POLITECNICO DI TORINO
Repository ISTITUZIONALE

Waypoint Generation in Row-Based Crops with Deep Learning and Contrastive Clustering

Original
Waypoint Generation in Row-Based Crops with Deep Learning and Contrastive Clustering / Salvetti, F., Angarano, S., Martini, M., Cerrato, S., Chiaberge, M.. - ELETTRONICO. - VI:(2023), pp. 203-218. (European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases 2022 Grenoble (France) September 19–23, 2022) [10.1007/978-3-031-26422-1_13].

Availability:
This version is available at: 11583/2977351 since: 2023-03-23T09:27:13Z

Publisher:
Springer

Published
DOI:10.1007/978-3-031-26422-1_13

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright
Springer postprint/Author's Accepted Manuscript

This version of the article has been accepted for publication, after peer review (when applicable) and is subject to Springer Nature's AM terms of use, but is not the Version of Record and does not reflect post-acceptance improvements, or any corrections. The Version of Record is available online at: http://dx.doi.org/10.1007/978-3-031-26422-1_13

(Article begins on next page)

01 November 2026

Waypoint Generation in Row-based Crops with Deep Learning and Contrastive Clustering

Francesco Salvetti[1,2,3] (✉), Simone Angarano[1,2], Mauro Martini[1,2], Simone Cerrato[1,2], and Marcello Chiaberge[1,2]

[1] Politecnico di Torino, Turin, Italy
name.surname@polito.it
[2] PIC4SeR (PoliTo Interdepartmental Centre for Service Robotics), Turin, Italy
[3] SmartData@PoliTo, Turin, Italy

Abstract. The development of precision agriculture has gradually introduced automation in the agricultural process to support and rationalize all the activities related to field management. In particular, service robotics plays a predominant role in this evolution by deploying autonomous agents able to navigate in fields while executing different tasks without the need for human intervention, such as monitoring, spraying and harvesting. In this context, global path planning is the first necessary step for every robotic mission and ensures that the navigation is performed efficiently and with complete field coverage. In this paper, we propose a learning-based approach to tackle waypoint generation for planning a navigation path for row-based crops, starting from a top-view map of the region-of-interest. We present a novel methodology for waypoint clustering based on a contrastive loss, able to project the points to a separable latent space. The proposed deep neural network can simultaneously predict the waypoint position and cluster assignment with two specialized heads in a single forward pass. The extensive experimentation on simulated and real-world images demonstrates that the proposed approach effectively solves the waypoint generation problem for both straight and curved row-based crops, overcoming the limitations of previous state-of-the-art methodologies.

Keywords: Deep learning · Clustering · Global Path Planning · Precision Agriculture.

1 Introduction

Agriculture 4.0 is introducing digital tools and technologies in precision farming. According to this innovative paradigm, Big Data, Artificial Intelligence and robotics play a key role in increasing the economic, environmental and social sustainability of agricultural processes, thanks to the efficient and automatic data collection and the processing tools they provide. In the last years, Deep Learning (DL) research has been substantially contributing to the development of new technologies for precision agriculture applications [12, 13, 33, 38]. In particular, several computer vision techniques have proven effective in supporting visual

agricultural tasks such as fruit detection and counting [23], plant disease identification [11, 25], land coverage and vegetation index classification from satellite and Unmanned Aerial Vehicles (UAVs) imagery [15, 21, 22].

Besides image processing systems, self-driving machines represent a crucial component to reduce the costs of agricultural processes by providing autonomous, full-time and weather-independent operators. In this regard, the confidence in solutions based on autonomous Unmanned Ground Vehicles (UGVs) and UAVs is drastically increasing [30, 31], considering the competitive advantage they can provide supporting tasks such as crop monitoring [35], spraying [8], grasping [14], and harvesting [19]. In this context, designing a reliable autonomous navigation system in constrained row-based crops such as vineyards and orchards is fundamental. So far, many works proposed local planners combining deep learning with computer vision [1, 2] or other sensor processing methods [3, 4, 28]. However, local planners provide a solution for intra-row navigation only, and therefore a global path generator is always needed. In a complex scenario such as a row-based environment, where traversing each row is the practical navigation goal, the problem of developing an efficient global path planner has been quite neglected by the research community. Existing solutions usually tackle the problem by clustering visual data obtained from satellites or UAVs. For example, in [39] authors use a classical clustering method to identify vineyard rows from a 3D model of the terrain reconstructed from UAV data and then compute the path accordingly. However, as pointed out in [34], the extraction of relevant information about rows geometry from images can be a complex task, in addition to being extremely computationally expensive. This limitation also holds considering other approaches besides clustering. For instance, in [7] authors adopted 3D point cloud aerial photogrammetry to detect the structure of vineyards.

Recently, the DeepWay method [24] has been proposed to efficiently combine deep learning and clustering for the generation of start and end row waypoints given an occupancy grid of the vineyard. Moreover, novel contributions adopted the same paradigm and training procedure to extend the coverage to arbitrary unstructured environments [17]. Despite being an important baseline for row-based path generation, DeepWay leaves substantial space for improvement. In particular, it applies DBSCAN clustering [9], followed by a complex heuristic geometrical post-processing heavily based on angle estimation, to discriminate start waypoints from end waypoints. However, this method performs poorly in a wide range of real-world situations, including curved crops and rows of different lengths.

In this work, we propose a novel solution for waypoint generation in row-based crops, combining deep learning with a contrastive clustering approach. To this end, we conceive a new DNN architecture to simultaneously predict the position of the navigation waypoints for each row and cluster them in a single forward step. Hence, we train our model with an additional contrastive loss on a synthetic dataset of top-view vineyard maps and test it on manually-labeled real satellite images. Our extensive experimentation demonstrates that the proposed

solution successfully predicts precise waypoints also in real-world crop maps. We also consider complex conditions such as curved rows, differently from previous solutions based on classical clustering algorithms.

