AI-based Precision Weed Detection and Elimination

Hovannes Kulhandjian[1], Devin Rocha[1], Blake Bennett[1], and Michel Kulhandjian[2]

[1]Department of Electrical and Computer Engineering, California State University, Fresno, CA, USA.

[2]Department of Electrical and Computer Engineering, Rice University, TX, USA.

Emails: hkulhandjian@csufresno.edu, devrocha01@mail.fresnostate.edu, vivalacrunch@mail.fresnostate.edu, michel.kulhandjian@rice.edu

A paper from the Proceedings of the
16th International Conference on Precision Agriculture
21-24 July 2024
Manhattan, Kansas, United States

Abstract.

Weeds are a significant challenge in agriculture, competing with crops for resources and reducing yields. Addressing this issue requires efficient and sustainable weed elimination systems. This paper presents a comprehensive overview of recent advancements in weed elimination system development, focusing on innovative technologies and methodologies. Specifically, it details the development and integration of a weed detection and elimination system based on the CoreXY architecture, implemented on the Smart Agricultural Robot bulldog (SARDOG). The system utilizes a Raspberry Pi 5, Ultralytics YOLOv8 deep learning model, and CoreXY configuration with Nema 17 stepper motors. The weed detection model, trained on the CropAndWeed dataset, guides the system to position a sprayer accurately over detected weeds for herbicide application. The paper discusses hardware-software integration, including testing and tuning phases, and outlines the coordinate system program, stepper driver program, and CoreXY integration. Finally, it describes the SARDOG integration and testing process, highlighting successful implementation and concluding the project's development. This research contributes to advancements in weed management practices and agricultural robotics, emphasizing the potential for sustainable agriculture.

Keywords.
Weed Detection, Weed Elimination, Artificial Intelligence (AI), Precision Agriculture, Farm Robotics, Machine Learning.

I. Introduction

Precision agriculture has been of significant importance to the farming industry in the past few years, revolutionizing traditional farming practices [1-4]. By utilizing advanced technologies such as GPS, IoT sensors, drones, robotic arms, and data analytics, precision agriculture enables farmers to optimize their use of resources, enhance crop yields, and reduce environmental impact. These technologies allow for real-time monitoring and management of crops, soil, and weather conditions, leading to more informed decision-making and efficient farm management. Additionally, precision agriculture supports sustainable farming practices by minimizing waste, reducing the use of pesticides and fertilizers, and conserving water. As the global demand for food continues to rise, the adoption of precision agriculture is proving to be crucial in meeting this demand while ensuring the long-term viability of farming.

Weeds pose significant challenges to agricultural productivity, competing with crops for resources and reducing yields. To address this issue, the development of efficient and sustainable weed elimination systems has become increasingly important. This paper provides a comprehensive overview of the latest advancements in weed elimination system development, focusing on innovative technologies, methodologies, and strategies employed in the field. From the integration of deep learning models for precise weed detection to the implementation of sophisticated hardware systems for targeted herbicide application, this paper explores the multifaceted approach undertaken to enhance weed management practices. Through a synthesis of recent research and development efforts, this paper aims to elucidate the current state-of-the-art in weed elimination system development and its potential implications for sustainable agriculture.

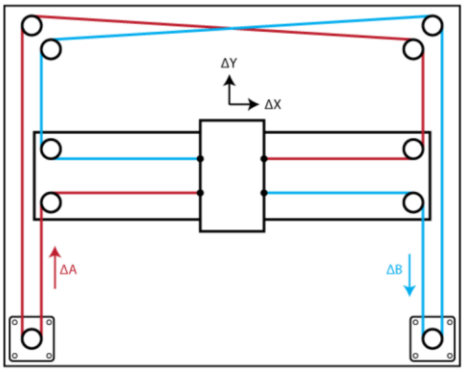
In this paper, we develop a weed detection and elimination system that is based on the XY-Core system. The developed system is installed on our newly developed Smart Agricultural Robot bulldog (SARDOG) [4]. The initial linear ball screw weed elimination design has been upgraded to a sophisticated dual belt system capable of XY movements. Employing the CoreXY configuration, favored in 3D printing and etching devices for its precision and responsiveness, the system utilizes two Nema 17 stepper motors within a square aluminum frame, driven by DM542 motor drivers. The Raspberry Pi 5 replaces the previous model for faster processing of weed detection software while maintaining rapid response from the elimination system. The weed detection component utilizes the Ultralytics YOLOv8 computer vision deep learning model, coupled with the Raspberry Pi 5 and ELP Sony USB Camera. This hardware-software integration enables the detection of weeds in ground images, guiding the subsequent actions of the elimination system. The coordinate system program translates pixel coordinates from the detection model into stepper motor steps, facilitating precise positioning of the sprayer over detected weeds.

Training the weed detection model involved a trial-and-error approach, utilizing the CropAndWeed dataset for ample data acquisition. Initial training sessions, conducted locally, were later shifted to Google Colab for enhanced performance and reduced training duration. Despite challenges in classification accuracy, the model effectively detects and targets weeds, aligning with the primary objective of the project. Integration with hardware involved meticulous wiring and configuration of stepper drivers and motors, adhering to specifications outlined in the DM542 datasheet. With hardware and software now integrated, extensive testing and tuning were conducted, ensuring optimal performance of the system. The final phase involved the integration of the CoreXY system into the SARDOG robot, marking the successful culmination of the project.

Through a synthesis of recent research and development efforts, this paper elucidates the current state-of-the-art in weed elimination system development, underscoring its potential implications for sustainable agriculture. The following sections provide detailed insights into the system's design, development, and performance, offering valuable contributions to the field of agricultural robotics and weed management practices.

