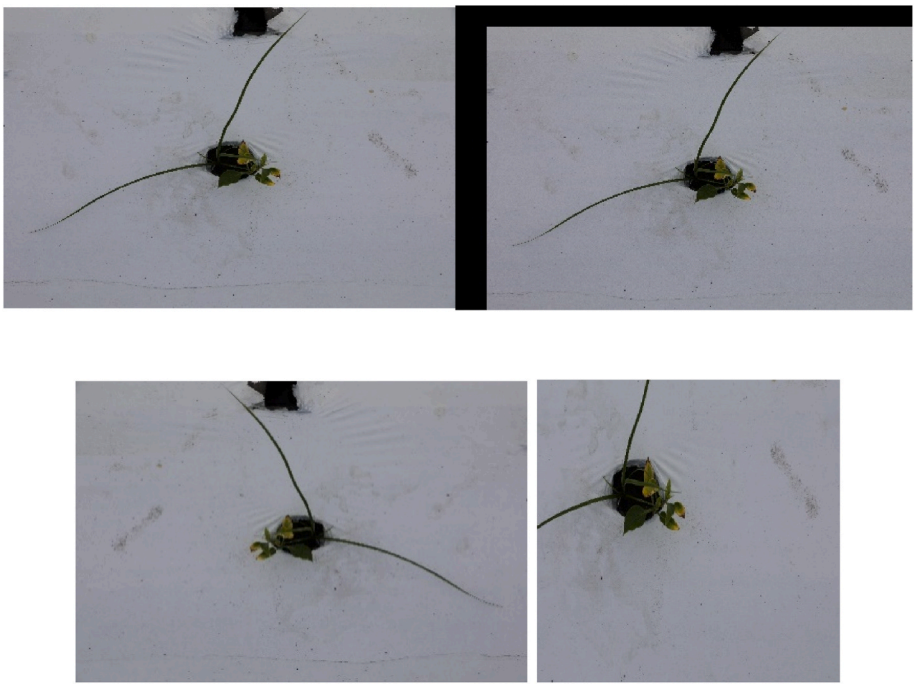


Fig. 6. An example image of sedge (top left) with its corresponding three augmentations (Clockwise from top right): SA1, SA2, SA3 (details in Table 4).

Table 4
The methods used for the dataset augmentation of the sedge class.

Name	Augmentation type	Parameter values	Number of images
SA1	Temporal augmentation 1 + Gaussian noise	shift limit= (0.05, 0.05); variance limit= (10, 50)	176
SA2	Horizontal flip + random brightness and contrast modification	Brightness limit = 0.25, contrast limit = 0.25	176
SA3	Random size image crop with bounding box	height = 460, width = 460	176

Table 5
Dataset distribution in terms of the number of images of each class and the total number of instances of the class.

Class name	Number of instances	Image per class
Broadleaf	3660	739
Sedge	1652	884
Grass	2316	633
Pepper	1379	551
tomato	1135	860

the grass class. To avoid such a scenario, only three augmentations (Fig. 6) are performed on the sedge dataset: temporal augmentation with Gaussian noise, horizontal flip with brightness and contrast modification, and random size image crop with bounding box preservation. Table 4 details the augmentations of the sedge dataset, with the parameters used and the number of images augmented.

The final dataset distribution (Table 5) post-augmentation is shown in Fig. 7. Post-augmentation, the image per class is much more balanced and has increased values for sedge and grass that were underrepresented in the original post-cleaned dataset. The percentage of broadleaf weeds is also less in terms of the number of images with at least one instance of broadleaf weeds and the total number of instances in the dataset.

2.3.3. Model training

The augmented dataset is used to train a YOLO v8 nano model. YOLO is a state-of-the-art, single-stage object detector widely used for real-time object detection tasks. YOLOv8, one of the latest and most advanced versions released in 2023, offers significant improvements in accuracy, speed, and flexibility over its predecessors (Liu et al., 2024; Sportelli et al., 2023; Li et al., 2024). Nano is the smallest version of YOLO v8, making it the most lightweight and fastest of all the existing YOLO v8 models. While newer object detection models like YOLOv10 and YOLOv11 have emerged in recent literature with incremental improvements in accuracy or speed (Karkee et al., 2024; Zhang et al., 2024), YOLOv8 remains a highly validated and robust model for real-time deployment, particularly in agricultural and resource-constrained environments. Previous studies have tested it on a range of weed datasets, including turfgrass (Sportelli et al., 2023), weeds in soybean fields (Li et al., 2023), weeds in cotton fields (Wang, Bochkovskiy, & Liao, 2023), and weeds in wheat fields (Qi & Wang, 2025), demonstrating consistently strong accuracy, low latency, and robustness to field variability. Beyond agriculture, YOLOv8 also performs competitively in real-time detection tasks in other domains, such as medical imaging (Zhou et al., 2024) and robotic inspection systems (Zhang et al., 2024). These results justify the use of YOLOv8 nano version for the real-time weed detection task in this study, where accuracy and low latency are equally important.

The customised YOLO v8 model is trained for 300 epochs with an input size of 640 × 640 pixels. The optimal epoch count (300) is determined through iterative training and early stopping with a patience of 100 epochs. Validation loss continued to improve past 100 epochs, and training converged around 300 without signs of overfitting, even in challenging cases. Standard metrics of precision, recall, F1 score, mAP50, and mAP50-95 are used as yardsticks for model evaluation. The performance of the model before and after augmentation is also compared. Post-training, the box loss and class loss curves were inspected to check for overfitting. The box loss is a type of localisation loss that measures the accuracy of the predicted bounding boxes as compared to the original annotated bounding boxes. The class loss is a classification loss that measures the performance of the model in detecting the right classes (Wang, Bochkovskiy, & Liao, 2023).