The contributions of this work can be therefore summarized as follows:

- we propose a novel deep learning model to simultaneously predict the position of navigation waypoints and cluster them in a unique forward step;
- we solve the limitations of classical clustering methodologies by adopting a contrastive loss function;
- we present a method for synthetic generation of realistic curved occupancy grids for row-based crops;
- we demonstrate that our model trained on synthetic data successfully generalizes to challenging real-world satellite images.

The article is organized in the following sections. In Sec. 2, we introduce our methodology, describing the backbone design, the waypoint estimation and the contrastive clustering separately. In Sec. 3, we propose a thorough description of our experimentation, defining the settings and procedures used to generate the synthetic dataset and train the DNN. In Sec. 4, we report and discuss the obtained results, comparing our solution with classical clustering baselines, and, in Sec. 5, we draw some conclusions.

2 Methodology

Due to its intrinsic nature, every row-based crop is characterized by a set of lines or curves that identify two regions comprising the starting and ending points of each row, respectively. In this scenario, a robotic path should cover the whole field, and it can be divided into intra-row segments, that connect the starting region to the ending region, and inter-row segments, that connect two starting or two ending points. Given an optimal estimation of these starting and ending waypoints, it is possible to plan a full-coverage path in the row-based environment simply by alternating intra-row and inter-row segments. Therefore, the planning process heavily relies on two main steps: waypoint estimation, which identifies candidates for the points of interest, and waypoint clustering, which assigns each estimated point to one of the two regions.

Following the same approach presented in [24], we frame the waypoint generation process as a regression problem, in which we estimate the coordinates of the points with a deep neural network, starting from a top-view map of the environment. The map consists of a 1-bit single-channel occupancy grid that identifies with 1 the plant rows and with 0 the free terrain. Therefore, this kind of estimation process can be easily applied to geo-referenced segmented masks of the target fields obtained from satellites or UAV imagery. The waypoints and the planned path can then be converted from the image reference system to a Global Navigation Satellite System (GNSS) reference frame to be used in real-world navigation. In addition to waypoint detection, differently from classical

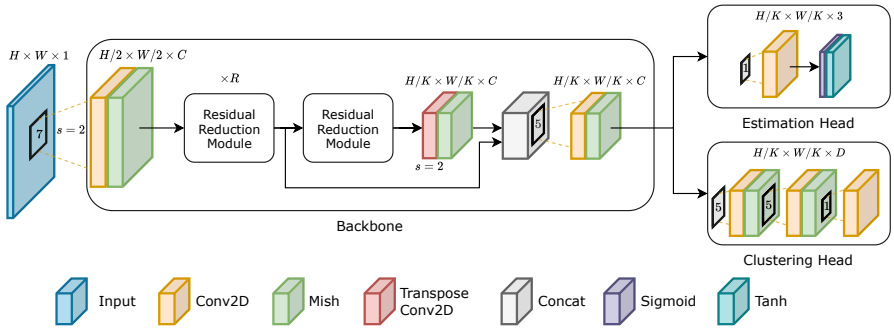


Fig. 1: Architecture of the backbone and the two regression heads. The number of residual reduction modules in the main block R determines the backbone compression factor $K = 2^{R+1}$.

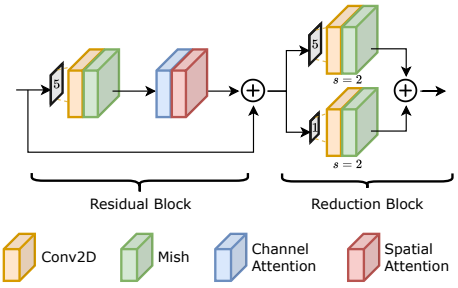


Fig. 2: Residual reduction module architecture. The channel and spatial attentions are implemented as in [36].

unsupervised methodologies for point clustering, we propose a supervised approach based on a contrastive loss to perform point assignment. Therefore, the proposed model simultaneously performs both estimation and clustering with a single forward pass, without the need for complex post-processing operations based on heuristic geometrical-based rules.

2.1 Backbone Design

We implement the model as a convolutional neural network characterized by a feature extraction backbone, followed by two specialized heads. A head is responsible for the estimation task, while the other deals with clustering.

The backbone is designed following the same architecture used in [24]. The basic block of the network is the residual module, characterized by a stack of a 2D convolution and spatial and channel attention [36]. Each residual block is followed by a reduction module characterized by convolutions with stride 2 that progressively halve the spatial dimensions. The backbone is a stack of R residual

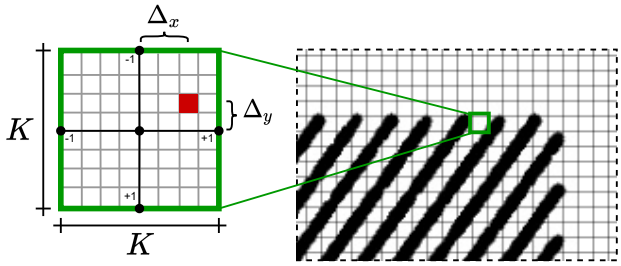


Fig. 3: The input occupancy grid is subdivided into a grid of $K \times K$ cells. For each cell, the waypoint estimation head outputs the probability p of a waypoint presence, as well as the relative horizontal and vertical displacements with respect to the cell center $\boldsymbol{\Delta} = (\Delta x, \Delta y)$.

reduction modules, made by combining a residual module and a reduction module. The final part of the network is made by an additional downsampling block, followed by a transposed convolution upsampling stage, all arranged in a residual fashion. This combination of compression and expansion has been proved very effective for different computer vision tasks such as segmentation [29] and representation learning [32]. Overall, the model performs a dimensionality compression of a factor of 2^{R+1}, where R is the number of residual reduction modules in the main block. The complete backbone structure is detailed in Fig. 1 and Fig. 2.