II. CoreXY Elimination System Development

The original linear ball screw weed elimination design has been upgraded to a sophisticated dual belt system capable of XY movements. The CoreXY configuration, favored in 3D printing and etching devices for its precision and responsiveness, was chosen. Unlike the singular Nema 23 stepper motor driving the ball screw actuator, the CoreXY system utilizes two smaller Nema 17 stepper motors. Constructed within a square aluminum frame, the CoreXY system includes idler pulleys, 30-tooth pulleys, GT2 timing belts, and a gantry system facilitating movement. The Nema 17 stepper motors are driven by DM542 motor drivers, which support a micro step of 400. Additionally, the Raspberry Pi 4 model B was replaced with the Raspberry Pi 5 for faster processing of weed detection software while maintaining rapid response from the elimination system. The frame features an extrusion with a central opening accommodating a gantry capable of sliding from end to end. Perpendicular gantries at the extrusion ends enable forward/backward movement in the Y direction, while the gantry on the extrusion slides in the X direction. Gantry movement is governed by simple kinematic equations: equal rotation in opposite directions moves the middle gantry forward/backward, equal rotation in the same direction moves it left/right, and single motor rotation causes diagonal movement.



Equations of Motion:

$\Delta X = {}^1/_2(\Delta A + \Delta B), \quad \Delta Y = {}^1/_2(\Delta A - \Delta B)$

$\Delta A = \Delta X + \Delta Y, \quad \Delta B = \Delta X - \Delta Y$

Fig. 1. CoreXY Belt Configuration and Kinematic Equations.

III. Detection System Development

A. Theoretical System Overview

The weed detection and elimination system design consists of the following hardware components:

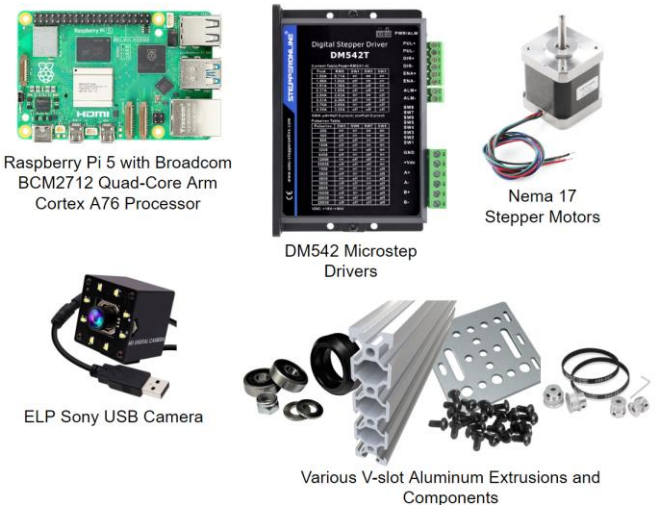


Fig. 2. Detection Hardware.

In the weed detection and elimination system, key hardware components include the Raspberry Pi 5 computer and the ELP Sony USB Camera. Software crucial for weed detection comprises the Ultralytics YOLOv8 computer vision deep learning model, Ultralytics libraries, Python scripts, and the publicly available CropAndWeed dataset.

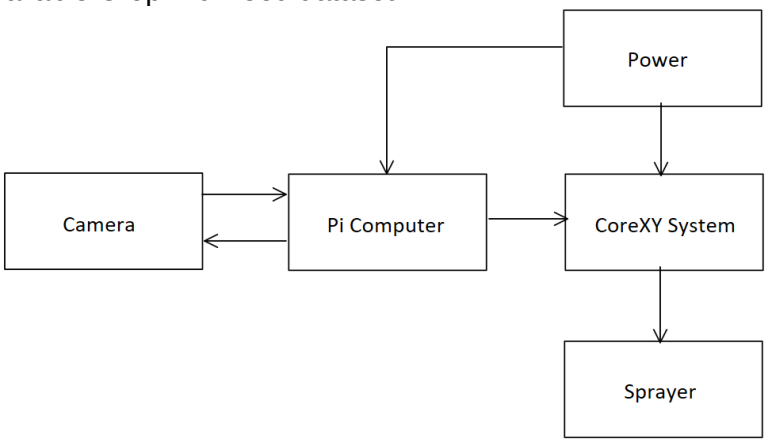


Fig. 3. Generic System Block Diagram.

The weed detection and elimination system design comprises several components outlined in the block diagram above. At the core is the Raspberry Pi 5 computer, serving as the central processing unit. The camera interfaces with the Raspberry Pi 5 to capture ground images, processed by scripts on the Pi. The Pi computer communicates with the Stepper drivers in the CoreXY system, driving them while processing images. SARDOG accommodates two 24V batteries connected to a central 24V output, along with 12V and 5V step-down converters. The Raspberry Pi 5 utilizes the 5V output, while the stepper drivers are powered by the 24V central output. Once integrated, the Raspberry Pi 5 captures ground images, processes them, and guides the CoreXY system to position the sprayer over weeds for herbicide application.

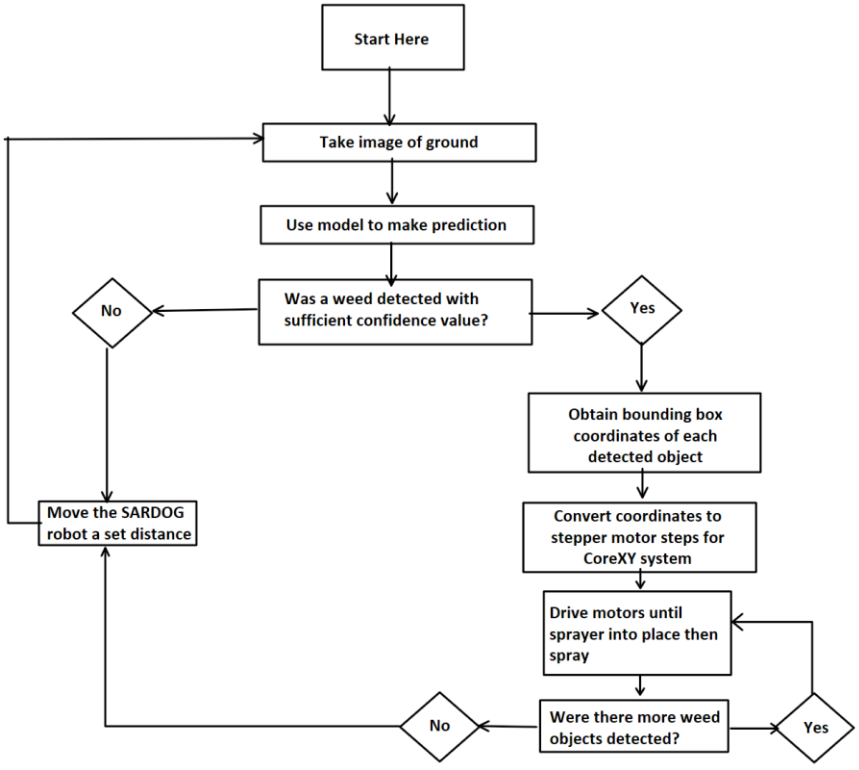


Fig. 4. System Behavior Flowchart.