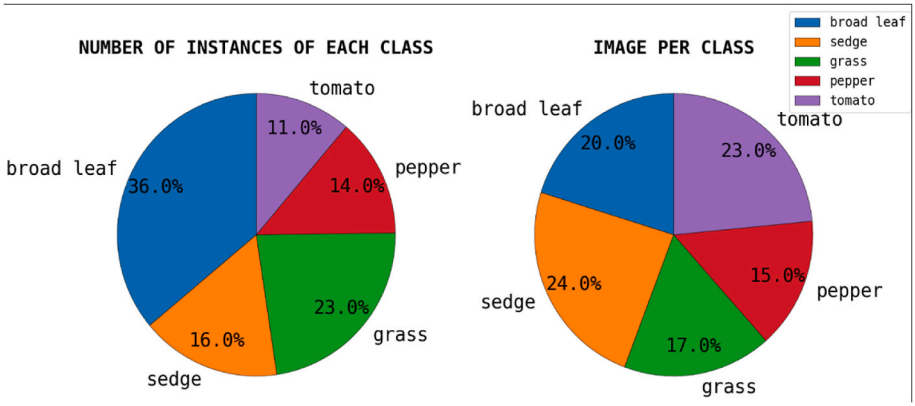


Fig. 7. Dataset distribution after augmentation of sedge and grass.

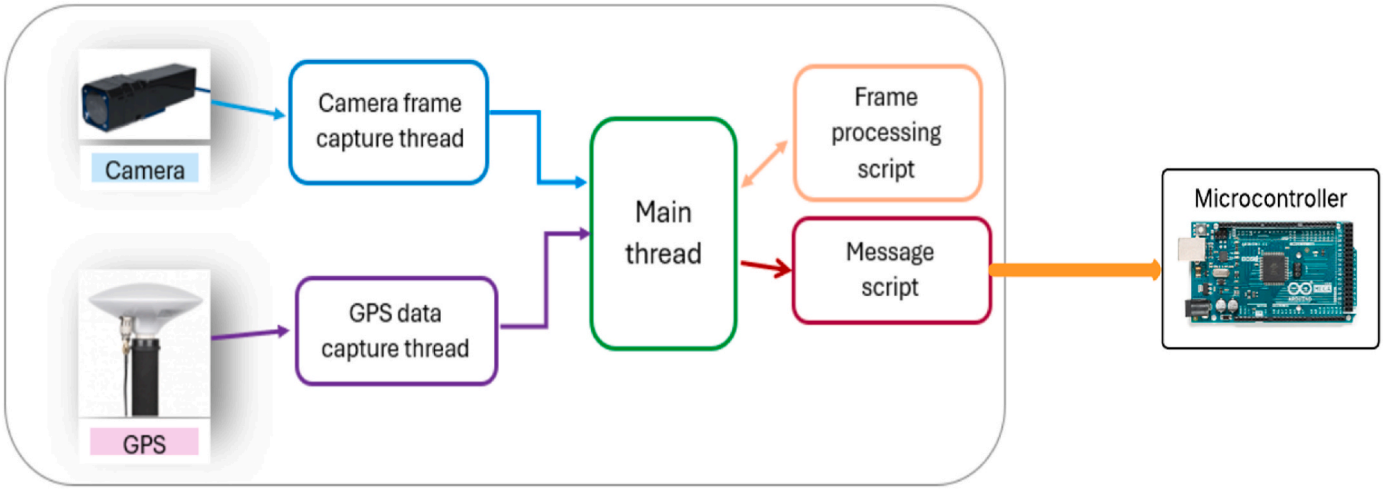


Fig. 8. Layout of the algorithm implemented to receive information from the sensors (camera and GNSS) and process it to get useful information to send to the actuators. Image source: Camera: https://www.get-cameras.com/; GPS: sparkfun.com; Arduino: https://store-usa.arduino.cc/.

Table 6
Number of frames that capture the target object (sedge) when the vehicle (robot) moves at three different speeds.

	Trial 1		Trial 2	
Set speed (m s^{-1})	Number of frames (N)	Average speed per frame (m s^{-1})	Number of frames (N)	Average speed per frame (m s^{-1})
0.45	14	0.47	12	0.45
0.89	7	0.89	7	0.88
1.34	4	1.35	4	1.33

In the initial stages of the project, datasets are used to train Faster R-CNN and YOLOv4, and their performance is evaluated based on detection accuracy and speed on a validation dataset. YOLOv4 outperforms Faster R-CNN and is initially selected for real-time implementation. However, as the dataset changes over time and newer versions of single stage detectors, such as YOLO are released, YOLO v8 is adopted to enhance efficiency and streamline detection. Prior to this selection, YOLOv8-small and YOLOv8-medium models were also evaluated. As the limitations of these models were already established, they were not re-evaluated on the current dataset to avoid unnecessary overhead. Additionally, given the abundance of existing literature comparing these models, repeating such baseline comparisons of multiple models is deemed redundant for the focus of this study.

2.3.4. Implementation algorithm

The trained model is implemented in a multi-threaded Python code to perform real-time detection using Ultralytics (Jocher et al., 2023). Ultralytics HUB is a user-friendly AI platform that enables users to build, train, and deploy machine learning models through a no-code interface, while also supporting popular deep learning frameworks. Images from the camera are captured continuously through a Python thread (camera frame capture thread in Fig. 8). The GNSS information is also captured in a separate thread. The main thread receives the information from each of these two threads when ready. The frame received from the camera frame capture thread is sent to the frame processing script to process the image and apply the trained detection model. Bounding box information obtained from the model's inference on the image provides useful information about the presence and location of weeds and plants in every frame. This information is then converted to information about the activation zones and returned to the main thread. Thus, the frame processing script receives and sends information to the main thread. The main thread, which controls all the threads and connected scripts, continuously sends this information to the message script responsible for communicating with the microcontroller through serial messages. The serial messages are sent as an array of 0's and 1's based on the information of the activation zones. The sensor threads (camera and GNSS) operate at a faster timescale than the main thread, allowing non-blocking data retrieval for processing and actuation. Despite frame processing being the most time-intensive task, the overall loop time remains within 40–45 ms without any synchronisation issues.

