machines

Article

Local Motion Planner for Autonomous Navigation in Vineyards with a RGB-D Camera-Based Algorithm and Deep Learning Synergy

Diego Aghi [1,2,*], Vittorio Mazzia [2,3,4] and Marcello Chiaberge [2,3]

1 Department of Environment, Land and Infrastructure Engineering, Politecnico di Torino, 10129 Turin, Italy
2 Politecnico di Torino Interdepartmental Centre for Service Robotics (PIC4SeR), 10129 Turin, Italy; vittorio.mazzia@polito.it (V.M.); marcello.chiaberge@polito.it (M.C.)
3 Department of Electronics and Telecommunications, Politecnico di Torino, 10129 Turin, Italy
4 SmartData@PoliTo—Big Data and Data Science Laboratory, 10129 Turin, Italy
* Correspondence: diego.aghi@polito.it

Received:20 April 2020; Accepted: 22 May 2020; Published: 25 May 2020

Abstract: With the advent of agriculture 3.0 and 4.0, in view of efficient and sustainable use of resources, researchers are increasingly focusing on the development of innovative smart farming and precision agriculture technologies by introducing automation and robotics into the agricultural processes. Autonomous agricultural field machines have been gaining significant attention from farmers and industries to reduce costs, human workload, and required resources. Nevertheless, achieving sufficient autonomous navigation capabilities requires the simultaneous cooperation of different processes; localization, mapping, and path planning are just some of the steps that aim at providing to the machine the right set of skills to operate in semi-structured and unstructured environments. In this context, this study presents a low-cost, power-efficient local motion planner for autonomous navigation in vineyards based only on an RGB-D camera, low range hardware, and a dual layer control algorithm. The first algorithm makes use of the disparity map and its depth representation to generate a proportional control for the robotic platform. Concurrently, a second back-up algorithm, based on representations learning and resilient to illumination variations, can take control of the machine in case of a momentaneous failure of the first block generating high-level motion primitives. Moreover, due to the double nature of the system, after initial training of the deep learning model with an initial dataset, the strict synergy between the two algorithms opens the possibility of exploiting new automatically labeled data, coming from the field, to extend the existing model's knowledge. The machine learning algorithm has been trained and tested, using transfer learning, with acquired images during different field surveys in the North region of Italy and then optimized for on-device inference with model pruning and quantization. Finally, the overall system has been validated with a customized robot platform in the appropriate environment.

Keywords: agricultural field machines; stereo vision; deep learning; autonomous navigation; edge ai; transfer learning

1. Introduction

Nowadays, with the continuous growth of the human population, agriculture industries and farmers have been facing the exponential augmentation of global demand of food production. According to the projections of growth established in 2017 by the United Nations [1], by 2050, the global population will be around 9.7 billion and it is expected to reach 11.1 billion in 2100. So, there is an incremental need of new techniques and technologies aimed at maximizing efficiency and productivity of every single land sustainably.

Over the years, precision agriculture [2] and digital farming [3] have gradually contributed with autonomous robotic machines and information collection to improve crop yield and resource management, to reduce the labor costs, and in part, to increase the production efficiency. This has led to equip harvesting machineries with driverless systems in order to maximize the navigation efficiency by reducing the number of intersections in the path, and therefore, the amount of fuel consumed [4]. Once endowed with the appropriate effectors, these robotic vehicles can harvest [5,6], spray [7–9], seed [10] and irrigate [11], and collect trees and crops data for inventory management [12–14]; when configured as platforms, they can carry laborers to prune and thin trees, hence reducing inefficiencies and injuries in the workplace [15]. Research on applications of mobile robotic systems in agricultural tasks has been increasing vastly [16]. However, despite the rising in investments and research activities on the subject, many implementations remain experimental and far from being applied on a large scale. Indeed, most of the proposed solutions require a combination of real-time kinematic GPS (RTK-GPS) [17,18] and costly sensors like three-dimensional multi-channel Light Detection and Ranging (LIDAR) [19]. Other than being very expensive, those solutions are unreliable and prone to failure and malfunction due to their complexity.

On the other hand, several recent works [20,21] focus their efforts on finding an affordable solution for the generation of a global map with related way-points. However, path following inside vineyard rows is still a challenging task due to localization problems and variability of the environment. Indeed, GPS receivers require to function in an open area with a clear view of the sky [22], hence, expensive sensors and solutions are needed in order to navigate through vineyards rows and follow the generated paths.

In this context, we present a low-cost, robust local motion planner for autonomous navigation in vineyards trying to overcome some of the present limitations. Indeed, our power-efficient solution makes use only of RGB-D camera without involving any other expensive sensor such as LIDAR or RTK-GPS receivers. Moreover, we exploit recent advancements in the Deep Learning [23] techniques and optimization practices for Edge AI [24] in order to create an overall resilient local navigation algorithm able to navigate inside vineyard rows without any external localization system. The machine learning model has been trained using transfer learning with images acquired during different field surveys in the North region of Italy and then we validated the navigation system with a real robot platform in the relevant environment.

2. Related Works

As far as autonomous navigation is concerned, classic autonomous systems capable of navigating a vineyard adopt high-precision RTK-GPS [25–28] or by the use of laser scanners combined with GPS [29,30]. However, the lack of GPS availability due to environmental conditions such as large canopies, the need for prior surveying of the area, and unreliable connectivity in certain scenarios make GPS-free approaches desirable [31]. On the other hand, more modern and recent approaches employ different types of sensors usually combined with each other [19]. For example, Zaidner et al. introduced a data fusion algorithm for navigation, which optimally fused the localization data from various sensors; GPS, inertial navigation system (INS), visual odometry (VO) and wheel odometry are fused in order to estimate the state and localization of the mobile platform [32]. However, as highlighted by the authors, there is a trade-off between cost and accuracy, and the data fusion algorithm could fail if each sensor highly differs from each other.

Regarding affordable and low-cost solutions, Riggio et al. proposed a low-cost solution based only on a single-channel LIDAR and odometry, but it is greatly affected by the type of canopy and condition of the specific vineyard [33]. Instead, in [16], they proposed a vision based-control system using a clustering algorithm and Hough Transform in order to detect the central path between two rows. However, it is extremely sensitive to illumination conditions and intra-class variations.

On the other hand, the emerging needs of automation in the agricultural production systems and in the crop life cycles, concurrent to the unstoppable expansion towards new horizons of the deep learning, led to the development of several architectures for a variety of applications in precision agriculture. For instance, in [34,35], authors proposed solutions based on known architectures such as AlexNet [36], GoogleNet [37] and CaffeNet [38] to detect diseases in plants and leaves respectively. Moreover, deep learning has been used for crop type classification [39–42], crop yield estimation [43–46], fruit counting [24,47,48], and even to predict the weather, forecasting temperature and rainfall [49].

We propose a novel approach based only on a consumer-grade RGB-D camera and latest innovations in the machine learning field to create a robust to noise local motion planner. The algorithm that exploits the depth map produces a proportional control and is supported, in case of failure, by a deep learning model that produces high-level primitives [50]. Moreover, the strict synergy between the two system blocks opens the possibility to easily create an incremental learning architecture where new labeled data coming from the field are used to extend the existing capabilities of the machine learning model.

The remainder of this paper is organized as follows. Section 3 introduces the materials and data used for this research. Sections 4 and 5 give a detailed overview of the proposed methodology with the obtained experimental results followed by the conclusion and future works.