The flowchart in Figure 4 outlines the operation of the weed detection and elimination system. Upon initiation, it captures an image of the ground and utilizes the trained YOLOv8 computer vision model for weed detection. If no weeds are detected, it repeats the process of image capture. If weeds are detected, the system retrieves the bounding box coordinates for each instance of the weed. These coordinates are then translated into motor steps to position the sprayer over the weed. After herbicide application, the system proceeds to the next detected weed until all are treated. Upon completion, the process restarts with another image capture.

B. Initial Hardware and Software Testing

The initial stage of the project was primarily setting up and researching the various components and software that were going to be used for the weed detection and elimination system. A large portion of the early setup and testing was becoming familiar with the Raspberry Pi computer, setting it up with the YOLOv8 software, testing the GPIO, and testing the stepper motors and drivers.

The first task was setting up the Raspberry Pi with YOLOv8 software and libraries. Once they were installed, basic programs were created to simultaneously test the software and become more familiar with it. For the purposes of early testing, a pre-trained YOLOv8 model (trained on the COCO dataset) was utilized to make testing the libraries possible while the actual weed detection model was being trained.

Fig. 5. Preliminary Coordinate System.

When first experimenting with the YOLOv8 pre-trained model and Raspberry Pi GPIO, a very basic version of the coordinate system was created so that a better understanding of the software could be obtained. This also enabled basic testing to be performed on the stepper motors. A basic program was written that activated an LED on the left, middle, or right side of a breadboard depending on where a smartphone object was detected in the coordinate system. Using a smartphone was selected arbitrarily to allow for ease of testing while waiting for the actual weed detection model to be trained. This was the first step towards creating the much more advanced version of the CoreXYdriver coordinate system that was implemented in the final product. This LED test was eventually modified into a stepper motor test, which allowed for the creation of a rudimentary detection and response system. This program allowed for testing of the stepper motors and drivers with the Pi GPIO, as well as developed a better understanding of how they would work with the detection model.

C. Weed Dataset Setup and Formatting

During machine learning and computer vision training, ample data is essential for model training and testing. Several datasets were evaluated before selecting the CropAndWeed Dataset as the optimal fit for this project [5]. This dataset comprises 8034 images featuring over 112k instances of crops and weeds captured from farm plots. The dataset categorizes weeds and crops into two distinct classes to assess performance differences.

Class	Species	Type	Class	Species	Type
1	Maize	Crop	13	Amaranth	Weed
2	Sugar Beet	Crop	14	Goosefoot	Weed
3	Sunflower	Crop	15	Potato Weed	Weed
4	Bean	Crop	16	Chamomile	Weed
5	Pea	Crop	17	Crucifer	Weed
6	Soy	Crop	18	Plantago	Weed
7	Potato	Crop	19	Poppy	Weed
8	Pumpkin	Crop	20	Corn Spurry	Weed
9	Grasses	Weed	21	Mercuries	Weed
10	Thistle	Weed	22	Solanales	Weed
11	Geranium	Weed	23	Chickweed	Weed
12	Knotweed	Weed	24	Labiate	Weed

Table 1. Dataset Configuration [5].

Table 1 displays the class numbers assigned to each crop and weed category used in the configuration file for training the detection model. The first configuration featured 100 individual classes, each representing a specific weed type. The alternative configuration included 24 classes, grouping certain weed types into broader categories. The 24-class setup was ultimately chosen as the optimal configuration. Another potential configuration for future exploration involves creating a single superclass encompassing all weed types, which may yield improved detection results, given that weed classification is not essential for this project. The data was split into training and validation sets with a ratio of 93/7 to maximize training data retention. Approximately 7500 images were randomly allocated to the training folder, with the remaining 534 used for model validation.

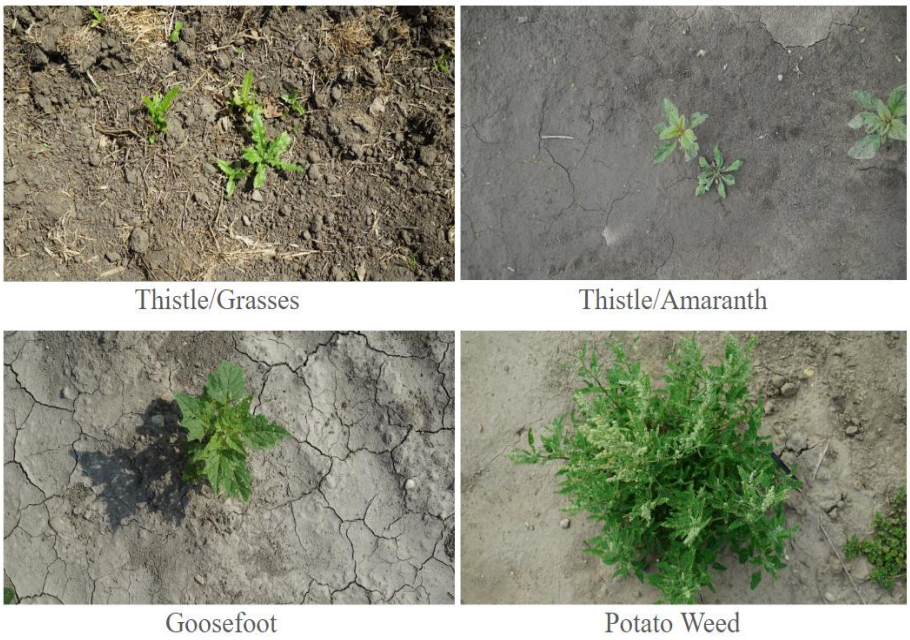


Fig. 6. Sample Weeds from the Dataset.