2.2 Waypoint Estimation

The waypoint estimation is framed as a regression problem, similarly to object detection approaches in computer vision [27]. In particular, given an input occupancy grid map X with dimensions $H \times W$, we subdivide it into a grid of $K \times K$ cells. Each cell is responsible for predicting the probability p that a waypoint falls in that region, as well as its relative horizontal and vertical displacements with respect to the cell center $\boldsymbol{\Delta} = (\Delta x, \Delta y)$. The displacements are defined in the range $[-1, +1]$ and represent a shift relative to half of the cell dimension, with -1 identifying the left/top borders and $+1$ the right/bottom ones. An example of prediction with its correspondent displacements is shown in Fig. 3. Given a prediction $\hat{\boldsymbol{p}}_{\boldsymbol{out}} = (\hat{x}_{out}, \hat{y}_{out})$ in the output reference frame, the waypoint coordinates in the input reference frame $\hat{\boldsymbol{p}}_{\boldsymbol{in}}$ can be reconstructed with the following equation:

$$\hat{\boldsymbol{p}}_{\boldsymbol{in}} = \hat{\boldsymbol{p}}_{\boldsymbol{out}}\, K + \frac{K}{2} + \boldsymbol{\Delta}\frac{K}{2} \tag{1}$$

The waypoint estimation head maps the high-level features extracted with the backbone to the output space with a 1x1 convolution. The backbone compression factor 2^{R+1} corresponds to the grid dimension K. Therefore, the output

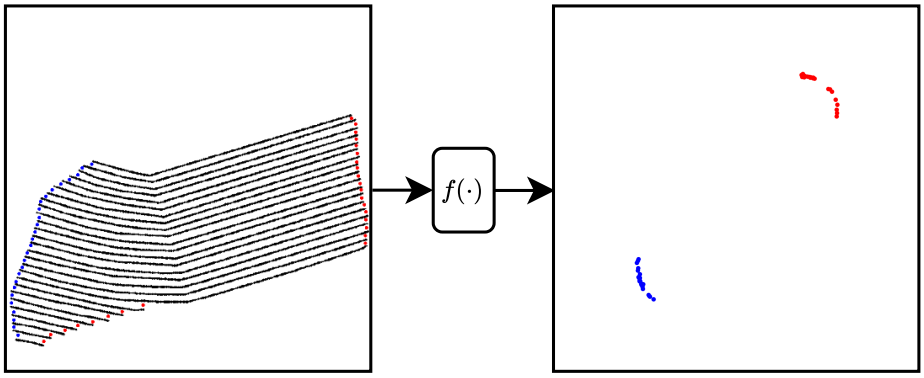


Fig. 4: In the latent space mapped by $f(\cdot)$, points of the same cluster appear closer together with respect to points of the other cluster. The mapping function $f(\cdot)$ is implemented with the backbone and the clustering head together. In this example, the latent space has a dimensionality $D = 2$.

tensor of the estimation branch has a dimension of $H/K \times W/K \times 3$. We apply a sigmoid activation to the probability output and a tanh activation to the displacement outputs. We optimize the network for the waypoint estimation task with a weighted mean squared error loss. For each output cell $\boldsymbol{u_{i,j}}$, the estimation loss is therefore computed as:

$$l_{i,j}^{\text{est}} = \mathbb{1}_{i,j}^{\text{wp}} \lambda \|\boldsymbol{u_{i,j}} - \hat{\boldsymbol{u}}_{\boldsymbol{i,j}}\|_2 + (1 - \mathbb{1}_{i,j}^{\text{wp}})(1 - \lambda)\|\boldsymbol{u_{i,j}} - \hat{\boldsymbol{u}}_{\boldsymbol{i,j}}\|_2 \tag{2}$$

where $\mathbb{1}_{i,j}^{\text{wp}} \in \{0, 1\}$ is an indicator Boolean function evaluating 1 if a waypoint is present in that cell, and λ is the relative constant that weights differently positive and negative cells.

At inference time, we get the list of predicted waypoints by considering all the cells with probability p over a certain threshold t_p. As in standard object detection methodologies, we also apply a suppression algorithm to decrease the number of redundant predictions that typically occur when multiple adjacent cells detect the same waypoint. The algorithm identifies all the groups of predictions with Euclidean distance within a certain threshold t_{sup} in the input reference frame. For each group, the point with highest confidence p is selected, while the remaining predictions are discarded.

2.3 Contrastive Clustering

Once the waypoints are detected, they should be assigned to starting or ending regions. This task can be seen as a simple binary classification, in which the labels represent the two clusters. However, in this scenario the actual assigned label is not relevant, as the only fundamental aspect is whether points of the same group are assigned the same label. The aim is to discriminate the points

of the two regions without caring about which of them is classified as starting or ending. Indeed, an optimal path can be successfully planned regardless of the choice of the starting cluster. This invariance cannot be guaranteed by supervised classification.

