Article

Assessment of tree detection methods in multispectral images

Dagoberto Pulido [1], Joaquín Salas[1,*], Matthias Rös [1], Klaus Puettmann[2] and Sertac Karaman[3]

[1] Instituto Politécnico Nacional
[2] Oregon State University
[2] Massachusetts Institute of Technology
* Correspondence: Joaquín Salas. jsalasr@ipn.mx. CICATA Querétaro, Instituto Politécnico Nacional. Cerro Blanco 141, Colinas del Cimatario, Querétaro, CP 76090, México.

Version July 9, 2020 submitted to Remote Sens.

Abstract: Detecting individual trees and quantifying their biomass is crucial for carbon accounting procedures at the stand, landscape, and national levels. A significant challenge for many organizations is the amount of effort necessary to document carbon storage levels, especially in terms of human labor. To advance towards the goal of efficiently assessing the carbon content of forest, we evaluate methods to detect trees from high-resolution images taken from unoccupied aerial systems (UAS). In the process, we introduce the Digital Elevated Vegetation Model (DEVM), a representation that combines multispectral images, digital surface models, and digital terrain models. We show that the DEVM facilitates the development of refined synthetic data to detect individual trees using deep learning-based approaches. As field validation, we carried out experiments in two tree fields located in different countries that demonstrate our approach's efficiency. Simultaneously~~At the same time~~, we perform comparisons among an array of classical and deep learning-based methods highlighting the precision and reliability of the DEVM.

Keywords: Tree Detection; Convolutional Neural Networks; Unocuppied Aerial Systems; Digital Elevated Vegetation Model; Synthetic Data Set

---

1. Introduction

Programs to reduce emissions from deforestation and forest degradation (*e.g.* , REDD+ [1]) intend to mitigate the effects of climate change by providing forest landowners with economic incentives reflecting the value of the carbon stored within the trees. However, despite advancements in remote sensing technology, ~~many measurements~~ manual labor still ~~need~~needs to ~~be accomplished by manual labor~~accomplish many measurements, such as estimating the overall vegetation biomass and the carbon stored in individual trees and forests. For example, it is common for field crews to travel to inventory plots and perform tasks such as counting and measuring tree sizes using visual estimations and manual measurements. This approach requires a considerable amount of time and resources, *e.g.*, the USDA Forest Service spends more than 75% of the inventory costs on data collection [2].

~~In this~~This describes ~~study, we describe~~ methodologies that efficiently detect trees automatically using remote sensing technology (see Figure 1). In our approach, we collected data using Unoccupied Aerial Systems (UAS) equipped with multispectral cameras sensitive to the green, red, red edge, and near-infrared wavelengths. Using structure from motion techniques (SfM) [3], we obtained 4-band orthophotos, digital surface models (DSM), and digital terrain models (DTM) [4] in the form of orthomosaics. Then, we calculated the Normalized Difference Vegetation Index (NDVI) [5] from the multispectral orthophotos. After the orthophotos registration~~ of the orthophotos~~, we utilized the DSM, DTM, and NDVI to obtain a Digital Elevated Vegetation Model (DEVM). We then generated a synthetic

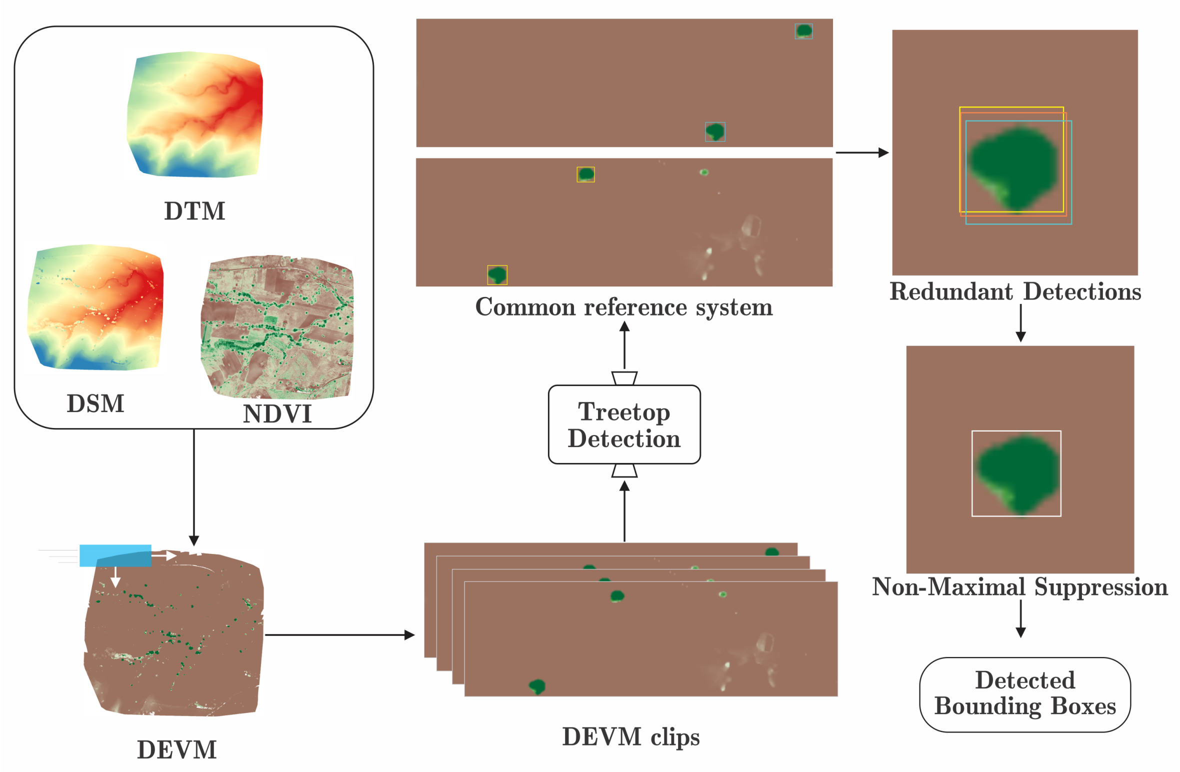


Figure 1. A Pipeline to Automatically Detect Tree Tops. Aerial photogrammetry, obtained from UAS using multispectral cameras, allows us to construct orthomosaics representing the DSM, DTM, and NDVI, from which we eventually build the DEVM. We split the DEVM into sub-images and evaluate them with an object detector, which predicts the bounding boxes for each sub-image. Then, we express the results in a common reference system. Once we obtain a prediction for the whole image, we apply non-maximal suppression to eliminate redundant detections.

dataset of DEVM images that we used to train classic and modern machine learning algorithms to detect trees. Finally, Performance tests using two tree plots in different countries indicated the precision of the new method. Our results show that CNN based methods have become the leading performer. Nonetheless, classic approaches remain competitive and may offer advantages in settings where data collection and available computing resources for training are an issue.

Our main contributions include:

- The introduction of the DEVM, an image representation that blends aboveground structural information and ~~the~~ quantification of vegetation suitable for the detection of trees;
- the development of a scheme to generate synthetic data sets of trees in DEVM space for training classical and modern tree detection methods;
- the assessment of classical and modern techniques, trained with synthetic images, to detect treetops.

We structure the rest of the document as follows. In the next section, we describe the current state-of-the-art and practice regarding tree detection. Then, in section 3, we formulate the foundation of the newly developed DEVM, provide further detail about the method to generate synthetic tree data sets, and detail the classical and modern techniques we benchmark in the paper. In section 4, we describe the ~~results~~effects of implementing the methods to detect treetops in two tree plots and compare their performance. We continue the paper discussing our results in section 5 and finally conclude summarizing our findings and delineating possible directions for future research.

2. Related Literature

3.1.1

~~The most significant divide among tree detection methods is whether they use classical approaches, where one manually engineers the design of features, or deep learning strategies, which compute features automatically.~~

This section reviews~~In this section, we review~~ the scientific literature describing classical and modern~~both of these~~ approaches for tree detection with particular emphasis on aerial images. Also, related to our work, we discuss the research aiming to generate synthetic images for training deep learning methods and describe the image sources for automatic tree detection.

2.1. Classical Tree Detection

Classical methods for detecting trees rely mostly on ~~the use of~~ crafted features (see [6–8] for reviews of approaches to detect individual tree crowns). The proposed methods include local maxima filtering, template matching, valley following, watershed region growing, circular structures fitting, and Support Vector Machines with Histograms of Oriented Gradients (HOG) features. Some recent research follows this trend. For instance, basing trees detection on edge votes required applying tree crown delineation for the candidates using watershed segmentation (Özcan *et al.* [9,10]). The eccentricity of the ellipses to fit these segments was used to discard human-made objects. Another method filtered out non-vegetation utilizing ~~the~~ NDVI (Ozdarici-Ok [6]). The selection of tree crows employed the gradient to detect high radial symmetry and increased diameter thresholds. A third method was based on the binarization of RGB images (Reza *et al.* [11]). An adaptive median filter removed noise and distortion before a morphological operation outlined the boundaries between the plants.

Alternatively, Maillard and Gomes [12], and Bao *et al.* [13] used template matching to detect trees. The former detects deciduous trees using a geometrical-optical model as a template, which includes parameters such as illumination angles, maximum and ambient radiance, and tree size specifications. The latter method selects several templates from the original image and computes mutual information for matching. When local maxima and watershed models were evaluated for the individual detection of trees, both approaches performed well for dominant and co-dominant trees but underperformed for small trees (Goldbergs *et al.* [14]). In contrast, Random Forest regression can estimate the number of trees using the local maxima and the result of a classification process which can distinguish between trees, soil, and shadows (Fassnacht *et al.* [15]). These features are fed to a Support Vector Machine (SVM) with a Radial Basis Function (RBF) to classify the tree species. Similarly, Wang *et al.* [16] first separated images between vegetation and non-vegetation with an SVM. After the extraction of HOG, these features were used to train an SVM to detect palms. This method appears limited to identifying palms, and it showed more reduced performance when the palms are intermingled with trees.

Recently, several approaches for tree detection used the local maximum filtering algorithm. For instance, Li *et al.* [17] implemented a Field-Programmable Gate Array (FPGA) for the detection of tree crowns, speeding up the computations considerably without loss of performance.~~On 12,188 × 12,576 pixel satellite images, the task was accomplished 18.75 times faster than the original algorithm without loss of performance. Marinelli *et al.* [18] proposed a Bayesian formulation to improve the detection of treetops in LiDAR data. This approach fuses bitemporal airborne acquired data improving the overall accuracy up to 8.6% with respect to single date detections.~~ Xiao *et al.* [19] used the DSM obtained from the 3D information provided by multiview satellite images to detect individual trees and delineate their crowns. Treetops are recognized from the local maxima, and outliers are eliminated with allometric equations.~~An alternative approach investigated optimal parameters to detect treetops from airborne LiDAR sensing. These parameters include the distance to the ground, the smoothing of the digital surface model, and the filtering of the output point cloud (Koledo and Ksepko [20]).~~ Finally, Garcia *et al.*[21] presented a framework for individual citrus tree detection based on Digital Surface Models that included a segmentation method based on Extended Maxima Transforms followed by a controlled-marker watershed for single tree segmentation. Other tree detection approaches include shallow neural networks. For instance, a two-stage method trained a backpropagation (BP) neural network to detect trees from color images in stage one. In the second stage, properties, such as energy, entropy, mean, skewness, and kurtosis, are used to correct the BP neural network and build a cascade neural network classifier (Tianyang *et al.* [22]).

2.2. Deep Learning for Plant Detection

Lately, there has been a surge in methods to detect and count plants using deep learning. Researchers have already~~So far, researchers have~~ employed ~~already~~ tested architectures [23], such as LeNet, VGG, AlexNet, or GoogLeNet for classification or regression. ~~For instance, Weinstein *et al.* [24] made combined use of LiDAR and RGB sensing information in a self-supervised approach, which employed an unsupervised delineation method to train a crown detection model. A RetinaNet convolutional neural network, basically a one-stage detector, is then refined using annotated RGB images.~~ Freudenberg *et al.* [25] developed a palm detection method using satellite images with 40 cm/pixel resolution. They employed a U-Net Convolutional Neural Network (CNN), which performs semantic segmentation between palms and background. This method performs particularly fast, especially when compared with traditional CNNs such as AlexNet. Similarly, Li *et al.* [26] detected oil palms in satellite images using a two-stage CNN. In the first stage, they classified land cover, and in the second, they detected the palms. For training, they employ 20,000 samples, and during operation, they apply a multiscale sliding window.

