Using Agents and Unsupervised Learning for Counting Objects in Images with Spatial Organization

Eliott Jacopin[1][a], Naomie Berda[1], Léa Courteille[1], William Grison[1], Lucas Mathieu[1], Antoine Cornuéjols[1][b] and Christine Martin[1][c]

[1] *UMR MIA-Paris, AgroParisTech, INRA,*
Université Paris-Saclay,
75005, Paris, France
{*antoine.cornuejols,christine.martin*}*@agroparistech.fr*

Keywords: Image Processing, Computer Vision, Counting Objects, Multi-Agent Systems, Unsupervised Learning.

Abstract: This paper addresses the problem of counting objects from aerial images. Classical approaches either consider the task as a regression problem or view it as a recognition problem of the objects in a sliding window over the images, with, in each case, the need of a lot of labeled images and careful adjustments of the parameters of the learning algorithm. Instead of using a supervised learning approach, the proposed method uses unsupervised learning and an agent-based technique which relies on prior detection of the relationships among objects. The method is demonstrated on the problem of counting plants where it achieves state of the art performance when the objects are well separated and tops the best known performances when the objects overlap. The description of the method underlines its generic nature as it could also be used to count objects organized in a geometric pattern, such as spectators in a performance hall.

1 INTRODUCTION

Object counting is an important task in computer vision motivated by a wide variety of applications such as crowd counting, traffic monitoring, ecological surveys, inventorying products in stores and cell counting. In agriculture, for instance, Unmanned aerial vehicles (UAVs) are very alluring (Sankaran et al., 2015). UAVs allow for cheaper image recording, enabling flexible and immediate image processing (Gnädinger and Schmidhalter, 2017). One critical challenge lies in the automatic counting of plants in fields, if possible at various stages of development. Indeed, the number of plants is important information to evaluate the physiological characteristics of the crop, and ultimately, the final yield.

However, many challenges are associated with object counting. Objects are often variable in terms of shape, size, pose and appearance. Also, they may be partially occluded. Taking again examples from agriculture, the performance of automatic plant counting is affected by the presence of weeds and blurry effects, and different estimates can be obtained for different growth stages.

The range of potential applications for the object counting task has motivated researchers across various fields to develop several methods. They can be categorized mainly into two classes: detection-based and regression-based.

In the *detection-based* approach, a classifier is learned to recognize the presence of the object(s) of interest in a sub-image or window, and then this window is scrolled through the image in order to count the number of recognized objects. There are however difficulties associated with this approach. First, it requires (very) numerous labeled training examples, often in the form of manually drawn bounding boxes or pixel annotations, which are notoriously costly to acquire. Second, classification of objects is itself a challenging task because of the variability of the appearances of objects, presence of noise and possible partial occlusions. Besides the selection of relevant descriptors, such as wavelets, shapeless, edgeless, and so on, it requires also the fine-tuning of the parameters of the classification algorithm. Finally, the choice of the size of a sliding window and of the scrolling process can be tricky.

In contrast, *regression-based* methods do not try

[a] https://orcid.org/0000-0002-4568-283X
[b] https://orcid.org/0000-0002-2979-3521
[c] https://orcid.org/0000-0003-3956-4789

to detect individual objects but, instead, attempt to directly estimate the number of objects of interest from an overall characterization of the image. The idea is to learn a mapping between features extracted from the images and the counts. This overcomes most of the difficulties of detection-based methods and, in recent years, these methods have defined the state-of-the-art performances, specially through the use of convolutional neural networks. However, these methods still require lots of training images, and advanced expertise to train deep neural networks. They also often assume fixed object sizes and have to be retrained when the objects of interest change.

In this paper, we introduce a novel approach, valid when the objects of interest have regular spatial relationships, like spectators in a performance hall, goods on the shelves of a retail store or plants in fields. It works in two phases. First, the approximate spatial relationships between objects are estimated. Second, based on the structure thus found, a multi-agent based approach is used where the structure determines the initial positions of the agents as well as a hierarchy of control agents and therefore a set of communication channels between the agents. Each agent is a weak classifier which guesses if it is positioned over an object of interest in the image and can confirm or deny its guess through exchanges with other agents. The second phase is iterative until the agents are no longer undergoing any changes. The number of final agents gives the number of detected objects.

The approach has been experimented on the plant counting task. We tested its value both on real images taken from UAVs (on sunflower fields) and on synthetic images that allow one to vary the conditions: size of the plants, proportion of weeds, mean distance between rows and between plants, lighting conditions and size of the shadows.

The advantages of the approach are that:

1. it does not require numerous training images since the determination of the structure is unsupervised and the agents themselves are simple detectors.
2. it easily adapts to various conditions on the structure, nature of the objects, their size and appearance
3. it achieves high performances over the variety of experimental conditions tested.

These good properties come from the assumption that a regular structure exists among objects. The approach should therefore not work on crowd counting, or on cells counting for instance. But when a regular structure exists, this knowledge brings a power that should not be wasted.

The paper is structured as follows. Section 2 describes the task of plant counting. Then, Section 3 presents the proposed approach. Information about the generation of synthetic datasets used in the experiments is provided in Section 4 and the results of the experiments are reported in Section 5. Finally, Section 6 concludes and gives perspectives on future works.

2 THE PLANT COUNTING TASK

Automatically counting plants from aerial images is challenging. Even when a field is planted with only one type of plant, such as sunflowers, the plants vary in size and shape. In addition, weeds are also present and sometimes quite abundantly. Finally, the ground is not always flat, which can introduce spurious variations in the lightning and appearance of the plants.

There exists quite a few studies that report the impact of the choice of camera and the UAV's flying height on the quality of the images that can be produced, and we do not delve into these considerations here (see for instance (Dvořák et al., 2015; Christiansen et al., 2017)). However, due to the lack of publicly available datasets for the task of plant counting, we have used a data set provided by the *Terres Inovia* research institute. Figure 1 provides an example of an aerial image of a sunflower field. One can see rows of plants, here in a rather late stage with overlap between plants, shadows of various sizes and patches of weeds, especially on the left side of the image.

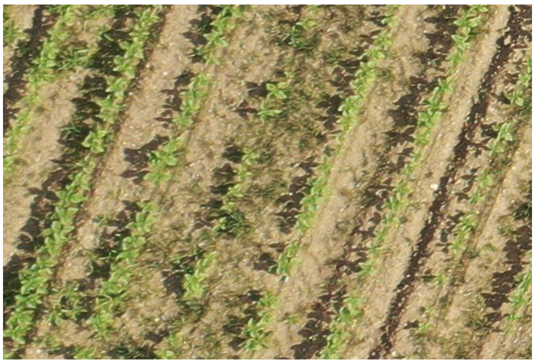


Figure 1: Example of an aerial image from a sunflower field.