For this reason, we model the clustering problem as a supervised representation learning process. Given the two sets of points $A = \{\boldsymbol{p} \mid \boldsymbol{p} \in \text{first cluster}\}$ and $B = \{\boldsymbol{p} \mid \boldsymbol{p} \in \text{second cluster}\}$, we want to find a non-linear mapping $f(\cdot)$ such that

$$d\big(f(\boldsymbol{p_i}), f(\boldsymbol{p_j})\big) \ll d\big(f(\boldsymbol{p_i}), f(\boldsymbol{p_k})\big) \quad \text{for} \quad \boldsymbol{p_i}, \boldsymbol{p_j} \in A \ , \ \boldsymbol{p_k} \in B \tag{3}$$

and vice versa, where d is a distance measure. In the latent space mapped by $f(\cdot)$, points of different clusters are well-separated according to distance d. This means that a simple clustering method such as K-means [20] can successfully discriminate the two groups in the latent space, as shown in Fig. 4. Inspired by the contrastive framework used for unsupervised learning in [6], we select as distance metric d the inverse of the cosine similarity:

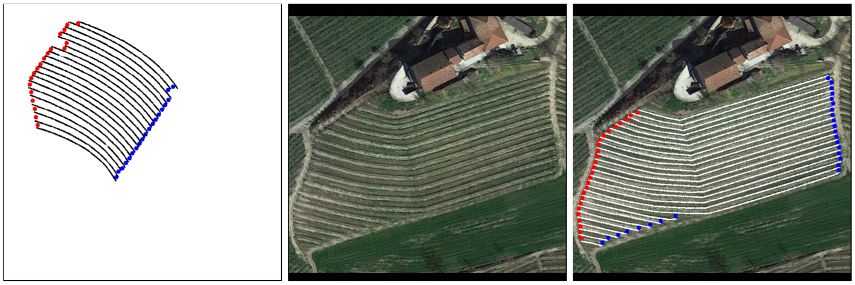
$$\text{sim}(\boldsymbol{u}, \boldsymbol{v}) = \frac{\boldsymbol{u}^\top \boldsymbol{v}}{\|\boldsymbol{u}\|_2 \, \|\boldsymbol{v}\|_2} \tag{4}$$

For each image, we consider the N ground-truth waypoints as independent samples. Given a point $\boldsymbol{p_i}$, we consider as positive examples all the other $N/2-1$ points in the same cluster, and as negative examples the $N/2$ points of the other cluster. Therefore, we define the clustering loss contribution for the sample i as:

$$\begin{aligned} l_i^{\text{clus}} = \frac{1}{N-1} \sum_{\substack{j=1 \\ j \neq i}}^{N} \Bigg[& \quad \mathbb{1}_{\substack{\boldsymbol{p_i}, \boldsymbol{p_j} \in A \\ \vee \, \boldsymbol{p_i}, \boldsymbol{p_j} \in B}} \log\Big(\text{sig}\big(\text{sim}\big(f(\boldsymbol{p_i}), f(\boldsymbol{p_j})\big)\big)\Big) + \\ & + \Big(1 - \mathbb{1}_{\substack{\boldsymbol{p_i}, \boldsymbol{p_j} \in A \\ \vee \, \boldsymbol{p_i}, \boldsymbol{p_j} \in B}}\Big) \log\Big(1 - \text{sig}\big(\text{sim}\big(f(\boldsymbol{p_i}), f(\boldsymbol{p_j})\big)\big)\Big) \Bigg] \end{aligned} \tag{5}$$

where $\mathbb{1}_{\substack{\boldsymbol{p_i}, \boldsymbol{p_j} \in A \\ \vee \, \boldsymbol{p_i}, \boldsymbol{p_j} \in B}} \in \{0, 1\}$ is an indicator function evaluating 1 if $\boldsymbol{p_i}$ and $\boldsymbol{p_j}$ are in the same cluster and 0 otherwise, while 'sig' represents the sigmoid function. Basically, this loss computes the binary cross-entropy of the cosine similarity in the latent space mapped by $f(\cdot)$ for the pair $(\boldsymbol{p_i}, \boldsymbol{p_j})$. $f(\cdot)$ is optimized to push the cosine similarity towards the maximum $+1$ if the points are in the same cluster and towards the minimum -1 otherwise. The final loss is computed over all the pairs (i, j) as well as (j, i) for each input image. This loss can be seen as a variation of the one used in [6,26,37], but instead of N groups with 2 elements each, optimized with categorical cross-entropy and softmax, we have 2 groups with $N/2$ elements each, optimized with binary cross-entropy and sigmoid.

The mapping $f(\cdot)$ is modeled by the clustering head in the output space reference system. The head is composed of two convolutional layers with Mish activation and one final 1x1 convolution with linear activation. The output tensor



(a) (b) (c)

Fig. 5: Examples of curved occupancy grids: synthetic (a) and real-world from Google Maps satellite database without (b) and with (c) manual annotation. Red and blue points are the ground-truth waypoints divided in the two clusters.

of the clustering branch has a dimension of $H/K \times W/K \times D$, where D is the latent space dimensionality.

At inference time, for each waypoint detected in the estimation phase, we select the correspondent feature from the clustering head output. We can predict the clustering assignment by fitting a K-means predictor with two centroids on the selected features. Since we use the cosine similarity in the loss computation, we are optimizing the clustering in the normalized latent space. For this reason, the features should be divided by their Euclidean norm before clustering. This normalization decreases by one the latent space dimensionality, and therefore the minimum number of dimensions D for the clustering head is 2.

3 Experimental Setting

In this section, we present all the details of our experimentation. We describe the datasets used for network training and testing as well as the main hyperparameters adopted during the training phase.

3.1 Dataset Description

Considering the lack of open datasets of row crops bird-eye maps and the time required to manually annotate a large set of real images, we define a method to build realistic synthetic occupancy grids to train the model. We modify the method presented in [24] to extend it to both straight and curved occupancy grids. The generation process can be summarized as follows:

1. sample a uniformly random number of rows $n \in [10, 50]$ and angle $\alpha \in [-\pi/2, \pi/2]$;
2. generate row centers with a random inter-row distance, along the line perpendicular to α and passing through the image center;

3. generate random field borders and find starting and ending points for each row with orientation α;
4. to create curved maps, add a random displacement to the row centers and compute a quadratic Bézier curve with the starting, ending and center points as control points; this ensures that the curves are continuous and smooth;
5. generate the occupancy grid by drawing circles with random radius $r \in [1, 2]$ pixels to model irregularities in the row width
6. create random holes in the rows to emulate segmentation errors or missing plants;
7. compute the $N = 2n$ ground-truth waypoints as the mean points of the lines connecting the ending points of the rows with the adjacent ones.