There are ample examples of the use of CNN to detect orchard trees such as citrus [27,28], coconut, oil palm [27,29–31], palm [32], and tobacco [33]. Also, species found typically in forests have been the subject of researchers' interest, such as spruce, birch, and pine [34,35]; while Pribe *et al.* [36] have studied the detection of urban trees. CNN architectures have received a lot of attention including LeNet [29,31,32], SqueezeNet [37], AlexNet [30,32,36], GoogLeNet [38] and DarkNet [39]. Windrim and Bryson [35] explored the combined use of candidates generation, with Faster R-CNN, and 3D detectors with VoxNet. Still, Zorte *et al.* [27,40], Fan *et al.* [33], Csilik *et al.* [28], and Trier *et al.* [34] studied the use of simple custom-made CNN architectures with two or three convolutional layers followed by two or three fully connected layers. Zortea *et al.* [27] first applied a CNN to detect tree rows, ~~then~~ located center lines, and finally used another CNN to detect trees. Puttemans *et al.* [39] showed that CNNs are a feasible alternative to boosted cascade [41] and aggregated channels [42]. Csillik *et al.* [28] utilized CNN on NDVI images to distinguish between trees, bare soil, and weeds; while Mubin *et al.* [31] detected and distinguished between mature and young trees. Zortea *et al.* [40] were the first to apply an ensemble of CNN-based classifiers. Windrim *et al.* [35] further separated the background class into shrubs, partial trees, and the tree class as foliage, lower stem, upper stem, and clutter components. Trier *et al.* [34] used the green-blue ratio to remove shadows and the NDVI image to remove dead vegetation and non-vegetation, similarly to Pribe *et al.* [36]. Finally, Fan *et al.* [33] selected their tree candidates using morphological operations. In contrast, Chen *et al.* [37] presented a pipeline for fruit counting, where they used a custom crowd-sourcing platform to label large data sets. After using a CNN to extract candidate blobs, they employed a secondary convolutional network to count. Finally, Ribera *et al.* [43], experimenting with AlexNet, Inception v2, Inception v3, and Inception v4, proposed a linear regressor to estimate the final fruit count.

In research similar to ours, Xiao *et al.* [44] used a Fully Convolutional Network (FCN) [45] to detect treetops in satellite imagery. They fused the NDVI values, the DSM, and the red band into a 3-channel input. To train the FCN, they obtained samples using the top-hat morphological operation on the DSM to detect the local maximum as treetops. In contrast, we used synthetic DEVM images to train the CNN. Regarding RGB images, Santos *et al.* [46] used a Deep Learning-based approach to detect and classify trees in aerial images. They captured and manually annotated a set of 392 images. They then trained and compared Faster-RCNN, YOLOv3, and RetinaNet, three different ~~models for~~ detection and classification models. Similarly, Fromm *et al.* [47] trained Faster R-CNN, SSD, and R-FCN CNN architectures to detect seedlings using images taken from UAVs along seismic lines. This brief overview of the different methods suggests an increasingly predominant role of CNN-based techniques to tackle ~~the problem of~~ tree detection.

2.3. Synthetic Dataset Generation

Deep learning commonly requires vast amounts of labeled data to train a CNN. As the manual labeling of images is very demanding, synthetic datasets are attractive for researchers working in machine learning. ~~Previous research efforts on this topic have resulted in datasets for detection of humans (particularly through the recognition of faces [48] and human bodies from unrestricted [49] and frontal view poses [50]), medical applications (particularly to generate Positron Emission Tomographies [51], synthetic ultrasound images for intravascular ultrasound simulation [52] and images of retinal vessel networks [53]), manufacturing (particularly to render scenarios to generate synthetic data for automotive applications [54] and synthetic depth images from CAD models [55]) and synthetic aperture radar data [56].~~ Using an approach similar to our work, Ubbens *et al.* [57] count leaves of *Arabidopsis thaliana* rosettes. They render 3D models of plants and use these to create data sets for training. Han and Kerekes [58] reviewed simulation methods for multispectral images, such as the ones used in our approach. They concluded that technological trends, including emerging computing power, powerful graphics processing units, and deep learning techniques, will continue to push for ~~the use of~~ more realistic images. Recently, Fassnacht *et al.* [59] introduced a method to simulate realistic tree canopy by combining the SILVA individual-tree forest simulator [60] with real LiDAR point clouds of individual trees. They employed their system to assess remote-sensing models for biomass estimation.

2.4. Image Sources

To obtain the data for the detection of trees, researchers have employed satellites [29–32] and airborne sources, such as UAS [27,28,33,40], helicopters [35] and piloted airplanes [34,36,38,39]. For instance, Millard and Gomes [12], Özcan *et al.* [9] and Fassnacht *et al.* [15] ~~have employed high-resolution satellite images taken from GoogleEarth and WorldView-2~~ have employed high-resolution images taken from the USGS's Landsat 8 satellite using the GoogleEarth platform and 3.2 WorldView-2 respectively to detect mango, orange and apple trees, as well as estimating stand density above-ground biomass. Ozdarici-Ok [6] detected and delimited citrus trees using the images obtained from the GeoEye-1 satellite, which image resolution is 50 cm/pixel. Bao *et al.* [13] also used GeoEye-1 to extract individual tree crowns from panchromatic satellite images, covering areas of 5 and 25 km^2. Using CNN, Li *et al.* [29] detected oil palm plants from satellite images in Malaysia.

Nowadays, UAS are becoming a popular tool for high-resolution, timely, and low-cost image capturing. For instance, Ribera *et al.* [43] counted plants using a regression CNN from images taken from a UAS flying over a sorghum field. Chen *et al.* [37] presented a pipeline for fruit counting in a supervised deep learning framework where they use a custom crowd-sourcing platform to label large data sets. They took their images from a multi-rotor UAS, and they evaluated their method's performance using ground truth produced by humans. ~~Wu *et al.* [61] assessed watershed, polynomial fitting, crown segmentation, and point cloud segmentation algorithms to estimate the canopy cover of individual trees in a planted forest. They obtained the data from a LiDAR mounted on a UAS and compared the results of the algorithms with field data.~~ Selim *et al.* [62] used an object-based method to detect trees from images obtained from UAS. Their approach got 1 (one) cm resolution scene reconstructions using SfM. They implemented a set of rules to identify trees based on their height, scale, shape, and integrity. Finally, Reza *et al.* [11] proposed a method to recognize and count rice plants using low altitude flying UAS.

Sensors employed to obtain information to detect trees from airborne platforms include RGB cameras [27,31–33,38–40], multispectral cameras [28–30,34], and LiDAR [35]. Using RGB images, Krisanski *et al.* [63] proposed a novel method to measure trees' diameter. They flew a UAS manually under the trees canopy while taking photos. Offline, they obtained a 3D representation from which they automatically measured the trees' diameter within a plot. Their results are promising and will undoubtedly boost the exploration of fully automatic approaches. Employing multispectral imagery, Qiu *et al.* [64] introduced an individual tree delineation map on multispectral images from cameras ~~mounted~~ on a UAV overflying a forest stand. Using the gradient map, they extracted treetops and refined the delineation employing spectral differences. They segmented the gradient map using watershed with the treetops as markers and improved the segmentation to yield the crown map. ~~Utilizing LiDAR sensors on UAS, Kuvzelka *et al.* [65] detected the individual tree stems and measured the stem diameters. They applied segmentation methods based on Hough transform, RANSAC, and robust least trimmed squares to Norway spruce and Scot pine with encouraging results.~~ Also, Picos *et al.* [66] detected and measured the height of Eucalyptus trees in a plantation. ~~For detection, they~~They investigated two methods for detection: One based on constructing overlapping polygons around each point in the stem cloud, and another employing density estimation with an axis-aligned bivariate quartic kernel. Finally, Yan *et al.* [67] observed that the fixed-bandwidth mean-shift based methods work well to extract the same size of individual trees. Thus, they introduced a self-adaptive bandwidth estimation method. Starting from the global maximum point, they divided the 3D space into angular sectors simulating the canopy surface. They employed the potential crown boundaries to estimate the crown width and from it, the kernel bandwidth. 3.1.4

Our literature review highlights the identification of trees from aerial images using either classic and deep learning-based methods remains an active area of research, with recent approaches competing in aspects such as detecting rate, computing time, and hardware requirements. Therefore, to assess their potential and limitations, ~~Overall,~~ there seems to be a need for comparisons~~between modern approaches based on deep learning and classical approaches~~. However, as~~As~~ deep learning methods deliver promising results, there is a requirement to develop databases for evaluation and improvement. As some recent work as precluded [24], shortly, there will be significant and rich-enough datasets of 3.1.3 trees taken from aerial images to cover data-hunger approaches. In the meantime, it is of immediate interest to generate synthetic ~~Of particular interest in this context is the synthetic generation of~~ data that fuses structural and multispectral information sources. Such novel and efficient representations allow testing different image capturing platforms, particularly those based on UAS. It is within these opportunities that we develop our approach.

3. Materials and Methods

In this section, we introduce DEVM, an image representation suitable for tree detection. Also, we present a model for the generation of synthetic images. Also, we describe the classical and modern methods we will use in our assessment.

3.1. DEVM: A Blended Representation of Structure and Multispectral Information

The database for this study is a set of multispectral images captured from UAS. We describe how these images were processed to generate the input for the convolutional neural networks we utilize in this study.

3.1.1. Characterizing Vegetation

Researchers have proposed several indices to measure the extent of vegetation in images [68]. ~~These indices include the perpendicular vegetation index (PVI), the soil adjusted vegetation index~~ 4.1 ~~(SAVI), the atmospheric resistant index (ARVI), the global environment monitoring index (GEMI), and their many variations [68]~~. It is still an open question under what conditions, which of these indices works best. It appears that their performance depends on several factors, including the atmospheric conditions, the presence of clouds, the plants' water content, the particular imaging viewpoint, and the sensitivity of the specific instrument used [69]. We employed the NDVI [70], a classical index that practitioners have used extensively because of the ample availability of sensors from which it can be extracted. For a pixel at position $\mathbf{x} \in \mathbb{R}^2$, Weier and Herring [71]) calculated the NDVI from the visible red (R, 640-680 nm) and near-infrared (NI, 770-810 nm) radiation as

$$\mathrm{NDVI}(\mathbf{x}) = \frac{\mathrm{NI}(\mathbf{x}) - \mathrm{R}(\mathbf{x})}{\mathrm{NI}(\mathbf{x}) + \mathrm{R}(\mathbf{x})}. \tag{1}$$

When using NDVI, we assume that healthy vegetation absorbs most of the radiation and, simultaneously, reflects a large portion of the near-infrared radiation [71]. Researchers have observed that the NDVI saturates rapidly in dense vegetation canopies. In these cases, one may employ a saturation adjusted NDVI, such as the ones proposed by Gu *et al.* [72] or Fang *et al.* [73]. 3.7

3.1.2. The Digital Elevated Vegetation Model

DSM are 2.5D pictures (2D images that facilitate the visual perception of depth) that represent the elevation over the terrain, *i.e.*, the land surface, vegetation, or human-made structures that one could obtain the DSM from images processed with SfM reconstruction techniques [3]. ~~Usually, it is calculated with reconstruction methods using Light Detection and Ranging (LiDAR), Interferometer Synthetic Aperture Radar (IFSAR), or photogrammetry [74]. Alternatively, one could obtain the DSM from images processed with SfM reconstruction techniques [3]~~. In contrast, DTM are 2.5D pictures that show the bare surface of the soil, ignoring any vegetation or human-made objects, leading to the challenges of computing the DTM from the DSM. ~~For instance, Unger *et al.* [75] used variational energy minimization to solve the problem. They employed the Huber norm for regularization to smooth surfaces and an L_1 norm for the data fidelity term.~~ In our approach, we used the orthomosaics (for the DSM and DTM models, and the NI and RE spectral bands) produced by Pix4D, a photogrammetry program for 3D reconstruction from a series of images. Using the NDVI, DTM and DSM orthomosaics, we expressed the concept of DEVM as

$$\mathrm{DEVM}(\mathbf{x}) = (\mathrm{DSM}(\mathbf{x}) - \mathrm{DTM}(\mathbf{x}))\mathrm{NDVI}(\mathbf{x}), \tag{2}$$

where the subtraction of the DTM from the DSM represents the objects over the terrain. Then, we multiplied the result by the NDVI, aiming to highlight those objects that correspond to vegetation above ground level (see Figure 2). Thus, the DEVM bundles characterizations of vegetation and terrain into a description which facilitates the generation of synthetic images for training. In its present form, the DEVM characterization gives a head start to the detection of trees algorithms. However, it also offers an ambivalence where tall/small trees with low/high NDVI values may be comparable. Adapting to that ambivalence may be a feature of a tree detection algorithm, resulting in a corresponding performance. 3.3 3.7

3.2. Synthetic Dataset Generation

Convolutional Neural Networks (CNNs) have become the dominant approach for object detection in computer vision. However, its application requires massive amounts of labeled images. The work needed to obtain the datasets tends to be costly, challenging, and error-prone. Even though people are using UAS in photogrammetry in recent years and have captured many pictures of terrain, it is still expensive to obtain a human-labeled training dataset of aerial multispectral images. Thus, we generated a batch of simulated and computer-labeled DEVM images designed to look similar to the real ones (see Figure 3).

