Dataset Diversity in Crop Row Detection based on CNN Models for Autonomous Robot Navigation

Igor Ferreira da Costa
Pontifical Catholic University of Rio de Janeiro
Rio de Janeiro, RJ - Brazil
ifcosta93@gmail.com

Antonio Candea Leite
Norwegian University of Life Sciences
Ås, Norway
antonio.candea.leite@nmbu.no

Wouter Caarls
Pontifical Catholic University of Rio de Janeiro
Rio de Janeiro, RJ - Brazil
wouter@ele.puc-rio.br

Abstract

Agricultural automation emerges as a vital tool to increase field efficiency, pest control, and reduce labor burdens. While agricultural mobile robots hold promise for automation, challenges persist, particularly in navigating a plantation environment. Accurate robot localization is already possible, but existing RTK-GNSS systems are costly, while also demanding careful and precise mapping. In response, onboard navigation approaches gain traction, leveraging sensors like cameras and LiDARs. However, the machine learning methods used in camera-based systems are highly sensitive to the training dataset used. In this paper, we study the effects of dataset diversity on a proposed deep learning-based visual navigation system. Leveraging multiple datasets, we assess the model robustness and adaptability while investigating the effects of data diversity available during the training phase. The system is presented with a range of different camera configurations, hardware, field structures, as well as a simulated environment. The results show that mixing images from different cameras and fields can improve not only system robustness to changing conditions, but also its single-condition performance. Real world tests were conducted which show that good results can be achieved with reasonable amounts of data.

Keywords — Agricultural Navigation, Agricultural Robot, Deep Learning, Dataset Mixing, Dataset Diversity, Row Following, Lane Detection

1 Introduction

Agricultural mobile robots can be implemented to achieve better field automation and efficiency (Aravind et al., 2017), but the plantation environment has plenty of challenges to overcome. Those challenges include the capability of reliable operation in a variety of field conditions, even in a range of fields, so farmers can take advantage of the systems mounted in their robots. Accurate robot localization is already available with an RTK-GNSS (Global Navigation Satellite System with Real-time Kinematic Positioning) system, but the network of base stations required and the precise mapping of the field is expensive and presents a challenge (Ahmadi et al., 2021). Simultaneous Localization and Mapping (SLAM) and other inertial techniques are prone to drift because of the long and non overlapping trajectories (Bakken et al., 2019).

Due to the challenges controlling the robot based on position estimates, a change in perspective to onboard navigation is commonly researched. In the onboard paradigm, the robot is developed to sense the field using cameras, LiDARs and other sensors, and guide itself relative to it, in a similar manner that humans are able to. Cameras are inexpensive sensors able to collect vast amounts of visual data from the environment (Xaud et al., 2019). So, with the right technique, the desired features can be detected and processed. The capabilities of CNNs (Convolutional Neural Networks) regarding the acquisition of high level features from images are well documented and researched (Bakken et al., 2019) (Ponnambalam et al., 2020). Thus, this paper aims to build on top of a proposed CNN model that predicts the direction which the robot should follow in a combination of fields, conditions and hardware.

The basic navigation system we employ was introduced in a simulated environment (da Costa and Caarls, 2023), containing large amounts of relatively homogeneous data available for training. In contrast, in the current study we investigate the possibility of training a system able to detect the correct direction with a limited amount of highly heterogeneous training data. To suffice this purpose, data taken from different occasions, cameras and two distinct fields, as well as previous data and simulations, were utilized. Hence, the aim is to study the effects of dataset diversity on visual navigation systems, verifying their impact on robustness and adaptability. We explore data variance due to a combination of fields explored, training strategies and hardware changes, while guiding the robot in several use case scenarios.

2 Related Work

The autonomous navigation task is a prerequisite for accomplishing a good automation of many processes that need to happen in a crop field. Even if the task on hand is already being done automatically, driving the vehicle for large amounts of time in a consistent manner is a demanding task (Bai et al., 2023). Extracting the direction of movement from a RGB image has already been proven possible. Ahmadi et al. (2020) and Martins et al. (2021) took advantage from the green color of the plants in contrast with the ground to obtain the path forward with a purely computer vision approach, which however can fail if an unforeseen situation arises.

Deep learning techniques have been also investigated and evaluated. Ponnambalam et al. (2020) and de Silva et al. (2023) implemented a non end-to-end approach, where the Deep Learning Model only produced masks for further use in line detection. Bakken et al. (2019), however, applied an end-to-end approach, where the network directly outputs the desired steering angle. They used a modified VGG16 model in a Polytunnel environment while measuring the impacts of an initial training with a dataset composed of natural trails followed by a refinement training with data from the target field. A positive result in this end-to-end approach indicates a better generalisation for the test case evaluated, when data from different sources were included in the training.

The experiments proposed by Ranftl et al. (2022) also confirmed that mixing data from complementary sources improved a depth estimation task. Taking advantage of 3D movies as datasets, it demonstrated the quality of the generalization in unseen datasets improved when training with multiple mixed datasets, even when converting initially incompatible annotations. Vincent et al. (2023) applied a mixed domain training set ensuring feature diversity. With this approach the pixel accuracy of the segmentation task of martian soil improved considerably when compared to single domain training. The predictions for minority or rare classes also show big improvements with a general conclusion that the mixed domain training strategy is a good tool to improve model capabilities in a multi-mission scenario.

Different from this earlier work, which focuses on improved generalization across different environments, we show that increasing dataset diversity can also improve the performance on a *single* environment. That is, training with data gathered in different tasks and under different conditions often has a higher performance than using only a single task, *even if that is the only task used for testing.*

In a similar way to Bakken et al. (2019), and with promising results by Ranftl et al. (2022), both the refinement of a base training and the mixture of different fields will be analyzed, though this time accounting for more variables such as camera positioning, the presence of a simulated environment, and the model of camera used for obtaining the dataset. The last one being considerably important, since different cameras can present different images of the same environment, regarding its color or lighting, possibly contributing to greater robustness of the model.

3 Methodology

In this section, relevant systems are presented, such as the line detection system and controller. The robots in which the system is analysed, all datasets employed during training and the techniques applied to evaluate the system are also given. Furthermore, for reliable navigation, it is essential to ensure a mostly correct lane detection. Thus, we also present the employed performance analysis method.

3.1 Dataset Description

The presented line detection system will be evaluated in four different image scenarios, each one with a particular set of challenges and features: a soybean field with seedlings and larger plants; images from a built simulation environment; elevated strawberry plants in a polytunnel; and an orchard field composed of different species of trees and shrubs. All datasets are made available for further research at kaggle[1].