To further increase variability, we randomly add displacement noise every time we sample a point coordinate during the generation process. We select $H = W = 800$ pixels as input dimension for all the generated images. To investigate the effect of including synthetic curved images in the training set, we randomly generate two independent datasets, one with straight rows only, the other with both straight and curved rows. Overall, each dataset contains 3000 images for training, 300 for validation, and 1000 for testing. In addition to the synthetic data, we manually annotate real row-based images of vineyards and orchards from Google Maps (100 straight and 50 curved). These satellite images are fundamental to test the ability of the network to generalize to real-world scenarios and to prove the effectiveness of the synthetic generation process. Figure 5 shows examples of both synthetic and manually-annotated images.

3.2 Network Training

To select the best hyperparameters, we perform a random search over a set of reasonable values. For all the convolutional layers, we set a kernel size of 5 and channel dimension $C = 16$. For the main block of the backbone, we set the number of residual reduction modules $R = 2$. Therefore, the backbone compression factor and output cell dimension is $K = 2^{(R+1)} = 8$. We set the clustering space dimensionality to $D = 3$. Thus, the output tensors have both a dimension of $100 \times 100 \times 3$. The resulting network is a lightweight model with less than 73,000 parameters. We select Adam [16] as optimizer with a constant learning rate of $\eta = 3\text{e}{-4}$ and batch size of 16. Experimentally, we find more effective to first train the estimation head and the backbone together with the loss of Eq. 2. We set the loss weight to $\lambda = 0.7$ to compensate for the high imbalance in the number of positive and negative cells and stabilize the training. We then freeze the backbone weights and train the clustering head only with the loss of Eq. 5. To highlight the challenge posed by curved scenarios, we independently train the model on both the straight and curved training sets. We train each model for a total of 200 epochs on an Nvidia 2080 Ti GPU using the TensorFlow 2 framework. To obtain significant statistics, we run each training session three times, so that the results can be described in terms of mean and standard deviation.

Table 1: Performance of waypoint estimation on both straight and curved test datasets. We first test the model on our synthetic datasets (Straight Synth, Curved Synth) and then validate the results on manually annotated occupancy grids obtained from real satellite images (Straight Real, Curved Real). For each test set, we compare the results of the model trained on straight rows with those obtained training on curved rows. We report the mean and standard deviation for the Average Precision AP_r, where r is the maximum accepted distance in pixels between predicted and ground-truth waypoints.

Test	**Train**	$\boldsymbol{AP_2}$	$\boldsymbol{AP_3}$	$\boldsymbol{AP_4}$	$\boldsymbol{AP_6}$	$\boldsymbol{AP_8}$
Straight Synth	Straight	0.6404 ± 0.0171	0.9284 ± 0.0088	0.9856 ± 0.0021	0.9991 ± 0.0001	0.9993 ± 0.0001
	Curved	0.5751 ± 0.0241	0.8921 ± 0.0107	0.9743 ± 0.0022	0.9979 ± 0.0001	0.9984 ± 0.0001
Straight Real	Straight	0.5191 ± 0.0288	0.8155 ± 0.0109	0.9116 ± 0.0032	0.9482 ± 0.0017	0.9507 ± 0.0024
	Curved	0.4597 ± 0.0166	0.7634 ± 0.0076	0.8788 ± 0.0089	0.9391 ± 0.0052	0.9433 ± 0.0049
Curved Synth	Straight	0.5143 ± 0.0193	0.8224 ± 0.0236	0.9232 ± 0.0166	0.9726 ± 0.0078	0.9768 ± 0.0065
	Curved	0.5664 ± 0.0226	0.876 ± 0.0066	0.9632 ± 0.0009	0.9937 ± 0.0006	0.9949 ± 0.0006
Curved Real	Straight	0.4685 ± 0.0906	0.7110 ± 0.0625	0.8125 ± 0.0625	0.8802 ± 0.0374	0.8891 ± 0.0355
	Curved	0.5327 ± 0.0269	0.8010 ± 0.0095	0.8881 ± 0.0094	0.9333 ± 0.0026	0.9374 ± 0.0033

4 Results

In this section, we report and comment the main results regarding both waypoint detection and clustering. Visual examples are included as well, to give a qualitative idea of the performance of our model. We extensively test our approach on both straight and curved rows, including a final evaluation on real satellite data. All the related code is open source and available online[4].

4.1 Waypoint Estimation

As regards waypoint estimation, we use Average Precision (AP_r) as principal metric, considering different values of the range threshold r, such that a waypoint is considered correctly detected if its Euclidean position error in pixels is smaller than r. In this way, we can highlight the precision of the model at different levels of proximity. The AP is commonly used for evaluating object detection tasks [10, 18] and is computed as the area-under-the-curve of the precision-recall plot obtained varying the confidence threshold t_p. The waypoint estimation results are reported in Table 1, where each value is detailed with its mean and standard deviation. All the tests are performed setting a waypoint suppression threshold equal to the minimum inter-row distance of the synthetic datasets, $t_{sup} = 8$ pixels.

The first important result is the model trained on curved crops being able to reach an AP_8 of about 94% on all four test scenarios. This achievement confirms the effectiveness of our model far beyond the synthetic training scenario, as real satellite data does not seem to create substantial performance drops (5.7% at

[4] www.github.com/fsalv/ClusterWay

Table 2: Performance of waypoint clustering on both straight and curved datasets, comparing our approach with K-means and the DBSCAN pipeline proposed by [24]. We first test models on our synthetic datasets (Straight Synth, Curved Synth) and then validate the results on real occupancy grids obtained from satellite images (Straight Real, Curved Real). For each test, we compare the results of models trained on straight rows with those obtained training on curved rows. We report the mean adjusted accuracy and clustering error with their standard deviations.