Figure 6 above showcases sample images from the dataset, each containing 10 to 100 different weeds, providing ample training data. The images feature various weed classes such as Thistles, Grasses, Goosefoot, Potato Weed, and Amaranth, among others detailed in Table 1. Each image in the dataset is accompanied by a corresponding CSV file containing multilabel annotations for every weed. These annotations include bounding box coordinates, weed class, and stem position. However, YOLOv8 requires annotations in a specific format: text files with normalized values structured as Class, X_Center, Y_Center, Width, and Height. To comply with this format, a MATLAB script was developed to convert all 8034 annotation files into YOLOv8-compatible text files.

Class = Label ID
X_Center = (Left + Right) / 2
Y_Center = (Top + Bot) / 2
Width = Right - Left
Height = Bot - Top

Fig. 7. Derived Formatting Conversion Equations.

Normalized X_Center = X_Center / ImageWidth
Normalized Y_Center = Y_Center / ImageHeight
Normalized Width = Width / ImageWidth
Normalized Height = Height / ImageHeight

Fig. 8. Derived Normalization Conversion Equations.

The two sets of equations above were devised to convert all annotations into the correct format. The MATLAB script processes the 8034 CSV files, consolidating them into a large table, and then utilizes a loop to iterate through each file. During each iteration, the script opens a CSV file, organizes its data into a tall matrix, applies equations from Figure 7 to specific columns, and saves the data in a new matrix. Subsequently, the data is normalized using equations from Figure 8 for each column. Finally, the processed data is exported into a text file with the same name. Following this iterative process for all 8034 files, the annotations are transformed into YOLOv8 format, ready for use. These annotations are then combined with crop and weed images in a folder, partitioned into training and validation sets (93/7 ratio as mentioned earlier), compressed into a zip file, and uploaded to Google Drive. The data is now prepared for training using the YOLOv8 computer vision model.

D. YOLOv8 Computer Vision Training

The actual training process of the weed detection deep learning model involved a trial-and-error approach. Initial training sessions, conducted locally on an Intel i9-12900k CPU, utilized a class configuration with 100 unique classifications of weeds and crops. Each session consisted of 100 epochs, with an image size of 640x480 and a patience value of 15 epochs. Despite taking nearly 30 hours to complete, with an average epoch time of approximately 18 minutes, the performance and training time of these models were analyzed.

Subsequently, it was decided to shift future training attempts to a Google Colab environment for enhanced performance and reduced training duration. Subsequent training trials in Google Colab utilized a dataset uploaded to Google Drive, with the class configuration adjusted to 24 unique classifications of weeds and crops. This adjustment aimed to increase data per class, thus improving model performance. The training was conducted using an NVIDIA Tesla V100 PCIe, with parameters adjusted to 300 epochs, an image size of 640x480, and a patience value of 15 epochs. Despite encountering early termination at approximately 160 epochs due to a lack of performance improvement, these sessions demonstrated significant improvement. With an average epoch time of approximately 90 seconds, subsequent training times were reduced by nearly 93% compared to the local environment.

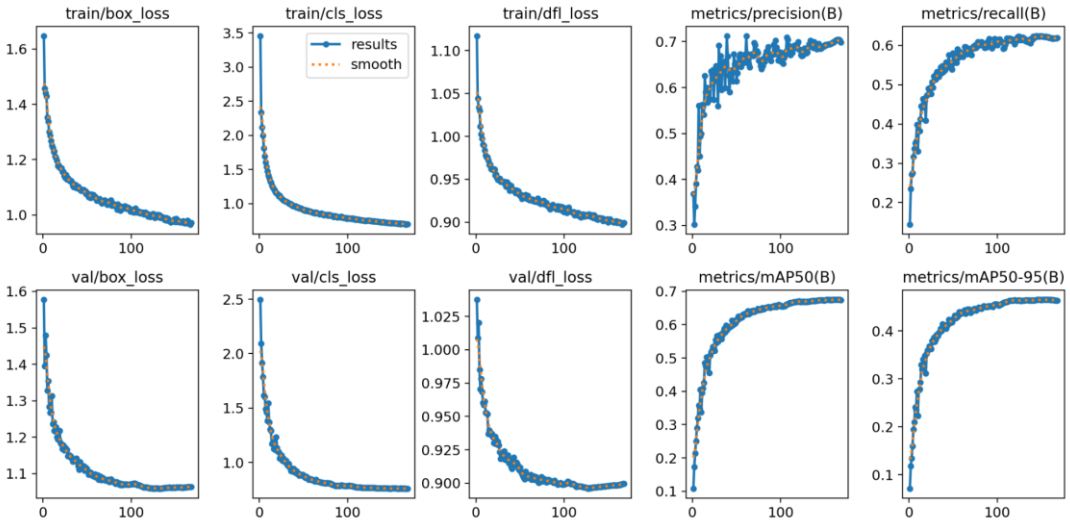


Fig. 9. Training and Validation Results.

The presented results illustrate the performance metrics of the trained deep learning model over the course of its training. At approximately 160 epochs (the end of each graph), the model demonstrates convergence into a stable state for the provided weed and crop dataset. While the model effectively detects real weeds with moderate accuracy, its performance in classification falls short. This discrepancy likely stems from limitations in the training data, as weeds exhibit significant variations in shape and size. Ideally, the dataset would contain tens of thousands of images per class, tailored to the local environment where the model will be deployed. Despite classification inaccuracies, the model excels in generic weed identification, which is the primary objective of this project. Ultimately, its function is to detect and spray weeds, rather than classify them.