All images have a resolution of 640×480 pixels and a color depth of 24 bits, stored in the JPEG format. For each image there is an accompanying file with the labels: line coordinates which the robot should follow, as well as a mask that can be used for segmentation as an auxiliary task if required during training or for applying different detection methods that rely on it. The datasets are summarized, with all relevant subdivisions for the present work, by Table 1, with examples shown in Figure 1.

3.1.1 Soybean Field

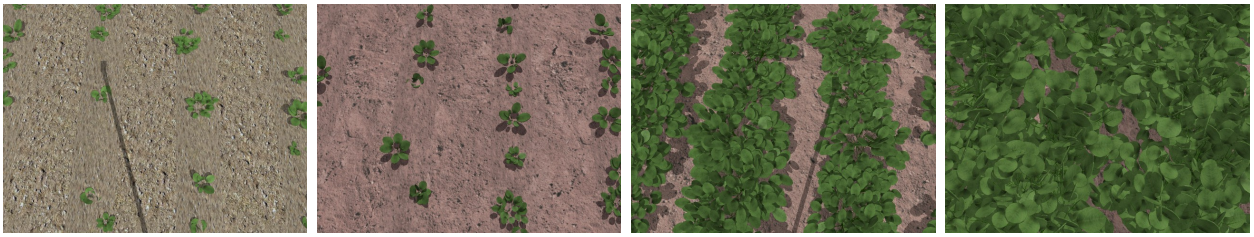
This dataset was acquired during previous tests of the Soybot robot (Martins et al., 2021), using a Logitech C270 camera. The images consist of a top-down view of the field, right in front of the robot, usually with two plant rows and the path between the rows in view. A range of different plant sizes, illumination conditions and a sample of the dataset can be seen in Figure 1a. The main challenge of this dataset is the variety of illumination conditions, soil appearance and ambiguity in choosing the correct lane to follow.

[1] `https://kaggle.com/datasets/d4af9154ac515572ea1686ee3d9f60398c0f2cf369643e563d4fd08f9fc15dea`

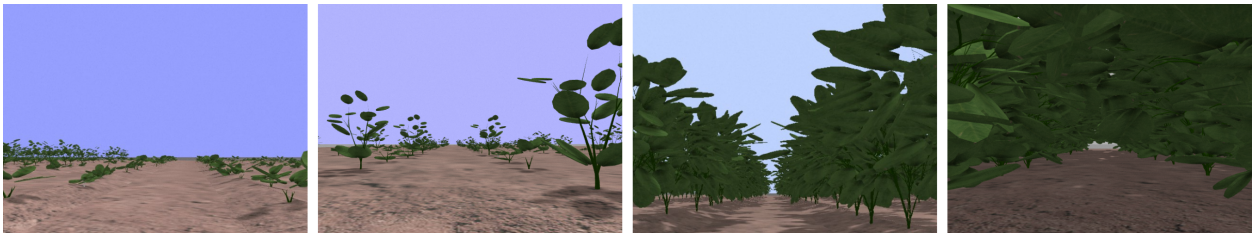
Table 1: Number of data points for each data subdivision explored

Data Group	Soybean Field	Simulated Environment					Strawberry Polytunnel			Orchard Field		
		Top View	Bottom View	Small Plants	Average Plants	Large Plants	Phone Camera	Robot Camera	Empty Tray	Phone Camera	Intel Camera	Lenovo Camera
Training	888	301	299	200	200	200	140	140	140	200	200	200
Validation	362	95	85	60	60	60	40	40	40	60	60	60
Test	152	45	45	30	30	30	20	20	20	30	30	30

(a) Soybot original dataset sample



(b) Sample of the top view from the simulated field dataset



(c) Sample of the bottom view from the simulated field dataset

(d) Strawberry trays dataset sample (3 different cameras)

(e) Orchard field dataset sample (3 different cameras)

Figure 1: Samples of the main dataset groups evaluated

3.1.2 Simulated Environment

In order to develop the proposed system, a simulated environment was built to reproduce a soybean field (da Costa and Caarls, 2023). Two datasets were acquired from this simulation: the first one mimics the top-down view of the soybean dataset and the second one uses a different camera position, closer to the ground. A sample of the top-down view and the near-ground view dataset can be seen, respectively, in Figure 1b and Figure 1c. In the top-down view the biggest challenge in this scenario are missing plants and plant size variation while the bottom view is created to deal with environments where a top-down view is unfeasible or does not provide enough visual information.

The simulated fields can be further divided in different growth stages and here will be evaluated in three different plant sizes: small (Field 1), medium (Field 2) and large (Field 3). The plants' sizes directly influence the capability of the model to detect the correct direction to follow and also allow a performance comparison between each camera position for a given plant size.

3.1.3 Strawberry Polytunnel

This dataset consists of images from two trays of elevated strawberry plants and an empty one, inside a polytunnel. Here the robot is expected to follow the row of plants instead of the path in between the rows, being only one row in the view of the camera.

This images were taken in the same place in two different occasions and using two different cameras. Initially, images were obtained with a Samsung S20 FE smartphone wide view camera and later, while testing, with the robot camera robot itself, an Axtel AX-FHD Webcam Pro. This polytunnel is property of the Norwegian University of Life Sciences and samples from the dataset can be seen in the Figure 1d.

The variation in the available camera hardware brings a new challenge to the line detection system. Color reproduction, field of view and general image clarity is considerably different between samples. This will represent a challenge for the learning model alongside the presence of empty trays, images taken with the robot camera, that the robot should also be able to transpose in order to reach a different plant section.

3.1.4 Orchard Field

In this last environment, also provided by the Norwegian University of Life Sciences, three different cameras were used to obtain images in different occasions. The first one was a smartphone camera and then an Intel Realsense D415i and a Lenovo FHDWC510 webcam were used, due to hardware availability. The robot is now expected to roam in between the rows of trees and small bushes.

The system must be able to detect the path forward with a camera close to the ground, around or below the plant level. This is possible by taking advantage of the visual structure of the environment, composed of two walls of trees, the ground in the middle and the sky on top, seen in the sample shown in Figure 1e. In a similar way to the strawberry dataset, the camera variation imposes a range of different image characteristics. The field is also uneven, with grass that can quickly change appearance and degrade detection quality.

3.2 Algorithm Structure

The complete guiding system has three steps: preprocessing, line detection and line controller. While running in simulation mode, there is a parallel step to the line detection aimed to get a theoretically correct line, using reference points, transforms and a projection. The whole process is described by Figure 2, where each relevant module and its internal steps are shown. As such, in the simulated environment, the robot controller can be evaluated both with the predicted line or with the ground truth line.