Inspired by the resulting structure of trees in DEVM space as observed from overhead, we created synthetic images using as a basis the multiple occurrences of a shape with closed-form analytical expression. In our procedure we generated an image $\mathbf{I}(\mathbf{x})$, for $\mathbf{x} = (x, y)$, where $x \in [1, w]$ and $y \in [1, h]$, containing at most n trees, where n is a random variable with probability density function (pdf) given by $n \sim \mathcal{U}(n_{\min}, n_{\max})$. In our case, $\mathcal{U}(u_i, u_f)$ represents a uniform distribution with value $1/(u_f - u_i)$ between the extremes of the interval $[u_i, u_f]$ and zero outside. To ensure that the DEVM representation of each tree is inside the image, we defined each tree center at $(\overline{x}_i, \overline{y}_i)$, where $\overline{x}_i$ and $\overline{y}_i$ are

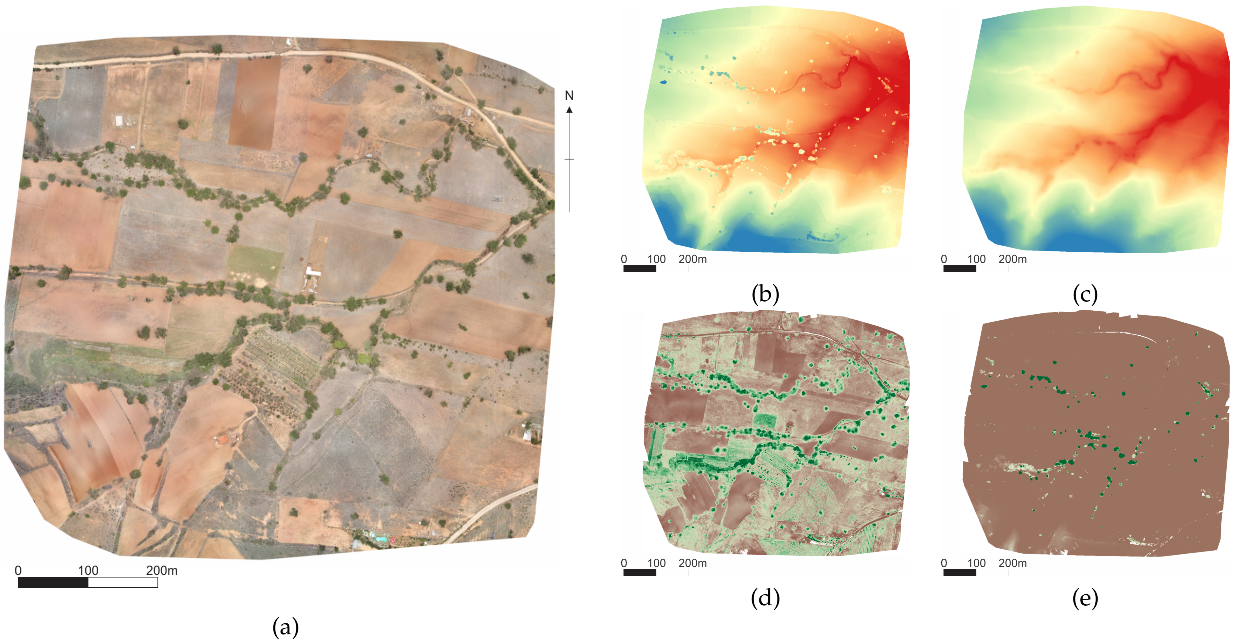


Figure 2. We generated orthomosaic models from multispectral images corresponding to (a) RGB, (b) DSM, (c) DTM, (d) NDVI, and (e) DEVM. The patch of 867m $\times$ 801m terrain corresponded to an agricultural landscape with scattered trees in Zimatlan, Oaxaca, Mexico.

random variables with pdf defined as $\overline{x}_i \sim \mathcal{U}(a_{\max}, w - a_{\max})$ and $\overline{y}_i \sim \mathcal{U}(b_{\max}, h - b_{\max})$, respectively. Here i refers to the i-th tree, and thus $i \in [1, \ldots, n]$. Each tree will have lateral orthogonal widths given by a_i and b_i, where a_i and b_i are random variables with pdf given by $a_i \sim \mathcal{U}(a_{\min}, a_{\max})$ and $b_i \sim \mathcal{U}(b_{\min}, b_{\max})$.

Also, we modeled each tree as a set of at most m_i overlapping domes (see (3)), where we defined m_i as a random variable with pdf given by $m_i \sim (m_{\min}, m_{\max})$. We defined the center of each dome $(\overline{x}_{ij}, \overline{y}_{ij})$ around the tree center as $\overline{x}_{ij} = \overline{x}_i + \Delta_x$ and $\overline{y}_{ij} = \overline{y}_i + \Delta_y$, where Δ_x and Δ_y are random variables defined as $\Delta_x \sim U(-\Delta_{xy}, \Delta_{xy})$ and $\Delta_y \sim U(-\Delta_{xy}, \Delta_{xy})$. Meanwhile, we randomly varied the lateral widths of each dome by a_{ij} and b_{ij} respectively, for $j \in [1, \ldots, m_i]$, where a_{ij} and b_{ij} were random variables with pdf given by $a_{ij} \sim \mathcal{U}(a_i - \Delta_{ab}, a_i)$ and $b_{ij} \sim \mathcal{U}(b_i - \Delta_{ab}, b_i)$, respectively.

For our method, we found it suitable to define the domes using the closed analytical form expressed as

$$\mathbf{D}(\alpha, \beta) = h_{ij} \cdot \cos\left(\frac{\alpha\pi}{2a_{ij}}\right) \cdot \cos\left(\frac{\beta\pi}{2b_{ij}}\right), \tag{3}$$

for given values of a_{ij}, b_{ij}, and h_{ij}, where $\alpha \in [-a_{ij}, a_{ij}]$ and $\beta \in [-b_{ij}, b_{ij}]$, and h_{ij} was a random variable with uniform pdf given by $h_{ij} \sim \mathcal{U}(h_{\min}, h_{\max})$. The dome could be conveniently represented in image space using the linear transformation $\mathbf{x} = \mathbf{K}\boldsymbol{\theta} + \overline{\mathbf{x}}$, where $\mathbf{x} = (x, y)^T$ are the coordinates of a point in the image, $\mathbf{K}_{2\times 2}$ was a matrix which diagonal contains k, a constant k that relates pixels in the image with metric units, $\boldsymbol{\theta} = (\alpha, \beta)^T$ contains the dome parameters, and $\overline{\mathbf{x}} = (x_{ij}, y_{ij})^T$ is the center of the dome. In Algorithm A1, we present a pipeline describing how we created domes. 4.2

Given an image resolution and a set of parameters defining bounds, we created synthetic images by randomly varying the number of trees, their width, their height, and with a random amount of domes with random location and diameter, which themselves depend on the parameters previously computed. Along with the images, we saved the bounding boxes' location, describing each tree's position.

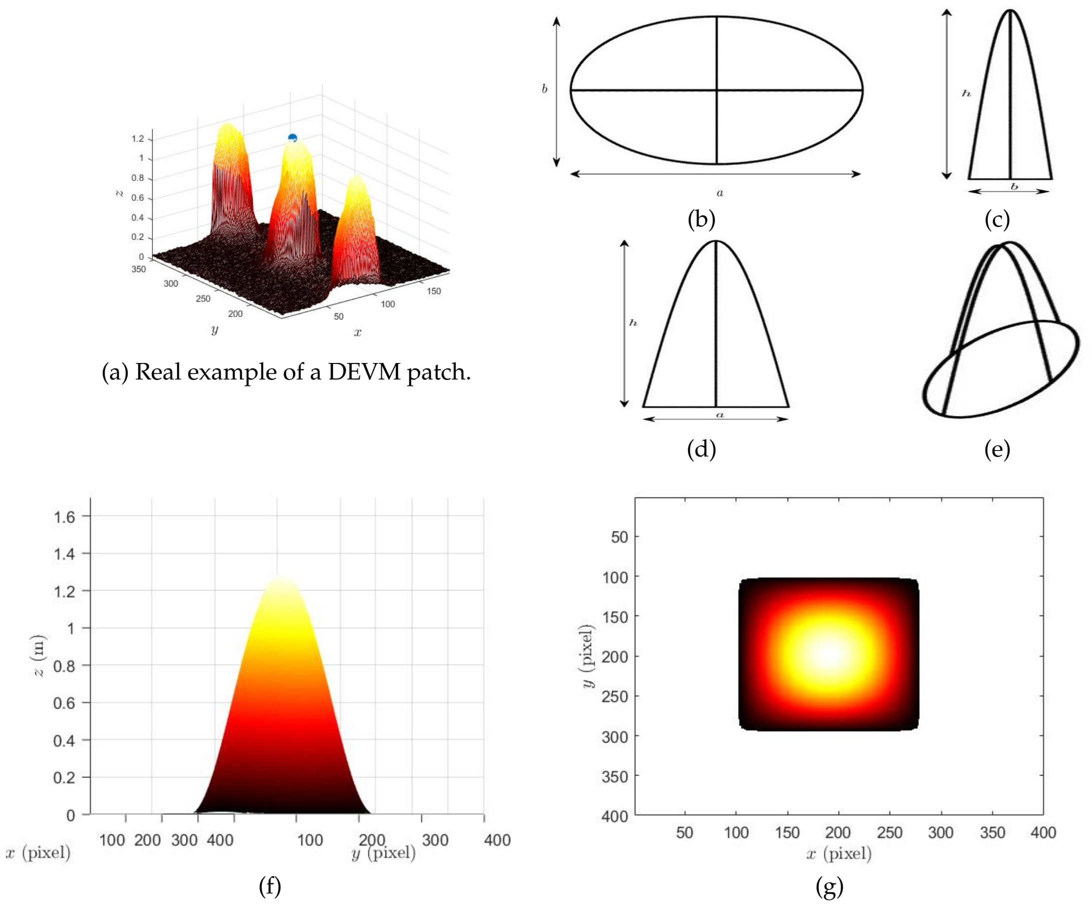


Figure 3. Example of a training dataset using DEVM to highlight trees. We built a synthetic data set of trees in the DEVM representation using random ellipses. In (a), we show an example of a DEVM image. We illustrate the orthogonal (b)-(d) and isometric (e) views of a single dome, which forms the basis for the construction of the synthetic representation of a tree in DEVM space. We show an example of the side (f) and top (g) views of the synthetic description.

3.3. Treetop Detection Methods

We implemented several classical and deep learning-based alternatives for treetop detection. In their comparison, ~~Also,~~ we employed DEVM images as inputs, using synthetic images when the methods required training, to establish a baseline to evaluate their~~the~~ performance. Thus, our results could differ from those reported in the literature because either the input images contain different information or our implementation changes in subtle details from other studies. We developed the approaches using Matlab, Nvidia DIGITS, and Tensorflow with the Google Object Detection API [76] for classical and deep learning-based methods, respectively. In all cases, we compared the inferred bounding boxes against the manually-obtained ground truth data.

3.1.2

Table 1. Constant values used for the generation of synthetic images in Algorithm A1

r	c	n_{min}	n_{max}	m_{min}	m_{max}	h_{min}	h_{max}	a_{min}	a_{max}	b_{min}	b_{max}	Δ_{xy}	Δ_{ab}
1248	384	2	7	5	10	1.4	2.3	65	75	65	75	5	20

3.3.1. Classical Methods

For our comparison, we have included implementations for Local Maxima Filtering, Correlation with a Template, HOG features with an SVM classifier, and a Hough-based circular structures detector.

Local Maxima Filtering (LMF). In this method (inspired by Pouliot *et al.* [77]), we detected trees as peaks in the DVEM image. First, we smoothed the DVEM image with a Gaussian filter, with $\sigma = 2$, and proceeded to find the regional maxima, which we defined as the set of connected pixels with equal value surrounded only by pixels with a lower value. Although rarely necessary, we selected a random pixel when several pixels have the same regional maximum values. We considered a successful tree detection when pixels survived a non-maxima suppression stage, where, starting from the highest valued pixel, we eliminated all those pixels within a neighborhood of radius $\tau \in [1, 500]$ that have a smaller value.