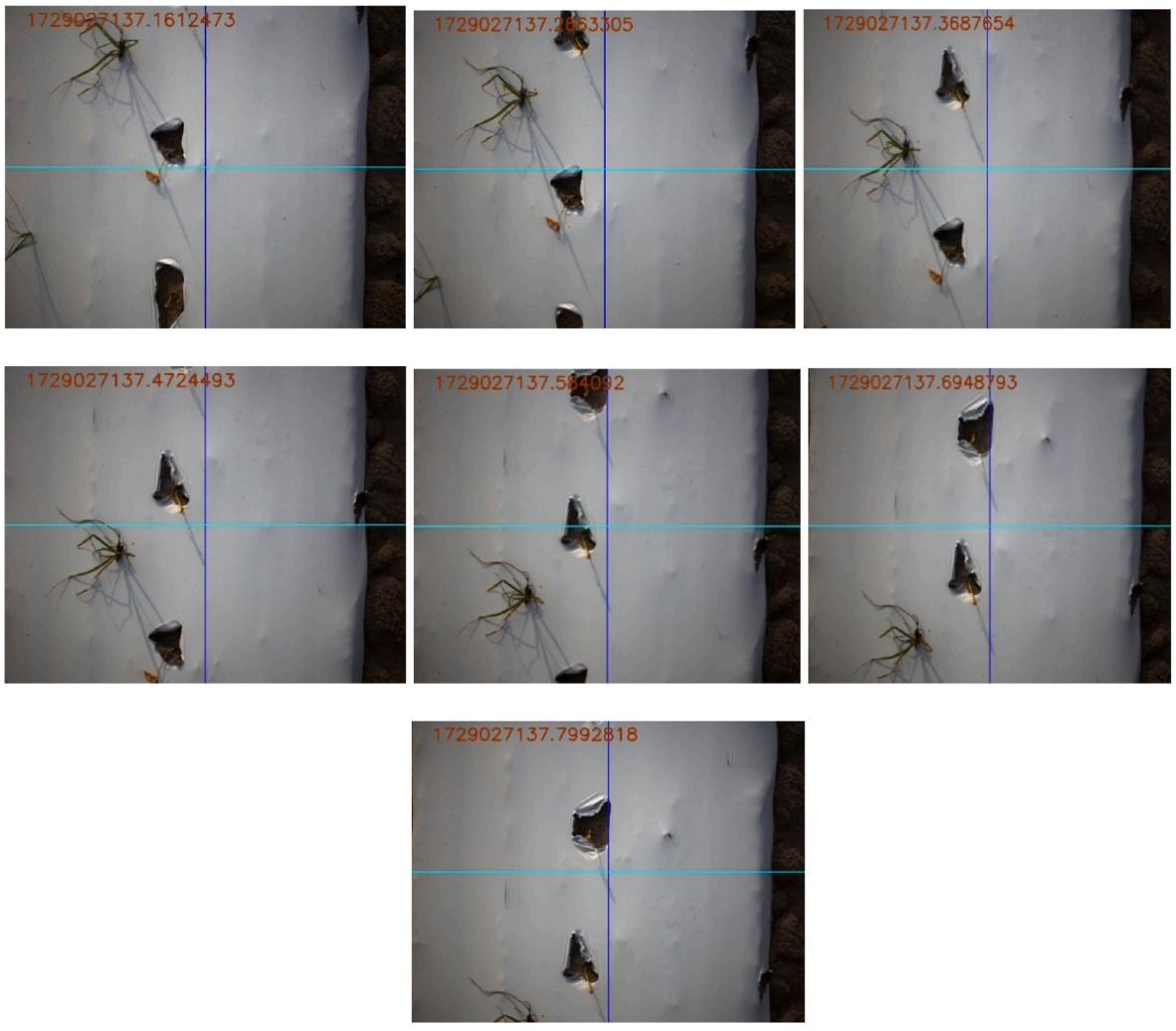


Fig. 9. Instances of a weed captured in consecutive image frames at a vehicle speed of a) 0.89 m s^{-1} (seven instances) and b) 1.34 m s^{-1} (four instances). Forward direction of travel for the robotic sprayer in both cases.

Exponential moving averages are applied to the GNSS data to mitigate any loss, and the system remains stable under varying lighting conditions. The layout of the implementation algorithm is shown in Fig. 8.

2.3.5. Real-time performance - speed

The trained model in Pytorch format is converted to a TensorRT FP16 format to speed up real-time processing. To evaluate the real-time performance of the code (with the trained model), the whole machine vision system is tested on a pepper and tomato bed with real-time input from the camera and the GNSS. The beds had pepper or tomato plants (planted per standard practice on raised beds) and weeds (sedge and broadleaf weeds). The machine vision Python code is run on a Dell Alienware 16 laptop with a Linux (Ubuntu 22.4) operating system. It includes a 4070 NVIDIA Geforce GTX graphics card with CUDA support.

Table 7
Performance of the model before and after class-specific augmentation evaluated on the test set.

Before Augmentation							
Class	Images	Instances	P	R	F1	mAP50	mAP50-95
all	344	1329	0.77	0.72	0.74	0.75	0.45
broad leaf	344	605	0.62	0.65	0.63	0.69	0.38
sedge	344	130	0.63	0.51	0.56	0.53	0.20
grass	344	163	0.70	0.47	0.56	0.56	0.27
pepper	344	255	0.97	0.99	0.98	0.99	0.74
tomato	344	176	0.90	0.97	0.93	0.98	0.67
After Augmentation							
Class	Images	Instances	P	R	F1	mAP50	mAP50-95
all	308	1322	0.88	0.83	0.86	0.86	0.61
broad leaf	308	703	0.75	0.65	0.70	0.69	0.39
sedge	308	85	0.88	0.80	0.84	0.80	0.56
grass	308	133	0.85	0.77	0.81	0.82	0.58
pepper	308	234	0.96	0.98	0.97	0.99	0.78
tomato	308	167	0.97	0.95	0.96	0.99	0.73

Using GPU processing capabilities improves the speed of model loading, model inference, and real-time performance. The time taken per frame for the main loop (Fig. 8) is analysed for images without detection and with detection in the region of interest. Each frame could have only plants, a combination of plants and weeds, or neither of the five classes. The model's performance in terms of the time taken for the entire main thread when tested in real-time on pepper and tomato beds is evaluated. Three trials are conducted on each bed. Each trial has the model run on 1500 frames for the tomato bed and 1700 frames for the pepper bed.