As explained below, in Section 4, we also coded a generator of synthetic images that closely mimics real images. This allows us to produce as many images as needed, and above all to control parameters such as the lightning conditions, the plants growth stage, and the distance between plants and between rows.

3 THE METHOD

3.1 Analyzing the Spatial Relationships

Crop fields usually exhibit a geometrical design resulting from the agricultural tool that helped making them. Whether they are traced with a hand tool or a machine, the rows of a crop field are indeed usually parallel to each other and evenly spaced. In addition, crops are planted on the basis of a target density which induces an even distance between two consecutive plants. Taking this into account potentially brings valuable information to help detecting, and therefore counting, plants on an image captured by an UAV.

One main theme of this paper is to underline the interest of researching and exploiting information on the geometry of the objects in the images to be analyzed. For crop fields images, in order to estimate the inter-rows and inter-plants distances, the method presented begins with i) isolating the green areas of the images; then ii) rotating the images enough for the rows to be collinear with the Y axis; and, finally, iii) applies a Fourier Transform (FT) analysis on the signal produced by projecting the coordinates of the green pixels on the X and Y axis.

3.1.1 Image Segmentation

Before estimating the inter-rows and inter-plants distances, it is necessary to identify the areas of the images corresponding to plants. To that end, we used the vegetation index *Excess Green* (*ExG*) in association with Otsu's automatic segmenting method (Otsu, 1979). The *ExG* vegetation index consists in replacing every pixels of an image by its amplified green value: $ExG = 2g - r - b$. The association of *ExG* with Otsu's segmentation method for crop fields image segmentation has been studied and validated in previous works (Guerrero et al., 2012; Guijarro et al., 2011; Pérez-Ortiz et al., 2016). At the end of the segmentation process, the RGB crop fields images are transformed into black and white images where the white pixels are expected to correspond to a plant (crop or weed). In the rest of this paper, these black and white images will be referred as *Otsu images*.

3.1.2 Vertically Adjusting the Images

To ease the estimation of the inter-rows and inter-plants distances, the rotation of all the images of the datasets was computed in order for the crop rows to be oriented along the Y axis. To do so, each Otsu images was rotated between $0°$and $180°$yielding an associated score for every $1°$. This score is the ratio of columns of the rotated image that are occupied by at least one white pixel to the total number of columns (number of pixels along the X axis) of this rotated image. It lies between 0 (for a pitch black image) and 1 (at least one white pixel per column). Since rows of crop fields are usually defined as lines, the minimum value of this score for a field indeed corresponds to the case when the rows are colinear with the Y axis. This method succeeds as long as two consecutive rows do not overlap with each other or weed do not cover all the inter-rows space. Should this happen, one can apply a filter to the Otsu images in order to only keep the skeleton of the crop rows in white. This can be implemented with, for example, the midpoint encoding suggested in (Han et al., 2004).

3.1.3 Estimating the Inter-Rows and Inter-Plants Distances

Crop fields present regular structures characterized by a periodical geometry among rows as well as among plants in each row. The problem is to automatically identify the rows and the plants from an (Otsu) image where neither are labeled. The approach relies on having a prior knowledge of the type of structure to expect, and then to devise a procedure to extract that type of signal from the input. Here, linear structures, corresponding to rows, are expected, with an almost constant inter-rows distance. The procedure depends therefore on first detecting lines, and then, thanks to a Fourier analysis, filtering out the lines that are periodically positioned and thus have a high likelihood of corresponding to rows, and not, say, to weeds.

Items **1** and **3** on Fig. 2 illustrate how a periodic signals is detected out of a vertically adjusted Otsu image. Since the rows are assumed to have been realigned with the Y axis, the periodicity of the positions of the rows appears on the X axis: the peaks of the density distribution of the white pixels on the X axis mirror the positions of the rows on the image (item **1**). The inter-rows distance is computed using a Fourier analysis on the density distribution and keeping the maximal frequency thus found. Once the period is identified, a search for the positions of the local maxima of the density distribution is undertaken yielding the estimated position of the rows along the X axis.

A second steps aims at estimating the inter-plants distance. This estimation is conducted as before, except this is now the projections on the Y axis of the white pixels attributed to each row that are considered (items **3** and **4**).

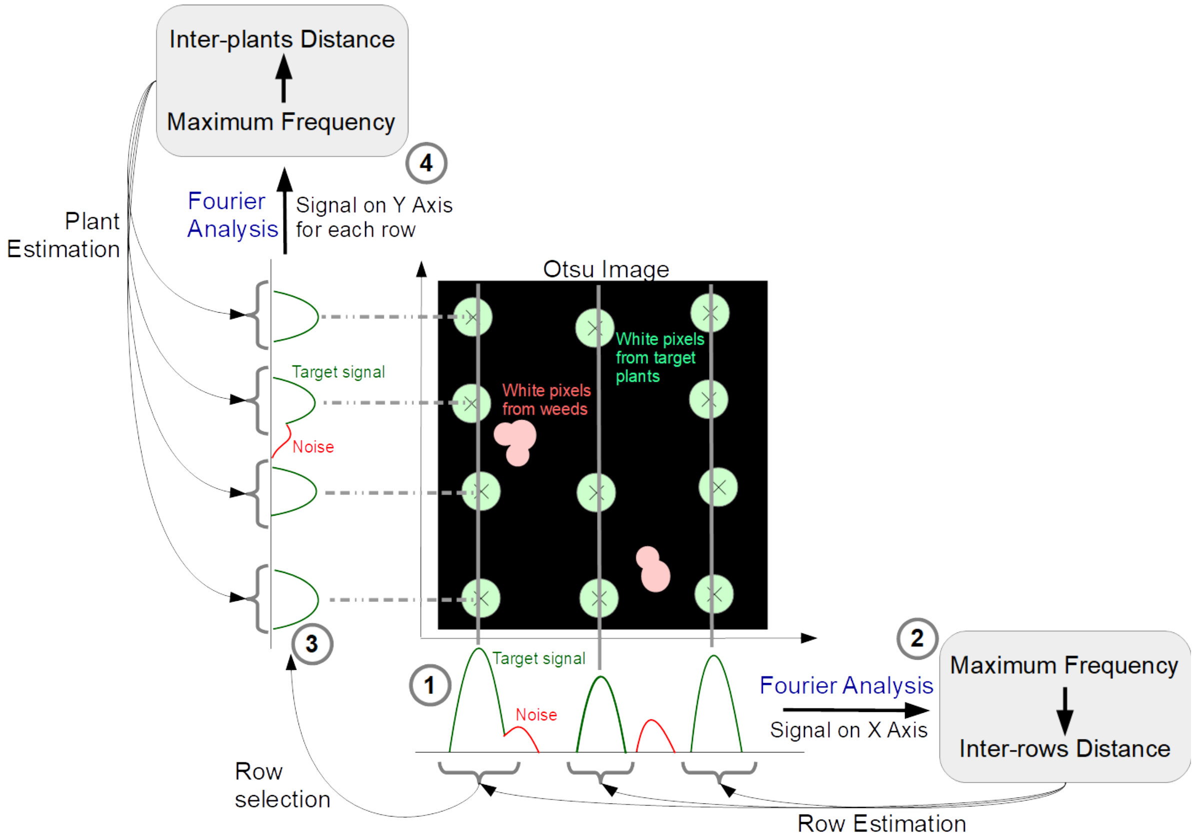


Figure 2: Fourier Analysis on the X and Y axis. The signal processed by the Fourier Transform is made from the projection of the white pixels of the Otsu images on the X and Y axis.