Correlation with a Template (Template). In this method (inspired by Ke and Quackenbush [78]), we compared portions of the DVEM image with a template we extracted from using Pearson's linear correlation coefficient. We selected the local maxima peaks as the centroids of the detected bounding boxes, with the same size as the template. We generated the templates using eCognition, a computer program aimed to determine detections from a set of sub-images extracted by the user from the orthomosaics. In eCognition, the user gives relevant feedback based on the proposed examples~~detections~~ to improve the detector's performance. The program defines the template as the average over the correct predictions.

HOG Features with an SVM Classifier (HOG+SVM). In this method (inspired by Wang *et al.* [16]), we characterized the DVEM image using HOG features [79] and used a SVM classifier to distinguish between the classes *tree* and *no-tree*. Using the DVEM image corresponding to the *Almendras*, we selected 64×64 ground truth bounding boxes corresponding to trees. Afterward, we chose areas randomly without trees to construct a dataset of true negatives. Then, we augmented the dataset with five images corresponding to rotations of 90°, 180° and 270° degrees, vertical and horizontal mirroring, resulting in 27,055 images. Using this dataset, we extracted HOG features for each image, resulting in a feature vector of 1,764 values. Next, we fit an SVM with a linear kernel that ended up with 578 support vectors. Using this classifier, we slid a window over all the test images to obtain the SVM score in each location. To get the position of the trees, we first detected the position of the maxima. Then, for a given SVM score threshold, we applied non-maximal suppression for those detections around it. To assess the performance, we varied this threshold from -11.28 to 10.05 in steps of 0.1.

Circular Structures (Hough). In this method (inspired by Ke and Quackenbush [78]), we detected trees by the similarity of the contours in the DVEM image with circular rings. Firstly, we computed a Canny edge detector. Then, we found the circles between a minimum and maximum radius. We estimated the parameters for the minimum and maximum threshold for the Canny edge detector, and the minimum and maximum radius for the circular rings, using the DVEM image for the *Mancañas* field. To evaluate the performance, we varied the minimum radius from 10 to 65 pixels and tested these parameters on the DVEM image for the *Almendras* field.

3.3.2. Deep Learning-based Methods

We used deep learning to detect the trees because this technique automatically extracts complex features, is well suited detecting~~for the detection of~~ objects~~,~~ and generalizes well in the presence of new data. Our deep learning-based alternative methods include implementations for DetectNET, Faster R-CNN with Inception v2, Faster R-CNN with ResNet-101, Single Shot Multibox with Inception v2, and R-FCN with ResNet-101.

DetectNET. Barker *et al.* [80] derived DetectNet from the classification engine GoogLeNet [81,82]. In turn, GoogLeNet corresponds to the incarnation of Inception v1. It is a 22 layer CNN that receives as input a 224×224 RGB image with mean subtraction. To detect multiple objects during training, DetectNET extracts the bounding boxes of each image from the annotations overlaid on the coverage map. Given the coverage map for object k, $C_k(\mathbf{x})$, for $\mathbf{x} = (x, y)^T$ and $1 \leq x, y \leq S$, we set positions to

1 where objects are present and 0 otherwise [80]. Once DetectNet predicts the coverage map and the bounding boxes, it expressed the result as a three-dimensional label format describing the class of the present object and the pixel coordinates of the ~~corners of the~~ bounding box's corners relative to the center of the grid. ~~Then, a~~A clustering function produces a list of M bounding boxes (see Figure 4). We trained with Nvidia/Caffe, a modified version of Berkeley's Caffe framework for deep learning, and used transfer-learning to establish the initial weights from a model previously trained with the KITTI dataset [83] to achieve faster convergence.

Faster R-CNN with Inception v2 (Faster R-CNN/Inception v2). Faster R-CNN consists of two stages, a Region Proposal Network (RPN) and a detection network [84]. The former simultaneously predicts bounding boxes and objectness scores at each position of its last feature map layer. In the latter, a detector attends the proposals and refines them, *i.e.*, one pools the features from bounding boxes from where one detects a class of objects. In our implementation, we scaled the image to 600 $\times$ 1024 pixels. We initialized the weights with a checkpoint of MSCOCO Dataset from Tensorflow's object detection zoo [76]. ~~Then, we~~ We then refined the model with our DEVM synthetic database for 30,000 steps, using the stochastic gradient descent (SGD) with momentum optimizer with an initial learning rate of 2×10^{-4} that changes to 2×10^{-5} after step 15,000. To evaluate the performance, we divided the validation DEVM maps into smaller overlapping images.

Faster R-CNN with ResNet-101 (Faster R-CNN/ResNet-101). In this case, we initialized the weights with a checkpoint of the KITTI dataset (cars and pedestrian) [83] from the Tensorflow's object detection zoo [76]. We then~~Then, we~~ trained the model with our DEVM synthetic database for 30,000 steps, using the momentum optimizer with an initial learning rate of 10^{-4} that changes to 10^{-5} after step 15,000. To evaluate the performance, we divided the validation DEVM maps into smaller overlapping images.

Single Shot Multibox Detector with Inception V2 (SSD/Inception v2). Similarly to Faster R-CNN, SSD consists of a neural network-based strategy where one extracts feature maps from images and infers bounding boxes and classes using a multi-scale bounding box predictor [85]. We initialized the weights with a checkpoint of MSCOCO Dataset from Tensorflow's object detection zoo [76]. Then, we refined the model with our DEVM synthetic database resized to 300 $\times$ 300 pixels for 30,000 steps, using RMSprop [86] optimizer with an initial learning rate of 4×10^{-3} that changes to 4×10^{-4} after 15,000 steps. To evaluate the performance, we divided the validation DEVM maps into smaller overlapping images.

R-FCN with ResNet-101 (R-FCN/ResNet-101). R-FCN is a method for object detection that uses a region-based, fully convolutional network (R-FNC) that proposes candidate regions of interest that are later voted to decide which one accurately covers the object. We initialized the weights with a ResNet-101 checkpoint of MSCOCO Dataset from Tensorflow's object detection zoo [76]. We refined the model with our DEVM synthetic database resized to 300 $\times$ 300 pixels for 30,000 steps, using a Stochastic Gradient Descent (SGD) with the momentum optimizer with an initial learning rate of 3×10^{-4} that changes to 3×10^{-5} after 15,000 steps. To evaluate the performance, we divided the validation DEVM maps into smaller overlapping images, which in turn, we resized to 300 $\times$ 300 pixels.

3.4. Image Acquisition

We mounted a Parrot Sequoia camera on a UAS and flew over the *Almendras* and *Mancañas*(see 3.4 4.1) field using a self-built multicopter for the former and over the with a 3DR Solo quadcopter for the later. For these settings, we performed NADIR double grid missions with an overlap of 85% at an 3.5 altitude of 25 meters. The Sequoia produced multispectral images with spectral response peaking in wavelengths of 550 nm (Green), 660 nm (Red, R), 735 nm (Red Edge), and 790 nm (Near Infrared, NI). Each of these images has a spatial resolution of 1280 (horizontal) $\times$ 960 (vertical) pixels~~. For~~for all the flight missions.

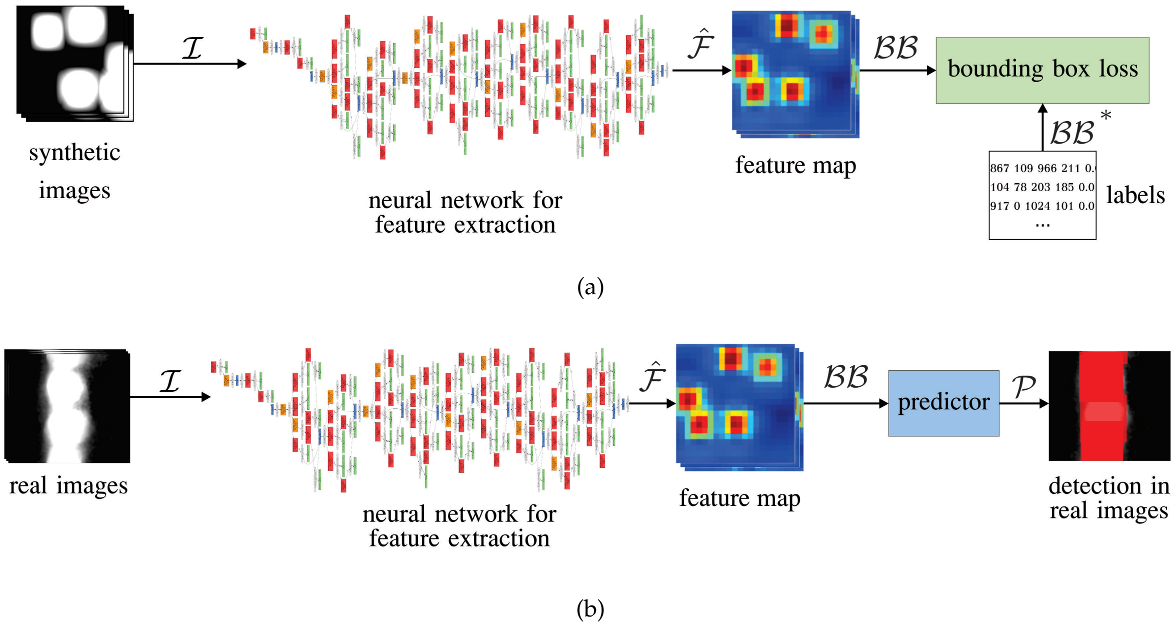


Figure 4. Training and inference of neural network models for treetop detection. We fed CNN with synthetic DEVM images $\mathcal{I}$. During training (a), we utilized synthetic DEVM images $\mathcal{I}$ to fine-tune the pre-trained weights of the fully connected layers. Then we compared the estimated bounding boxes $\mathcal{BB}$ with the respective ground truth $\mathcal{BB}^*$ to compute the loss. During evaluation (b), the CNN infers the bounding boxes for real DEVM images to generate predictions $\mathcal{P}$.

4. Experimental Results

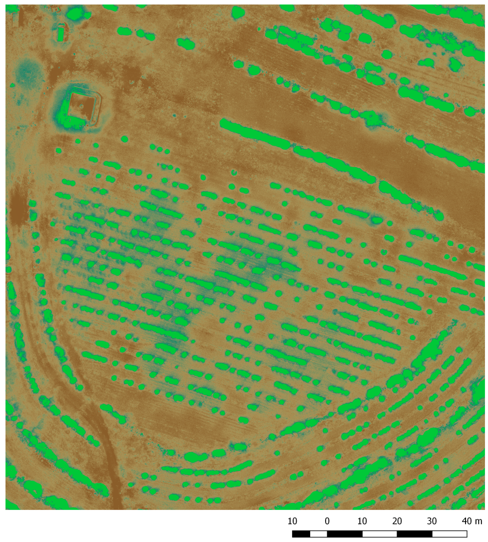
To assess the effectiveness of treetop detection methods, we implemented the algorithms described above, set experimental environments to gather data, and evaluated their performance. ~~To achieve this, we~~We divided the validation images into multiple overlapping sub-images of 1248×384 pixels and trained using the different methods. Since we partitioned the original image into sub-images, ~~the integration of~~integrating the results in a common reference frame can give rise to multiple boxes for the same tree. We used non-maximal suppression to select the bounding box corresponding to the highest confidence score from the overlapping bounding boxes with IoU ≥ 0.5 [41] (see Figure 1).

4.1. Experimental Setup

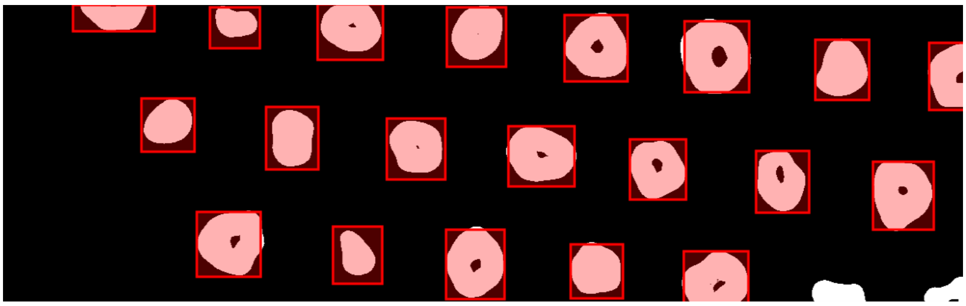
For our experiments, we flew over two different locations (see Figure 5): *Almendras* and *Mancañas*. The *Almendras* is a 3.5 hectare (ha) leaf-on almond (*Prunus dulcis*) tree plantation with a mean distance between the trees of 7.9 m located near Valencia, Spain. The *Mancañas*, in Guanajuato, Mexico, is a 0.76 ha leaf-on pine (*Pinus greggii*) with rows of trees and a mean distance between rows of 5.9 m. However, within the rows, the trees have a mean distance of ≈ 1 m.