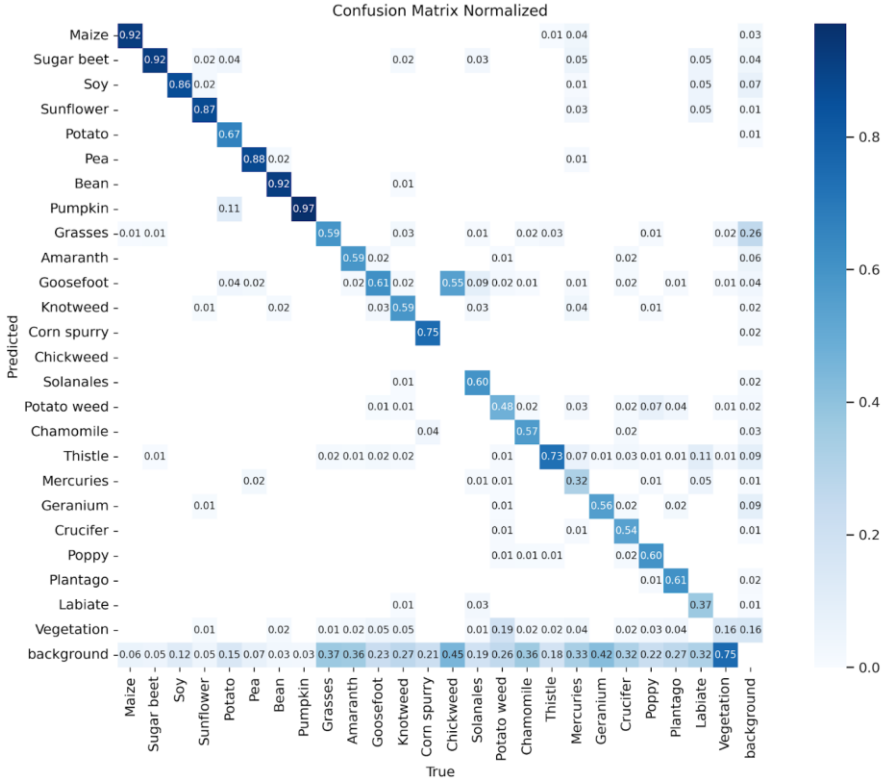


Fig. 10. Normalized Confusion Matrix.

The normalized confusion matrix above visually represents the deep learning model's classification predictions compared to the actual classifications of weeds. While the model performs well in many classes, it's important to note that these results are based solely on validation data. Its performance on general data is only moderate, suggesting a need for additional local data inclusion during model training to improve classification accuracy. However, it's essential to emphasize that classification is secondary to the primary objective of weed detection in this project.

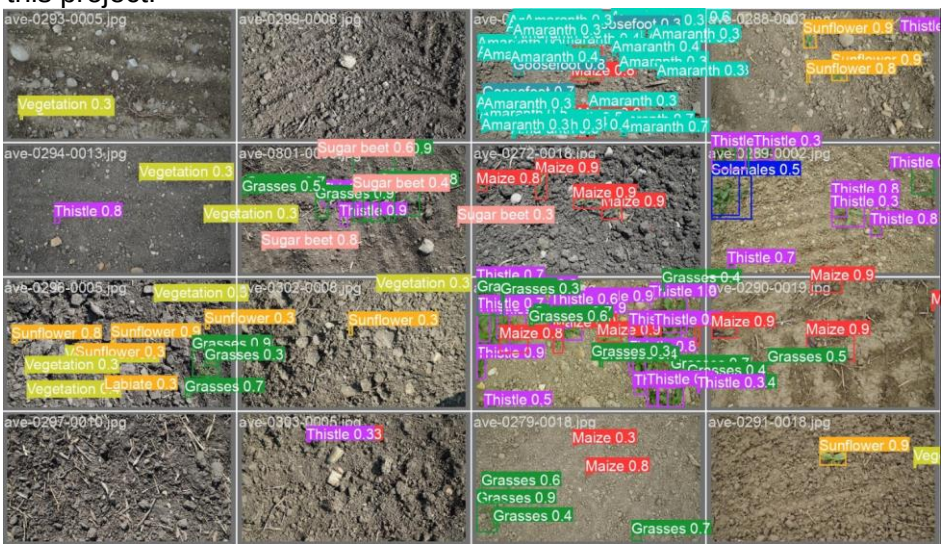


Fig. 11. Validation Batch Example.

The validation process is a crucial training step where the model refines its weights based on validation accuracy. The provided example validation batch illustrates multiple predictions made by the trained model. It predicts the location of weeds in the image, draws bounding boxes around them, and assigns a confidence value indicating its certainty about the weed type. This detection capability is invaluable for activating the sprayer, and the bounding box coordinates play a pivotal role in guiding the hardware. A detailed discussion of how this is managed will follow in the subsequent section.

E. Coordinate System Program

Having trained the weed detection computer vision model, the next significant task was integrating it with both software and hardware. Initially, the programs were developed for the Raspberry Pi 4 Model B. However, with the availability of the Raspberry Pi 5, which offers significantly faster performance, the code was later adapted for this platform. Due to differences in GPIO hardware mappings between the two Raspberry Pi models, the Python libraries were incompatible, necessitating adaptation of the coordinate system program to utilize the gpiozero library. The primary objective of the coordinate system was to synchronize the hardware, capable of moving a sprayer anywhere in an XY plane, with the weed detection model, which provides bounding box coordinates for detected weed objects. This was achieved by breaking down the image into its pixels and the sprayer XY plane into stepper motor steps.

$$Spp = \frac{S_{\max}}{P_{max}}$$

Where Spp is the Steps per pixel ratio
Where $S_{\max}$ *is the maximum steps*
Where P_{max} *is the maximum pixels*

Fig. 12. Derived Steps Per Pixel Equation.

The "Steps Per Pixel" ratio (SPP) illustrated above is determined by dividing the maximum number of stepper motor steps the sprayer can travel in one direction by the total number of pixels in the image. Since the image size is 640x480, and the maximum number of steps the sprayer can travel varies for the X and Y directions of the CoreXY, a unique SPP ratio is calculated for each direction, X and Y.

$$d = |P_T - P_c|$$

Where d is pixel distance
Where P_T is target pixel
Where P_c is current pixel

Fig. 13. Derived Pixel Distance Equation.

Given that the sprayer begins at a known location within the XY plane, the number of steps required to move the sprayer to the desired location in the image can be calculated by determining the absolute difference between the current sprayer pixel location and the target weed pixel location.

$$S = Spp \cdot d$$

Where S is steps
Where Spp is Steps per pixel ratio
Where d is pixel distance

Fig. 14. Derived Steps Equation.