3. Materials and Data

In order to acquire a dataset for training and testing the deep neural network, we performed field surveys in two distinct rural areas in the North part of Italy; Grugliasco near the metropolitan city of Turin in the Italian region of Piedmont and Valle San Giorgio di Baone in the Province of Padua in the Italian region Veneto. The collected video samples present different types of terrains, wine quality, and they were acquired at a different time of the day, with diverse meteorological conditions. Videos were shot at 1080p with a 16:9 ratio in order to have more flexibility during the data processing process.

On the other hand, to acquire images and compute the depth map on the platform, we employed the stereo camera Intel RealSense Depth Camera D435i (https://www.intelrealsense.com/depth-camera-d435i/). It is a vision system equipped with an RGB camera and two infrared cameras which computes the depth of each pixel of the acquired frame up to 10 m.

Finally, for the practical in-field evaluations, the stereo camera has been installed on an unmanned ground vehicle (UGV): the model Jackal from Clearpath Robotic (https://clearpathrobotics.com/jackal-small-unmanned-ground-vehicle/) endowed with an Intel Core i3-4330TE. The camera mounted on the chosen robotic platform is depicted in Figure 1.

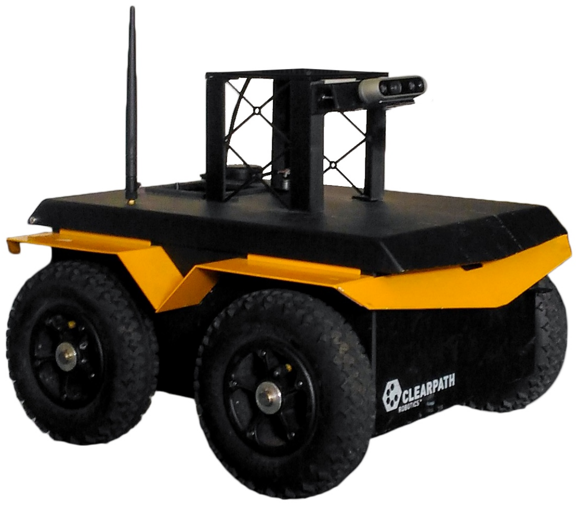

Figure 1. UGV Jackal from Clearpath Robotics endowed with an Intel RealSense D435i.

4. Proposed Methodology

Our goal is to develop a real-time local motion planner with an ultra-light computational load able to overcome practical problems faced by the GPS device when carrying out an autonomous navigation along a vineyard row.

The workflow of our proposal is the following: first, the stereo camera acquires the frames with the RGB camera and, simultaneously, it provides a depth map computed through the two infrared cameras. Successively, a light depth-map-based algorithm processes the depth maps detecting the end of the vineyard row and, consequently, it calculates the control values with a proportional controller on both linear and angular velocities. Unfortunately, in particular weather and lightning conditions, the depth map generation is unreliable and prone to error. Indeed, as in many outdoor applications, the sunlight influences negatively the quality of the results and compromises the control given by the local navigation algorithm. To face this problem, as a back-up solution, we implemented a Convolutional Neural Network (CNN) trained at classifying whether the camera is pointing at the center of the end of the vineyard row or at one of its sides. Once an output prediction is obtained, we can route the path of the robot properly to avoid collisions with the sides of the vineyard. Moreover, we exploited the latest advancement in model optimization techniques in order to obtain an efficient and lightweight neural network able to inference in real-time on a low-cost, low-power device with limited computational capabilities. The overall algorithm pseudo-code is reported in Figure 1. We integrated the proposed algorithm with the open-source Robot Operating System (ROS) to apply the generated control to the actuators of the selected UGV. Finally, to prevent the robot platform from colliding with unexpected obstacles that obstruct its way, we use the depth-map provided by the stereo camera and we apply a simple threshold value in order to immediately stop the motion in case of an impending collision.

The resulting system is a low-cost, power-efficient, and connection-free local path planner that can be easily integrated with a global system achieving fully autonomous navigation in vineyards.

4.1. Continuous Depth Map Control

In order to obtain a proportional control, we detect the center of the end of the vineyard row exploiting the depth-map provided by the stereo camera. Subsequently, the control values for the linear velocity and the angular velocity are calculated proportionally to the horizontal distance between the center of the end of the vineyard row and the longitudinal axis of the UGV.

To this end, we compute the largest area that gathers all the points beyond a certain depth value and then, we bound that area with a rectangle that will be used to compute the control values.

The depth-map is a single-channel matrix with the same dimensions of the image resolution, where each entry represents the depth in millimeters of the corresponding pixel in the camera frame. The limits of the depth computation are 0 and 8 meters; therefore, the values in the depth matrix range from 0 to 8000.

The main steps of the proposed methodology, described in Algorithm 1, are shown in detailed with the following points:

1. *Matrix normalization:* In order to have a solution adaptable to different outdoor scenarios, we need to have a dynamic definition of near field and far-field. Therefore, we employ a dynamic threshold computed proportionally to the maximum acquired depth value. Hence, by normalizing the matrix, we obtain a threshold that changes dynamically depending on the values of the depth map.
2. *Depth threshold:* We apply a threshold on the depth matrix, obtained through a detailed calibration, in order to define which is the near field (represented with a "0") and the far field (represented with a "1"). At this point the depth matrix is a binary mask.
3. *Bounding operation:* We perform edge detection on the binary image, extrapolating the contours of the white areas, and then, we bound these contours with a rectangle.

4. *Back-up solution:* If no white area is detected or in case the area of the largest rectangle is less than a certain threshold, we activate the back-up model based on machine learning.
5. *Window selection:* On the other hand, if there are multiple detected rectangles, we evaluate only the biggest one in order to get rid of the noise. The threshold value for the area is obtained through a calibration and it is used to avoid false positive detection. In fact, the holes on the sides of the vineyard row can be detected as large areas with all the points beyond the distance threshold, and therefore they can lead to a wrong command computation. To prevent the system from performing an autonomous navigation using an erroneous detection of the end of the vineyard row, we calibrated the threshold to reduce the possibility that this eventuality occurs drastically. From now on, with the term window we will refer to the largest rectangle detected in the processed frame which area is greater than the area threshold.
6. *Control values:* The angular velocity and the linear velocity values are both proportional to the horizontal distance (in pixel) between the center of the detected window and the center of the camera frame and based on a parabolic function. The distance d is computed as:

$$d = X_w - X_c \tag{1}$$

where X_w is the horizontal coordinate of the center of the detected rectangle and X_c is the horizontal coordinate of the center of the frame. Figure 2 shows a graphical representation of the computation of the distance d.