The hardware employed to run the computer vision and image analysis algorithms consisted of a computer to implement the classical approaches and a second one for the deep learning-based methods (see §3.3 and Table 2). The former consists of a Windows 8.1 machine with an i7-3770 CPU at 3.4 GHz, 16 GB of RAM. The latter is a computer running Ubuntu 16.04 xenial with a liquid-cooled Intel Xeon E5-2650 CPU, 32 GB of RAM, four Nvidia Titan X Pascal GPUs, each one with 12 GB of video memory.

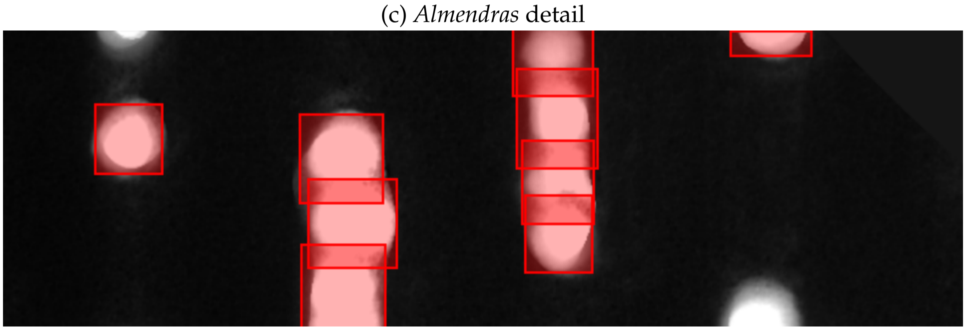
~~We flew over the *Almendras* using a self-built multicopter and over the *Mancañas* with a 3DR Solo quadricopter. In these settings, the Parrot-Sequoia produced multispectral images with spectral response peaking in wavelengths of 550 nm (Green), 660 nm (Red, R), 735 nm (Red Edge) and 790 nm (Near Infrared, NI). Each of these images has a spatial resolution of 1280 (horizontal) × 960 (vertical) pixels.~~



(a) *Almendras* (b) *Mancañas*



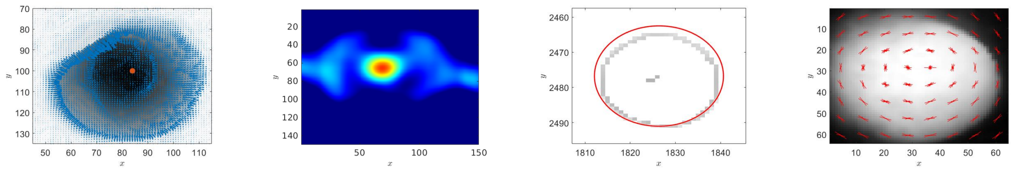
(c) *Almendras* detail



(d) *Mancañas* detail

Figure 5. Test Fields. In (a) and (b), we present the orthomosaics for the whole area of the two places we used to test our method. The *Almendras* (a) corresponds to almond (*Prunus dulcis*) trees in Spain, and the *Mancañas* (b) corresponds to pine (*Pinus greggii*) trees in Mexico. The DEVM shows that while the trees in the *Almendras* are isolated (c), in the *Mancañas* (d) the rows of trees are isolated between them but clustered together within. The bounding boxes in (c) and (d) show the detections with our method. Please note that viewed from above, almond trees seem to have a hole in the middle.

~~For our experiments, we mounted a Parrot Sequoia Micasense camera on a UAS and flew over two different locations (see Figure 5): *Almendras* and *Mancañas*. The *Almendras* is a 3.5 hectare (ha) leaf-on almond (*Prunus dulcis*) tree plantation with a mean distance between the trees of 7.9 m located near Valencia, Spain. The *Mancañas*, in Guanajuato, Mexico, is a 0.76 ha leaf-on pine (*Pinus greggii*) with rows of trees and a mean distance between rows of 5.9 m. However, within the rows, the trees have a mean distance of ≈ 1 m. We flew over the *Almendras* using a self-built multicopter and over the *Mancañas* with a 3DR Solo quadricopter. In thes~~



(a) LMF (b) Template-matching (c) Hough (d) HOG+SVM

Figure 6. Tree Detection with Classic Methods. In (a), the gradient points toward the maximum, where we place a red dot. In (b), we show the correlation between trees and a template made with synthetic images in the *Almendras* field, in (c), we show the edges (gray) and a circle fitting the edge points. In (d), we show the HOG descriptors superimposed on the corresponding DEVM patch.

4.2. Tree Detection

We trained our treetop detection methods using synthetic images and evaluated the performance on the images produced at the *Almendras* and *Mancañas* test fields. To train the deep learning-based approaches for treetop detection, we generated a synthetic-labeled dataset of 12,500 synthetic DEVM images of 1248 × 384 pixels that simulate a resolution of 1 cm/pixel with the values described in Table 1. ~~At refinement, we~~We split the 12,500 images synthetic dataset into two subsets of 10,000 images for training and 2,500 images for validation at refinement. We refined the neural network weights during ten epochs. The structure of the neural network models that we tested require three-channel images. Thus, to feed the network, we converted the DEVM to RGB images using OpenCV's *cvtColor* function, which replicates the DVEM image in each of the three channels. This process facilitated using~~the use of~~ the synthetic database on off-the-shelf neural network models requiring three-channel images, of course, at the expense of additional weights in the first convolutional layer.

We tested the efficiency of the different methods in the *Almendras* and *Mancañas* DEVM orthomosaic images containing pine (*Pinus greggii*) and almond (*Prunus dulcis*) trees, respectively. We evaluated ~~the performance of~~ our approach's performance by comparing the detections with manually obtained ground truth data (see Figure 7). We considered a detection when the Intersection of the Union (IoU) is at least 0.50 between the predicted and the ground truth bounding boxes.

We processed the individual images from our test fields with Pix4D to generate GeoTIFF orthomosaics with a size of $3,843 \times 4,386$ and $7,063 \times 8,410$ pixels for the *Almendras* and *Mancañas*, respectively. Since these images are too large for the computer's memory, we divided them into smaller overlapping clips of 1248 × 384 pixels (resulting in 176 images for the *Almendras* and 504 images for the *Mancañas*) which in turn were fed to the different methods for treetop detection. Afterward, we expressed the results on a global reference system and applied non-maximal suppression.

4.3. Results

~~For our results, we~~We trained and fine-tuned the algorithms using the synthetic dataset, while employed the *Mancañas* and the *Almendras* datasets to test without making a change to the parameters to obtain the respective performance metrics. To evaluate the performance of the different algorithms involved in the comparison, we applied the methods to the *Almendras* and *Mancañas* tree stands and evaluated different metrics, including Precision, Recall, Average Precision, Average Recall, and F_1. In Figure 8, we show the precision-recall curves resulting from varying the acceptation threshold for detection. We obtained each point of the curve by discarding those detections whose confidence score was under the threshold. The companion Table 2 highlights quantitatively some characteristics of the curves in Figure 8. In particular, it provides indicators such as the Average Precision, AP, Average Recall, AR, and the metric $F1$. The columns $AP_{0.5}$, $AR_{0.5}$, and $F1_{\max}$ follow the Pascal VOC [87] criterion, where an object is correctly detected when the IoU between its prediction and the ground truth bounding boxes is larger or equal to 0.5. Thus, $F1_{\max}$ corresponds to the maximum value of the $F1$ metric for the criterion IoU$\geq$ 0.5.

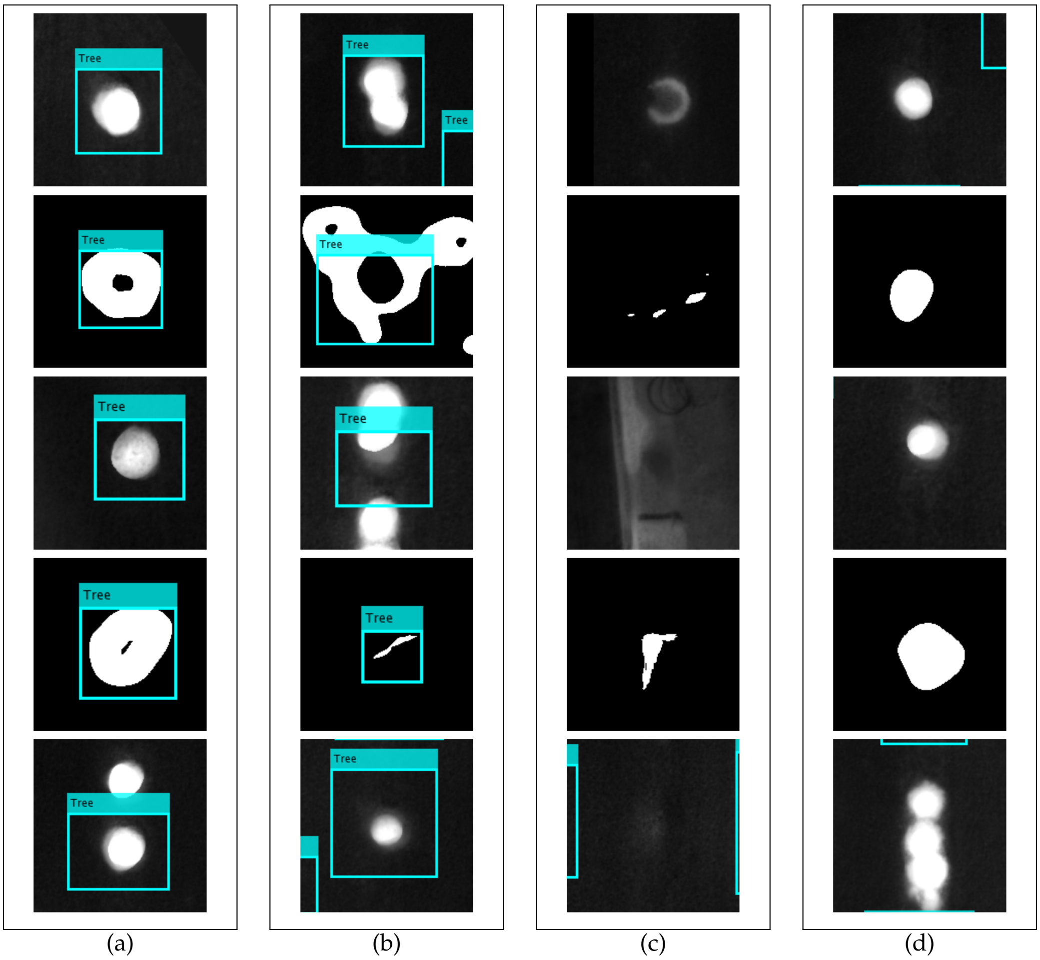
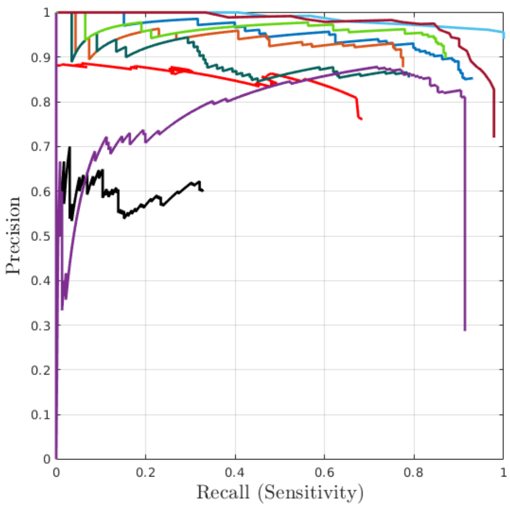


Figure 7. Tree Detection. We show the trees in DEVM images with the detected bounding boxes on it. The columns show examples of (a) true positive, (b) false-positive, (c) true-negative, and (d) false negative detection.

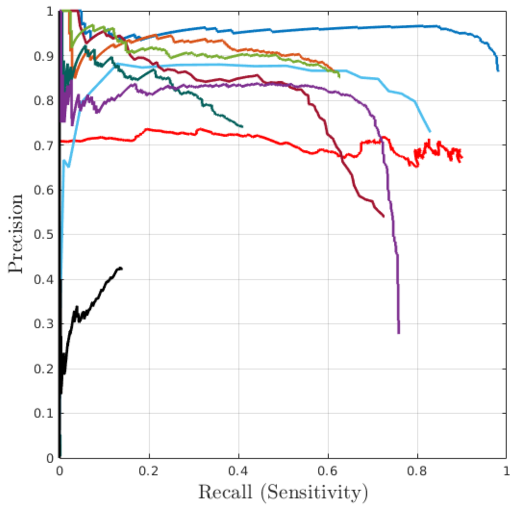
For the Template method, we selected the correlation template from the synthetic DEVM database and applied it to both *Almendras* and *Mancañas* fields. Note that consistently, the *Almendras* tree stand gave better results than the *Mancañas* tree stand for the $AP_{0.5}$, $AR_{0.5}$, and $F1_{\max}$ metrics.