The coordinate system program determines the required stepper motor steps to position the sprayer at the target location by multiplying the calculated pixel distance value with its corresponding SPP ratio. These equations played a crucial role in the development of the coordinate system program. The program initiates by importing the YOLOv8 libraries and loading the trained weed detection model. It then iterates through an infinite loop, processing the results of the weed detection model. The X and Y pixel coordinates of the bounding boxes are stored in variables. Subsequently, the coordinate system calculates the pixel distance from the sprayer to the target weed in both the X and Y directions. Utilizing the previously discussed equations, the coordinate system converts the pixel distance to steps for the X and Y directions. These step values are then transmitted to the Stepper Driver program, managing the actual movement of the sprayers. Once the sprayer is in position, the Sprayer Driver program is invoked to control the pump. The coordinate system repeats this process for each detected weed, ensuring to return of the sprayer to its initial location at the conclusion of the process. This cycle continues indefinitely until the weed detection and elimination system is deactivated.

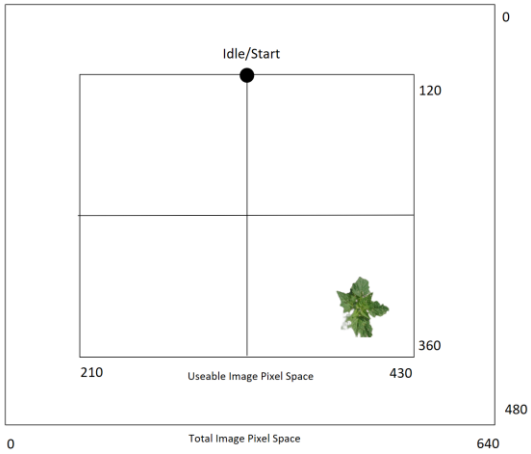


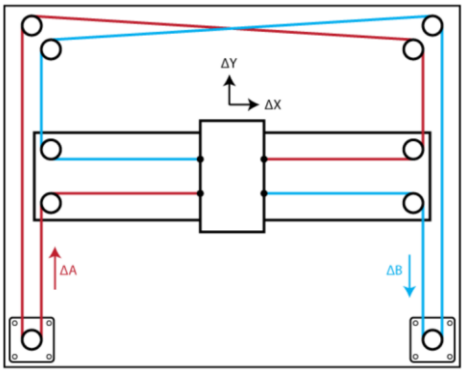
Fig. 15. Coordinate System Visualization.

The depicted image illustrates the functional aspect of the coordinate system. The idle/start location denotes the initial position of the sprayer relative to the entire image. Due to the camera's slight elevation above the CoreXY elimination system, there exists unusable space at the image border, inaccessible to the sprayer. The coordinate system compensates for this, preventing the motors from directing the sprayer beyond its operational range. This adjustment enhances sprayer precision by refining the accuracy of the steps per pixel ratio.

In essence, the coordinate system program serves as the primary control mechanism, orchestrating the operation of the weed detection model, computing the steps between the sprayer and the weed, and executing all requisite function calls.

F. Stepper Driver Program

The stepper driver program plays a pivotal role as a subprogram within the coordinate system. Following the completion of image processing and step calculations by the coordinate system, it invokes the stepper driver program. This program is furnished with the precise number of steps required to maneuver the sprayer in both the X and Y directions. Direct interaction with the Raspberry Pi GPIO facilitates the control of the stepper drivers.



Equations of Motion:

$\Delta X = {}^1/_2(\Delta A + \Delta B), \quad \Delta Y = {}^1/_2(\Delta A - \Delta B)$

$\Delta A = \Delta X + \Delta Y, \quad \Delta B = \Delta X - \Delta Y$

Fig. 16. CoreXY Architecture and Kinematic Equations.

The functionality of the stepper driver is intricate, particularly due to the CoreXY architecture of the elimination system. Upon configuring GPIO pins for interaction with the stepper drivers' pulse and direction inputs, the function undertakes steps in X and Y directions, adjusting them based on the camera's orientation. Motor A's direction and step count are determined by summing the X and Y steps to ascertain its delta, while Motor B's direction and step count involve subtracting Y steps from X steps to find its delta. This conversion is imperative as CoreXY employs both motors collaboratively to maneuver the gantry carriage, unlike conventional X and Y motion systems utilizing individual motors for each direction. To ensure operational feasibility, negative step values are converted to their absolute values, with the direction pin set counterclockwise as necessary. Subsequently, the stepper driver program initiates a loop, pulsing each motor until it reaches its designated destination, employing CoreXY kinematics principles.

G. CoreXY Integration

After training the weed detection model and finalizing the stepper driver and coordinating system programs, integrating with the hardware was essential for comprehensive testing and fine-tuning of the system. The wiring of stepper drivers and motors followed the specifications outlined in the DM542 datasheet.

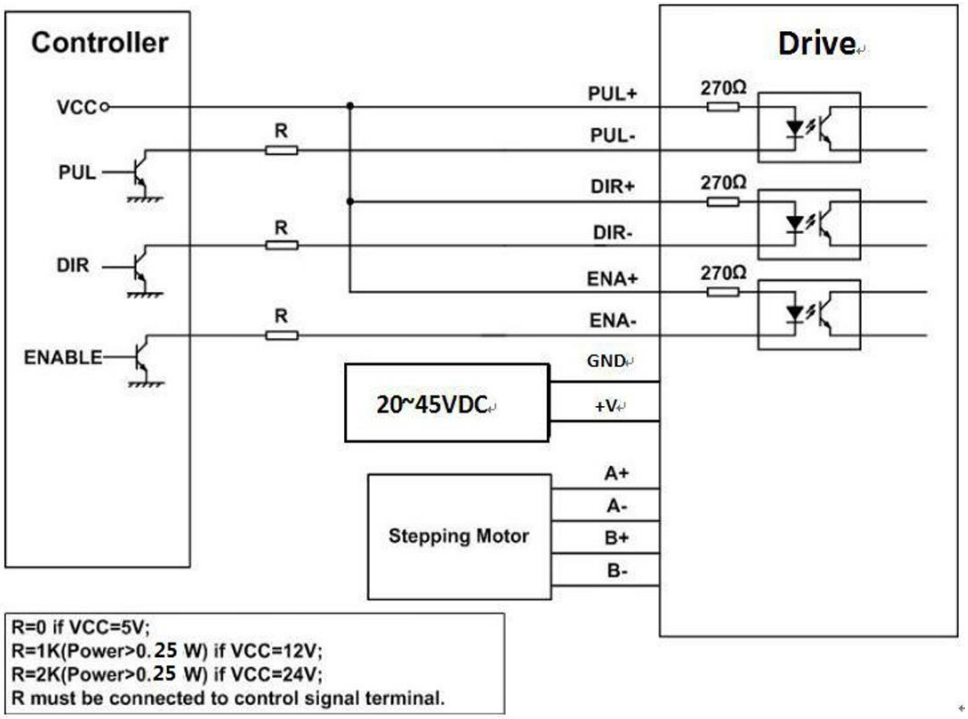


Fig. 17. DM542 Datasheet Connection.