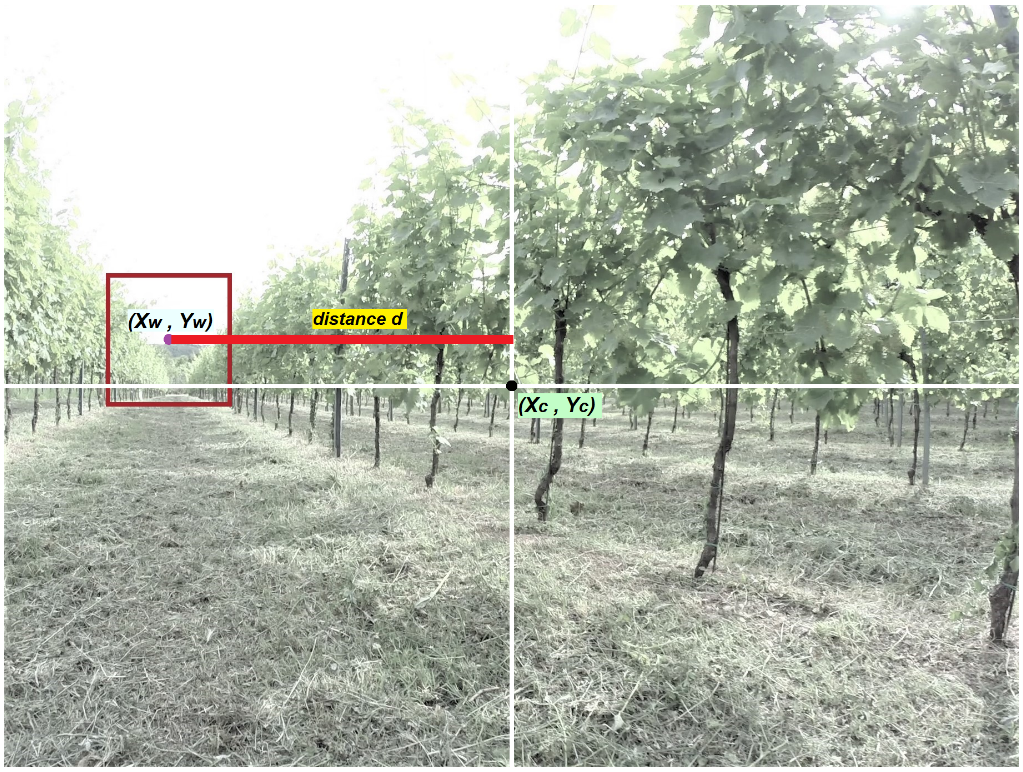

Figure 2. Depth-map algorithm scheme. The end of the vineyard row detected is represented by the red rectangle. The value d, on which the control algorithm is based on, is computed as the horizontal distance between the center of the camera frame and the center of the detected window.

The controller value for the angular velocity (ang_vel) is calculated through the following formula:

$$ang\_vel = \begin{cases} -max\_ang\_vel \cdot \left(\frac{d^2}{(\frac{w}{2})^2}\right), & \text{if } d \geq 0 \\ max\_ang\_vel \cdot \left(\frac{d^2}{(\frac{w}{2})^2}\right), & \text{if } d < 0 \end{cases} \tag{2}$$

where max_ang_vel is the maximum angular velocity achievable and w is the width of the frame.

Algorithm 1 RGB-D camera based algorithm

```
Input: $D_{h\times w}$: Depth Matrix provided by the camera
Input: $F_{h\times w\times 3}$ RGB frame acquired by the camera
Input: $T_{distance}$ threshold on the distance
Input: $T_{area}$ threshold on the area of the rectangles
Input: $X_c$ horizontal coordinate of the center of the camera frame
Input: $X_w$ horizontal coordinate of the center of the detected window
Output: Control commands for the autonomous navigation
 1: $D \longleftarrow \|D\|_2$
 2: for i = 1,··· h and j = 1,··· w do
 3:     if $D_{i\times j} > T_{distance}$ then
 4:         $D_{i\times j} = 1$
 5:     else
 6:         $D_{i\times j} = 0$
 7:     end if
 8: end for
 9: cont[] ← contours($D_{i\times j}$)
10: rect[] ← boundingRect(cont[])
11: if max(area(rect[]))< $T_{area}$ or rect[].isEmpty() then
12:     continue from line 18
13: else
14:     angular_velocity()
15:     linear_velocity()
16:     acquire next frame and restart from line 1
17: end if
18: $I_{1\times rh\times rw\times 3}$← preprocessing($F_{h\times w\times 3}$)
19: model_prediction($I_{1\times rh\times rw\times 3}$ )
20: ML_controller()
```

As far as the linear velocity (lin_vel) control function is concerned, it is still be a parabola, but this time the lower is the distance d the higher its value gets. Therefore the formula is:

$$lin\_vel = max\_lin\_vel \cdot \left(1 - \left(\frac{d^2}{(\frac{w}{2})^2}\right)\right) \tag{3}$$

where max_lin_vel is the maximum linear velocity achievable and w is the width of the frame. Both control characteristics curve are depicted in Figure 3.

4.2. Discrete CNN Control

Navigating in an outdoor environment can be extremely challenging. Among several troubles, we noticed that sunlight can be very deceptive when performing edge detection and it could lead to hazardous situations using a total camera-based navigation system. Therefore, besides the depth-map based algorithm, we propose a back-up solution that exploits machine learning methodologies in order to assist the main algorithm in case of failure. These are the last points described in Algorithm 1.

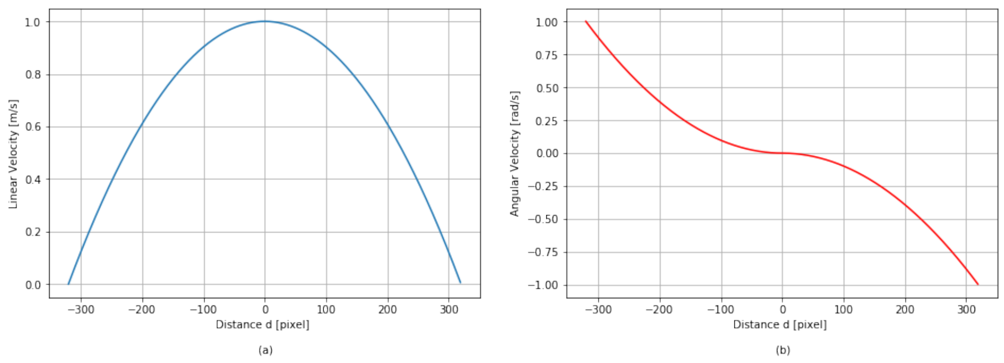

Figure 3. Considering a frame resolution of 640 × 480, a maximum linear velocity of 1 m/s and a maximum angular velocity of 1 rad/s the plot of the linear velocity control function is shown in Figure 2a, whereas, the plot of the angular velocity control function is shown in Figure 2b.

Greatly inspired by Giusti et al. [51] our second approach relies on a convolutional neural network (CNN) that classifies the frames acquired by the camera into the three following classes: left, center and right. So, in a vineyard scenario, the class center describes the view of the camera when the vehicle is pointing at the end of the vineyard row, whereas the classes left and right indicates whether the vehicle is pointing at the left side or at the right side of the vineyard row, respectively.

Successively, using the predictions of the trained network, we designed a basic control system to route the path of the robot through vineyards rows. Moreover, we exploited latest advancements in model optimization techniques in order to obtain an efficient and lightweight network able to inference in real-time on a low-cost edge AI platform.

4.2.1. Network Architecture

We have carefully selected a deep learning architecture from the literature that reaches high performance by also containing computational requirements and hardware costs. MobileNet [52] network, due to its efficient design, works reasonably fast on mobile devices and embedded systems without too much memory allocation. The structure of the MobileNet, illustrated in Figure 4, consists on a first convolutional layer with $n = 32$ filters and stride $s = 2$ followed by 13 layers that include depthwise (dw) and pointwise convolutions. That largely reduces the number of parameters and inference time while still providing reasonable accuracy level. After each convolution, batch normalization [53] and ReLU activation function [54] are applied. Every two blocks the number of filters is doubled while reducing the first two dimensions with a stride greater than one. Finally, an average pooling layer resizes the output of the last convolutional block and feeds a fully connected layer with a softmax activation function that produces the final classification predictions.