3.2 A Multi-Agent Approach

We advocate the use of a multi-agent system which takes advantage of the knowledge gathered on the geometry in the image. In the context of the plant counting task, we identified four types of agents that follow the organization policy of a corporate hierarchy, as shown in Fig. 3. The agent at the top of the system is called the *Director Agent* (DA), then come the *Row Agents* (RAs), the *Plant Agents* (PAs) and finally the *Pixel Agents* (PXAs). Following the corporate hierarchy scheme, each agent of one layer either acts on its own or receive orders from an agent of the upper layer: there is no communication between agents of the same layer. The environments in which the agents act are the vertically adjusted Otsu images.

3.2.1 The Director Agent

The DA can initialize or destroy RAs, and decides when to stop the simulation. Typically, the DA initializes or destroys RAs at the beginning of the simulation according to the predictions made using the Fourier analysis performed on the Otsu images (see 3.1.3). Since the DA has access to all the information provided by the agents of the lower levels, it is also the one that computes the *inter-plants critical distance* (IPCD) (see below).

Managing the Row Agents At the beginning of the simulation, the DA analyses the rows detected using the Fourier analysis in an attempt to exclude the false positives: rows that are only made out of weeds. These will be positioned in between real RAs (rows consisting in plants). Therefore, everywhere a false RA exists, the inter-rows distance will be decreased. Thus, the DA clusters the candidate RAs resulting from the Fourier analysis into two groups according to the distance separating the i-th and the $(i+1)$-th RA. The k-means clustering method is used on the 1-D array built with all the observable inter-rows distance. Once the two groups are formed, a two-sample t-Test is applied under the classic null hypothesis that the groups' means are equal. If the hypothesis is rejected for a p-value threshold π, the DA eliminates the RAs involved in the cluster that has the minimal mean.

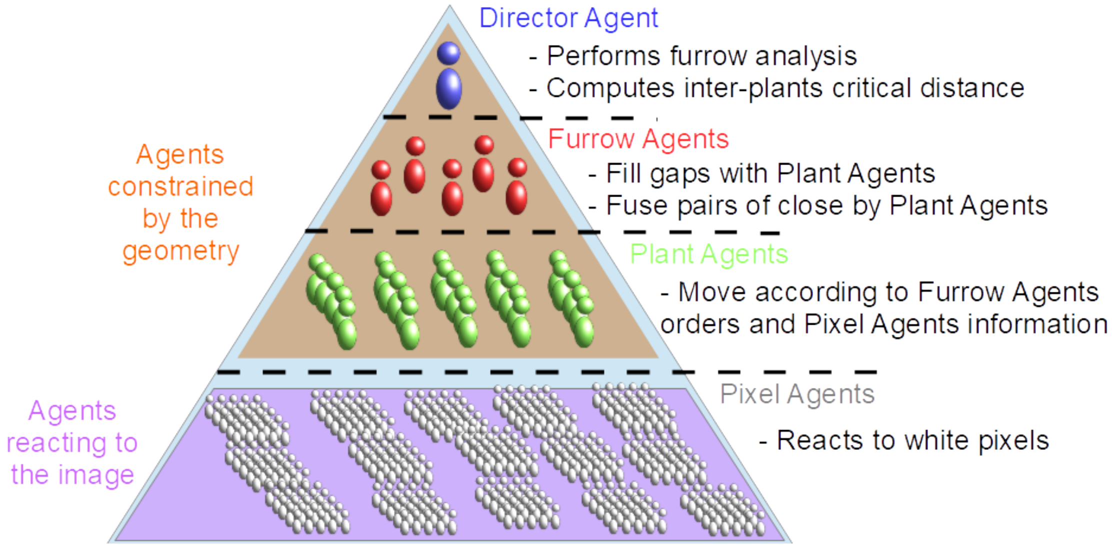

Director Agent
- Performs furrow analysis
- Computes inter-plants critical distance
Furrow Agents
- Fill gaps with Plant Agents
- Fuse pairs of close by Plant Agents
Plant Agents
- Move according to Furrow Agents orders and Pixel Agents information
Pixel Agents
- Reacts to white pixels
Agents constrained by the geometry
Agents reacting to the image

Figure 3: Hierarchical architecture of the multi-agent system.

Computing the Inter-Plants Critical Distance (IPCD). Most of the decision functions used by the agents depend on the IPCD. It is set equal to the maximum of the density distribution of the inter-plant distances.

3.2.2 The Pixel Agents

The PXAs, at the lowest level of the hierarchy, and sensing the Otsu image. They are instantiated by a PA. They become *activated* if they are positioned on a white pixel and their position is determined by the PA they are dependent upon.

3.2.3 The Plant Agents

The PAs are ultimately the most important agents for the plant counting task. The number of PAs at the end of the simulation determines the number of plants detected in the frame of the image. Each PA has under its supervision a group of PXAs that is centered on the position of the PA. The role of the group of PXAs is to guide the PA toward the most white parts of an Otsu image (i.e. guiding them toward plants). Therefore, at step $i+1$ of the simulation, a PA moves on the mean point of all its activated PXAs at step i:

$$(PA_x^{i+1}, PA_y^{i+1}) = (\frac{1}{n}\sum_{PXA\in\mathcal{A}} PXA_x^i, \frac{1}{n}\sum_{PXA\in\mathcal{A}} PXA_y^i) \quad (1)$$

with $\mathcal{A}$ the set of activated PXAs. The x and y are the positions of the agents. Finally, a PA can decide to decrease or increase its sensing area by eliminating PXAs or by initializing new PXAs. In our simulations, we set the goal of the PA to have between 20% and 80% of its PXAs activated. The hope is that the PA will frame the area covered by a plant on the Otsu image.