Despite low averages for $AP_{0.5}$ and $AR_{0.5}$, the LMF method, with 0.918, obtained the second highest $F1_{\max}$ value for the *Almendras*. Its behavior in the *Mancañas* observed just slight fluctuations with values 0.700, 0.774 and 0.797 for $AP_{0.5}$, $AR_{0.5}$ and $F1_{\max}$, respectively. The Hough method obtained the highest $AR_{0.5}$ value at 0.950 in the *Almendras*. Interestingly, in the same metric had an abrupt decrement, at 0.442, in the *Mancañas*. It is worth noting that both methods are easy to code and exhibit low computing complexity. Template-matching had a regular performance in both the *Almendras* and the *Mancañas*, perhaps justifying the common practice of selecting the template from samples of the same image where it is going to operate but underscoring its fragility to diversity. For HOG+SVM, we computed the HOG features using training examples from the synthetic dataset and tested on the *Almendras* and the *Mancañas* fields, performing better across our metrics in the former (0.794, 0.914, 0.92) than in the latter (0.659, 0.644, 0.663) for $AP_{0.5}$, $AR_{0.5}$, and $F1_{\max}$, respectively. These results show the ability of the DEVM synthetic database to generalize well.

Applying the deep learning-based methods, SSD/Inception v2 observed the lowest performance for both tree stands. DetectNET performed better for the *Almendras* in terms of $AP_{0.5}$, at 0.880, but was outperformed by R-FCN/ResNet-101 in terms of $F1_{\max}$, at 0.922. In all other cases, the deep learning methods performance was weaker for the *Mancañas* dataset than for the *Almendras* one. A noticeable



(a) *Almendras*.



(b) *Mancañas*.

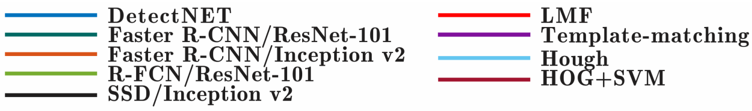


Figure 8. Precision-Recall curves for the methods described in §3.3 for the *Almendras* (a) and the *Mancañas* (b) tree stands (best seen in color). The curves describe the precision of the methods at different ~~levels of~~ recall levels, varying the confidence score to discard those detections under the threshold. See Table 2 for some quantitative highlights describing the performance.

Table 2. Performance Results. We tested in the *Almendras* and *Mancañas* tree plantations. AP, AR and $F1_{\max}$ stand for the Average Precision, Average Recall, and maximum $F1$ measures. Bold numbers correspond to the maximum per column. The methods in this table include Local Maximum Filtering (LMF), Template-matching correlation, Hough Transform to detect circles (Hough), and HOG as features followed by an SVM Classifier (HOG+SVM), DetectNet, Faster Region-based Convolutional Neural Network with Inception v2 as the backbone (Faster R-CNN/Inception v2), Faster Region-based Convolutional Neural Network with ResNet-101 as the backbone (Faster R-CNN/ResNet-101), Single-Shot Multibox Detector with Inception v2 as the backbone (SSD/Inception v2), and Region-based Fully Convolutional Networks with ResNet-101 as the backbone (R-FCN/ResNet-101). The columns show the Average Precision for an IoU of 0.5, $AP_{0.5}$, Average Recall for an IoU of 0.5, $AR_{0.5}$, and the maximum value for the $F1$ metric, $F1_{\max}$. *Computing* Time columns show the time that it takes to execute the different algorithms during training(CT_{train}) and validation(CT_{train}) stages.

	Method	*Computing Time*		*Almendras*			*Mancañas*		
		CT_{train}	CT_{val}	$AP_{0.5}$	$AR_{0.5}$	$F1_{\max}$	$AP_{0.5}$	$AR_{0.5}$	$F1_{\max}$
Classic Methods	LMF	00:00:00	10:17	0.786	0.548	0.918	0.700	0.774	0.797
	Template-matching	**00:26:00**	07:53	0.719	0.792	0.863	0.611	0.655	0.733
	Hough	12:00:00	21:53	0.779	**0.950**	0.719	0.702	0.422	0.796
	HOG+SVM	07:27:01	**00:02**	0.794	0.914	0.920	0.659	0.644	0.663
Deep Learning	DetectNet/GoogleNet	02:45:26	32:24	**0.880**	0.855	0.907	**0.920**	**0.906**	**0.940**
	F. R-CNN/Inception v2	01:06:46	02:59	0.820	0.720	0.881	0.563	0.568	0.718
	F. R-CNN/ResNet-101	07:28:26	11:52	0.796	0.722	0.873	0.343	0.357	0.526
	SSD/Inception v2	02:59:34	02:02	0.128	0.258	0.251	0.047	0.091	0.208
	R-FCN/ResNet-101	01:26:30	09:34	0.872	0.821	**0.922**	0.571	0.577	0.723

exception was the DetectNet method, which actually had a better performance for the *Mancañas* with $AP_{0.5} = 0.920$, $AR_{0.5} = 0.906$, and $F1_{\max} = 0.940$. Interestingly, Faster R-CNN, both with the Inception v2 and ResNet-101 backbones, has comparable performance in the *Almendras* but the Inception v2 backbone performed better in the *Mancañas* tree stand. R-FCN with ResNet-101 backbone was the best for the $F1_{\max}$ metric, at 0.922, for the *Almendras* but its $AP_{0.5}$, $AR_{0.5}$ and $F1_{\max}$ declined sharply for the *Mancañas* at 0.571, 0.577 and 0.723, respectively. In terms of computing time, besides LMF that does not require a training stage, the Template method took the least time to obtain the template used for detection, being less than any of the neural-network-based approaches where Faster R-CNN/Inception got trained in the least time. For evaluation, HOG+SVM was the fastest, taking only two seconds to process. 4.3 and 3.8

5. Discussion

The DEVM representation permitted us to synthesize structural and contextual information efficiently. An alternative may be to employ single RGB or multispectral images. However, the resulting system may require a large dataset of manually annotated data to work with the neural network approaches [23] and correspondingly expensive infrastructure. ~~An alternative may be to employ LiDAR and multispectral imaging sensing [88]. However, the resulting system may require a large aircraft and a correspondingly costly infrastructure.~~ Our results confirm that the DVEM representation facilitates the generation of synthetic images, which can be used effectively to train classical and modern tree detection methods. This observation aligns with Perez *et al.* [89], who have highlighted the importance of incorporating the NDVI as an input to foster the performance of automatic tree detection algorithms. However, for a precise estimation of vegetation indices, one needs to consider some crucial factors, including illumination geometry and flying height, which may play a significant role in surface reflectance determination [90]. In our work, we used the built-in Pix4D conversion formula to obtain the derived NDVI, but we may need to investigate further whether a more robust radiometric calibration could enhance tree detection performance [91]. 3.9 3.6

Our synthetic dataset includes images with a wide range of tree spacing and crown characteristics, including size, height, and shape. Therefore, it seems that neural network-based methods may generalize well for different forest types. In contrast to template matching, such approaches do not require producing a template describing a particular experimental forest [78]. We believe that the deep learning approach provides a unified framework where different tree models interact, making it easier to generalize. One could consider a CNN as a generalization of template matching where training supports the estimation of the most appropriate convolutional masks for detection [92]. This interpretation could explain the similar performance to DetectNET obtained in the *Almendras* tree stand.

Simple methods, such as LMF and Hough, performed consistently well. They can be programmed easily using widely available computer vision libraries and require humble computing platforms. From our perspective, this result confirms the often neglected value of classic approaches [93]. Perhaps, the most surprising result was the differential performance exhibited by the deep learning methods in the *Mancañas* test field. Certainly, detecting individual trees in the *Mancañas* is more difficult than in the *Almendras* site. (see Figure 5 (c)-(d)). In the latter, the trees are isolated, while in the former, the rows are isolated, but the trees within are clustered. Our results show the value of DetectNet as an object detection model based on coverage maps rather than the anchor-based models such as Faster R-CNN, SSD, and R-FCN [94]. Although they share the same backbone architecture in some cases, the bounding box extraction strategy may show different performance for specific scenarios [95]. However, further studies are needed to understand the effects of deep learning architectures in the generation of covering maps [96]. In our experiments, the DetectNet representation exhibited better performance than the region-based proposal architectures.

Conclusion

In this paper, we describe new methods to assess treetop detection methodologies efficiently. The DEVM representation made it possible to develop a strategy to construct synthetic ground truth data useful for training, alleviating the need for the task of labeling images. The representation compares well when benchmarked with classic and deep learning-based algorithms. Our experiments with datasets from two different forests provide support for our claims.

Although our results suggest that the methods can accommodate a limited degree of tree overlaps, further research is required to extend them to more challenging scenarios, such as the conditions of overlapping crowns often found in dense forests. Nonetheless, our experiments were successful in two different settings, including broadleaf and evergreen (needle leaf) trees, suggesting that we can apply the methods to other scenarios.

We plan~~are planning~~ to formulate the input to the CNN models with the raw data constituted by the near-infrared and red images, and the DMT and DSM maps. We expect that the neural network will unveil an optimized combination of the inputs to improve performance. We will also~~Also, we will~~ incorporate tree detection methods to a pipeline where SfM reconstruction and tree identification could be combined with allometric equations to obtain estimates of the stored carbon dioxide. Finally, we will continue ~~the development of~~developing algorithms for the detection of trees in more complex scenarios, such as urban areas or forests.

Author Contributions: Conceptualization, MR and JS; methodology, JS; software, DP and JS; validation, DP, KP and JS; formal analysis, DP, MR, and JS; investigation, DP and JS; resources, SK and JS; data curation, DP; writing–initial draft preparation, DP, JS; writing–review and editing, DP, JS, MR, KP, SK; visualization, DP and JS; supervision, JS, MR, and SK; project administration, JS; funding acquisition, SK and JS All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially funded by SIP-IPN 20201357 for Joaquín Salas.

Acknowledgments: We thank Bert Rijk, from Aurea Imaging, for providing us the images from Valencia. Also, we thank Othón González for piloting the UAS from where we extracted some of the images. Thanks to RAPTOR for their Phantom-3.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Appendix A

Algorithm 1 Synthetic DEVM images of trees. We generated domes to define individual tree shapes. The pdf $\mathcal{U}(a,b)$ generated random real values from a uniform distribution in the range between a and b, inclusive. $\lfloor\cdot\rfloor$ is the `floor` function.

Call: $\mathbf{I} \leftarrow$ **Synthetic_DEVM_Image**(r,c)
Input: The number of rows r and columns c for the output image
Output: Synthetic DEVM image, $\mathbf{I}_{r\times c}$

▷ Assume the existence of global constants $\mathbf{n} = (n_{\min}, n_{\max})$, $\mathbf{m} = (m_{\min}, m_{\max})$, and $\mathbf{h} = (h_{\min}, h_{\max})$, the minimum and maximum number of trees, number domes per tree, and height of the trees, respectively; $\mathbf{a} = (a_{\min}, a_{\max})$ and $\mathbf{b} = (b_{\min}, b_{\max})$, the minimum and maximum lateral widths of each dome, correspondingly; $\boldsymbol{\Delta} = (\Delta_{xy}, \Delta_{ab})$, the maximum displacement of the centroid and the change of the width, respectively; and $\mathbf{K}_{2\times 2}$ and $\bar{\mathbf{x}}$, a diagonal matrix with the relationship between meters and pixels, and the center of the dome in the image.