We have modified the original final fully connected layers of the MobileNet by substituting them with two fully connected layers of 256 and three neurons, respectively. The resulting model is a CNN network with an overall depth of 90 layers and with just 3,492,035 parameters.

Moreover, we optimized the network model and we sped up the inference procedure by using the framework provided by NVIDIA TensorRT [55].

4.2.2. Pre-Processing

In the pre-processing phase, before feeding the network, we normalize and resize the images to the expected input dimensions $rh \times rw$ of the model. Indeed, the last two fully connected layers chain the network at a fixed input size of the raw data. More specifically for our modified MobileNet, the input dimensions are 224 × 224.

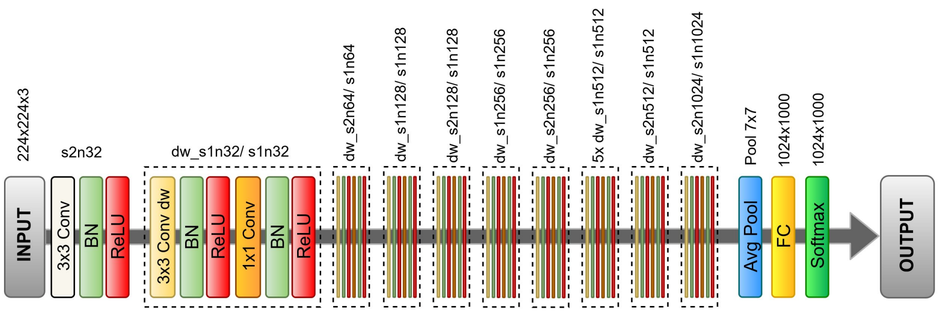

Figure 4. MobileNet network architecture. Depthwise and pointwise convolutions are used through out the entire model drastically reducing the number of parameters required. Coonvolutions with strides *s* two are used in substitution of Max-Pooling operations.

5. Experimental Discussion and Results

In this section, we discuss the details of the deep learning model training with its dataset generation and evaluation. Furthermore, we introduce the optimization adjustments applied to the network in order to boost the frequency control to 47.15 Hz. Finally, we conclude with the experimentation data and results gathered during the field tests.

5.1. Dataset Creation

We used a dataset of 33.616 images equally balanced along with the three previously introduced classes. In order to create the training dataset, as previously introduced in Section 3, we took several videos in a variety of vineyards rows with a 1080 p resolution camera in order to have more flexibility during the pre-processing phase. In particular, for the first video of the *center* class, we recorded rows with the camera pointing at its center. Whereas, for the other two videos, classes *left* and *right*, we registered with the camera rotated of 45 degrees with respect to the longitudinal axis of the row towards the left and the right side, respectively. Eventually, we took each video as a streaming of images and we selected the best frame every six consecutive ones using a Laplacian filter to detect the less blurring one. Figure 5 shows an example for each class.

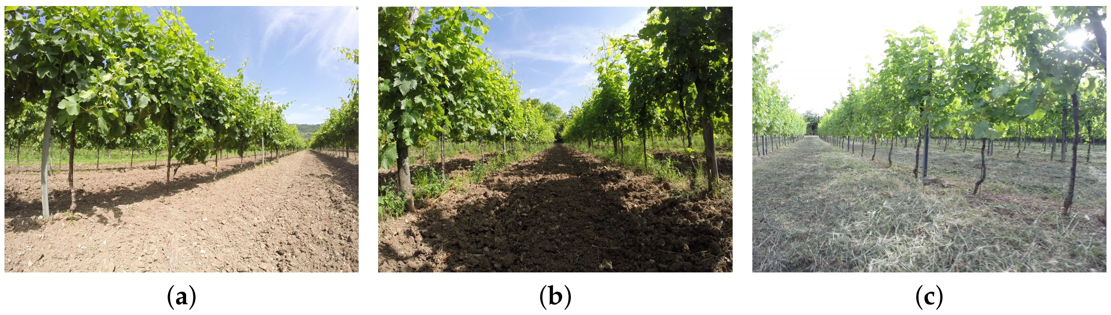

Figure 5. Three samples of the dataset used to train the network, one for each class. (**a**) is an example of the left class, (**b**) of the center class, and (**c**) of the right class. Dataset samples have been collected with different weather conditions and at a different time of day. The resulting heterogeneous training set is aimed at giving generality and robustness to the model.

5.2. Model Training

As already introduced, we trained the network using a technique known as transfer learning [56]; instead of starting to train with weights randomly initialized, we used variables obtained with an earlier training session. In particular, we exploited weights obtained fitting MobileNet with the ImageNet classification dataset [57]. Using this technique, we were able to take advantage of previous low-level features, learned by the network, highly reducing the number of images and epochs required for the training. Indeed, edges, contours, and basic textures are general-purpose features that can be reused for different tasks. In order to properly train, validate and test the model, we randomly divided the dataset into three subsets as follow: 70% for the training set, 15% for the development set, and the remaining 15% for the test set. We trained the resulting network for only six epochs with a batch size of 64. To increment the robustness of the network and to overcome possible problems of overfitting, we used different techniques such as dropout [58], weight decay [59] and data augmentation with changes in zoom and brightness [60]. Finally, we used mini-batches with RMSprop optimizer [61], accuracy metric, and cross entropy as a loss function.

5.3. Machine Learning Model Evaluation and Optimization

The implemented model has been trained and tested with the subdivision of the dataset introduced in previous Section 5.2, giving an accuracy of 1.0 over the test set. Therefore, this model is the one employed for the navigation.

In order to inspect the model and justify the high accuracy of it, we plotted the intermediate activations of the trained network and we adopted Grad-CAM [62], to highlight important regions in the image for predicting the correct class. In Figure 6 are shown some feature maps at different level of depth: immediately after the first convolution, at an intermediate point and before the average pooling layer. It is clear how the deep learning model is able to generate robust feature maps already after the first convolution. Later those representations are exploited in order to produce disentangled representations that easily allow the model to predict the three different classes with high level of confidence. Instead, in Figure 7, are presented the regions of interest for the three different classes. With Grad-CAM we can visually validate that the network is activating around the proper patterns of the input image and that it is not exploiting short cuts to achieve a high level of accuracy. Indeed, we can easily assess that the model, trained with transfer learning, is exploiting the vineyard rows and their vanishing point to obtain an effective generalization power.