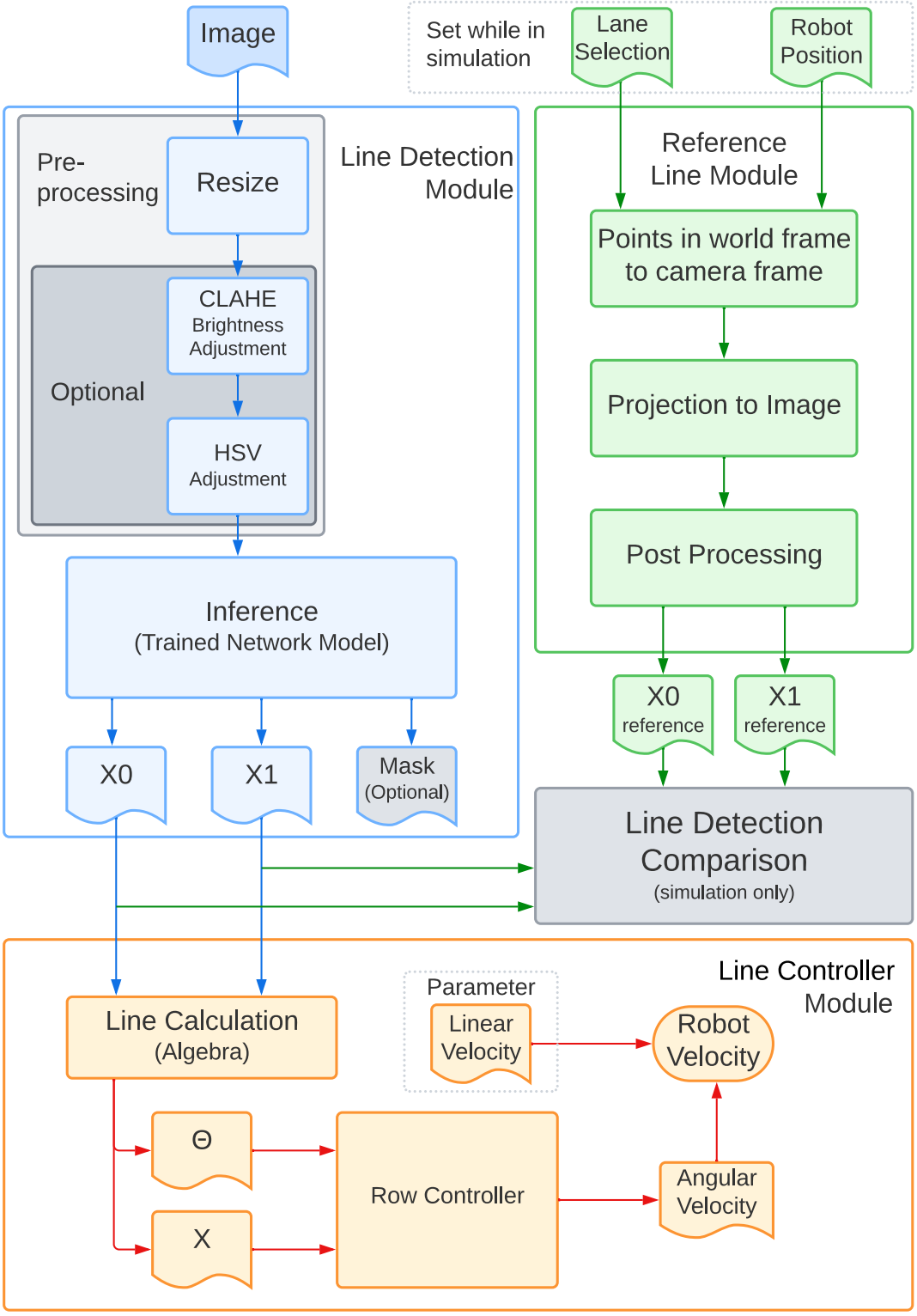


Figure 2: Relevant modules with its basics components: In blue, the line detection module; In green, the module responsible for the reference line in the simulations; In orange, the controller that converts the line data to angular velocities for the robot

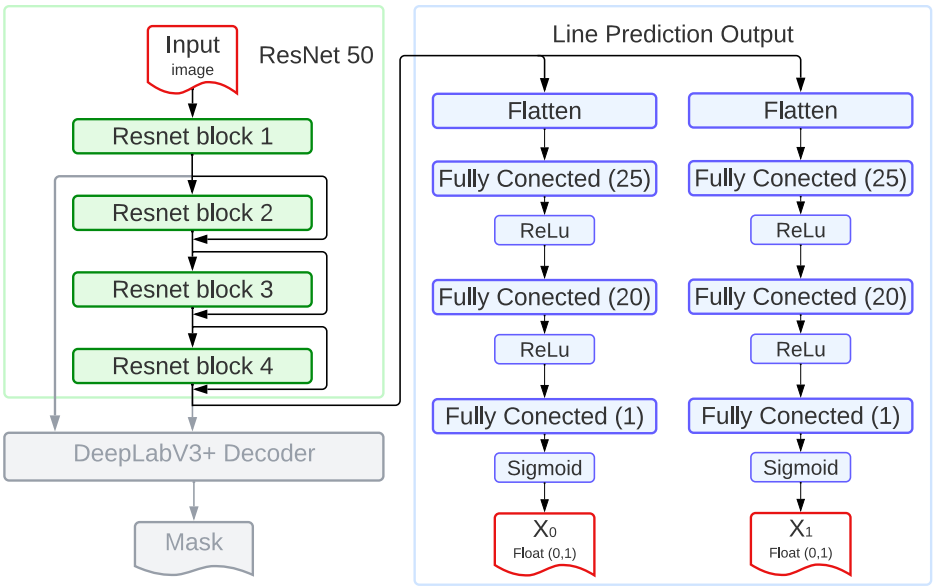


Figure 3: Layer Block Diagram of the model applied with the disabled section grayed out

3.2.1 Line Detection

The presented line detection system is based on a modified version of the DeepLabV3+ model (Chen et al., 2018). The base model evaluated, displayed in Figure 3, has a ResNet 50 (He et al., 2015) backbone for base feature extraction and separate outputs to predict the line. The original mask output of the DeepLab model can be retained as an optional auxiliary task for the training process (da Costa and Caarls, 2023), however it will not be applied in this study.

The line is inferred by the model as two values X_0 and X_1 used to define two points, one at the top and one at the bottom of the input image (see Figure 5). Considering H and W as the images' height and width, respectively, these points are defined as $(X_0, 0)$ and (X_1, H). The X_0 and X_1 output from the model are normalized to $(0, 1)$ while in the image it was defined to range in $(-W, 2W)$ interval, measured in pixels (see the next Section).

This strategy aims at easier implementation and ease of data annotation, by defining the line with only two parameters and allowing lines that start or finish in the vertical borders of the image. While this approach does not cover all possible lines (for example, nearly horizontal ones), those are not expected during normal navigating inside the crop rows, requiring a recovery controller in such events.