3. Results

3.1. Camera performance under three vehicle speeds

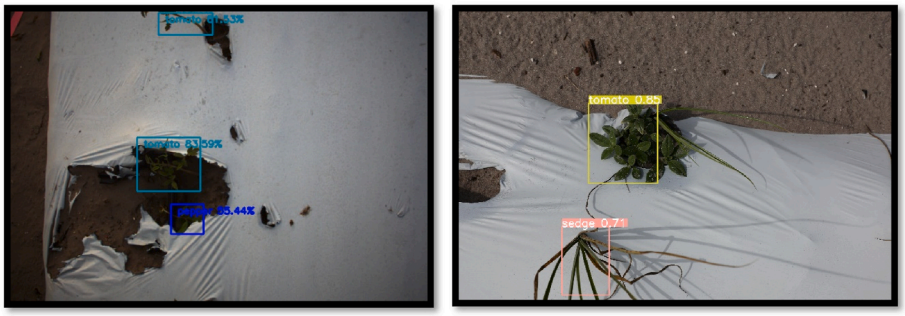
The number of consecutive frames (N) that include a specific object (in this case, the target weed sedge) was identified for all three vehicle (robot) speeds. For each speed trial, the average speed per frame is noted. It was calculated by averaging the speeds recorded for each of the 'N' frames. The value of N varied for each speed, as is demonstrated in Table 6.

These results show that the number of frames that can capture the presence of the object decreased with an increase in speed. With 14 frames, a speed of 0.45 m s^{-1} provided the best scenario for detection as it increased the probability for the object to not only be detected but also be located in the region of interest where the actuators (nozzles in this case) are triggered. This region of interest and its extent were choices that were part of the decision-making logic implemented in the code. Speeds greater than 0.89 m s^{-1} (and up to 1.12 m s^{-1}) could also work as long as the fps of the software was in an acceptable range. If the number of frames dropped below six, it may have led to missed actuations, as the object might not appear consistently within the region of interest across consecutive frames, especially at higher vehicle speeds. Examples of continuous frames captured at 0.89 m s^{-1} and 1.34 m s^{-1} are shown in Fig. 9.

3.2. Model performance: detection

The performance of the model on the test set before and after selective data augmentation is shown in Table 7. It is evident from the results in Table 7 that selective data augmentation on sedge and grass significantly improved the detection performance. The F1 score jumped from 56 % to 84 % for sedge and from 56 % to 81 % for grass. Similar improvements were also seen in mAP50, where the model's performance on sedge improved from 53 % to 80 %, and on grass from 56 % to 82 %. Even though there was no class-focused augmentation for broadleaf weeds, the model's performance improved from 63 % to 70 % for the F1 score. This was partly because of data cleaning and the presence of some instances of broadleaf weeds in some images of sedge and grass. The performance of the model on the plant classes improved marginally for tomato and was almost unchanged for pepper. However, the model still performed very well on both these classes post-augmentation. One of the main reasons for the performance being high for the plant classes was that the plant classes had the 'cleanest' images, i.e., images with little occlusion, very little variation in the background, and almost predictable positions of the classes within the image, i.e., along the centre of the bed for tomato plants and either along two rows or one row through the middle for pepper plants. The mAP50-95, which evaluates the mAP at multiple levels of bounding box overlap, also increased post-augmentation for all the classes. A few instances of false positives (FP) were observed (Fig. 10), primarily due to two factors: weeds being misclassified as crops because of a darker, wet soil background, and the morphological similarity between broadleaf weeds and late-stage tomato plants being wrongly perceived by the detection model. However, such misclassifications were rare during the tests conducted on the raised beds. They were more frequently observed in the row-middle tests, where different weed types were occasionally misclassified among one another. This type of error is less critical, as it does not pose a risk of crop injury and still results in a valid spray activation.

Despite the broadleaf class having the highest number of images and total instances in the dataset, it yielded the lowest detection performance among all classes. One of the main reasons for the low



a) b)

Fig. 10. Examples of model misclassifications (false positives) during testing: a) broadleaf weed identified as pepper (blue box), and b) broadleaf weed (yellow box) identified as tomato. (For interpretation of the references to colour in this figure legend, the reader is referred to the Web version of this article.)