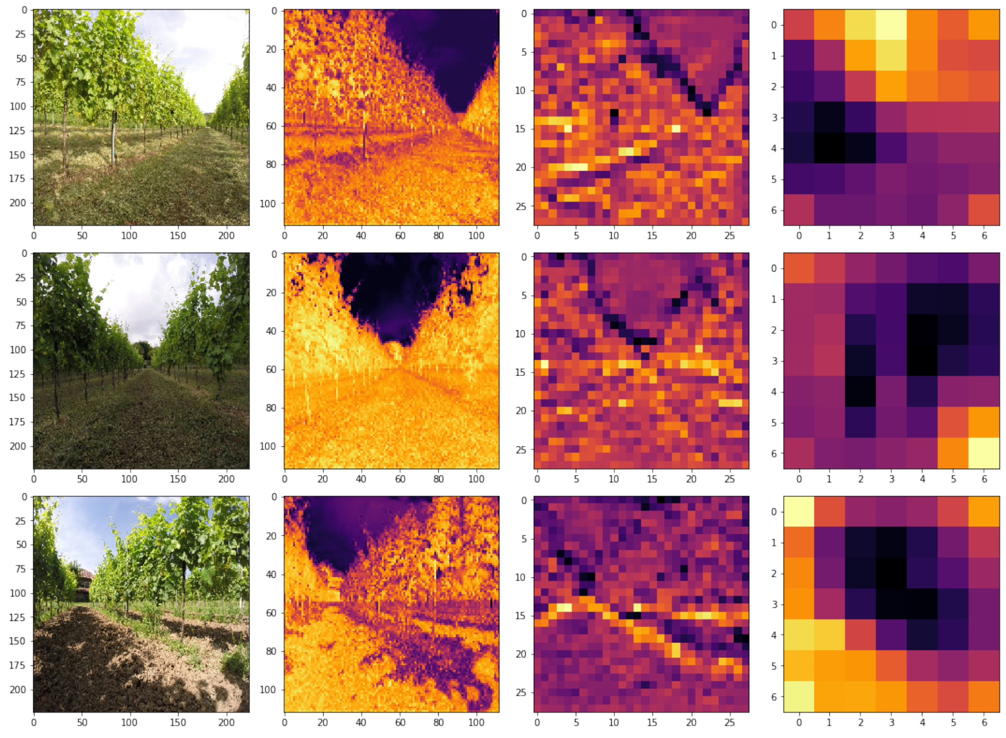

Figure 6. Three input images belonging to different classes, with their respective activation maps taken at different level of depth of the network. Already on early stages, the network, pre-trained on ImageNet, is able to extract useful representations that lead at robust, disentangle activations in the final layers. It is possible to notice how the two spacial dimensions are increasingly reduced.

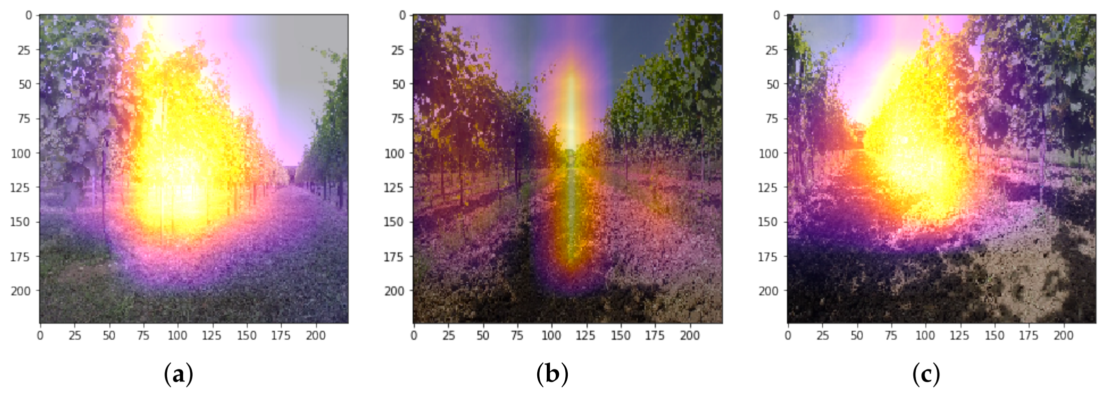

Figure 7. Gradient information, flowing into the last convolutional layer of MobileNet, is used to understand each neuron for a decision of interest. It is possible to assess that for either the three classes, (**a**–**c**), the network is "looking" at the vineyard roads and their vanishing point.

Moreover, in order to evaluate the robustness of the network over new scenarios, and prove how transfer learning is so effective for this specific application, we performed an experimentation, training the model only with a small part of the available dataset. So, we trained the architecture with just a vineyard type and tested the resulting model with five completely different scenarios with diverse wine quality and weather conditions. In particular, we used only 18% as training examples, corresponding to 6.068 images, due to the amount of images available for each region of the available dataset. Consequently, we tested the new trained network with the remaining 27,458 samples.

As shown in Table 1, an accuracy of 0.94 is achieved by the re-trained model in this second case. That is an optimal result considering the fact that the network has been trained with a very small dataset and it has been tested with a completely different vineyard scenario. This clearly demonstrates how transfer learning, for this specific task, is very effective at providing good generalization capabilities with also a small training set. We also compared, using this last split, the selected network with other notable architectures of the literature. As it is clear from Table 2 MobileNet is the right balance between average accuracy and computational request. However, in the presence of a platform with more flexible computational constraints, EfficientNet-B1, or networks with higher compound coefficient ϕ [63], would be much more likely to generalize over new scenarios maintaining an optimal level of efficiency.

Table 1. Results of the second evaluation of the trained CNN model.

Class	Precision	Recall	f1-Score
right	0.850	1.000	0.919
left	1.000	0.899	0.947
center	1.000	0.924	0.961
micro avg	0.941	0.941	0.941
macro avg	0.950	0.941	0.942
weighted avg	0.950	0.941	0.942

Finally, as previously introduced, the employed network has been optimized, discarding all redundant operations and reducing the floating point precision from 32 to 16 bits, using the framework TensorRT. The optimization process, besides not affecting the accuracy of the predictions, it gives a significant increment to the number of frames elaborated per second by our model, using the same hardware supplied with the robot. In fact, the control frequency using Tensorflow with a frozen graph, computational graph of the network without optimization and tranining nodes, was 21.92 Hz, whereas, with the performed optimization, we reached 47.15 Hz.

Table 2. Comparison between different CNN architectures using 18% of the available dataset and transfer learning to train the networks. MobileNet, for this specific application, is a good trade-off between performance and computational cost.

Model	**Parameters**	**GFLOPSs**	**Avg. Acc.**
MobileNet [52]	4,253,864	0.579	94.7%
MobileNetV2 [64]	3,538,984	0.31	91.3%
EfficientNet-B0 [63]	5,330,571	0.39	93.8%
EfficientNet-B1 [63]	7,856,239	0.70	96.8%
ResNet50 [65]	25,636,712	4.0	93.1%
DenseNet121 [66]	8,062,504	3.0	95.3%

5.4. Field Experimentation

As far as the deployment is concerned, the system has been implemented in a ROS-oriented robot platform (https://www.ros.org/). The robot in which the local planner has been tested is an unmanned ground vehicle: the model Jackal from Clearpath Robotics (Figure 8) introduced in Section 3.

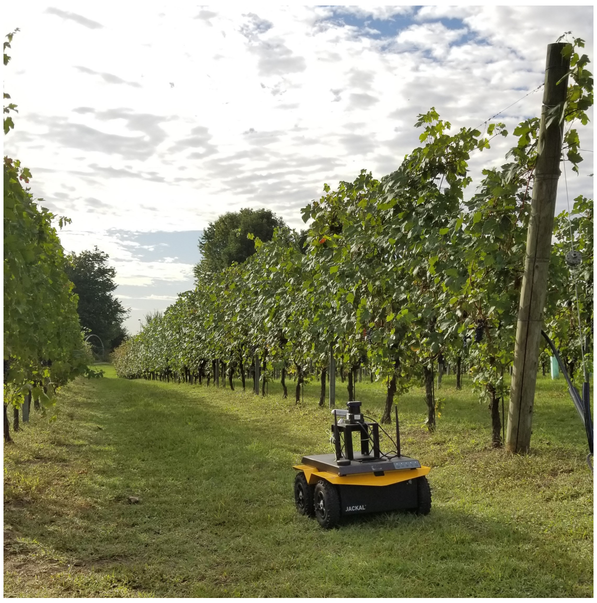

Figure 8. Jackal model robotic platform on the test site during the field experimentation.