$\mathbf{I} \leftarrow \mathbf{0}_{w\times h}$; ▷ Initialize DEVM image to zero
$n \leftarrow \lfloor \mathcal{U}(n_{\min}, n_{\max}) \rfloor$; ▷ Define the number of trees
for $i \leftarrow 1 : n$ **do**
    $h_i \leftarrow \mathcal{U}(h_{\min}, h_{\max})$; ▷ i-th tree height
    $a_i \leftarrow \mathcal{U}(a_{\min}, a_{\max})$;
    $b_i \leftarrow \mathcal{U}(b_{\min}, b_{\max})$; ▷ i-th tree width
    $x_i \leftarrow \mathcal{U}(a_{\max} + \Delta_{xy}, c - a_{\max} - \Delta_{xy})$;
    $y_i \leftarrow \mathcal{U}(b_{\max} + \Delta_{xy}, r - b_{\max} - \Delta_{xy})$; ▷ i-th tree center
    $m_i \leftarrow \lfloor \mathcal{U}(m_{\min}, m_{\max}) \rfloor$; ▷ number of domes for the i-th tree
    for $j \leftarrow 1 : m_i$ **do**
        $h_{ij} \leftarrow h_i + \mathcal{U}(-\Delta_h, \Delta_h)$; ▷ height for the j-th domes of the i-th tree
        $x_{ij} \leftarrow x_i + \mathcal{U}(-\Delta_{xy}, \Delta_{xy})$;
        $y_{ij} \leftarrow y_i + \mathcal{U}(-\Delta_{xy}, \Delta_{xy})$; ▷ center of the j-th dome of the i-th tree
        $a_{ij} \leftarrow \mathcal{U}(a_i - \Delta_{ab}, a_i)$;
        $b_{ij} \leftarrow \mathcal{U}(b_i - \Delta_{ab}, b_i)$; ▷ width of the j-th dome of the i-th tree
        $\mathbf{D}(\alpha,\beta) = h_{ij} \cdot \cos\left(\frac{\alpha\pi}{2a_{ij}}\right) \cdot \cos\left(\frac{\beta\pi}{2b_{ij}}\right)$; ▷ create a dome using (3)
        $\mathbf{J}(\mathbf{K}\boldsymbol{\theta} + \bar{\mathbf{x}}) = \mathbf{D}(\alpha,\beta)$; ▷ transform domains, $\boldsymbol{\theta} = (\alpha,\beta)^T$
        $\mathbf{I} \leftarrow \max(\mathbf{I}, \mathbf{J})$; ▷ Add dome to image $\mathbf{I}$
    end for
end for

Appendix B

Algorithm 2 Detection of trees in images using a CNN and a synthetic dataset for training. The symbols $\oslash$ and $\odot$ represent the point-wise division and multiplication, respectively.

Call: $\mathcal{BB}$List $\leftarrow$ DETECTION(NI, R, DSM, DTM)
Input: Near Infrared (NI), Red (R), Digital Surface Map (DSM), Digital Terrain Map (DTM)
Output: $\mathcal{BB}$List ▷ Trees in the image
NDVI $\leftarrow$(NI - R) $\oslash$ (NI + R) ▷ Compute NDVI
DEVM $\leftarrow$ (DSM - DTM) $\odot$ NDVI ▷ Digital Elevated Vegetation Model
$\Delta w \leftarrow \cdots$, $\Delta r \leftarrow \cdots$ ▷ image patch size
ROIList $\leftarrow$ `Partition`(DEVM, $\Delta w, \Delta r$); ▷ divide DEVM into ROIs
$\mathcal{BB}$List $\leftarrow \{\}$
for $j \leftarrow$ 1:`length( ROIList)` **do** ▷ for all the ROIs in the image
    ROI $\leftarrow$ ROIList(j) ▷ obtain current ROI from DEVM
    $\langle BB, S \rangle \leftarrow$ `DetectNet`(ROI); ▷ bounding boxes and confidence score
    if BB is not empty **then**
        $\mathcal{BB} \leftarrow$ `GlobalCoordinates` (BB) ▷ BB in global coordinates
        $\mathcal{BB}$List $\leftarrow$ `Append` ($\mathcal{BB}$List, $\mathcal{BB}, S$) ▷ add to global list
end for
$\mathbb{BB}$List $\leftarrow$ `selectStrongestBB`($\mathcal{BB}$List) ▷ Non-Maximal Suppression

References

1. Kelly, A. Improving REDD+ (Reducing Emissions from Deforestation and Forest Degradation) Programs. PhD thesis, University of Washington, 2017.
2. Lund, G.; Thomas, C. A Primer on Stand and Forest Inventory Designs. Technical Report WO-54, US Department of Agriculture Forest Service, 1989.
3. Özyeşil, O.; Voroninski, V.; Basri, R.; Singer, A. A Survey of Structure from Motion. *Acta Numerica* **2017**, *26*, 305–364.
4. Zhang, Y.; Zhang, Y.; Yunjun, Z.; Zhao, Z. A Two-Step Semiglobal Filtering Approach to Extract DTM From Middle Resolution DSM. *IEEE Geoscience and Remote Sensing Letters* **2017**, *14*, 1599–1603.
5. Carlson, T.; Ripley, D. On the Relation between NDVI, Fractional Vegetation Cover, and Leaf Area Index. *Remote sensing of Environment* **1997**, *62*, 241–252.
6. Ozdarici-Ok, A. Automatic Detection and Delineation of Citrus Trees from VHR Satellite Imagery. *International Journal of Remote Sensing* **2015**, *36*, 4275–4296.
7. Gomes, M.; Maillard, P. Detection of Tree Crowns in Very High Spatial Resolution Images. In *Environmental Applications of Remote Sensing*; InTech, 2016.
8. Koc-San, D.; Selim, S.; Aslan, N.; San, B. Automatic Citrus Tree Extraction from UAV Images and Digital Surface Models using Circular Hough Transform. *Computers and Electronics in Agriculture* **2018**, *150*, 289–301.
9. Özcan, A.; Hisar, D.; Sayar, Y.; Ünsalan, C. Tree Crown Detection and Delineation in Satellite Images using Probabilistic Voting. *Remote Sensing Letters* **2017**, *8*, 761–770.
10. Shafarenko, L.; Petrou, M.; Kittler, J. Automatic Watershed Segmentation of Randomly Textured Color Images. *IEEE Transactions on Image Processing* **1997**, *6*, 1530–1544.
11. Reza, N.; Na, S.; Lee, K. Automatic Counting of Rice Plant Numbers after Transplanting using Low Altitude UAV Images. *International Journal of Contents* **2017**, *13*, 1–8.
12. Maillard, P.; Gomes, M. Detection and Counting of Orchard Trees from VHR Images using a Geometrical-Optical Model and Marked Template Matching. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* **2016**, *3*, 75.
13. Bao, Y.; Tian, Q.; Chen, M.; Lin, H. An Automatic Extraction Method for Individual Tree Crowns based on Self-Adaptive Mutual Information and Tile Computing. *International Journal of Digital Earth* **2015**, *8*, 495–516.
14. Goldbergs, G.; Maier, S.; Levick, S.; Edwards, A. Efficiency of Individual Tree Detection Approaches Based on Light-Weight and Low-Cost UAS Imagery in Australian Savannas. *Remote Sensing* **2018**, *10*, 161.

15. Fassnacht, F.; Mangold, D.; Schäfer, J.; Immitzer, M.; Kattenborn, T.; Koch, B.; Latifi, H. Estimating Stand Density, Biomass and Tree Species from Very High Resolution Stereo-Imagery–Towards an All-in-One Sensor for Forestry Applications. *Forestry: An International Journal of Forest Research* **2017**, pp. 1–19.
16. Wang, Y.; Zhu, X.; Wu, B. Automatic Detection of Individual Oil Palm Trees from UAV Images using HOG Features and an SVM Classifier. *International Journal of Remote Sensing* **2018**, pp. 1–15.
17. Li, W.; He, C.; Fu, H.; Zheng, J.; Dong, R.; Yu, L.; Luk, W. A Real-Time Tree Crown Detection Approach for Large-Scale Remote Sensing Images on FPGAs. *Remote Sensing* **2019**, *11*, 1025.
18. Marinelli, D.; Paris, C.; Bruzzone, L. An Approach to Tree Detection Based on the Fusion of Multitemporal LiDAR Data. *IEEE Geoscience and Remote Sensing Letters* **2019**.
19. Xiao, C.; Qin, R.; Xie, X.; Huang, X. Individual Tree Detection and Crown Delineation with 3D Information from Multi-view Satellite Images. *Photogrammetric Engineering & Remote Sensing* **2019**, *85*, 55–63.
20. Kolendo, Ł.; Ksepko, M. Selection of Optimal Tree Top Detection Parameters in a Context of Effective Forest Management. *Ekonomia i Środowisko* **2019**.
21. García, D.; Caicedo, J.; Castellanos, G. Individual Detection of Citrus and Avocado Trees Using Extended Maxima Transform Summation on Digital Surface Models. *Remote Sensing* **2020**, *12*, 1633.
22. Tianyang, D.; Jian, Z.; Sibin, G.; Ying, S.; Jing, F. Single-Tree Detection in High-Resolution Remote-Sensing Images Based on a Cascade Neural Network. *ISPRS International Journal of Geo-Information* **2018**, *7*, 367.
23. Guo, Y.; Liu, Y.; Oerlemans, A.; Lao, S.; Wu, S.; Lew, M. Deep Learning for Visual Understanding: A Review. *Neurocomputing* **2016**, *187*, 27–48.
24. Individual tree-crown detection in RGB imagery using semi-supervised deep learning neural networks, author=Weinstein, Ben G and Marconi, Sergio and Bohlman, Stephanie and Zare, Alina and White, Ethan. *Remote Sensing* **2019**, *11*, 1309.
25. Freudenberg, M.; Nölke, N.; Agostini, A.; Urban, K.; Wörgötter, F.; Kleinn, C. Large Scale Palm Tree Detection In High Resolution Satellite Images Using U-Net. *Remote Sensing* **2019**, *11*, 312.
26. Li, W.; Dong, R.; Fu, H.; others. Large-Scale Oil Palm Tree Detection from High-Resolution Satellite Images Using Two-Stage Convolutional Neural Networks. *Remote Sensing* **2019**, *11*, 11.
27. Zortea, M.; Nery, M.; Ruga, B.; Carvalho, L.; Bastos, A. Oil-Palm Tree Detection in Aerial Images Combining Deep Learning Classifiers. IEEE International Geoscience and Remote Sensing Symposium. IEEE, 2018, pp. 657–660.
28. Csillik, O.; Cherbini, J.; Johnson, R.; Lyons, A.; Kelly, M. Identification of Citrus Trees from Unmanned Aerial Vehicle Imagery using Convolutional Neural Networks. *Drones* **2018**, *2*, 39.
29. Li, W.; Fu, H.; Yu, L.; Cracknell, A. Deep Learning based Oil Palm Tree Detection and Counting for High-Resolution Remote Sensing Images. *Remote Sensing* **2016**, *9*, 22.
30. Li, W.; Fu, H.; Yu, L. Deep Convolutional Neural Network based Large-Scale Oil Palm Tree Detection for High-Resolution Remote Sensing Images. IEEE International Geoscience and Remote Sensing Symposium. IEEE, 2017, pp. 846–849.
31. Mubin, N.; Nadarajoo, E.; Shafri, H.; Hamedianfar, A. Young and Mature Oil Palm Tree Detection and Counting using Convolutional Neural Network Deep Learning Method. *International Journal of Remote Sensing* **2019**, pp. 1–16.
32. Cheang, E.; Cheang, T.; Tay, Y. Using Convolutional Neural Networks to Count Palm Trees in Satellite Images. *arXiv preprint arXiv:1701.06462* **2017**.
33. Fan, Z.; Lu, J.; Gong, M.; Xie, H.; Goodman, E.D. Automatic Tobacco Plant Detection in UAV Images via Deep Neural Networks. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **2018**, *11*, 876–887.
34. Trier, Ø.; Salberg, A.B.; Kermit, M.; Rudjord, Ø.; Gobakken, T.; Næsset, E.; Aarsten, D. Tree Species Classification in Norway from Airborne Hyperspectral and Airborne Laser Scanning Data. *European Journal of Remote Sensing* **2018**, *51*, 336–351.
35. Windrim, L.; Bryson, M. Forest Tree Detection and Segmentation using High Resolution Airborne LiDAR. *arXiv preprint arXiv:1810.12536* **2018**.
36. Pibre, L.; Chaumont, M.; Subsol, G.; Ienco, D.; Derras, M. How to Deal with Multi-Source Data for Tree Detection based on Deep Learning. GlobalSIP: Global Conference on Signal and Information Processing, 2017.