3.2.2 Normalization

The X_0 and X_1 values that convey lines which start and finish on the top and bottom of the image, that is the subsection $[0, W]$ of $[-W, 2W]$, are more relevant to the navigation task. This range of values happens often, and as such it is desirable to have a greater section of the $(0, 1)$ interval dedicated to it, with the center of this interval being the most important area as it denotes the target line, mostly vertical and centered.

The sigmoid family of functions has a general shape that matches this objective; it has a higher slope region surrounded by smaller slope regions. Since the line coordinates are defined in pixels, thus a natural number, there is a finite precision available. Defining the symmetry point of the sigmoid curve as its middle, the higher slope near the middle make errors in lines close to the center of the image more relevant to the model.

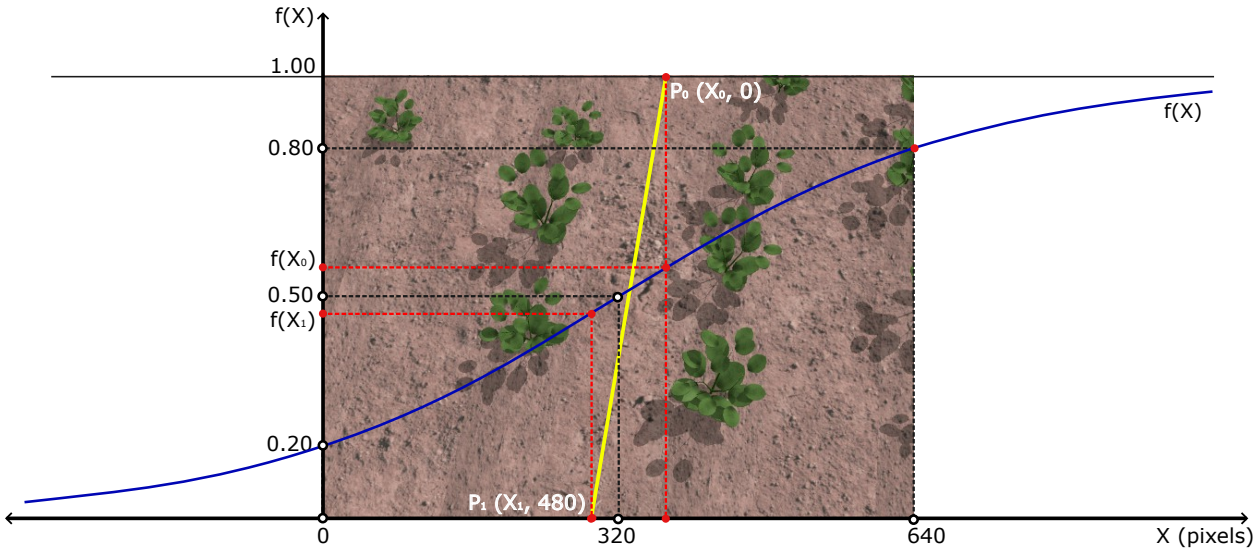


Figure 4: Sigmoid curve applied to normalize the X_0 and X_1 values to $(0, 1)$ interval

This approach can be summarized by the Figure 4, where some meaningful characteristics of the desired function are shown. Let us define the sigmoid curve

$$X_{norm} = \frac{1}{1 + e^{-\frac{X_{pixel} - \mu}{\sigma}}}, \tag{1}$$

where μ is an offset to define the curve center and σ a factor that compresses or dilates the curve in the X axis. In order to achieve the desired output present in Figure 4, an offset of $\frac{W}{2} = 320$ and a suitable σ should be assigned. Applying $\mu = \frac{W}{2}$ to (1) and solving for σ, we obtain

$$\sigma = \frac{\frac{W}{2} - X_{pixel}}{\ln\left(\frac{1}{X_{norm}} - 1\right)}. \tag{2}$$

From Figure 4, when $X_{pixel} = 0$, the desired output is $X_{norm} = 0.2$. Applying these values to (2), the desired σ can be calculated. For the chosen values, $\sigma = 231$.

In order to obtain the controller input from the model output, the normalization process needs to be reversed. Solving the sigmoid equation, (1), for X_{pixel}, we obtain

$$X_{pixel} = \mu - \sigma \ln\left(\frac{1}{X_{norm}} - 1\right), \tag{3}$$

after which the X and θ values used in (8) are given by

$$X = X_1 - \frac{W}{2} \tag{4}$$

$$\theta = \arctan\left(\frac{X_1 - X_0}{H}\right). \tag{5}$$

3.2.3 Line Controller

The line controller is an updated version of the previously implemented controller, described initially by Cherubini et al. (2008), and developed to be used in the Soybot II mobile robot (Barbosa, 2022). The controller sets the desired angular velocity, ω_d, given the input line. It expects the robot to move with a constant linear velocity, v_d, and receives two parameters defining the line: a linear offset, X, from the vertical and an angular offset, θ, also from the vertical, as shown in Figure 5. The target of this control scheme is to achieve a desired configuration of $X = \theta = 0$. When this desired configuration is achieved, the robot is expected to follow the intended path detected in the image.

Given the interaction matrix $L_S = \begin{bmatrix} L_x & L_y & L_\theta \end{bmatrix}^T$, the velocity transform from the robot frame to the camera frame $T_R = \begin{bmatrix} T_v & T_w \end{bmatrix}^T$ and applying the problem constraints detailed by Cherubini et al. (2008), the system equations are defined by

$$\begin{bmatrix} \dot{X} \\ \dot{\theta} \end{bmatrix} = A_r v + B_r w, \tag{6}$$

where

$$A_r = \begin{bmatrix} L_x \\ L_\theta \end{bmatrix} T_v \qquad\qquad B_r = \begin{bmatrix} L_x \\ L_\theta \end{bmatrix} T_w. \tag{7}$$

The selected control law is

$$\omega_d = -B_r^\dagger \left(\begin{bmatrix} \lambda_x X \\ \lambda_\theta \theta \end{bmatrix} + A_r v_d \right), \tag{8}$$

being λ_x and λ_θ positive gains, $\dagger$ is the Moore-Penrose pseudo inverse and $B_r \neq 0$.