3.2.4 The Row Agents

RAs are instantiated by the DA according to the rows detected by the Fourier Analysis (Fig. 2, item **2**). In turn, each RA first initializes as many PAs as were detected using the Fourier analysis (Fig. 2, item **4**). In a second step, it instantiates additional PAs at its edges in order to cover the whole length of the image. The Fourier analysis may indeed miss plants at the edges of the rows detected. The additional instantiated PAs are evenly spaced at 1.1ν times the IPCD, ν being the PAs fusing factor (see next paragraph). At every step of the simulation, RAs will eliminate the PAs that are located in black areas of the Otsu image: PAs with less than a proportion δ of activated PXAs .

Filling and Fusing PAs
During the simulation, one RA may consider that the distance between two consecutive PAs is either too large or too small. It then decides to either fill in the gaps with new PAs of fuse the two involved PAs according to the following decision function:

$$Decision = \begin{cases} \text{Fill} & if\ |PA_y^{i+1} - PA_y^i| > \mu\, IPCD \\ \text{Fuse} & if\ |PA_y^{i+1} - PA_y^i| < \nu\, IPCD \end{cases} \quad (2)$$

with μ and ν the filling and fusing factor respectively. The PAs instantiated during a *Fill* action are evenly spaced at 1.1ν times the IPCD from the i-th PA and as long as they do not overcome the $(i+1)$-th PA.

Constraining PAs Movements

In a crop field, the rows usually exhibit a linear shape, aligned with the Y axis when adjusting the images (Section 3.1.2). The plants that are part of the same row are thus expected to be aligned. As a consequence, a RA can constrain the moves of the PAs that it supervises in order to keep them as aligned as possible. At each step of the simulation, the PAs are first free to move in the direction that their PXAs are guiding them. Then, the RA analyses the moves of its PAs to compute whether most of them moved to the right or to the left of the detected alignment. All the PAs that moved in the opposite direction of the majority are re-positioned on the mean X coordinate of the PAs that are part of the majority. That way, the PAs move as a group in the same direction with regards to the X axis.

3.2.5 Running the Simulation

The simulation consists in a sequence of actions that the agents carry out in a deterministic order (Algo. 1). The final count of the plants could be determined when the number of PAs remain constant, with no destruction nor initialization between the i-th and $(i+1)$-th steps.

4 SYNTHETIC DATASETS

Counting objects in an image is a difficult task, and solving it by automatic means requires using large data sets with at the very least hundreds of images, with thousands of objects, each of them to be labeled. In the case of plant counting, there are no publicly available data sets. This entails a lack of labeled training data and a problem of reproducibility of experiments.

The solution we adopted is to use a virtual environment engine to generate artificial crop fields. Game engines are indeed nowadays able to generate very realistic images, with the advantage to have an automatic labeling of the objects of interest. Here we chose to use the game engine Unity (Technologies, 2020) in which we simulated an UAV capturing pictures. In the following sections, we provide an overview of the implementation of the generator.

4.1 The Field Generator

Unity is a game engine based on the notion of *game objects* (GO) to which are attached *components* (CP). Different GOs exhibit different behaviours based on the nature of the CPs attached to them. There is a wide variety of built-in CPs in Unity which help setting up the physics (e.g. gravity), the lights, 3D models, artificial intelligence, etc. At a high level, a virtual environment in Unity is only a set of GOs that behave as the designer of the environment had decided. It is also possible to attach C# scripts to GOs that govern CPs when one wishes for something more specific than the built-in CPs. In our case, the crop field generator (CFG) is a C# script that we attached to a single GO in the virtual environment. The script takes as input a set of parameters which we can modify at will to quickly generate a wide range of crop fields. The parameters mainly manage the surface of the field, the virtual crop, the weed, the sun and the simulated drone.

Algorithm 1: Simulation

```
Input: max_nb_steps, $\mu$, $\nu$, $\delta$, $\pi$
initialize DA, RAs, PAs, PXAs
/* Sec. 3.2.1                                  */
AnalyseRows($\pi$)
ComputeIPCD()
AnalyseRowsEdges($\nu$, IPCD)
StopSimu ⟵ False
RE_Eval ⟵ False
i ⟵ 1
while $i \leq max\_nb\_steps\ \&\ StopSimu = False$ do
    /* Sec. 3.2.3                              */
    MoveToMeanPoint()
    /* Sec. 3.2.4                              */
    ConstrainPAsXMovement()
    FillOrFusePAs($\mu$, $\nu$, IPCD)
    /* Sec. 3.2.3                              */
    AdaptSize()
    /* Sec. 3.2.4                              */
    DestroyLowActivityPAs($\delta$)
    if $Nb\_PAs_i - Nb\_PAs_{i-1} = 0$ then
        if $RE\_Eval = False$ then
            DA_ComputeIPCD()
            RE_Eval ⟵ True
        else
            StopSimu ⟵ True
        end
    else
        RE_Eval ⟵ False
    end
    i⟵ i+1
end
```

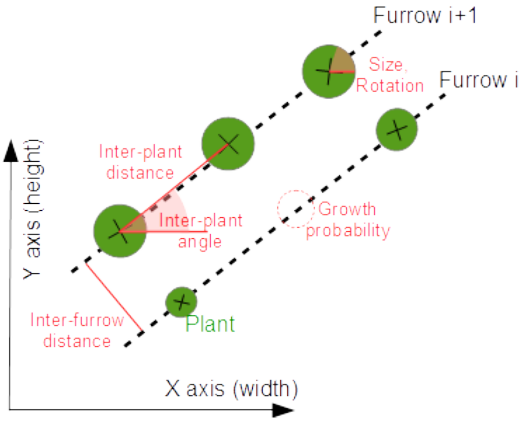
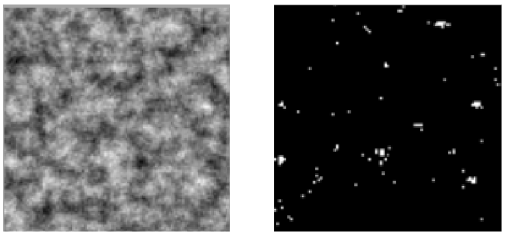


Figure 4: Parameters involved in the placement of crops along rows. The red labels are parameters undergoing randomization.



(a) Perlin Noise Texture (b) Thresholded Perlin Noise Texture

Figure 5: Example of a Perlin Noise Texture used to place weeds.