The tests have been carried out in a new vineyard scenario. In order to correctly perform navigation the stereo camera has been installed in such a way that the center of the camera frame corresponds to the longitudinal axis of the vehicle.

The proposed solution, after several trials with different vineyard rows but similar weather conditions, proved to be able to perform an autonomous navigation along the given paths, even lowering down the resolution of the camera to 640 × 480. More specifically, for the two infrared cameras with which the camera computes the depth-map the resolution has been set to 640 × 480, whereas, for the RGB images processed by the machine learning model we started with a resolution of 1280 × 720 and then we gradually reduced it until 640 × 480 as mentioned. Moreover, when acquiring images we used the default calibration provided by Intel with the camera lens distortion based on the Brown–Conrady model [67]. The intrinsic parameters for the final configuration of the camera in both the so-called depth and color modes are showed in Table 3. All our tests showed precise trajectories, comparable with ones obtained with data fusion techniques that make use of several expensive sensors to maintain the correct course.

Table 3. Intrinsic parameters of the camera for each of the exploited modes. Principal point (PPX and PPY) and focal length (Fx and Fy) for a resolution of 640 × 480.

Mode	Depth	Color
PPX	321.910675048828	316.722351074219
PPY	236.759078979492	244.21875
Fx	387.342498779297	617.42242431640
Fy	387.342498779297	617.789978027344

As noticeable from the depth maps samples in Figure 9 taken during the field experimentation, the first method can detect the end of the vineyard independently from the direction of the longitudinal axis of the robot. The rectangle is successively used for control signals generation. In case of a fault of this solution, as previously introduced, the machine learning based algorithm takes control. Finally, it is possible to exploit the distance value d to easily collect new, already labeled, sample data from the operational work of the robotic platform. Indeed, due to the nature of all mini-batch gradient descent based optimizer, it is possible to continuously use new data points to extend the existing model's knowledge obtaining a more robust and prone to generalize neural network.

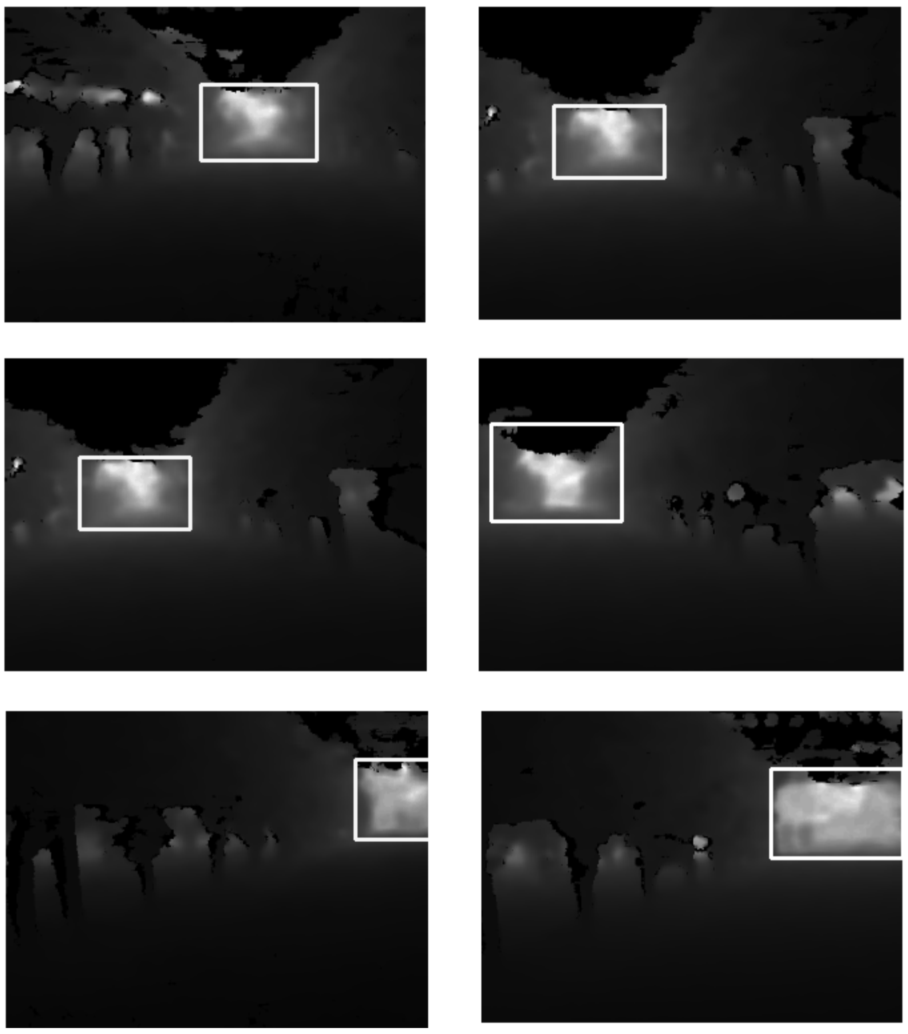
Figure 9. Instances of the depth-map based algorithm while performing tests in the vineyards. Wherever the robot is pointing at, it is capable of correctly detecting the end of the vineyard rows.

6. Conclusions

We proposed a local motion planner for vineyards rows autonomous navigation. We exploited the stereo vision properties of an RGB-D camera and latest advancements in deep learning optimization techniques in order to obtain a lightweight, power-efficient algorithm able to run on a low-cost hardware.

The proposed overall methodology provides a real-time control frequency using only hardware with limited computational capabilities containing costs and required resources. The back-up trained neural network is robust to different factors of variation, and after the optimization procedure, it provides a control frequency of 47.15 Hz without the need of external hardware accelerators.

Finally, the proposed local motion planner has been implemented on a robotic platform and tested on the relevant environment, demonstrating to scale real working conditions even with a low resolution.

As future work, we plan to integrate the presented work with a concrete application and extent the methodology to orchards and any other analogous scenario.