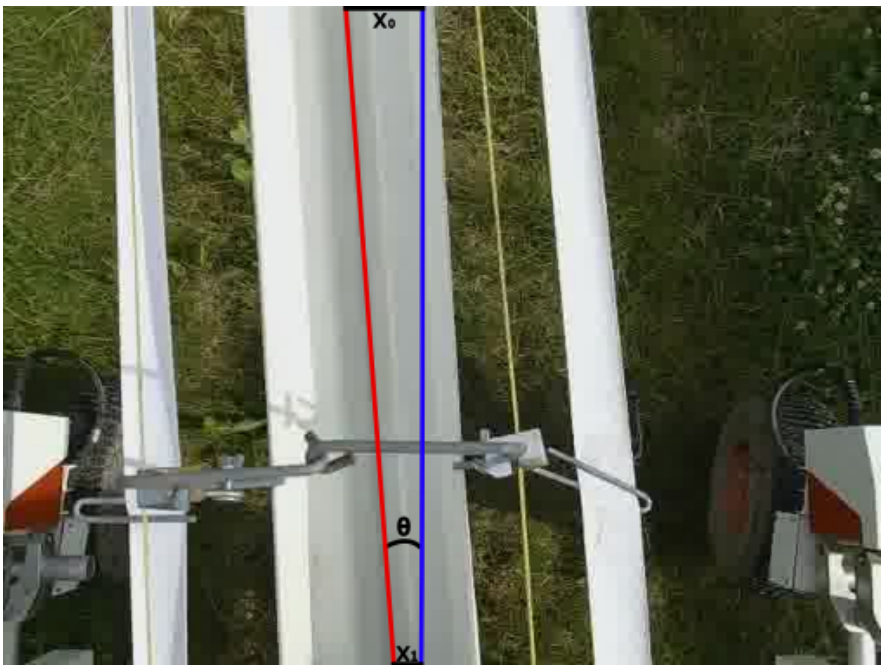


Figure 5: Line that should be detected by the model (red) and vertical reference (blue), with the model outputs shown (X_0 and X_1), and the derived θ offset

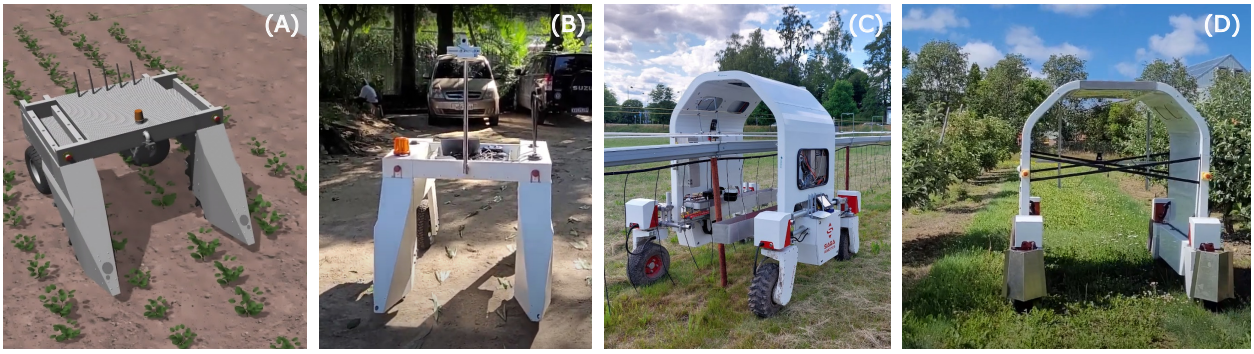


Figure 6: (a) Simulated Model of the Soybot-II robot; (b) Soybot-II Robot; (c) Polytunnel Thorvald in an empty and open tray; (d) Reinforced Thorvald in one lane of an orchard field

3.3 Robots Evaluated

Three different robots were evaluated in different environments. The first one, shown in Figure 6a, was used in the simulated environment, and is a model of the second version of the Soybot robot (Oliveira et al., 2019), developed by the Pontifical Catholic University of Rio de Janeiro in cooperation with Solinftec (Martins et al., 2021), shown in Figure 6b. In the simulation environment, the Soybot robot evaluation occurs with two different camera positions, the original top-down configuration and the camera near the ground.

The second and third robots are both Thorvald robots (Grimstad and From, 2017) built by Saga Robotics. The robot number two was deployed in a strawberry plantation tunnel with a camera mounted in a top-down view of the target crop tray. This Thorvald model is known as the polytunnel variant and can be seen in one of the target fields in Figure 6c. The last robot is a modified version, reinforced and wider, for larger fields with possibly taller plants or grass, such as wheat and orchard fields. This robot will navigate through an orchard field with cameras mounted below the top of the trees, looking forward, as shown in Figure 6d.

3.4 Test Methodology

The first evaluation targets the capability of the system to correctly detect the direction line in each obtained dataset. Here, the impact of the different cameras used across the plantation fields and the variation in camera position is explored to determine the best set of data to be used during the training for each scenario.

To evaluate the line, the percentile average of the absolute error of X_0 and X_1, relative to the image width, will be taken into account. With the combined error of each image, the average can help judge the overall performance of the model in the target test dataset. Figure 7 shows four examples, three of them demonstrating incorrect lines and the types of errors and its approximated value for each one observed.

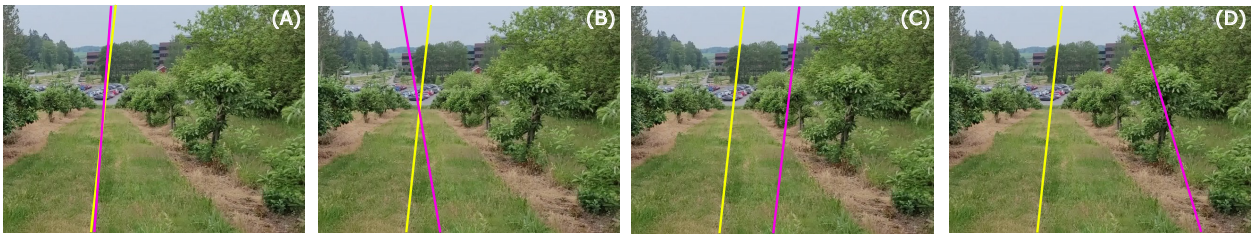


Figure 7: Detection example and reference line, pink and yellow, respectively: (a) Good Detection (error 1,4% of image width); (b) Error in θ (10,7%); (c) Error in X (17,9%); (d) Error in θ and X (39,4%)

This metric was chosen for its simplicity and good response to errors in both inputs of the controller, X and θ. However, Figure 7 also shows that even some detection error can still point in an overall correct direction. As such, evaluating robot performance is the final goal.

The simulated environment allows access to the entire system performance since it is possible to obtain the true position of the robot and lanes precisely. This will be employed while evaluating the impact of the different plant sizes and camera positioning on the performance, according to images presented during the training phase. In those comparisons, the performance is evaluated as the success rate of the crossing without going outside of the lane, and the average error from the center of the path during a successful crossing. This way, the impact of the lane detection system precision can be observed in the system performance.

Finally, we also use an indirect evaluation of the system performance in the field. With one of the trained models, the proposed robot will be tested in the field and its performance analyzed by segmenting, manually and blindly, the camera image registered during the crossing. This way, a comparison is mode with what a theoretical human remote operator would consider correct.