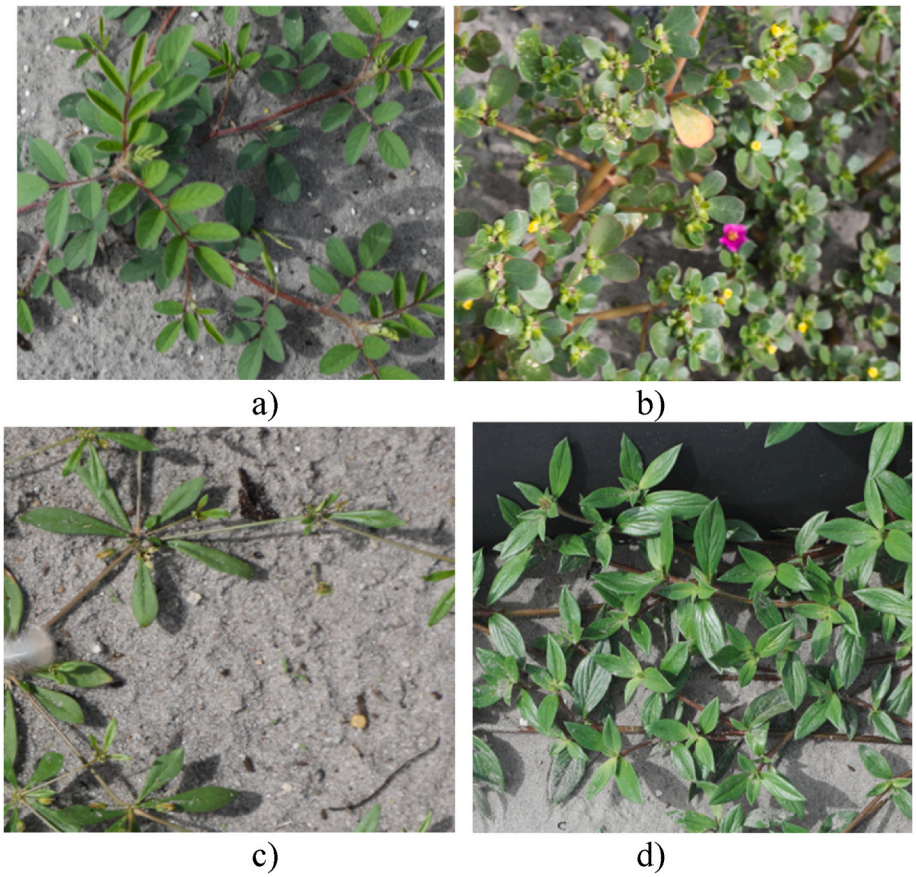


Fig. 11. Different varieties of the broadleaf class found in the dataset (common names presented): a) Spreading sandmat, b) Sea purslane, c) Green carpetweed, and d) Richardia. (For interpretation of the references to colour in this figure legend, the reader is referred to the Web version of this article.)

performance was the large variety of broadleaf weeds found in the field. Overall, there were at least four different broadleaf weed varieties (Fig. 11) in the dataset. Consequently, there weren't enough representations of all four varieties, and the model struggled to identify some. Unfortunately, these varieties grow randomly around the edges of the beds, and their presence in the dataset was inevitable. It was also difficult to eliminate a specific variety out of these and construct the dataset with only a few of them. While such a practice might result in a better performance of the model on the broadleaf class, it would be farther from the actual representation of the distribution of broadleaf weeds in the field.

Despite training the model on augmented images of sedge and grass, it did not reach a stage of overfitting during training. This can be confirmed by inspecting the loss curves of training and validation (Fig. 12). Both these curves show a decreasing trend as training progresses through the epochs.

3.3. Model performance – speed

The model's performance in terms of the time taken for the whole pipeline when tested in real-time on pepper and tomato beds is shown in Fig. 13a and b, respectively. The plots show the speed for three trials, each containing results for 1500 frames for the tomato bed and 1700 frames for the pepper bed. The tests on pepper bed had many instances of no detection as well as multiple detections in a frame (even up to nine detections in a frame). Fig. 13a shows that the loop times varied between 40 and 80 ms when there were detections in the frame. For zero detections (i.e., no classes of interest in the frame), the loop time varied between 15 and 40 ms. There was a jump in the average time taken per frame when there were detections in the frame compared to no detections. This was expected as the model had to extract useful features from the image and subsequently use the bounding box data for future tasks, such as nozzle actuation based on weed location. Table 8 presents the average time taken per loop for different numbers of detections for the trials on the pepper bed. The average value for zero detection in the frame was 28.5 ms (for three trials). The average value increased to ~55 ms for one or more detections. It is important to note that loop times did not increase with the increasing number of detections.

The tests on the tomato bed had no cases of zero detection in a frame for the three trials conducted on the bed. However, the zero detection values were similar to the pepper beds during other field trials. In cases with multiple detections per frame, loop times ranged from 35 ms to 85 ms when there were one or two detections per frame. The loop times generally stayed around 60 ms for frames with more detections. Table 9 presents the average loop time for different numbers of detections during the trials on the tomato bed. The average loop time for frames with one or more detections was approximately 60 ms. This was higher than the results on the pepper beds, as the trials on the tomato beds included additional print statements for debugging purposes. Without these debug outputs, the loop times are estimated to average closer to 55 ms for frames with one or more detections. Like the pepper bed trials, the loop times did not increase with more detections per frame.

It is important to note that in both trials, the output process included saving the processed images to a folder for debugging, along with multiple lines of print outputs. This additional step added approximately 12–15 ms to the total loop time. Therefore, the actual loop times for each trial would be ~12–15 ms shorter than the values shown in the charts and tables.

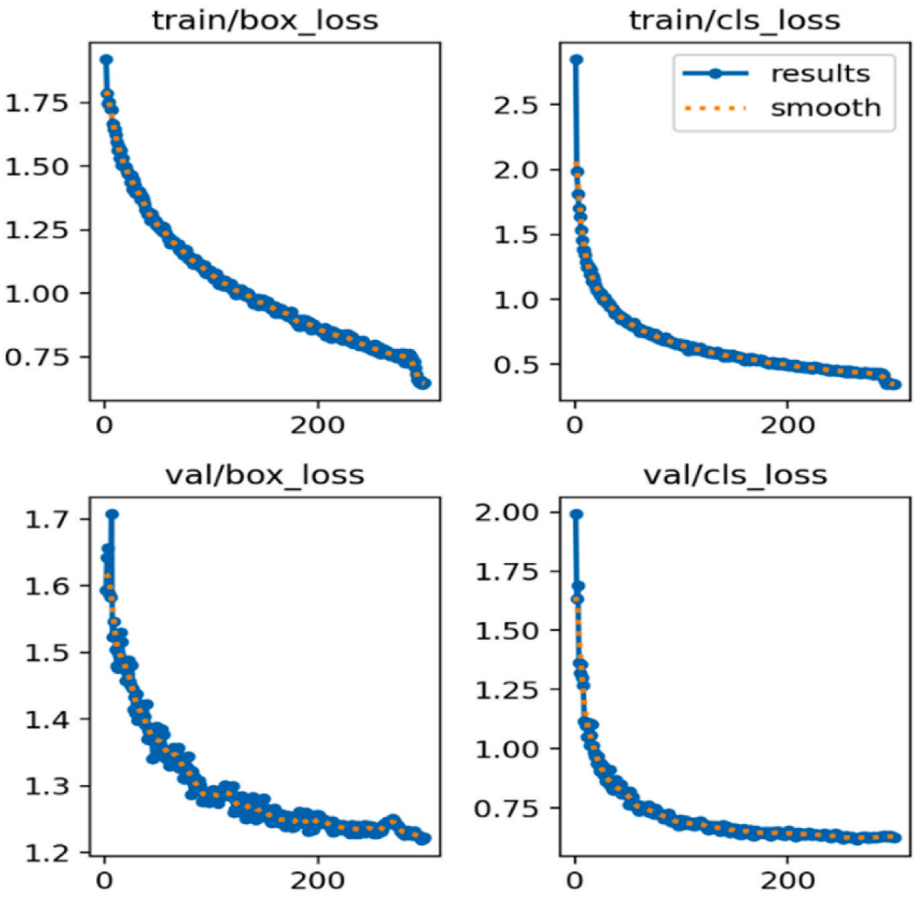


Fig. 12. Bounding box loss and classification loss for the training and validation sets.