Test	Method	Train	Adjusted Accuracy	Clustering Error
Straight Synth	K-means	Straight	**1.0000 ± 0**	**0 ± 0**
		Curved	0.9913 ± 0.0076	0.6667 ± 0.5774
	DBSCAN	Straight	**1.0000 ± 0**	**0 ± 0**
		Curved	0.9724 ± 0.0240	2.0000 ± 1.7321
	Ours	Straight	0.9994 ± 0.0003	0.0187 ± 0.0114
		Curved	0.9985 ± 0.0006	0.0527 ± 0.0219
Straight Real	K-means	Straight	0.4243 ± 0.1037	26.3333 ± 7.0238
		Curved	0.4635 ± 0.0873	26.0000 ± 5.1962
	DBSCAN	Straight	0.9532 ± 0.0429	2.3333 ± 2.0817
		Curved	0.9585 ± 0.0026	2.0000 ± 0
	Ours	Straight	0.9707 ± 0.0135	1.0400 ± 0.5197
		Curved	**0.9716 ± 0.0123**	**0.7700 ± 0.3012**
Curved Synth	K-means	Straight	0.9714 ± 0.0336	1.0000 ± 1.0000
		Curved	0.9885 ± 0.0199	0.3333 ± 0.5774
	DBSCAN	Straight	0.9563 ± 0.0757	1.3333 ± 2.3094
		Curved	0.8898 ± 0.0337	3.0000 ± 1.0000
	Ours	Straight	0.9823 ± 0.0138	0.3414 ± 0.3278
		Curved	**0.9992 ± 0.0006**	**0.0127 ± 0.0038**
Curved Real	K-means	Straight	0.2443 ± 0.0984	73.3333 ± 29.2632
		Curved	0.2721 ± 0.1493	70.0000 ± 19.5192
	DBSCAN	Straight	0.7247 ± 0.2734	27.0000 ± 25.5343
		Curved	0.5181 ± 0.1061	45.3333 ± 6.6583
	Ours	Straight	0.8571 ± 0.0924	3.4667 ± 2.4437
		Curved	**0.9344 ± 0.0116**	**1.1933 ± 0.1858**

worst). Looking at lower values of r, the synthetic-to-real gap rises to 11.5%, showing how the model is able to estimate synthetic waypoints with higher precision. The model trained on straight crops achieves excellent performance on its corresponding test set and even on real satellite data, but generalizes poorly on curved rows: the precision drop reaches 11% on AP_8 and even 22% considering AP_3. On the contrary, the model trained on curved crops scales very

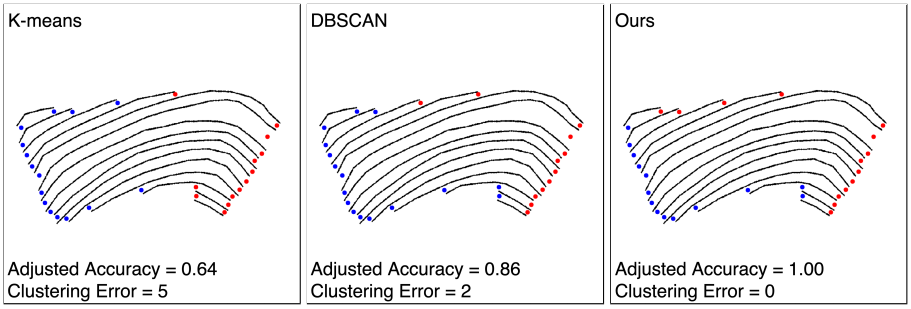


Fig. 6: Examples of clustering on a real-world curved sample: K-means and DBSCAN pipeline [24] are not able to correctly cluster the predicted waypoints; on the other hand, the proposed method correctly assigns the points.

well on straight scenarios. This outcome confirms the importance of training on curved crops to obtain robust models able to cope with challenging situations.

4.2 Waypoint Clustering

As regards waypoint clustering, we adopt two separate metrics. The first is an adjusted binary accuracy, assigning a score of 0 to the worst outcome (all the points in the same cluster, meaning 50% of the points correctly clustered) and 1 to perfect clustering. However, the number of waypoints in a crop is variable and accuracy alone does not give an insight of the distribution of errors among different samples. For example, crops with a small number of waypoints tend to be easier to cluster than dense ones. Considering the fact that full-coverage path planning is possible only if every waypoint is correctly clustered, we add a clustering error metric computing the average number of wrongly labeled points per image. The results are detailed in Table 2. To have a baseline, we compare our approach with the K-means algorithm directly applied in the image reference system and the DBSCAN clustering with geometrical assignment approach proposed by [24]. All the clustering tests are performed setting the confidence threshold to $t_p = 0.4$ and the waypoint suppression threshold to $t_{sup} = 8$ pixels. As for the previous results, each value is reported with its mean and standard deviation.

Our methodology achieves remarkable results, outperforming or at least matching existing solutions in all the testing scenarios. In particular, both the training strategies (based on straight and curved crops) approach perfect clustering on the synthetic straight dataset and generalize well to real crops. On the contrary, K-means, which perfectly works for the well-separated synthetic samples, loses more than half of its adjusted accuracy and presents a very high clustering error when switching to real test rows, mainly due to the irregular shapes typical of real-world vineyards. The DBSCAN pipeline, instead, is able to generalize to straight satellite crops, since the methodology was specifically designed to cope with real-world straight rows.