4.1.1 Placing the Crops

Crops position in the field are based on several parameters shown in red on Figure 4. All parameters except the *growth probability* are drawn randomly over an interval centered on a value specified as part of the CFG options. For example, the *inter-row distance* between a pair $(i, i+1)$ of rows is decided as a random draw when it is time to instantiate the $(i+1)$-th row. Similarly, the *inter-plant distance* as well as the *inter-plant angle* between a pair $(j, j+1)$ of crops are chosen randomly when instantiating the $(j+1)$-th crop. During instantiation, we test whether the crop actually grows according to the *growth probability*. If the test is successful, the *rotation* as well as the *size* of the 3D model are also decided by a random draw on centered intervals. The field is filled with rows and plants as long as the calculated coordinates of the 3D models are within the boundaries of the plane making up the virtual field.

4.1.2 Placing the Weeds

Weeds cannot be expected to follow any geometry at the scale of the field but they can regularly be found clustered together. This is why we used the Perlin Noise (Perlin, 1985) to generate spaces on the crop field where the weeds would be present. Perlin Noise was originally implemented to create 3D textures that would feel real (Perlin, 1985) and has since been widely used in visual effects. Perlin Noise is a type of gradient Noise that can be adapted to any dimension. For this work, we used the 2D perlin noise and generated textures such as those shown in Fig. 5a. In this texture, the gray colors are scaled along the noise values between 0 and 1. Once the texture is obtained, a threshold (a parameter of the field generator) determines the spaces containing weeds. Figure 5b shows the thresholded version of the texture of Figure 5a, each white pixel being a potential weed. The actual presence of the weed at this location is decided on the basis of a *weed growth probability*.

4.1.3 Placing the Sun

In Unity, the sun is a type of light source among others and its elevation and azimuth can be simulated using the rotation parameters available through the *Transform* CP. Therefore, in order to generate a variety of differently shaded crop fields in our synthetic datasets, we added a couple of parameters to control both the elevation and the azimuth.

4.1.4 Simulating the UAV

The altitude of the UAV is the height at which the camera takes pictures. The focal length of the camera and the size of the sensor can also be configured using two parameters. In addition, the flight plan of a UAV generally takes into account the proportions of overlap on the Y and X axes between the images it captures to avoid having crops cut in half at the edge of the images. Two more parameters control this overlap. Figure 6 describes the UAV flight plan.

4.2 Content of the Datasets

Plants may overlap as the plants grow. Plants at an early stage of development are supposed to be well separated from each other. However, during growth, the foliage of one plant may reach and then overlap that of its neighbours. It is assumed that the overlap interferes with the signal used by the counting method, and previous studies on automatic counting of plants from UAV images have raised that the difficulty of the task increases with the proportion of crop overlap (García-Martínez et al., 2020). In order to assess this effect on our method, we generated three datasets with three different levels of overlap between crops.

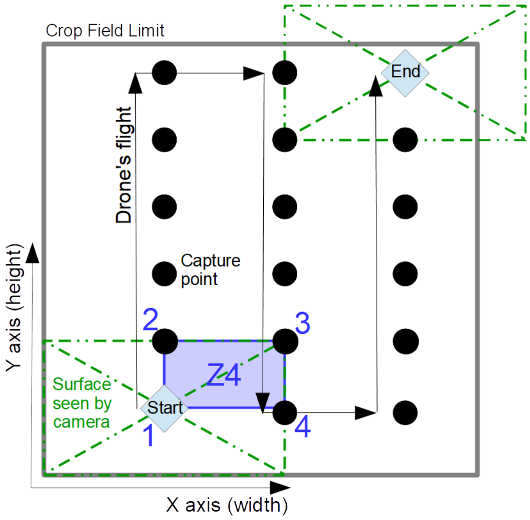


Figure 6: Scheme of a UAV flight plan above the virtual crop field. The start position is calibrated to capture the bottom left corner of the field. The other capture points are calculated depending on the image overlap configured on the X and Y axis (here, 50% on both). As a result, the images of the upper and right limit of the field may go over these. The area named *Z4* is subsequently captured four times, one by each of the four captured points numbered in blue.

The plants are separated (S) from each other in the first dataset; they overlap for some leaves and do not overlap for others (B) in the second datase; and finally, the third dataset exhibits overlap (O) between neighbouring plants. The dataset (S) is considered easy, (B) is intermediate and (O) is difficult. Aside from varying the scale of the plant 3D model to simulate its growth, the parameters used to generate the fields are similar for all three datasets. Each crop field was generated with an inter-rows distance of 70 cm and an inter-plants distance of 20 cm with 5% variability. This yields a target average of 7 $plants/m^2$ which matches typical sunflower crop fields. The plant growth probability was set to 0.8. The Perlin noise threshold used to generate the surfaces where weed grows was set to 0.75 while the weed growth probability was set to 0.6. In each of these datasets, 100 crop fields were generated, and from each of them four images were taken. So, each dataset contains 400 images which amounts to 1200 images in total. To take pictures of the virtual fields, we simulated a short drone flight plan that covers the lower left corner of the field as it moves once along the height and width of the field (see the blue numbers on Fig. 6). We have configured the motion of the simulated drone to overlap the image by 50% along both their height and width, as is usual with images from UAVs.

Fig. 7 gives an example of an image of a virtual crop field. Fig. 7b is the same image after an Otsu filter has been applied and the image has been reoriented so that the rows are aligned with the Y axis. (see sections 3.1.1 and 3.1.2).

5 EXPERIMENTS AND RESULTS

The method we propose aims at counting objects that are linked by spatial relationships in an image. It is a two steps method with the first phase that detects and estimates the spatial structure, and the second phase which, starting from this structure identifies the objects.

The goal of the experiments carried out is threefold. *First*, to assess the performance of the first phase alone in counting plants, *second*, to measure the added value of the second phase based on a multi-agent approach, and, *third*, to look at the gain of performance, if any, when parts of a field are covered by multiple passes of the UAV and a redundancy of information follows (see area *Z*4 in Figure 6 for an example).

But first, we briefly present the rules under which we considered that the method had successfully detected a plant and how the counting performance was measured.

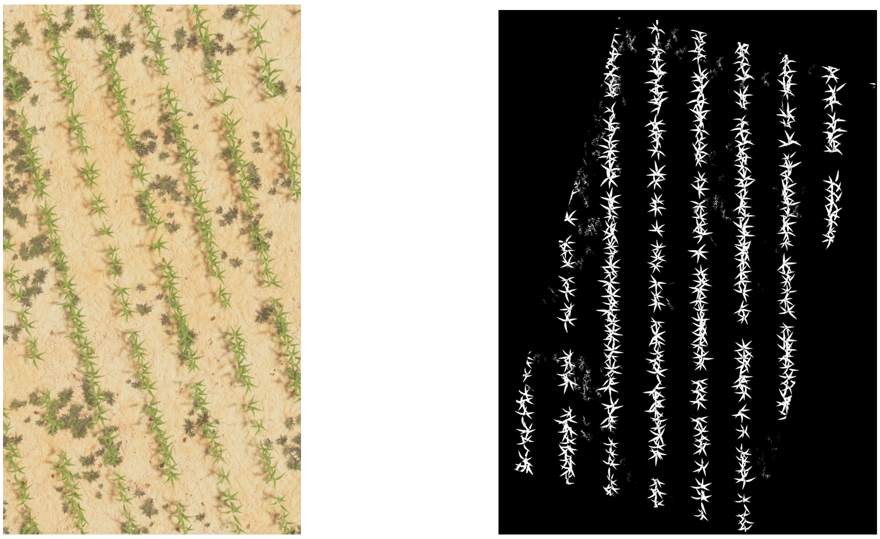
5.1 Plant Detection Rules and Performance Scores

In order to measure the performance of the Fourier analysis alone, the rule is that if the plant position, which is known in synthetic data sets, falls within a 40 square pixel area of a predicted position, then this is counted as a true positive (TP).