4. Discussion

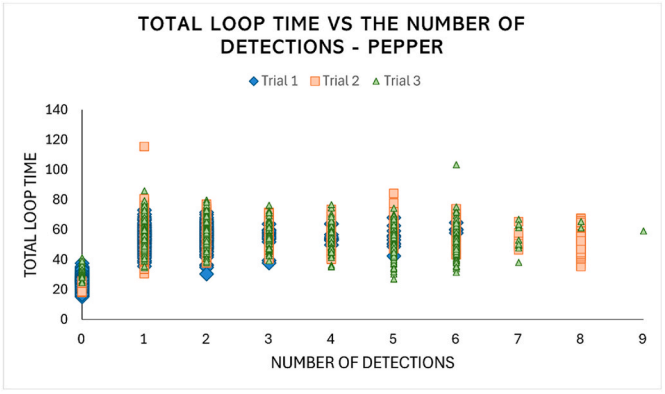
4.1. Camera performance under three vehicle speeds

For real-time implementation, it was important to not only ensure that all the classes were detected with a high accuracy, but also to have a high probability of those detected classes being in the region of interest to perform the targeted actuation, such as spraying the nozzles. Having a higher number of frames with the target class gave it a higher probability of getting detected in the region of interest. As evident from Table 6, with an increase in the speed of the vehicle, the number of continuous frames that captured a target object decreased. Since the region of interest comprised the region along the centre of the image (horizontal axis), with 30 pixels on each half, there must be at least six continuous frames with a target class to give it a good chance of being detected in the target zone. If it was detected in all the frames except for the frame where it was in the target zone, there would be no actuation, and the nozzles would not be triggered. Hence, any speed greater than 1.12 m s^{-1} is to be avoided. Speeds below 0.89 m s^{-1}, where the number of continuous frames was at least seven, are preferred.

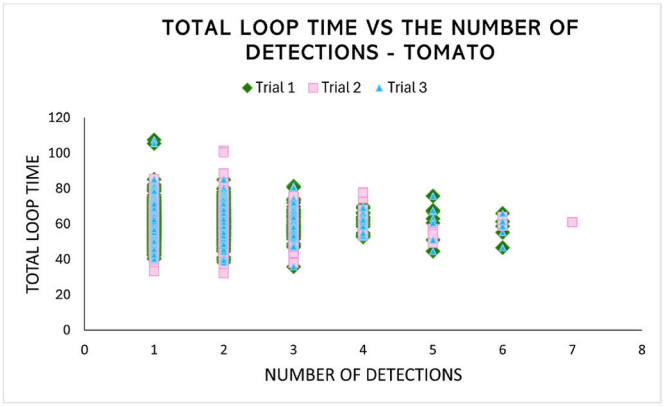
4.2. Model performance: detection

The trained model achieved very high performance in detecting and localising both crop classes (pepper and tomato), with an F1 score above 96 % and an mAP of 99 %. This bodes well for real-time herbicide spraying applications because the model detected the crop classes correctly almost every time. This significantly reduced any chance of misidentifying the plants as weeds and wrongly spraying them. The latter situation is undesirable as growers want to avoid spraying the plants at all costs. A missed identification of a weed is significantly less critical than a misclassification of a crop as a weed. In the first case, the weed is simply not detected and therefore not sprayed. In the latter, however, the crop is mistaken for a weed and sprayed with herbicide, which can result in crop damage or loss. In this respect, the model exhibited very good performance in the crop classes.

The performance of the model on weeds is relatively lower than that on crops because of the variation in the weed dataset. Crops in the dataset were always on a white background of plastic beds, had almost predictable positions in an image as they were planted in standardised holes in the beds, and there was very little occlusion from other classes onto the crop class instances, especially in early growth stages. This made the crop dataset easier for the model to learn. In contrast, the weed classes did not have any fixed, predictable position in the image, as their distribution was random. Sedge had a higher probability of being found on the beds. But it could also be found near the edge of beds and in the row middle. Grass and broadleaf weeds had a lower probability than sedge of piercing the plastic beds and growing on the beds. However, they were found near the crop holes and edges of the bed. Grass and sedge at certain growth stages were easily misidentified because of similar longitudinal leaf structures. They can also grow in clusters together with broadleaf weeds. Such issues made it difficult for the model to identify the weed classes under certain difficult conditions, such as small size, low light, occlusion, etc. However, the F1 score was still greater than 81 % for both sedge and grass and around 70 % for broadleaf weeds. The variation in the types of broadleaf weeds present in the field was one of the primary reasons for its low score. However, it could be improved with class-focused fine-tuning methods, such as lower-class weighting. Testing the system with lighter detection models trained on datasets from multiple vegetable farms across multiple seasons could improve the detection performance. Using faster frameworks and improved neural network architectures to improve real-time



a)



b)

Fig. 13. Speed of the main code when tested on pepper and tomato beds in real time; a) shows the speed in terms of the total loop time vs the number of detections per frame on the pepper bed, and b) for the tomato bed.

Table 8
Average loop time (ms) for the total number of detections per frame for the pepper bed.

	Average loop time (ms) – Pepper bed							
Trial	0	1	2	3	4	5	6	7
1	26.3	55.1	55.42	55.67	55.8	53.9	55.6	54.1
2	27.1	57.9	57.4	57.5	57.05	57.8	58.1	54.1
3	32.1	60	59.3	57.4	55.8	53.7	54.2	54.8

Table 9
Average loop time (ms) for the total number of detections per frame for the tomato bed.