The A and B coils of the stepper motors were linked to the respective stepper drivers. Additional wires were connected to the stepper drivers' +V and GND terminals to supply power. Throughout the testing phase, a DC bench power supply furnished 24V to the stepper drivers. Settings were adjusted to microstep at 400 steps per revolution, maintain a 50% current during idle, and limit each motor's current to 1 amp. Idle motor consumption was approximately 0.15A, rising to around 0.3A during operation. With motors and drivers installed on the CoreXY frame, control signals were then linked to the Raspberry Pi 5 GPIO using GPIO cables.

```
#Pin setup
PUL_A = 4  #board 7
#PUL_A- = board 9
PUL_B = 11  #board 23
#PUL_B- = board 25
DIR_A = 18  #board 12
#PUL_A- = board 14
DIR_B = 24  #board 18
#PUL_B- = board 20
```

Fig. 18. Stepper Driver Pins Code Snippet.

The connections for the stepper driver pulse and direction were configured as indicated in the provided code snippet. Each negative pulse and direction connection was linked to an individual ground terminal on the Raspberry Pi GPIO. While the specific pin values were somewhat arbitrarily chosen, the primary aim was to have the positive pulse and direction connections adjacent to available ground pins.

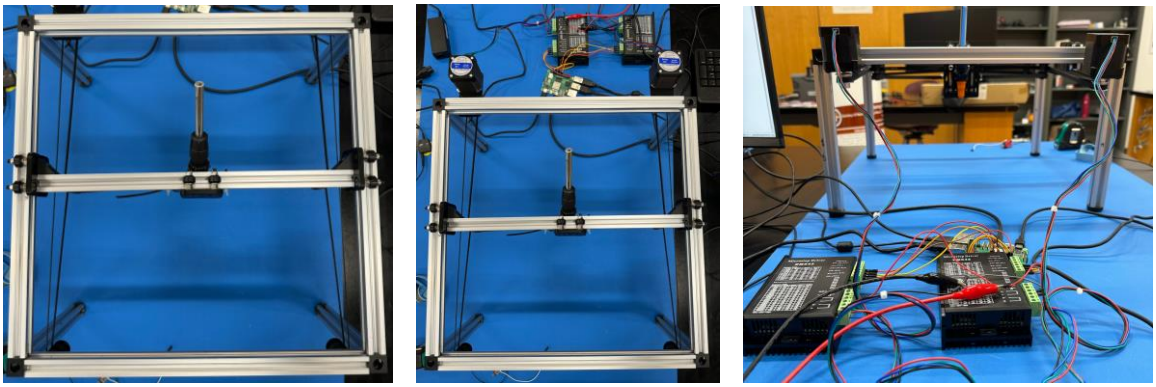


Fig. 19. Perspectives of Integrated Detection and CoreXY.

With hardware and software now integrated, the testing and tuning phase of the project began. A variant of the CoreXY driver program, known as CoreXYTest, was developed for manual testing of the CoreXY elimination system, allowing users to provide X and Y steps manually. During this phase, everything performed better than expected. Users could move the carriage with high precision using only the software, requiring minimal tuning for optimal hardware and software performance. The stepper motor pulse width was adjusted from 1ms down to 0.1ms in increments of 0.1ms, with an experimentally determined optimal pulse width of 0.7ms providing a balance of speed and system stability. Additionally, the test program helped determine the maximum number of steps the sprayer could travel in the Y and X directions—around 1800 steps and 1200 steps respectively—before reaching the edge of the plane. This information was crucial for accurate tuning of the CoreXY driver program before integration into the SARDOG robot.

IV. SARDOG Integration and Testing

Integrating the CoreXY weed elimination system into the SARDOG robot involved a step-by-step tuning process. Initially, the robot was widened to accommodate the CoreXY frame, which was then secured in place using zip ties around each corner. Since the robot undergoes frequent remodeling and relocation, a permanent mounting method wasn't suitable as it would hinder movement through narrow spaces. The zip ties provided flexibility, allowing the elimination system frame to slide in and out of the robot when needed. Next, the stepper drivers and Raspberry Pi 5 were secured beside the frame on a small wooden board mounted in one corner of the robot. The stepper drivers were powered by the robot's 24V output, while the camera was fixed to the front end, providing an elevated view of the underside. With all hardware integrated, the final phase involved software testing and calibration.

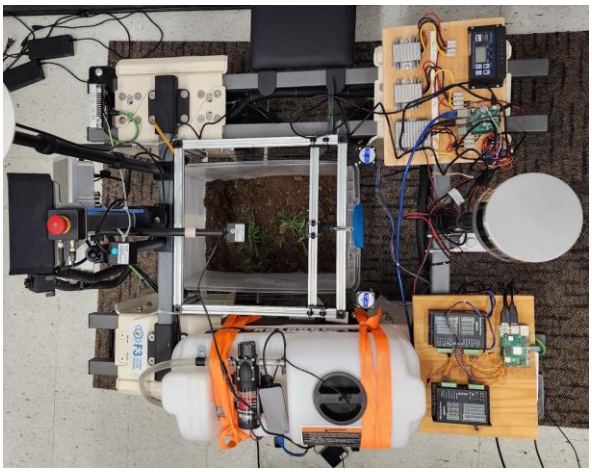


Fig. 20. Entire System Integrated on SARDOG.

In the image above, the CoreXY elimination frame is centrally mounted on the robot. At the bottom right, the two stepper drivers and the Raspberry Pi 5 computer are securely fastened to the wooden board. The camera is positioned to look straight down beneath the robot, providing a clear view of the weeds below. Additionally, a box containing soil and weeds is placed directly under the elimination system to test its performance on real weeds.