For the multi-agent system, we considered that a PA detected a plant if that plant was located within the sensing area defined by the PXAs of the PA. If two PA happen to detect the same plant, then only one PA is counted as TP and the other is counted as a false positive (FP). Additionally, a PA or a prediction from the Fourier analysis that does not contain a plant in their sensing area are also considered as FP. Finally, a plant that has not been detected is counted as a false negative (FN). In addition to these three indicators, three scores are computed:

$$\text{Detection Accuracy} = \frac{TP}{\text{Total number of PAs}} \tag{3}$$

$$\text{Detection Recall} = \frac{TP}{\text{Total number of Plants}} \tag{4}$$



(a) An image of a virtual field (b) Otsu image vertically adjusted

Figure 7: Example of a an synthetic image and its vertically adjusted Otsu image.

$$\text{Counting Accuracy} = \frac{\text{Total number of PAs}}{\text{Total number of Plants}} \quad (5)$$

These scores are later referenced as *DAc*, *DR* and *CA* respectively.

The performance of the proposed algorithms is measured on areas of the field that are covered four times due to the 50% overlap between images on the X and Y axis. For instance, in the example of Figure 6, the area *Z4* is a candidate image satisfying these constraints.

In the following, we compare the performances of the Fourier analysis alone (Section 5.2), of the multi-agent approach from a single image of the area (Section 5.3), and of a technique that takes into account that several images (up to four) can cover a given area (Section 5.4).

5.2 Detecting the Spatial Structure and Counting

As explained in Section 3.1.3, we use Fourier analysis to approximate the spatial structure in an image. We first try to discover the rows and then to locate plants within the presumed rows. This relies on the analysis of the density distribution of the projection of the white pixels from an Otsu image on the X or Y axis (Fig. 8 shows such a density distribution (in yellow) as well as the detected peaks (in blue)). Notice that the largest peaks indeed correspond to rows, but that weeds can also produce peaks, albeit smaller ones.

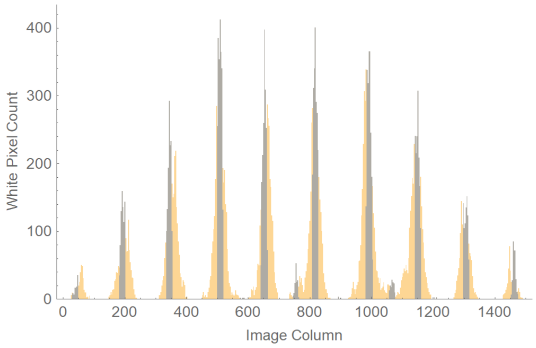


Figure 8: Example of row detection thanks to Fourier analysis. The histogram in yellow results from the projection of the white pixels of an Otsu Image on the X axis. The blue parts of the histogram are the detected rows.

The results obtained for the three scores are summarized in Table 1 in the line *Fourier* while Fig. 9 provides details on the distribution of the counting accuracies (CAs) (violet boxes indicate the results of the Fourier analysis).

It is apparent that the Fourier analysis alone tends to underestimate the number of plants on dataset (S), (the well separated plants) (12% on average) while over estimating this number on datasets (B) (between separated and overlapping) (by 3%) and (O) (overlapping plants) (by 7% on average). Why is it so?

For dataset (S), the plants are well separated, but this also entails that the peaks of the histogram used by the Fourier analysis are rather narrow, and one consequence is that if a peak is slightly off a predicted position by the analysis, it may be entirely missed by it. This may result in ignoring existing rows or plants within a row.

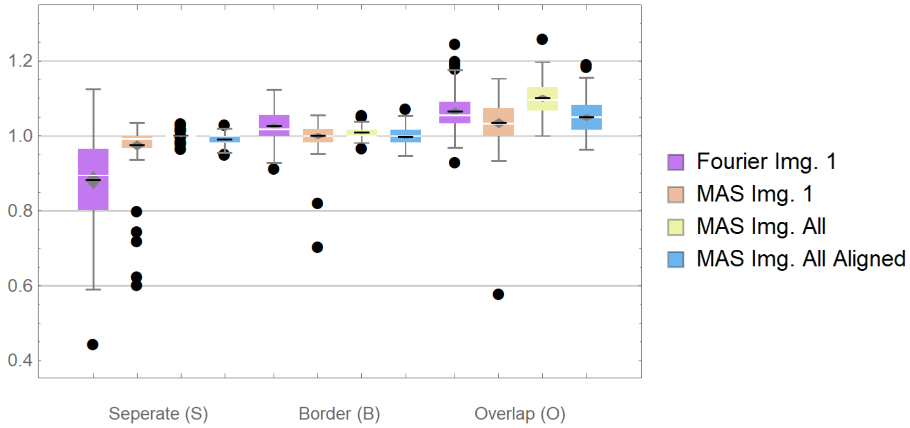


Figure 9: Results on Counting Accuracy (CA). The colors of the whisker boxes indicate the method used to count the number of plants. With *Fourier Img. 1* we counted the plants with the Fourier analysis on one imagefor each of the 100 fields of the dataset. The same images were used with *MAS Img. 1* that counts the plants using the multi-agent system. *MAS Img. All* and *MAS Img. All Aligned* are methods that exploit the redundancy when several images cover the same area in a field. The black dots represent outliers. The boxes' lower and upper limits indicate the 0.25-th and 0.75-th percentile respectively. The median is represented on each box by a white line mark while the mean is represented as a black line mark. The grey diamond represents the interval of confidence. Non-overlapping diamond between pairs of boxes are equivalent to rejecting the null hypothesis of equal means of a two-sample t-Test.

For datasets (B) and (O), the overlapping leaves between plants induces noise that leads the Fourier analysis dedicated to the plants identification to find a slightly higher frequency than the actual target. This results in overestimating the number of plants.