3.5 Training Setup

The model training was conducted for up to 300 epochs, with early stopping set up for 50 epochs without improvement, and the ADAM optimizer with a learning rate of 10^{-5} was applied. As both X_0 and X_1 are numerical real values, the loss function chosen is the sum of the mean absolute error for each output. Three data augmentation techniques were applied, each with an independent 50% chance. The image can be flipped horizontally, have its brightness increased or decreased randomly, up to 25%, and also its contrast in the same way. The weight values of the last epoch or from the early stopping is kept for further use.

The input images are resized, from 640×480 to 256×256, for performance reasons both during the training phase and execution phase. It is also important to note that the image is not cropped, just resized, which causes a small aspect ratio distortion. Other image sizes were evaluated previously (da Costa and Caarls, 2023), with 256×256 displaying good results overall.

The dataset mixing is done within same data in the same group, training, validation or test data. This way, a image from the test group will always be in the test group of mixed variants. The Table 2 lists the mixed datasets created from Table 1 data, in this process no image was removed or added.

4 Results and Discussion

In this section, as different datasets are being contrasted, a method to compare the expected difficulty of each dataset is also provided. For all tables present in this section, the Center Line entry defines the error measured if all the predictions of the line detection system were a vertical centralized line. Thus, it can also be interpreted as the average error from the vertical of the lines present in the dataset, with a higher value being a more varying, and possibly tougher, dataset.

Table 2: Number of data points for each additional mixed data group explored

Data Group	Soybean Field with Sim	Simulated Environment Mixed	Strawberry Polytunnel Mixed	Orchard Field Mixed	All Data Mixed
Training	1488	600	420	600	3108
Validation	542	180	120	180	842
Test	242	90	60	90	482

Table 3: Cross test error for each camera in the orchard field. Best results are bold highlighted

Train Data \ Test Data	Intel camera	Lenovo camera	Phone camera	Mixed camera
Intel camera	**3.3**	9.5	16.5	9.9
Lenovo camera	9.7	5.3*	15.3	10.4
Phone camera	11.0	7.8	8.5*	9.6
Mixed camera	3.4*	**4.8**	**7.9**	**5.4**
Center Line	11.6	7.4	12.4	11.0

*Statistically equivalent within 95% confidence interval of the best result.

4.1 Line detection assessment for different cameras across the orchard datasets

In the first analyzed scenario, the datasets extracted from the orchard field, one training scenario was performed with each camera and evaluated using the other cameras as test data. One extra group was created, being a combination of all cameras images. Table 3 shows the observed error for each combination between training and test.

An essential point is the model's ability to surpass the center line reference value when trained and tested in the same dataset, as expected. Some weak generalization between cameras can be seen in the results, but interestingly the best results were obtained when training with all cameras combined. It is well known that increasing dataset diversity helps generalization across different scenarios, but our results indicate that it also increases robustness in a single scenario. Rather than degrading performance, adding out-of-test-distribution samples to the training set increases it.

4.2 Line detection comparison for tray occupancy and camera change in the strawberry dataset

In strawberry dataset, in addition to using different cameras, a comparison is made by analyzing the system with an empty tray. The tray can only be seen occasionally in the general dataset, but other items, such as the soil in the background and the side tapes from the tray, are always visible and comparable. Table 4 shows, in a similar way as before, the results obtained for the cross test run with these groups of data.

As can be seen from Table 4, the model was always able to get good results in its own test group and good generalization can be seen from the phone to the robot camera dataset. The model trained only with the empty tray has the weaker results of this set, while joining all the images from different sources and situations again gave the best overall results. Mixing empty and full tray images also improves the performance over using only full-tray images, even though the empty-tray images are even further out of the test distribution than using a different camera.

Table 4: Cross test error for each camera in the strawberry polytunnel. Best results are bold highlighted

Train Data \ Test Data	Phone camera	Robot camera	Empty Tray	Mixed Images
Phone camera	5.1	4.8	17.5	9.1
Robot camera	18.1	4.4*	22.9	15.1
Robot camera empty tray	18.2	12.1	**2.7**	11.0
Robot camera mix empty/full tray	13.6	3.12*	3.6	6.8
Mixed Images	**4.2**	**2.9**	2.9*	**3.3**
Center Line	14.3	16.2	13.7	14.7

*Statistically equivalent within 95% confidence interval of the best result.

Table 5: Error comparison of the mixed trained model with the dedicated model for each camera. Best results are highlighted in bold

Train Data \ Test Data	Bot sim Dataset	Top sim Dataset	Soybot Dataset
Top Sim Dataset	73.5	4.9	10.3
Bot Sim Dataset	10.4*	8.0	12.1
Mixed Sim	10.3*	4.8*	10.0
Soybot Dataset	29.4	4.6	5.7*
Soybot and Sim	**10.3**	**3.7**	**5.3**
Center Line	12.1	8.7	10.6

*Statistically equivalent within 95% confidence interval of the best result.

4.3 Line detection evaluation for camera placement in the simulated field

Having shown the benefit of mixing data of different cameras and different environments (presence of strawberry plants in the tray or not), we now treat different camera positions. Using a simulated soybean field, Table 5 shows the system performance for two camera positions: from the top or near the bottom of the robot. In this case, training with a mixed dataset is not significantly different, although it has a bias towards slightly better results.

However, mixing simulated data with real-world data from the Soybot robot does significantly improve the results on the top camera dataset. This is to be expected, as the Soybot data was gathered with a top camera. At the same time, it does not decrease the performance on the bottom camera. Again, mixing datasets from different scenarios is often beneficial, and never detrimental.

4.4 Line detection and controller performance assessment in the simulated environment

The simulation is also able to give a measure of the whole system performance, and verify if this performance observed in the test group could manifest in the field. For this test, we always train with mixed top and bottom camera data.

The results shown in Table 6 state the inability of the model to correctly guide the robot with a top-view of field 3 for any training set, as expected due to the size of the plants. In a similar manner, the model trained with the data from field 3 is the worst at guiding the robot from this perspective. Meanwhile, the model trained with data from field 2 only excels at its own field, just failing twice. And, lastly, the model trained with data from field 1 only achieves 6% and 12% in its best case scenario.