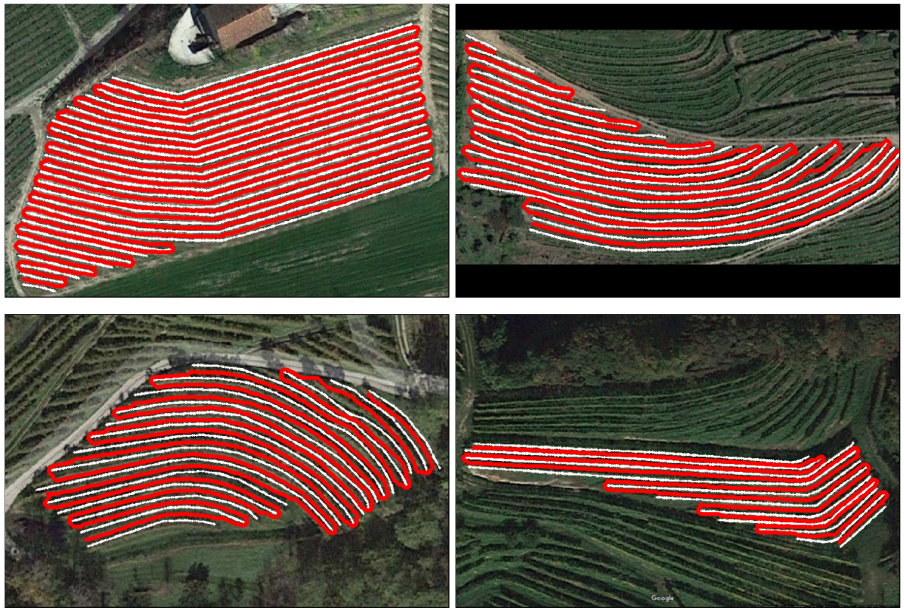


Fig. 7: Examples of full-coverage path planning in real-world curved vineyards taken from Google Maps satellite database.

As regards curved test sets, K-means clustering is totally unable to generalize to the real dataset. At the same time, also the DBSCAN pipeline results drop significantly when switching to real samples, due to its heavy dependence on angle estimation. Our model, trained on straight rows, obtains 0.98 adjusted accuracy and 0.34 clustering error on synthetic data, outperforming both the baselines. However, it struggles to generalize to real crops, reaching an adjusted accuracy of 0.86. On the other hand, the model trained on curved data outperforms the baselines in synthetic and real data, where it achieves an adjusted accuracy of 0.93. This result can be considered extremely positive, taking into account the strong challenges present in satellite data. In particular, a clustering error of 1.19 is remarkably smaller than those obtained by K-means and DBSCAN. In conclusion, these results confirm how the proposed methodology, combined with a well-devised generation process of curved synthetic samples, allows path planning even in challenging scenarios.

4.3 Qualitative Results

To give further insight into the performance of the proposed methodology, we present some qualitative examples on real-world curved samples. Fig. 6 shows a comparison between the three clustering methodologies. K-means and the DBSCAN pipeline are clearly unable to correctly assign points in challenging scenarios. Finally, Fig. 7 shows some examples of full-coverage path planning. The

planning is performed by selecting the points in an A-B-B-A fashion and using the planner proposed by [5]. With geo-referenced maps, the planned path can be converted from the image reference system to a Global Navigation Satellite System (GNSS) reference frame to be used in real-world navigation. All the tests are performed with the model trained on the curved dataset and setting the confidence threshold to $t_p = 0.4$ and the waypoint suppression threshold to $t_{sup} = 8$ pixels.

5 Conclusions

In this work, we propose a novel solution for global path generation in row-based crops using deep learning and contrastive clustering. The problem of path planning in geometrically constrained environments such as vineyards and orchards has been solved through the identification of waypoints at the end of each row. Our deep learning model can simultaneously predict the position of navigation waypoints and cluster them in a unique feed-forward step. To this aim, we train the network on a synthetic dataset of top-view occupancy grids and test it on real-world satellite images, outperforming previous methodologies based on classical clustering by adopting a contrastive loss function. Our extensive experimentation demonstrates that this model successfully generalizes to challenging realistic conditions, including curved and incomplete rows.

Future works may seek the integration of the proposed waypoint generator in a complete pipeline for row-based vineyard and orchard navigation, composed of a first segmentation step to obtain the occupancy grid of the parcel from satellite or UAV imagery and a local planner for intra-row navigation.

Acknowledgements This work has been developed with the contribution of the Politecnico di Torino Interdepartmental Centre for Service Robotics (PIC4SeR) and SmartData@Polito.

References

1. Aghi, D., Mazzia, V., Chiaberge, M.: Autonomous navigation in vineyards with deep learning at the edge. In: International Conference on Robotics in Alpe-Adria Danube Region. pp. 479–486. Springer (2020)
2. Aghi, D., Mazzia, V., Chiaberge, M.: Local motion planner for autonomous navigation in vineyards with a rgb-d camera-based algorithm and deep learning synergy. Machines **8**(2), 27 (2020)
3. Astolfi, P., Gabrielli, A., Bascetta, L., Matteucci, M.: Vineyard autonomous navigation in the echord++ grape experiment. IFAC-PapersOnLine **51**(11), 704–709 (2018)
4. Barawid Jr, O.C., Mizushima, A., Ishii, K., Noguchi, N.: Development of an autonomous navigation system using a two-dimensional laser scanner in an orchard application. Biosystems Engineering **96**(2), 139–149 (2007)