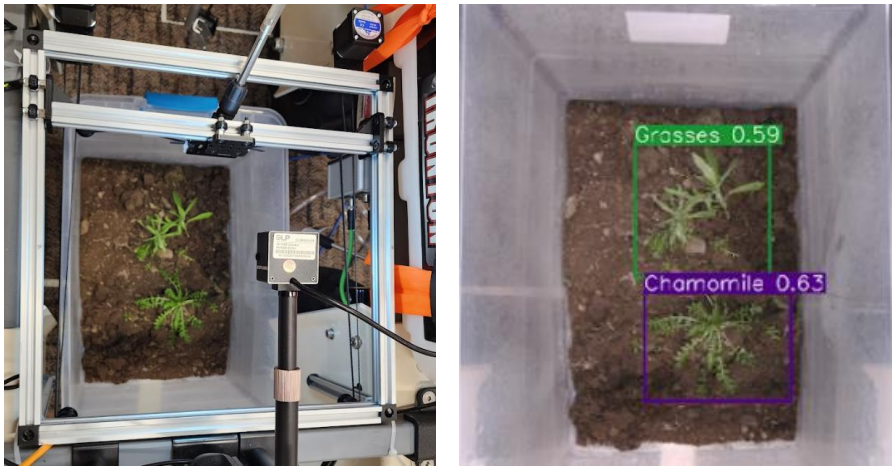


Fig. 21. Working System Close Up.

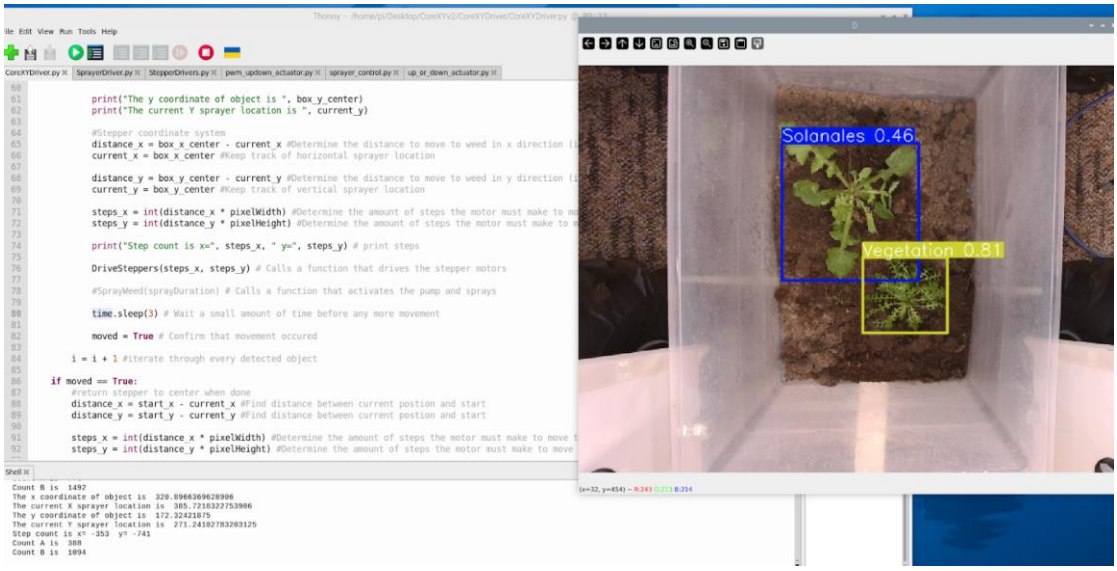


Fig. 22. CoreXY Driver Program Performing Predictions.

In Figure 21, the camera offers an optimal view of the ground beneath the robot. While the CoreXY driver program is operational, the weed detection model conducts inferences on images every few hundred milliseconds. Upon detecting weeds, as shown in the image on the right, the CoreXY driver program retrieves the bounding box coordinates, positions the sprayer over each weed, activates the sprayer and then returns it to the starting location. Calibrating the camera to ensure alignment with the weed required some adjustments initially, but after fine-tuning, the system operated smoothly. This marks the successful development and implementation of a fully functional weed detection and elimination system, thereby concluding the research and development of this senior project.

Conclusion

We've created an autonomous weed detection and elimination system for SARDOG. Utilizing a dual belt gantry system in a CoreXY configuration, we've enhanced precision and responsiveness while reducing maintenance needs in field settings. The system employs a trained YOLOv8 deep learning model for weed identification and classification, coupled with a complex coordinate system program to trigger the elimination process. Extensive research and design efforts were invested in perfecting the detection programs and deep learning model. The resulting system significantly reduces the risk of chemical exposure for farmers and field workers. SARDOG's compact size enables it to perform various functions efficiently, making it a valuable asset for weed detection and elimination.

Acknowledgments

We want to give a special thanks to DPS Telecom for generously providing funding for the project as well as to the Economic Development Administration (EDA) Project Grant 077907908, Fresno-Merced Future of Food Innovation (F3) Coalition.

References

[1] H. Kulhandjian, B. Irineo, J. Sales, M. Kulhandjian, “Low-Cost Tree Health Categorization and Localization Using Drones and Machine Learning,” in Proc. of IEEE International Conference on Computing, Networking and Communications (ICNC), Big Island, Hawaii, Feb. 2024.

[2] H. Kulhandjian, D. Rocha, B. Bennett, N. Amely, M. Kulhandjian, “AI-based Pollinator using XY-Core Robot,” in Proc. of 16th International Conference on Precision Agriculture, Manhattan, Kansas, July. 2024.

[3] H. Kulhandjian, N. Amely, M. Kulhandjian, “AI-based Fruit Harvesting using a Robotic Arm,” in Proc. of 16th International Conference on Precision Agriculture, Manhattan, Kansas, July. 2024.

[4] H. Kulhandjian, Y. Yang and N. Amely, "Design and Implementation of a Smart Agricultural Robot bullDOG (SARDOG)," in Proc. of IEEE International Conference on Computing, Networking and Communications (ICNC), Big Island, Hawaii, Feb. 2024.

[5] D. Steininger, A. Troll, G. Croonen, J. Simon, and V. Widhalm, “The crop and weed dataset: A multi-modal learning approach for efficient crop and weed manipulation,” 2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), Jan. 2022. doi:10.1109/wacv56688.2023.00372