Table 6: Average center line deviation, in mm, for successful crossings with its associated fail rate in parenthesis for 100 crossings, in each field and camera, when trained with images from a given field

Train Data \ Test Field	Field 1 Bot View	Field 1 Top View	Field 2 Bot View	Field 2 Top View	Field 3 Bot View	Field 3 Top View
Mixed view Field 1	36 (6)	21 (12)	76 (43)	134 (92)	52 (68)	- (100)
Mixed view Field 2	73 (79)	- (100)	21 (2)	65 (56)	48 (48)	- (100)
Mixed view Field 3	83 (70)	- (100)	25 (1)	- (100)	18 (**0**)	- (100)
Mixed Fields	**15** (**0**)	48 (5)	**16** (**0**)	42 (**4**)	**16** (**0**)	**49** (**90**)
Multi-ROI	- (100)	**17** (**1**)	- (100)	**38** (24)	- 100	87 (92)
Ground truth line	9 (0)	24 (0)	9 (0)	23 (0)	9 (0)	22 (0)

The model trained with data from all fields was able to perform well in all scenarios, with the trained network able to learn from both camera positions in each field growth stage, again outperforming training on the specific field being tested. The Multi Region of Interest algorithm, previously deployed in the Soybot robot, was able to perform well in the first field, as it was designed to, but started to fail with larger plants. The same result was observed in real-world field tests (data not shown).

A crossing success rate above 95% with an average position deviation smaller than 50 millimeters is obtainable for all but one scenarios. In this one, Field 3 Top View, the poor performance is expected, as there is a lack of direction information in the image due to the plant size. Controlling using the ground truth line shows that the chosen controller gains lead to a naturally higher error in the top-down view.

4.5 Training strategy comparison for line detection performance

Instead of mixing datasets, another strategy is to perform fine-tuning, further training the model to be applied in the robot with the target dataset on top of a base model previously trained in a similar scenario. Table 7 compares the model results across three different methods: training the model from the ground up on the target dataset, mixing the data or performing fine-tuning. In this last method, the model is first trained on the Soybot dataset, with added simulations, and then further trained with the target dataset.

The fine-tuning shifted the focus of the model, improving performance on the target dataset to the detriment of performance in the other scenarios. However, fine-tuning shows consistently better results than only training on the target dataset. As expected from the previous results, mixing the training data gave statistically similar results to fine-tuning, but with the added advantage of generalization across scenarios. Even though in this case the scenarios are vastly different, the increased dataset diversity consistently leads to lower errors (although not statistically significant).

4.6 Real world robot performance experiments

The Thorvald robots were deployed in two of the scenarios discussed previously, the strawberry polytunnel and the orchard field, both with networks trained for the target field. In both cases the robot crossing was successful[2] and, by blindly annotating the output image from the camera, it is possible to compare the output line of the model with the expected one by a human annotator, which would guide the robot correctly.

[2] https://www.youtube.com/playlist?list=PL521NS6JaNgtC2N_2eOBkHiA7b_sOOVca

Table 7: Error from training strategy comparison table. Best results are highlighted in bold

Test Data / Train Data	Soybot Dataset	Strawberry Dataset	Orchard Dataset
Soybot + Sim Model	5.1*	15.8	11.0
Orchard	11.2	14.5	5.4
Fine-tune on Orchard	7.8	16.1	4.5*
Strawberry	12.3	3.3	15.3
Fine-tune on Strawberry	10.1	2.4*	26.2
Full Mixed Training	**4.9**	**2.3**	**4.3**
Center Line	10.6	14.7	11.0

*Statistically equivalent within 95% confidence interval of the best result.

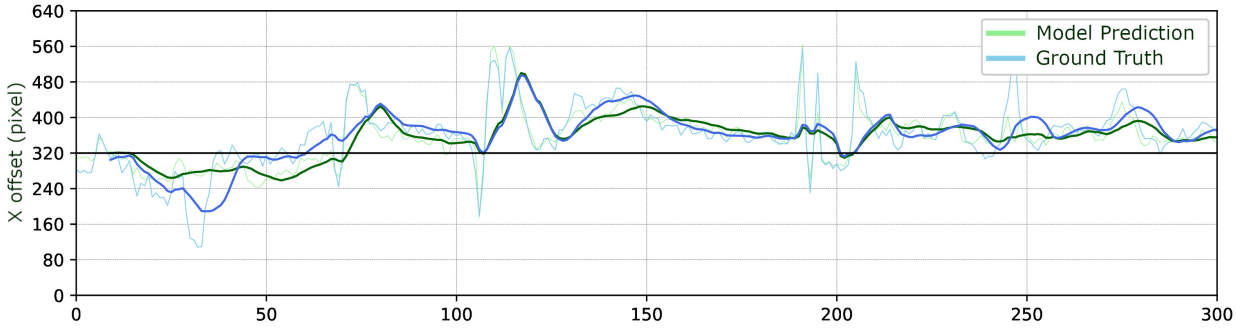


Figure 8: Comparison of the X value from the detected line by the network and a manual (blind) defined line, green and blue, respectively. Darker lines were drawn as an average of 10 values in a rolling window for easier visualization.

Figure 9: Examples of the predicted line of the model and the line drawn by a human, pink and yellow, respectively, showing the error observed in a off center view

4.6.1 System performance evaluation in the orchard field

The network output error in the orchard fields shown in Figure 8. This field also proves that the bottom camera placement works in a real orchard environment, although the orchard has a more challenging landscape view and the trees are further apart.

The predicted line followed well the expected line by the human segmentation. However, in some instances, it presented a conservative behavior toward extreme angles and positions, as seen in Figure 9. This event had the same effect as a dampening in the output, since the controller output still had the correct direction, demanding a low speed operation.

4.6.2 System performance evaluation in the strawberry field

For the strawberry polytunnel environment, the error in the line detection, in one of the crossings, can be seen in Figure 10. Even though the robot crossed the field successfully, a small deviance can be observed in the graph. This offset in the detection made the robot run off center, as shown in Figure 11. The trained model presented a bias toward keeping the detected line in the center of the image, only moving it further while away from the center.

From both observed scenarios, camera alignment also played an important role. It impacted in the quality of the observed crossing, being enough to make the robot also be off center in certain conditions. This way, it is essential to ensure robustness in the camera mount, as it can become loose in operation and lead to a non ideal performance over time.

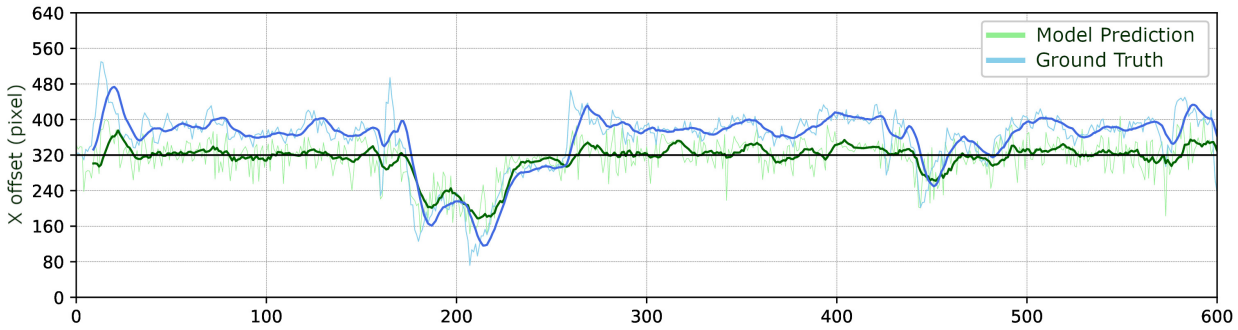


Figure 10: Comparison of the X value from the detected line by the network and a manual (blind) defined line, green and blue, respectively. Darker lines were drawn as an average of 10 values in a rolling window for easier visualization.