Overall, still, taking into account that the Fourier analysis is in fact used only to estimate the spatial relationships between plants on crop fields, the counting results are surprisingly good.

5.3 Effect of the Multi-Agents Analysis

The multi-agent stage initializes the PAs using the predictions made by the detector of spatial relationships, and then let the PAs evolve and converge towards presumed plants. The question is: how much this can improve the counting performance? In which way can it correct false positives and false negatives?

In our experiments on plant counting, we ran the simulations with the following parameters values: max_nb_steps $= 50$, $\mu = 1.5$, $\nu = 0.5$, $\delta = 0.01$ and $\pi = 0.0001$. *max_nb_steps* has been set as an upper limit of the number of steps of the simulation which has never been reached in our experiments. The values μ and ν were chosen for geometric reasons. ν is the PAs' fusion factor; a value of 0.5 means that two PAs perfectly positioned on consecutive plants will absorb a wrongly positioned PA in-between them which is desirable. μ is the PAs' filling factor; if two PAs are perfectly positioned on plants but another plant has been missed in-between them, then a value of 2 should allow its detection. However a value of 1.5 proved to be better during tests. Lowering the values of δ and π will lead the simulation to overestimate the number of plants while raising them will lead to underestimation. These values were optimized by repeatedly testing the system on training synthetic datasets. The reported results have been obtained on test datasets, different from the training ones.

As can be seen in Fig. 9 and in Table 1, the results show that the multi-agent phase significantly improves the counting performance. For the (S) and (B) datasets, the mean value is closer to the value 1 (approximately 0.98 instead of 0.87 for the Fourrier analysis alone), which means that the estimated number of plants is close to the correct one, and the confidence interval is much narrowed (standard deviation of 0.04 instead of 0.11). The gain is less pronounced on the (O) dataset. Even if the distribution of the results are very similar between the Fourier analysis and the multi-agent one (violet and orange boxes on Fig. 9), the average for the multi-agent analysis is significantly lower than the average of the Fourier analysis as indicated by the fact that the grey diamonds on the boxes do not overlap (non-overlapping diamonds mean that the null hypothesis of equal means can be rejected using a 2-sample t-Test).

It is thus apparent that the proposed two step method: first detecting a structure, then using a multi-agent system to refine the counting, gives very promising results. But, most of the areas of a crop field are covered by several different images from UAVs (up to four times in the example of Figure 6). Is it possible then that even these good results can be improved by resorting to the redundancy thus offered?

5.4 Exploiting Image Overlapping

A common practice when acquiring images of crop fields is to let consecutive images overlap each other. One of the main motivation for this is to avoid that plants located at the edges of an image are only partially visible, and thus ignored. Another motivation is the hope that the mistakes made on an image can be compensated on another image that partially covers the same area. In our case, the synthetic datasets were built with 50% overlap on the height and width of the images. As an illustration, in our example, it exists an area (e.g. *Z4*) that is covered by all four images. In order to combine the information coming from N (e.g. $N = 4$) analyses for a given area, we used an iterative method to form clusters of PAs that can be of size $n \in [1, N]$. These clusters can contain at most one PA from each analyses.

The results are presented under the name *MAS Img. All* in Table 1 and Fig. 9.

Another variant of this algorithm (called *MAS Img. All Aligned*) was introduced with the motivation that aligning the N images covering a given area could help the clustering procedure to gather relevant PAs. One analysis from the multi-agent approach is chosen as a reference and then the algorithm attempts to align the PAs of the $N-1$ remaining analyses to that reference.

The results reported in Table 1 and in Figure 9 show that combining information from the analysis of several images brings improvement in the counting accuracy for the (S) and (B) datasets. For the (O) dataset, the variant *MAS Img. All Aligned* is to be preferred to the *MAS Img. All* method, while *MAS Img. All* is better than *MAS Img. All Aligned* on the (S) and (B) datasets. If the counting accuracy of the combined method is slightly lower than for the method analyzing only one image for the (O) datasets (1.05 instead of 1.03), on the other hand the detection accuracy (DA) is significantly improved from 0.83 to 0.90 which means that the plants are better recognized.

Overall then, combining information from several images seems to be a good strategy.

5.5 Application to Real Images

We also applied the method to a subset of the dataset of real crop fields provided by *Terres Inovia*. The images were taken in two regions in France, one near Toulouse in the south-west, and one near Niort, a city in the west of France. The dataset used in our experiments contains a mosaic of images of a field near Toulouse which all together make one composite image of an entire sunflower field. These images are therefore non-overlapping.

In total, the dataset contains 2111 non-labelled images from which we randomly extracted 50 that were manually labeled and used to test our method. Not all the sunflowers in an image have grown at the same rate, so the images mix areas where the plants are well separated and areas where the leaves of one plant overlap with those of its neighbors in the same row. In addition, the drone captured the original images at an altitude of $30m$ (compared to $10m$ for the synthetic data) and the sunflowers overlap with many weeds in some images, making it sometimes difficult, even for a human, to visually identify the sunflowers. It is thus fair to say that the chosen subset of data contains images comparable to the ones of the (S), (B) and (O) synthetic datasets.

Our counting method yielded an average counting accuracy of 1.03 for a standard deviation of 0.12 on the 50 images subset. The detection accuracy and detection Recall fared at 0.87 and 0.90 respectively for a standard deviation of 0.14 for both. These scores are at least as good as the ones reported in the state of the art (see Section 5.6). Furthermore, they are quite close to the results obtained on the synthetic dataset even if the standard deviation is larger.

This confirms that using synthetic datasets for tuning the method we propose is a promising procedure, effectively leading to good results on real data.

5.6 Comparison with the State of the Art

There are strong economic incentives for being able to automatically count plants in fields at various stages of growth. Aerial images from UAVs offer new possibilities to do so and the scientific literature reflects this interest with a increasing number of publications in recent years. This is in line with similar concerns for automatic object counting in other contexts, in particular counting people or vehicles.

Counting objects can be done through the detection of the objects, or it can be done from a density estimate, usually directly from an analysis at the pixel level of the image.

In the first case, object detection relies either on some prior knowledge of the shape of the objects to be counted or on machine learning to recognize objects. Deciding which templates are useful is generally difficult, while using supervised learning requires (very) many labeled images and large computing resources, for example using deep neural networks.