Author Contributions: Conceptualization, M.C., D.A. and V.M.; methodology, D.A. and V.M.; software, D.A. and V.M.; validation, D.A. and V.M.; data curation, D.A. and V.M.; writing–original draft preparation, D.A. and V.M.; writing–review and editing, D.A., V.M. and M.C.; project administration, M.C.; All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. DeSA, U. *World Population Prospects: The 2012 Revision*; Population Division of the Department of Economic and Social Affairs of the United Nations Secretariat: New York, NY, USA, 2013; Volume 18.
2. Mulla, D.J. Twenty five years of remote sensing in precision agriculture: Key advances and remaining knowledge gaps. *Biosyst. Eng.* **2013**, *114*, 358–371. [CrossRef]
3. R. Shamshiri, R.; Weltzien, C.; Hameed, I.A.; J. Yule, I.; E. Grift, T.; Balasundram, S.K.; Pitonakova, L.; Ahmad, D.; Chowdhary, G. Research and development in agricultural robotics: A perspective of digital farming. *Int. J. Agric. Biol. Eng.* **2018**. [CrossRef]
4. Payne, C. Technologies for efficient farming. In Proceedings of the Electrical Insulation Conference and Electrical Manufacturing Expo, Indianapolis, IN, USA, 23–26 October 2005; pp. 435–441.
5. Bac, C.W.; van Henten, E.J.; Hemming, J.; Edan, Y. Harvesting robots for high-value crops: State-of-the-art review and challenges ahead. *J. Field Robot.* **2014**, *31*, 888–911. [CrossRef]
6. Rath, T.; Kawollek, M. Robotic harvesting of Gerbera Jamesonii based on detection and three-dimensional modeling of cut flower pedicels. *Comput. Electron. Agric.* **2009**, *66*, 85–92. [CrossRef]
7. Berenstein, R.; Shahar, O.B.; Shapiro, A.; Edan, Y. Grape clusters and foliage detection algorithms for autonomous selective vineyard sprayer. *Intell. Serv. Robot.* **2010**, *3*, 233–243. [CrossRef]
8. Shapiro, A.; Korkidi, E.; Demri, A.; Ben-Shahar, O.; Riemer, R.; Edan, Y. Toward elevated agrobotics: Development of a scaled-down prototype for visually guided date palm tree sprayer. *J. Field Robot.* **2009**, *26*, 572–590. [CrossRef]
9. Monta, M.; Kondo, N.; Shibano, Y. Agricultural robot in grape production system. In Proceedings of the 1995 IEEE International Conference on Robotics and Automation, Nagoya, Japan, 21–27 May 1995; Volume 3, pp. 2504–2509.
10. Katupitiya, J.; Eaton, R.; Yaqub, T. Systems engineering approach to agricultural automation: New developments. In Proceedings of the 2007 1st Annual IEEE Systems Conference, Honolulu, HI, USA, 9–13 April 2007; pp. 1–7.
11. Kohanbash, D.; Valada, A.; Kantor, G. Irrigation control methods for wireless sensor network. In *2012 Dallas, Texas, July 29–August 1, 2012*; American Society of Agricultural and Biological Engineers: St. Joseph, MI, USA, 2012; p. 1.
12. Virlet, N.; Sabermanesh, K.; Sadeghi-Tehran, P.; Hawkesford, M.J. Field Scanalyzer: An automated robotic field phenotyping platform for detailed crop monitoring. *Funct. Plant Biol.* **2017**, *44*, 143–153. [CrossRef]

13. Nuske, S.; Achar, S.; Bates, T.; Narasimhan, S.; Singh, S. Yield estimation in vineyards by visual grape detection. In Proceedings of the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, San Francisco, CA, USA, 25–30 September 2011; pp. 2352–2358.
14. Wang, Q.; Nuske, S.; Bergerman, M.; Singh, S. *Automated Crop Yield Estimation for Apple Orchards. Experimental Robotics*; Springer: Berlin, Germany, 2013; pp. 745–758.
15. Davis, B. CMU-led automation program puts robots in the field. *Mission Crit.* **2012**, 38–40.
16. Sharifi, M.; Chen, X. A novel vision based row guidance approach for navigation of agricultural mobile robots in orchards. In Proceedings of the 2015 6th International Conference on Automation, Robotics and Applications (ICARA), Queenstown, New Zealand, 17–19 February 2015; pp. 251–255.
17. Guzmán, R.; Ariño, J.; Navarro, R.; Lopes, C.; Graça, J.; Reyes, M.; Barriguinha, A.; Braga, R. Autonomous hybrid GPS/reactive navigation of an unmanned ground vehicle for precision viticulture-VINBOT. In Proceedings of the 62nd German Winegrowers Conference, Stuttgart, Germany, 27–30 November 2016.
18. Dos Santos, F.N.; Sobreira, H.; Campos, D.; Morais, R.; Moreira, A.P.; Contente, O. Towards a reliable robot for steep slope vineyards monitoring. *J. Intell. Robot. Syst.* **2016**, *83*, 429–444. [CrossRef]
19. Astolfi, P.; Gabrielli, A.; Bascetta, L.; Matteucci, M. Vineyard autonomous navigation in the echord++ grape experiment. *IFAC-PapersOnLine* **2018**, *51*, 704–709. [CrossRef]
20. Santos, L.; Santos, F.N.; Magalhães, S.; Costa, P.; Reis, R. Path planning approach with the extraction of topological maps from occupancy grid maps in steep slope vineyards. In Proceedings of the 2019 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC), Porto, Portugal, 24–26 April 2019; pp. 1–7.
21. Zoto, J.; Musci, M.A.; Khaliq, A.; Chiaberge, M.; Aicardi, I. Automatic Path Planning for Unmanned Ground Vehicle Using UAV Imagery. In *International Conference on Robotics in Alpe-Adria Danube Region*; Springer: Cham, Switzerland, 2019; pp. 223–230.
22. Ma, C.; Jee, G.I.; MacGougan, G.; Lachapelle, G.; Bloebaum, S.; Cox, G.; Garin, L.; Shewfelt, J. Gps signal degradation modeling. In Proceedings of the International Technical Meeting of the Satellite Division of the Institute of Navigation, Salt Lake City, UT, USA, 11–14 September 2001.
23. LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436–444. [CrossRef] [PubMed]
24. Mazzia, V.; Khaliq, A.; Salvetti, F.; Chiaberge, M. Real-Time Apple Detection System Using Embedded Systems with Hardware Accelerators: An Edge AI Application. *IEEE Access* **2020**, *8*, 9102–9114. [CrossRef]
25. Ruckelshausen, A.; Biber, P.; Dorna, M.; Gremmes, H.; Klose, R.; Linz, A.; Rahe, F.; Resch, R.; Thiel, M.; Trautz, D.; et al. BoniRob: an autonomous field robot platform for individual plant phenotyping. *Precis. Agric.* **2009**, *9*, 1.
26. Stoll, A.; Kutzbach, H.D. Guidance of a forage harvester with GPS. *Precis. Agric.* **2000**, *2*, 281–291. [CrossRef]
27. Thuilot, B.; Cariou, C.; Cordesses, L.; Martinet, P. Automatic guidance of a farm tractor along curved paths, using a unique CP-DGPS. In Proceedings of the 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No. 01CH37180), Maui, HI, USA, 29 October–3 November 2001; Volume 2, pp. 674–679.
28. Ly, O.; Gimbert, H.; Passault, G.; Baron, G. A fully autonomous robot for putting posts for trellising vineyard with centimetric accuracy. In Proceedings of the 2015 IEEE International Conference on Autonomous Robot Systems and Competitions, Vila Real, Portugal, 8–10 April 2015; pp. 44–49.
29. Longo, D.; Pennisi, A.; Bonsignore, R.; Muscato, G.; Schillaci, G. A multifunctional tracked vehicle able to operate in vineyards using gps and laser range-finder technology. Work safety and risk prevention in agro-food and forest systems. In Proceedings of the International Conference Ragusa SHWA2010, Ragusa, Italy, 16–18 September 2010.
30. Hansen, S.; Bayramoglu, E.; Andersen, J.C.; Ravn, O.; Andersen, N.; Poulsen, N.K. Orchard navigation using derivative free Kalman filtering. In Proceedings of the 2011 American Control Conference, San Francisco, CA, USA, 29 June–1 July 2011; pp. 4679–4684.
31. Marden, S.; Whitty, M. GPS-free localisation and navigation of an unmanned ground vehicle for yield forecasting in a vineyard. In *Recent Advances in Agricultural Robotics, International Workshop Collocated with the 13th International Conference on Intelligent Autonomous Systems (IAS-13)*; Springer: Berlin, Germany, 2014.
32. Zaidner, G.; Shapiro, A. A novel data fusion algorithm for low-cost localisation and navigation of autonomous vineyard sprayer robots. *Biosyst. Eng.* **2016**, *146*, 133–148. [CrossRef]