Figure 11: Lateral shift observed while crossing the strawberry polytunnel

5 Conclusion

In this work we showed the benefits of dataset diversity in training a navigation system for crop-row following. In particular, we show that mixing data from different conditions (camera models, camera positioning, crop presence and growth stages) and even completely different scenarios (orchard and strawberry polytunnels) is never detrimental and often beneficial. We therefore advise practitioners to not only gather as much data as possible from the target scenario, but also mix in other data sets. Changes made to the robotic system, such as changing the camera model or position, do not invalidate datasets gathered before, but may even enhance performance. Increased dataset diversity not only increases generalization across scenarios, but also accuracy and robustness in a single target environment.

Seasonal effects in the appearance of the crop was not considered in this work, such as the possibility of snow in the winter, although we can expect it to behave similarly as two different fields. In the same way as evaluated here, mixing the data across the whole season cycle should improve the system altogether. When deploying the system in a completely new environment, even data collected by hand can be useful for training, and then subsequent crossings of the robot can supply more data for better robustness and reliability.

Simulation and real world tests validated that the trained network output can be successfully used in a robot controller to successfully follow crop rows, whether they be the row itself, as in the strawberry scenario, or the path between the rows, as in the soybean and orchard scenarios.

Future Work

As noted previously, season long tests could not be performed due to time constraints, and this factor still requires investigation and measurement of its impacts. The current assumption is that the robot is already inside the desired row, however a maneuver is needed to engage in the next row as it reaches the end of it. Current strategies for this maneuver still rely in open loop control or, at least, simple position based control. Hence, developing strategies for a vision based end-of-row maneuver is still needed.

Further developments are also possible in the robot line controller, improving its response and output stability in uneven and slippery terrain. Auxiliary systems should also be added to aid in the detection of people in the robot path, for security reasons, and fail safe conditions to minimize crop damage in case of failure, be it software or hardware related.

Acknowledgments

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001; The National Council for Scientific and Technological Development – CNPq under project number 314121/2021-8; Fundação de Apoio a Pesquisa do Rio de Janeiro (FAPERJ) - APQ1 Program - E-26/010.001551/2019; and The Norwegian Directorate for Higher Education and Skills (HK-dir) through the project UTFORSK Enabling Technologies and People for Next-Generation Precision Agriculture (EnTechAgri), project number UTF-2021-10160.

References

Ahmadi, A., Halstead, M., and McCool, C. (2021). Towards autonomous visual navigation in arable fields. *International Conference on Intelligent Robots and Systems (IROS)*.

Ahmadi, A., Nardi, L., Chebrolu, N., and Stachniss, C. (2020). Visual servoing-based navigation for monitoring row-crop fields. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4920–4926. IEEE.

Aravind, K. R., Raja, P., and Pérez-Ruiz, M. (2017). Task-based agricultural mobile robots in arable farming: A review. *Spanish Journal of Agricultural Research*, 15(1):e02R01.

Bai, Y., Zhang, B., Xu, N., Zhou, J., Shi, J., and Diao, Z. (2023). Vision-based navigation and guidance for agricultural autonomous vehicles and robots: A review. *Computers and Electronics in Agriculture*, 205:107584.

Bakken, M., Moore, R. J., and From, P. (2019). End-to-end learning for autonomous crop row-following. *IFAC-PapersOnLine*, 52(30):102–107.

Barbosa, G. B. P. (2022). Robust vision-based autonomous crop row navigation for wheeled mobile robots in sloped and rough terrains. Dissertação de mestrado, Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Engenharia Elétrica. v., 67 f: il. color. ; 30 cm.

Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., and Adam, H. (2018). Encoder-decoder with atrous separable convolution for semantic image segmentation.

Cherubini, A., Chaumette, F., and Oriolo, G. (2008). An image-based visual servoing scheme for following paths with nonholonomic mobile robots. In *2008 10th International Conference on Control, Automation, Robotics and Vision*, pages 108–113. IEEE.

da Costa, I. F. and Caarls, W. (2023). Crop row line detection with auxiliary segmentation task. In *Lecture Notes in Computer Science*, page 162–175. Springer Nature Switzerland.

de Silva, R., Cielniak, G., Wang, G., and Gao, J. (2023). Deep learning-based crop row detection for infield navigation of agri-robots. *Journal of Field Robotics*.

Grimstad, L. and From, P. (2017). The thorvald ii agricultural robotic system. *Robotics*, 6(4):24.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep residual learning for image recognition.

Martins, F. F., Carvalho, T. M., Celecia, A., Oliveira, A. I. S., Barbosa, G. B. P., Vellasco, M. M. B., Caarls, W., Figueiredo, K., and Leite, A. C. (2021). Sistema de navegação autônoma para o robô agrícola soybot. In *Procedings do XV Simpósio Brasileiro de Automação Inteligente*, SBAI2021, pages 701–707. SBA Sociedade Brasileira de Automática.

Oliveira, A. I. S., Carvalho, T. M., Martins, F. F., Leite, A. C., Figueiredo, K. T., Vellasco, M. M. B. R., and Caarls, W. (2019). On the intelligent control design of an agricultural mobile robot for cotton crop monitoring. In *2019 12th International Conference on Developments in eSystems Engineering (DeSE)*, pages 563–568. IEEE.

Ponnambalam, V. R., Bakken, M., Moore, R. J. D., Gjevestad, J. G. O., and From, P. J. (2020). Autonomous crop row guidance using adaptive multi-ROI in strawberry fields. *Sensors*, 20(18):5249.

Ranftl, R., Lasinger, K., Hafner, D., Schindler, K., and Koltun, V. (2022). Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(3):1623–1637.

Vincent, G., Yepremyan, A., Chen, J., and Goh, E. (2023). *Mixed-Domain Training Improves Multi-mission Terrain Segmentation*, page 96–111. Springer Nature Switzerland.

Xaud, M. F. S., Leite, A. C., and From, P. J. (2019). Thermal image based navigation system for skid-steering mobile robots in sugarcane crops. In *2019 International Conference on Robotics and Automation (ICRA)*. IEEE.