	Average loop time (ms) – Tomato bed					
Trial	1	2	3	4	5	6
1	60.7	61.17	60.2	60.9	61.4	58.4
2	59.2	60.7	58.7	62.1	53.4	60.1
3	60.7	61.17	60.18	60.9	61.5	58.4

performance under varying lighting conditions would also be beneficial. Techniques such as hierarchical classification using two stages of classifiers, the first stage for broad-leaf and the next stage for the varieties of broadleaf, could allow the model to learn the finer distinctions within the broadleaf class better.

4.3. Model performance - speed

Model performance, in terms of speed, evaluated on images collected by the robotic smart sprayer in real-time test conditions, provided an insight into the bottlenecks of real-time implementation. Trials conducted on pepper and tomato beds in real-time allowed evaluation of the Python code's execution time under actual operating conditions. When there were classes of interest (three weed classes and two crop classes for this case) in a frame, it was detected by the model, and the average loop time was ~55 ms. Without any detection, the loop time was ~28 ms for trials on the pepper bed. And similar for tomato beds as well. The only difference was a 5 ms increase in loop times due to different output statements. For both crop beds, trials involved saving processed images for debugging, which increased the loop times by 12–15 ms. Without this step, the real-time loop times would be around 40–45 ms, resulting in a speed of 22–25 fps for the main thread. For both beds, an increase in the number of detections per frame did not increase the loop times, indicating that multiple numbers and classes of detections do not strain the model.

5. Conclusion

In this study, the machine vision system of a precision smart sprayer was developed. The system's camera selection criteria were presented along with its calibration and performance under different vehicle/robot speeds. Evaluating the instances of objects captured in consecutive frames of the camera provided an upper-speed limit of 1.12 m s^{-1} for the robot, beyond which the machine vision system would suffer through the loss of instances of objects in the frames. Also, beyond this speed, the probability of actuation for the given camera setup on the vehicle would be lower. Ideally, the speed should stay between 0.45 and 0.89 m s^{-1} to give it the best possible chance to be in the region of interest to allow the algorithm to trigger the actuator.

The data collection was done for five classes of objects (two crop types and three weed classes). The initial images collected were augmented using class-focused augmentation for sedge and grass to improve their performance. Post-augmentation, the model showed improvement in all the weed classes and showed no signs of overfitting. The intra-class variation in the broadleaf class contributed to the model's relatively lower detection performance. The model was employed in a multithreaded Python code. The main thread received continuous information from the camera and GPS sensors and processed the image in real time. The speed of Python code utilised in the model was also evaluated in real-time on tomato and pepper beds. The actual loop times for the main thread were noted, and the variation in the speed of the code was analysed for a higher number of detections in a frame. Future work should explore lighter models, faster frameworks, improved neural network architectures, as well as techniques such as hierarchical classification. In addition, a focused investigation of how environmental variability affects detection accuracy and processing speed would provide valuable insights for enhancing real-time performance.

CRediT authorship contribution statement

Vinay Vijayakumar: Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Yiannis Ampatzidis:** Writing – review & editing, Validation, Supervision, Resources, Project administration, Methodology, Investigation, Funding acquisition, Conceptualization. **Christian Lacerda:** Writing – review & editing, Supervision, Methodology, Formal analysis, Data curation. **Tom Burks:** Writing – review & editing, Supervision, Methodology, Conceptualization. **Won Suk Lee:** Writing – review & editing, Supervision, Methodology, Conceptualization. **John Schueller:** Writing – review & editing, Supervision, Methodology, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This material was made possible, in part, by a Cooperative Agreement from the U.S. Department of Agriculture, National Institute of Food and Agriculture, Award #2020-67021-30761. Its contents are solely the responsibility of the authors and do not necessarily represent the official views of the USDA.

References

Ahmad, A., Saraswat, D., Aggarwal, V., Etienne, A., & Hancock, B. (2021). Performance of deep learning models for classifying and detecting common weeds in corn and soybean production systems. *Computers and Electronics in Agriculture, 184*, Article 106081. https://doi.org/10.1016/J.COMPAG.2021.106081

Aitkenhead, M., Dalgetty, I. A., Mullins, C., McDonald, A. J. S., & Strachan, N. J. C. (2003). Weed and crop discrimination using image analysis and artificial intelligence methods. *Computers and Electronics in Agriculture, 39*(3), 157–171. https://doi.org/10.1016/S0168-1699(03)00076-0

Bossu, J., Gée, C., & Truchetet, F. (2007). Wavelet transform to discriminate between crop and weed in agronomic images. *Proceedings of SPIE, the International Society for Optical Engineering/Proceedings of SPIE, 6763*, 67630R. https://doi.org/10.1117/12.735568

Brown, R. B., & Noble, S. D. (2005). Site-specific weed management: Sensing requirements- what do we need to see? *Weed Science, 53*(2), 252–258. https://doi.org/10.1614/WS-04-068R1

Buslaev, A., Iglovikov, V. I., Khvedchenya, E., Parinov, A., Druzhinin, M., & Kalinin, A. A. (2020). Albumentations: Fast and flexible image augmentations. *Information, 11*(2). https://doi.org/10.3390/info11020125. Article 2.

Dang, F., Chen, D., Lu, Y., & Li, Z. (2023). YOLOWeeds: A novel benchmark of YOLO object detectors for multi-class weed detection in cotton production systems. *Computers and Electronics in Agriculture, 205*, Article 107655. https://doi.org/10.1016/j.compag.2023.107655

Diao, Z. H., Guo, P., Zhang, B., Zhang, D., Yan, J., He, Z., Zhao, S.-Z., Zhao, C., & Zhang, J. (2023). Navigation line extraction algorithm for corn spraying robot based on improved YOLOv8s network. *Computers and Electronics in Agriculture, 212*, Article 108049. https://doi.org/10.1016/j.compag.2023.108049

Dinh, H., Wang, Q., Tu, F., Frymire, B., & Mu, B. (2023). Evaluation of motion blur image quality in video frame interpolation. *Electronic Imaging, 35*(8), 262–265. https://doi.org/10.2352/ei.2023.35.8.iqsp-262

Gerhards, R., Sökefeld, M., Schulze-Lohne, K., Mortensen, D. A., & Kühbauch, W. (1997). Site specific weed control in winter wheat. *Journal of Agronomy and Crop Science, 178*(4), 219–225. https://doi.org/10.1111/J.1439-037X.1997.TB00494.X

Inoue, M., Jiang, M., Matsumoto, Y., Takaki, T., & Ishii, I. (2017). Motion-blur-free video shooting system based on frame-by-frame intermittent tracking. *Robomech Journal, 4*(1). https://doi.org/10.1186/s40648-017-0096-0

Jocher, G., Qiu, J., & Chaurasia, A. (2023). Ultralytics YOLO. *Ultralytics* [Computer software] Version 8.0.0. https://github.com/ultralytics/ultralytics.