Datasets	Separate (S)			Border (B)			Overlap (O)		
Scores	DAc	DR	CA	DAc	DR	CA	DAc	DR	CA
Fourier Img. 1	0.93 (0.04)	0.82 (0.11)	0.88 (0.12)	0.87 (0.06)	0.89 (0.05)	1.03 (0.04)	0.81 (0.05)	0.86 (0.05)	1.07 (0.05)
MAS Img. 1	0.99 (0.01)	0.97 (0.07)	0.97 (0.07)	0.98 (0.02)	0.98 (0.04)	**1.00** (0.04)	0.83 (0.05)	0.86 (0.06)	**1.03** (0.07)
MAS Img. All	0.99 (0.01)	0.99 (0.01)	**1.00** (0.01)	0.99 (0.02)	1.00 (0.01)	1.01 (0.02)	0.88 (0.04)	0.96 (0.02)	1.10 (0.05)
MAS Img. All Aligned	0.99 (0.01)	0.98 (0.01)	0.99 (0.02)	0.99 (0.02)	0.98 (0.02)	**1.00** (0.02)	0.90 (0.04)	0.94 (0.03)	1.05 (0.05)

Table 1: Average scores results on the three datasets. Standard deviation is in parenthesis. Values were rounded to the second digit.

On the other hand, density estimation seems simpler but it still requires large training sets and yields coarser estimates of the number of objects in an image. Both approaches, object-based and density-based, are subject to large errors when objects are occluded or overlapping.

For plant counting, (García-Martínez et al., 2020) is an example of the template approach. In their maize plant counting experiments, they selected 4 to 12 templates and used a Normalized Cross-Correlation technique to estimate the number of plants. The method requires that representative plants in the images be chosen, and no recipe is given for this. They obtain a percentage or error of 2.2% when using 12 templates, but acknowledge that the performance drops to 25.7% when the plants overlap.

In their paper, (Ribera et al., 2017) use deep neural networks to learn how to recognize sorghum plants. They describe the rather involved preprocessing and formatting steps that are necessary before learning can take place. They also had to develop a technique to increase the number of labelled training images. Learning itself took between 50,000 and 500,000 iterations which entails a very heavy computing load. They obtained a Mean Absolute Percentage Error of 6,7%. It is not possible to know if the data sets used included overlapping plants or not.

The density-based approach is illustrated in (Gnädinger and Schmidhalter, 2017). They first eliminate what can be presumed to be weeds and parasitic signals using a clustering method. Then they set thresholds on different wavelengths in order to classify pixels as belonging to plants or not. This requires some fine tuning. They obtain error rates around 5% with fairly large standard deviations. Here too, plant overlapping lead to a deterioration in performance.

It must be noticed that it is difficult to make fair comparisons between the approaches proposed by various authors due to the lack of publicly available data sets and thus of a common basis for performance evaluation.

6 CONCLUSIONS

With the arrival of new devices for taking pictures, it is increasingly important to be able to automatically count objects of interest in these images. This paper has introduced a new method to achieve this. It is applicable when objects are spatially organized according to a regular pattern. The method first detects the pattern and then uses it to seed agents in a multi-agent system. The method is simple, requiring no complex fine tuning of parameters, the tricky definition of templates or costly learning. In fact, it requires very modest computing resources.

In a series of extensive experiments on controlled data sets and real aerial images of crop fields, the method yielded state of the art or better performance when the objects are well-separated while exceeding the best known performances when the objects overlap.

Future work will include the incorporation of plant growth models into the synthetic data generator so that more in-depth experiments can be conducted on counting plants from sowing to harvest. On the one hand, we expect that the method presented in this paper will naturally lend itself to successive plant growth counts in the same field. Indeed, the agents existing at the end of a counting process at a given time can be used as seeds for the counting process at the next stage (for example in images of the same field taken one month later). On the other hand, this would contribute to set up a repository of realistic images of crop fields thus allowing systematic comparisons to be made between different plant counting methods. In addition, we plan to test the method on other other object counting problems such as counting people in stadiums or performance halls or vehicles in parking lots.

REFERENCES

Christiansen, M. P., Laursen, M. S., Jørgensen, R. N., Skovsen, S., and Gislum, R. (2017). Designing and Testing a UAV Mapping System for Agricultural Field Surveying. Sensors, 17(12). Number: 12 keywords = point cloud, aerial robotics, canopy estimation, crop monitoring, winter wheat mapping, pages = 2703,.

Dvořák, P., Müllerová, J., Bartaloš, T., and Brůna, J. (2015). Unmanned aerial vehicles for alien plant species detection and monitoring. The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, XL-1/W4:83–90.

García-Martínez, H., Flores-Magdaleno, H., Khalil-Gardezi, A., Ascencio-Hernández, R., Tijerina-Chávez, L., Vázquez-Peña, M. A., and Mancilla-Villa, O. R. (2020). Digital count of corn plants using images taken by unmanned aerial vehicles and cross correlation of templates. Agronomy, 10(4):469.

Gnädinger, F. and Schmidhalter, U. (2017). Digital counts of maize plants by unmanned aerial vehicles (uavs). Remote sensing, 9(6):544.

Guerrero, J. M., Pajares, G., Montalvo, M., Romeo, J., and Guijarro, M. (2012). Support vector machines for crop/weeds identification in maize fields. Expert Systems with Applications, 39(12):11149–11155.

Guijarro, M., Pajares, G., Riomoros, I., Herrera, P., Burgos-Artizzu, X., and Ribeiro, A. (2011). Automatic segmentation of relevant textures in agricultural images. Computers and Electronics in Agriculture, 75(1):75–83.

Han, S., Zhang, Q., Ni, B., and Reid, J. (2004). A guidance directrix approach to vision-based vehicle guidance systems. Computers and Electronics in Agriculture, 43(3):179–195.

Otsu, N. (1979). A Threshold Selection Method from Gray-Level Histograms. IEEE Transactions on Systems, Man, and Cybernetics, 9(1):62–66. Conference Name: IEEE Transactions on Systems, Man, and Cybernetics.

Pérez-Ortiz, M., Peña, J. M., Gutiérrez, P. A., Torres-Sánchez, J., Hervás-Martínez, C., and López-Granados, F. (2016). Selecting patterns and features for between-and within-crop-row weed mapping using uav-imagery. Expert Systems with Applications, 47:85–94.

Perlin, K. (1985). An image synthesizer. ACM SIGGRAPH Computer Graphics, 19(3):287–296.

Ribera, J., Chen, Y., Boomsma, C., and Delp, E. J. (2017). Counting plants using deep learning. In 2017 IEEE global conference on signal and information processing (GlobalSIP), pages 1344–1348. IEEE.

Sankaran, S., Khot, L. R., Espinoza, C. Z., Jarolmasjed, S., Sathuvalli, V. R., Vandemark, G. J., Miklas, P. N., Carter, A. H., Pumphrey, M. O., Knowles, N. R., et al. (2015). Low-altitude, high-resolution aerial imaging systems for row and field crop phenotyping: A review. European Journal of Agronomy, 70:112–123.

Technologies, U. (2020). Unity 2019.4.1.