5. Cerrato, S., Aghi, D., Mazzia, V., Salvetti, F., Chiaberge, M.: An adaptive row crops path generator with deep learning synergy. In: 2021 6th Asia-Pacific Conference on Intelligent Robot Systems (ACIRS). pp. 6–12. IEEE (2021)
6. Chen, T., Kornblith, S., Norouzi, M., Hinton, G.: A simple framework for contrastive learning of visual representations. In: International conference on machine learning. pp. 1597–1607. PMLR (2020)
7. Comba, L., Biglia, A., Aimonino, D.R., Gay, P.: Unsupervised detection of vineyards by 3d point-cloud uav photogrammetry for precision agriculture. Computers and Electronics in Agriculture **155**, 84–95 (2018)
8. Deshmukh, D., Pratihar, D.K., Deb, A.K., Ray, H., Bhattacharyya, N.: Design and development of intelligent pesticide spraying system for agricultural robot. In: International Conference on Hybrid Intelligent Systems. pp. 157–170. Springer (2020)
9. Ester, M., Kriegel, H.P., Sander, J., Xu, X., et al.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: Kdd. vol. 96, pp. 226–231 (1996)
10. Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A.: The pascal visual object classes (voc) challenge. International journal of computer vision **88**(2), 303–338 (2010)
11. Ferentinos, K.P.: Deep learning models for plant disease detection and diagnosis. Computers and Electronics in Agriculture **145**, 311–318 (2018)
12. Kamilaris, A., Kartakoullis, A., Prenafeta-Boldú, F.X.: A review on the practice of big data analysis in agriculture. Computers and Electronics in Agriculture **143**, 23–37 (2017)
13. Kamilaris, A., Prenafeta-Boldú, F.X.: Deep learning in agriculture: A survey. Computers and electronics in agriculture **147**, 70–90 (2018)
14. Kang, H., Zhou, H., Wang, X., Chen, C.: Real-time fruit recognition and grasping estimation for robotic apple harvesting. Sensors **20**(19), 5670 (2020)
15. Khaliq, A., Mazzia, V., Chiaberge, M.: Refining satellite imagery by using uav imagery for vineyard environment: A cnn based approach. In: 2019 IEEE International Workshop on Metrology for Agriculture and Forestry (MetroAgriFor). pp. 25–29. IEEE (2019)
16. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
17. Lei, T., Luo, C., Jan, G., Bi, Z.: Deep learning-based complete coverage path planning with re-joint and obstacle fusion paradigm. Front. Robot. AI 9: 843816. doi: 10.3389/frobt (2022)
18. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European conference on computer vision. pp. 740–755. Springer (2014)
19. Luo, L., Tang, Y., Lu, Q., Chen, X., Zhang, P., Zou, X.: A vision methodology for harvesting robot to detect cutting points on peduncles of double overlapping grape clusters in a vineyard. Computers in Industry **99**, 130–139 (2018)
20. MacQueen, J., et al.: Some methods for classification and analysis of multivariate observations. In: Proceedings of the fifth Berkeley symposium on mathematical statistics and probability. pp. 281–297. Oakland, CA, USA (1967)
21. Martini, M., Mazzia, V., Khaliq, A., Chiaberge, M.: Domain-adversarial training of self-attention-based networks for land cover classification using multi-temporal sentinel-2 satellite imagery. Remote Sensing **13**(13), 2564 (2021)

22. 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 **20**(9), 2530 (2020)
23. 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 **8**, 9102–9114 (2020)
24. Mazzia, V., Salvetti, F., Aghi, D., Chiaberge, M.: Deepway: a deep learning waypoint estimator for global path generation. Computers and Electronics in Agriculture **184**, 106091 (2021)
25. Mohanty, S.P., Hughes, D.P., Salathé, M.: Using deep learning for image-based plant disease detection. Frontiers in plant science **7**, 1419 (2016)
26. Van den Oord, A., Li, Y., Vinyals, O., et al.: Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748 **2**(3), 4 (2018)
27. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 779–788 (2016)
28. Riggio, G., Fantuzzi, C., Secchi, C.: A low-cost navigation strategy for yield estimation in vineyards. In: 2018 IEEE International Conference on Robotics and Automation (ICRA). pp. 2200–2205. IEEE (2018)
29. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical image computing and computer-assisted intervention. pp. 234–241. Springer (2015)
30. Sparrow, R., Howard, M.: Robots in agriculture: prospects, impacts, ethics, and policy. precision agriculture **22**(3), 818–833 (2021)
31. Tripicchio, P., Satler, M., Dabisias, G., Ruffaldi, E., Avizzano, C.A.: Towards smart farming and sustainable agriculture with drones. In: 2015 International Conference on Intelligent Environments. pp. 140–143. IEEE (2015)
32. Tschannen, M., Bachem, O., Lucic, M.: Recent advances in autoencoder-based representation learning. arXiv preprint arXiv:1812.05069 (2018)
33. Ünal, Z.: Smart farming becomes even smarter with deep learning—a bibliographical analysis. IEEE Access **8**, 105587–105609 (2020)
34. Vidović, I., Scitovski, R.: Center-based clustering for line detection and application to crop rows detection. Computers and electronics in agriculture **109**, 212–220 (2014)
35. Virlet, N., Sabermanesh, K., Sadeghi-Tehran, P., Hawkesford, M.J.: Field scanalyzer: an automated robotic field phenotyping platform for detailed crop monitoring. Functional Plant Biology **44**(1), 143–153 (2017)
36. Woo, S., Park, J., Lee, J.Y., So Kweon, I.: Cbam: Convolutional block attention module. In: Proceedings of the European conference on computer vision (ECCV). pp. 3–19 (2018)
37. Wu, Z., Xiong, Y., Yu, S.X., Lin, D.: Unsupervised feature learning via non-parametric instance discrimination. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3733–3742 (2018)
38. Zhai, Z., Martínez, J.F., Beltran, V., Martínez, N.L.: Decision support systems for agriculture 4.0: Survey and challenges. Computers and Electronics in Agriculture **170**, 105256 (2020)
39. 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. pp. 223–230. Springer (2019)