Karkee, M., Zhang, Q., Gong, X., Gongal, A., & Zhang, J. (2024). Lightweight and real-time fruit detection model for robotic harvesting using improved YOLOv8 architecture. *Computers and Electronics in Agriculture, 212*, Article 108595. https://doi.org/10.1016/j.compag.2024.108595

Lee, W. S., Slaughter, D. C., & Giles, D. K. (1999). Robotic weed control system for tomatoes. *Precision Agriculture, 1*, 95–113.

Li, H., Gu, Z., He, D., Wang, X., Huang, J., Mo, Y., Li, P., Huang, Z., & Wu, F. (2024). A lightweight improved YOLOv5s model and its deployment for detecting pitaya fruits in daytime and nighttime light-supplement environments. *Computers and Electronics in Agriculture, 220*, Article 108914. https://doi.org/10.1016/j.compag.2024.108914

Li, J., Zhang, W., Zhou, H., Yu, C., & Li, Q. (2023). Weed detection in soybean fields using improved YOLOv7 and evaluating herbicide reduction efficacy. *Computers and Electronics in Agriculture, 209*, Article 107911. https://doi.org/10.1016/j.compag.2023.107911

Liu, J., Abbas, I., & Noor, R. S. (2021). Development of deep learning-based variable rate agrochemical spraying system for targeted weeds control in strawberry crop. *Agronomy, 11*(8). https://doi.org/10.3390/agronomy11081480. Article 8.

Liu, B., Li, R., Li, H., You, G., Yan, S., & Tong, Q. (2019). Crop/weed discrimination using a field imaging spectrometer system. *Sensors, 19*(23), 5154. https://doi.org/10.3390/S19235154

Liu, Y., Zeng, F., Diao, H., Zhu, J., Ji, D., Liao, X., & Zhao, Z. (2024). YOLOv8 model for weed detection in wheat fields based on a visual converter and multi-scale feature fusion. *Sensors, 24*(13). https://doi.org/10.3390/s24134379. Article 13.

Oriade, C. A., King, R. P., Forcella, F., & Gunsolus, J. L. (1996). A bioeconomic analysis of site-specific management for weed control. *Applied Economic Perspectives and Policy, 18*(4), 523–535. https://doi.org/10.2307/1349587

Partel, V., Charan Kakarla, S., & Ampatzidis, Y. (2019). Development and evaluation of a low-cost and smart technology for precision weed management utilizing artificial intelligence. *Computers and Electronics in Agriculture, 157*, 339–350. https://doi.org/10.1016/j.compag.2018.12.048

Qi, Y., & Wang, R. (2025). PMDNet: An improved object detection model for wheat field weed detection. *Agronomy, 15*(3), 250. https://doi.org/10.3390/agronomy15030250

Ruigrok, T., van Henten, E., Booij, J., van Boheemen, K., & Kootstra, G. (2020). Application-specific evaluation of a weed-detection algorithm for plant-specific spraying. *Sensors, 20*(24). https://doi.org/10.3390/s20247262. Article 24.

Shapira, U., Herrmann, I., Karnieli, A., & Bonfil, D. J. (2013). Field spectroscopy for weed detection in wheat and chickpea fields. *International Journal of Remote Sensing, 34*(17), 6094–6108. https://doi.org/10.1080/01431161.2013.793860

Sportelli, M., Apolo-Apolo, O. E., Fontanelli, M., Frasconi, C., Raffaelli, M., Peruzzi, A., & Perez-Ruiz, M. (2023). Evaluation of YOLO object detectors for weed detection in different turfgrass scenarios. *Applied Sciences, 13*(14). https://doi.org/10.3390/app13148502. Article 14.

Vijayakumar, V., Ampatzidis, Y., Schueller, J. K., & Burks, T. (2023). Smart spraying technologies for precision weed management: A review. *Smart Agricultural Technology, 6*, Article 100337. https://doi.org/10.1016/j.atech.2023.100337

Wang, C.-Y., Bochkovskiy, A., & Liao, H. M. (2023). YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *IEEE/CVF conference on computer vision and pattern recognition (CVPR)* (pp. 7464–7475). https://doi.org/10.1109/CVPR52729.2023.00721. Vancouver, BC, Canada.

Wang, A., Zhang, W., & Wei, X. (2019). A review on weed detection using ground-based machine vision and image processing techniques. *Computers and Electronics in Agriculture, 158*, 226–240. https://doi.org/10.1016/j.compag.2019.02.005

Zhang, F., & Fu, L. S. (2013). Application of computer vision technology in agricultural field. *Applied Mechanics and Materials, 462–463*, 72–76.

Zhang, Y., Liu, X., Chen, L., & Yang, Z. (2024). Improved YOLOv8-based multi-scale feature fusion for defect detection in crop monitoring robots. *Advanced Technology in Agriculture, 4*, Article 100648. https://doi.org/10.1016/j.atech.2024.100648

Zhang, N., Wang, N., Elfaki, M. S., Chaisattapagon, C., & Fan, G. (2006). Weed identification using imaging technology and spectroscopy. *Environment Control in Biology, 44*(3), 161–171. https://doi.org/10.2525/ecb.44.161

Zhou, X., Lin, D., & Chen, J. (2024). Comparative study of YOLOv7 and YOLOv8 in pulmonary carcinoma detection. *Computers in Biology and Medicine, 167*, Article 107716. https://doi.org/10.1016/j.compbiomed.2024.107716