33. Riggio, G.; Fantuzzi, C.; Secchi, C. A Low-Cost Navigation Strategy for Yield Estimation in Vineyards. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; pp. 2200–2205.
34. Mohanty, S.P.; Hughes, D.P.; Salathé, M. Using deep learning for image-based plant disease detection. *Front. Plant Sci.* **2016**, *7*, 1419. [CrossRef] [PubMed]
35. Sladojevic, S.; Arsenovic, M.; Anderla, A.; Culibrk, D.; Stefanovic, D. Deep neural networks based recognition of plant diseases by leaf image classification. *Comput. Intell. Neurosci.* **2016**, *2016*. [CrossRef] [PubMed]
36. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*; The MIT Press: Cambridge, MA, USA, 2012; pp. 1097–1105.
37. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
38. Jia, Y.; Shelhamer, E.; Donahue, J.; Karayev, S.; Long, J.; Girshick, R.; Guadarrama, S.; Darrell, T. Caffe: Convolutional architecture for fast feature embedding. In Proceedings of the 22nd ACM international conference on Multimedia, Orlando, FL, USA, 3–7 November 2014; pp. 675–678.
39. Kussul, N.; Lavreniuk, M.; Skakun, S.; Shelestov, A. Deep learning classification of land cover and crop types using remote sensing data. *IEEE Geosci. Remote Sens. Lett.* **2017**, *14*, 778–782. [CrossRef]
40. Mortensen, A.K.; Dyrmann, M.; Karstoft, H.; Jørgensen, R.N.; Gislum, R. Semantic segmentation of mixed crops using deep convolutional neural network. In Proceedings of the International Conference of Agricultural Engineering (CIGR), Aarhus, Denmark, 26–29 June 2016.
41. Rebetez, J.; Satizábal, H.F.; Mota, M.; Noll, D.; Büchi, L.; Wendling, M.; Cannelle, B.; Pérez-Uribe, A.; Burgos, S. Augmenting a convolutional neural network with local histograms-A case study in crop classification from high-resolution UAV imagery. In Proceedings of the ESANN 2016, European Symposium on Artifical Neural Networks, Bruges, Belgium, 27–29 April 2016.
42. Mazzia, V.; Khaliq, A.; Chiaberge, M. Improvement in Land Cover and Crop Classification based on Temporal Features Learning from Sentinel-2 Data Using Recurrent-Convolutional Neural Network (R-CNN). *Appl. Sci.* **2020**, *10*, 238. [CrossRef]
43. Kuwata, K.; Shibasaki, R. Estimating crop yields with deep learning and remotely sensed data. In Proceedings of the 2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Milan, Italy, 26–31 July 2015; pp. 858–861.
44. Minh, D.H.T.; Ienco, D.; Gaetano, R.; Lalande, N.; Ndikumana, E.; Osman, F.; Maurel, P. Deep Recurrent Neural Networks for mapping winter vegetation quality coverage via multi-temporal SAR Sentinel-1. *arXiv* **2017**, arXiv:1708.03694.
45. Mazzia, V.; Comba, L.; Khaliq, A.; Chiaberge, M.; Gay, P. UAV and Machine Learning Based Refinement of a Satellite-Driven Vegetation Index for Precision Agriculture. *Sensors* **2020**, *20*, 2530. [CrossRef] [PubMed]
46. Khaliq, A.; Mazzia, V.; Chiaberge, M. Refining satellite imagery by using UAV imagery for vineyard environment: A CNN Based approach. In Proceedings of the 2019 IEEE International Workshop on Metrology for Agriculture and Forestry (MetroAgriFor), Portici, Italy, 24–26 October 2019; pp. 25–29.
47. Chen, S.W.; Shivakumar, S.S.; Dcunha, S.; Das, J.; Okon, E.; Qu, C.; Taylor, C.J.; Kumar, V. Counting apples and oranges with deep learning: A data-driven approach. *IEEE Robot. Autom. Lett.* **2017**, *2*, 781–788. [CrossRef]
48. Bargoti, S.; Underwood, J. Deep fruit detection in orchards. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 3626–3633.
49. Sehgal, G.; Gupta, B.; Paneri, K.; Singh, K.; Sharma, G.; Shroff, G. Crop planning using stochastic visual optimization. In Proceedings of the 2017 IEEE Visualization in Data Science (VDS), Phoenix, AZ, USA, 1 October 2017; pp. 47–51.
50. Mousavian, A.; Toshev, A.; Fišer, M.; Košecká, J.; Wahid, A.; Davidson, J. Visual representations for semantic target driven navigation. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 8846–8852.
51. Giusti, A.; Guzzi, J.; Cireşan, D.C.; He, F.L.; Rodríguez, J.P.; Fontana, F.; Faessler, M.; Forster, C.; Schmidhuber, J.; Di Caro, G.; et al. A machine learning approach to visual perception of forest trails for mobile robots. *IEEE Robot. Autom. Lett.* **2015**, *1*, 661–667. [CrossRef]

52. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv* **2017**, arXiv:1704.04861.
53. Ioffe, S.; Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv* **2015**, arXiv:1502.03167.
54. Nair, V.; Hinton, G.E. Rectified linear units improve restricted boltzmann machines. In Proceedings of the 27th International Conference on Machine Learning (ICML-10), Haifa, Israel, 21–24 June 2010; pp. 807–814.
55. Vanholder, H. Efficient Inference with TensorRT. 2016. Available online: http://on-demand.gputechconf.com/gtc-eu/2017/presentation/23425-han-vanholder-efficient-inference-with-tensorrt.pdf (accessed on 22 May 2020).
56. Tan, C.; Sun, F.; Kong, T.; Zhang, W.; Yang, C.; Liu, C. A survey on deep transfer learning. In *International Conference on Artificial Neural Networks*; Springer: Berlin, Germany, 2018; pp. 270–279.
57. Deng, J.; Dong, W.; Socher, R.; Li, L.J.; Li, K.; Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255.
58. Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **2014**, *15*, 1929–1958.
59. Loshchilov, I.; Hutter, F. Fixing Weight Decay Regularization in Adam. Avaliable online: https://openreview.net/forum?id=rk6qdGgCZ (assessed on 25 May 2020).
60. Perez, L.; Wang, J. The effectiveness of data augmentation in image classification using deep learning. *arXiv* **2017**, arXiv:1712.04621.
61. Tieleman, T.; Hinton, G. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA Neural Netw. Mach. Learn.* **2012**, *4*, 26–31.
62. Selvaraju, R.R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; Batra, D. Grad-cam: Visual explanations from deep networks via gradient-based localization. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 618–626.
63. Tan, M.; Le, Q.V. Efficientnet: Rethinking model scaling for convolutional neural networks. *arXiv* **2019**, arXiv:1905.11946.
64. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.C. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4510–4520.
65. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016; pp. 770–778.
66. Huang, G.; Liu, Z.; Van Der Maaten, L.; Weinberger, K.Q. Densely connected convolutional networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 4700–4708.
67. Duane, C.B. Close-range camera calibration. *Photogramm. Eng.* **1971**, *37*, 855–866.