37. Chen, S.; Shivakumar, S.; Dcunha, S.; Das, J.; Okon, E.; Qu, C.; Taylor, C.; Kumar, V. Counting Apples and Oranges with Deep Learning: A Data-Driven Approach. *IEEE Robotics and Automation Letters* **2017**, *2*, 781–788.
38. Zakharova, M. Automated Coconut Tree Detection in Aerial Imagery Using Deep Learning. PhD thesis, The Katholieke Universiteit Leuven, Löwen, Belgium, 2017.
39. Puttemans, S.; Van Beeck, K.; Goedemé, T. Comparing Boosted Cascades to Deep Learning Architectures for Fast and Robust Coconut Tree Detection in Aerial Images. International Conference on Computer Vision Theory and Applications, 2018.
40. Zortea, M.; Macedo, M.; Britto, A.; Ruga, B. Automatic Citrus Tree Detection from UAV Images based on Convolutional Neural Networks. Conference on Graphics, Patterns and Images, 2018.
41. Viola, P.; Jones, M. Rapid Object Detection using a Boosted Cascade of Simple Features. IEEE Conference on Computer Vision and Pattern Recognition, 2001, Vol. 1, pp. I–I.
42. Dollár, P.; Appel, R.; Belongie, S.; Perona, P. Fast Feature Pyramids for Object Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence* **2014**, *36*, 1532–1545.
43. Ribera, J.; Chen, Y.; Boomsma, C.; Delp, E. Plant Leaf Segmentation for Estimating Phenotypic Traits. IEEE International Conference on Image Processing, 2017.
44. Xiao, C.; Qin, R.; Huang, X.; Li, J. A Study of using Fully Convolutional Network for Treetop Detection on Remote Sensing Data. *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences* **2018**, *4*.
45. Long, J.; Shelhamer, E.; Darrell, T. Fully Convolutional Networks for Semantic Segmentation. IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3431–3440.
46. Santos, A.d.; Marcato, J.; Araújo, M.S.; Di Martini, D.; Tetila, E.; Siqueira, H.; Aoki, C.; Eltner, A.; Matsubara, E.; Pistori, H.; others. Assessment of CNN-Based Methods for Individual Tree Detection on Images Captured by RGB Cameras Attached to UAVs. *Sensors* **2019**, *19*, 3595.
47. Fromm, M.; Schubert, M.; Castilla, G.; Linke, J.; McDermid, G. Automated Detection of Conifer Seedlings in Drone Imagery Using Convolutional Neural Networks. *Remote Sensing* **2019**, *11*, 2585.
48. Zhao, J.; Xiong, L.; Jayashree, K.; Li, J.; Zhao, F.; Wang, Z.; Pranata, S.; Shen, S.; Yan, S.; Feng, J. Dual-Agent GANs for Photorealistic and Identity Preserving Profile Face Synthesis. Advances in Neural Information Processing Systems, 2017, pp. 65–75.
49. Liu, J.; Mian, A. Learning Human Pose Models from Synthesized Data for Robust RGB-D Action Recognition. *arXiv:1707.00823* **2017**.
50. Huang, R.; Zhang, S.; Li, T.; He, R. Beyond Face Rotation: Global and Local Perception GAN for Photorealistic and Identity Preserving Frontal View Synthesis. *arXiv:1704.04086* **2017**.
51. Bi, L.; Kim, J.; Kumar, A.; Feng, D.; Fulham, M. Synthesis of Positron Emission Tomography (PET) Images via Multi-channel Generative Adversarial Networks (GANs). In *Molecular Imaging, Reconstruction and Analysis of Moving Body Organs, and Stroke Imaging and Treatment*; Springer, 2017; pp. 43–51.
52. Tom, F.; Sheet, D. Simulating Patho-realistic Ultrasound Images using Deep Generative Networks with Adversarial Learning. *arXiv:1712.07881* **2017**.
53. Costa, P.; Galdran, A.; Meyer, M.; Niemeijer, M.; Abràmoff, M.; Mendonça, A.; Campilho. End-to-End Adversarial Retinal Image Synthesis. *IEEE Transactions on Medical Imaging* **2017**.
54. Tsirikoglolu, A.; Kronander, J.; Wrenninge, M.; Unger, J. Procedural Modeling and Physically based Rendering for Synthetic Data Generation in Automotive Applications. *arXiv:1710.06270* **2017**.
55. Planche, B.; Wu, Z.; Ma, K.; Sun, S.; Kluckner, S.; Chen, T.; Hutter, A.; Zakharov, S.; Kosch, H.; Ernst, J. DepthSynth: Real-Time Realistic Synthetic Data Generation from CAD Models for 2.5D Recognition. *arXiv:1702.08558* **2017**.
56. Malmgren-Hansen, D.; Kusk, A.; Dall, J.; Nielsen, A.; Engholm, R.; Skriver, H. Improving SAR Automatic Target Recognition Models With Transfer Learning From Simulated Data. *IEEE Geoscience and Remote Sensing Letters* **2017**, *14*, 1484–1488.
57. Ubbens, J.; Cieslak, M.; Prusinkiewicz, P.; Stavness, I. The Use of Plant Models in Deep Learning: An Application to Leaf Counting in Rosette Plants. *Plant Methods* **2018**, *14*, 6.
58. Han, S.; Kerekes, J. Overview of Passive Optical Multispectral and Hyperspectral Image Simulation Techniques. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* **2017**, *10*, 4794–4804.

59. Fassnacht, F.; Latifi, H.; Hartig, F. Using Synthetic Data to Evaluate the Benefits of Large Field Plots for Forest Biomass Estimation with LiDAR. *Remote Sensing of Environment* **2018**, *213*, 115–128.
60. Pretzsch, H.; Biber, P.; Ďurskỳ, J. The Single Tree-based Stand Simulator SILVA: Construction, Application and Evaluation. *Forest Ecology and Management* **2002**, *162*, 3–21.
61. Wu, X.; Shen, X.; Cao, L.; Wang, G.; Cao, F. Assessment of Individual Tree Detection and Canopy Cover Estimation using Unmanned Aerial Vehicle based Light Detection and Ranging (UAV-LiDAR) Data in Planted Forests. *Remote Sensing* **2019**, *11*, 908.
62. Selim, S.; Sonmez, N.K.; Coslu, M.; Onur, I. Semi-automatic Tree Detection from Images of Unmanned Aerial Vehicle Using Object-Based Image Analysis Method. *Journal of the Indian Society of Remote Sensing* **2019**, pp. 1–8.
63. Krisanski, S.; Taskhiri, M.S.; Turner, P. Enhancing Methods for Under-Canopy Unmanned Aircraft System based Photogrammetry in Complex Forests for Tree Diameter Measurement. *Remote Sensing* **2020**, *12*, 1652.
64. Qiu, L.; Jing, L.; Hu, B.; Li, H.; Tang, Y. A New Individual Tree Crown Delineation Method for High Resolution Multispectral Imagery. *Remote Sensing* **2020**, *12*, 585.
65. Kuželka, K.; Slavík, M.; Surovỳ, P. Very High Density Point Clouds from UAV Laser Scanning for Automatic Tree Stem Detection and Direct Diameter Measurement. *Remote Sensing* **2020**, *12*, 1236.
66. Picos, J.; Bastos, G.; Míguez, D.; Alonso, L.; Armesto, J. Individual Tree Detection in a Eucalyptus Plantation Using Unmanned Aerial Vehicle (UAV)-LiDAR. *Remote Sensing* **2020**, *12*, 885.
67. Yan, W.; Guan, H.; Cao, L.; Yu, Y.; Li, C.; Lu, J. A Self-Adaptive Mean Shift Tree-Segmentation Method using UAV LiDAR Data. *Remote Sensing* **2020**, *12*, 515.
68. Xue, J.; Su, B. Significant Remote Sensing Vegetation Indices: A Review of Developments and Applications. *Journal of Sensors* **2017**, *2017*.
69. Kotchi, S.; Viau, A.; Barrette, N.; Gond, V.; Jang, J.; Mostafavi, M. *Uncertainty Assessment and Comparison of Vegetation Indices, Surface Emissivity Models and Split-Window Algorithms used to Estimate Surface Temperature from Satellite Images*; Nova Science Publisher, 2017.
70. Rouse, J.; Haas, R.; Schell, J.; Deering, D. Monitoring Vegetation Systems in the Great Plains with ERTS. NASA. Goddard Space Flight Center 3d ERTS-1 Symp, 1974.
71. Weier, J.; Herring, D. Measuring Vegetation (NDVI & EVI), 2000.
72. Gu, Y.; Wylie, B.K.; Howard, D.; Phuyal, K.; Ji, L. NDVI Saturation Adjustment: A New Approach for Improving Cropland Performance Estimates in the Greater Platte River Basin, USA. *Ecological Indicators* **2013**, *30*, 1–6.
73. Liu, F.; Qin, Q.; Zhan, Z. A Novel Dynamic Stretching Solution to Eliminate Saturation Effect in NDVI and its Application in Drought Monitoring. *Chinese Geographical Science* **2012**, *22*, 683–694.
74. Rahmes, M.; Allen, J.; Yates, J.; Kelley, P. Production System for Autonomous 3-Dimensional Modeling with LiDAR, IFSAR, and Photogrammetric DSM Data. *ASPRS, May* **2007**.
75. Unger, M.; Pock, T.; Klaus, A.; Bischof, H. A Variational Approach to Semiautomatic Generation of Digital Terrain Models. International Symposium on Visual Computing. Springer, 2009, pp. 1119–1130.
76. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, L. Microsoft COCO: Common Objects in Context. European Conference on Computer Vision. Springer, 2014, pp. 740–755.
77. Pouliot, D.; King, D.; Bell, F.; Pitt, D. Automated Tree Crown Detection and Delineation in High-Resolution Digital Camera Imagery of Coniferous Forest Regeneration. *Remote Sensing of Environment* **2002**, *82*, 322–334.
78. Ke, Y.; Quackenbush, L. A Review of Methods for Automatic Individual Tree-Crown Detection and Delineation from Passive Remote Sensing. *International Journal of Remote Sensing* **2011**, *32*, 4725–4747.
79. Dalal, N.; Triggs, B. Histograms of Oriented Gradients for Human Detection. IEEE International Conference on Computer Vision and Pattern Recognition, 2005, Vol. 1, pp. 886–893.
80. Barker, J.; Sarathy, S.; Tao, A. DetectNet: Deep Neural Network for Object Detection in DIGITS. *Nvidia,(2016-11-30), https://tinyurl.com/detectnet* **2016**.
81. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. IEEE Conference on Computer Vision and Pattern Recognition, 2016, pp. 779–788.
82. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going Deeper with Convolutions. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1–9.

83. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets Robotics: The KITTI Dataset. *International Journal of Robotics Research* **2013**.
84. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. Advances in Neural Information Processing Systems, 2015, pp. 91–99.
85. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A. SSD: Single-Shot Multibox Detector. European Conference on Computer Vision. Springer, 2016, pp. 21–37.
86. Tieleman, T.; Hinton, G. Lecture 6.5—RmsProp: Divide the Gradient by a Running Average of its Recent Magnitude. Coursera: Neural Networks for Machine Learning, 2012.
87. Everingham, M.; Van Gool, L.; Williams, C.; Winn, J.; Zisserman, A. The Pascal Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision* **2010**, *88*, 303–338.
88. Kukkonen, M.; Maltamo, M.; Korhonen, L.; Packalen, P. Multispectral Airborne LiDAR Data in the Prediction of Boreal Tree Species Composition. *IEEE Transactions on Geoscience and Remote Sensing* **2019**, *57*, 3462–3471.
89. Pérez-Bueno, M.; Pineda, M.; Vida, C.; Fernández-Ortuño, D.; Torés, J.; de Vicente, A.; Cazorla, F.; Barón, M. Detection of White Root Rot in Avocado Trees by Remote Sensing. *Plant Disease* **2019**, pp. PDIS–10.
90. Stow, D.; Nichol, C.; Wade, T.; Assmann, J.; Simpson, G.; Helfter, C. Illumination Geometry and Glying Height Influence Surface Reflectance and NDVI Derived from Multispectral UAS Imagery. *Drones* **2019**, *3*, 55.
91. Fawcett, D.; Panigada, C.; Tagliabue, G.; Boschetti, M.; Celesti, M.; Evdokimov, A.; Biriukova, K.; Colombo, R.; Miglietta, F.; Rascher, U.; Anderson, K. Multi-Scale Evaluation of Drone-Based Multispectral Surface Reflectance and Vegetation Indices in Operational Conditions. *Remote Sensing* **2020**, *12*, 514.
92. Goodfellow, I.; Bengio, Y.; Courville, A.; Bengio, Y. *Deep Learning*; Vol. 1, MIT press Cambridge, 2016.
93. Marcus, G. Deep Learning: A Critical Appraisal. *arXiv preprint arXiv:1801.00631* **2018**.
94. Huang, J.; Rathod, V.; Sun, C.; Zhu, M.; Korattikara, A.; Fathi, A.; Fischer, I.; Wojna, Z.; Song, Y.; Guadarrama, S. Speed/Accuracy Trade-offs for Modern Convolutional Object Detectors. IEEE Computer Vision and Pattern Recognition, 2017, pp. 7310–7311.
95. Wang, H.; Yu, Y.; Cai, Y.; Chen, X.; Chen, L.; Liu, Q. A Comparative Study of State-of-the-Art Deep Learning Algorithms for Vehicle Detection. *IEEE Intelligent Transportation Systems Magazine* **2019**.
96. Lenc, K. Representation of Spatial Transformations in Deep Neural Networks. PhD thesis, University of Oxford, 2018.