MDPI

Technical Note

DIRT: The Dacus Image Recognition Toolkit

Romanos Kalamatianos [1,*], Ioannis Karydis [2,3], Dimitris Doukakis [4] and Markos Avlonitis [5]

[1] Dept. of Informatics, Ionian University, 49132, Kerkyra, Greece; rkalam@ionio.gr
[2] Dept. of Informatics, Ionian University, 49132, Kerkyra,Greece; karydis@ionio.gr
[3] Creative Web Applications P.C., 49131, Kerkyra, Greece; jonjon@cwa.gr
[4] Dept. of Informatics, Ionian University, 49132, Kerkyra, Greece; di.doukakis@gmail.com
[5] Dept. of Informatics, Ionian University, 49132, Kerkyra, Greece; avlon@ionio.gr
* Correspondence: rkalam@ionio.gr; Tel.: +302661087752

Academic Editor: name
Version October 25, 2018 submitted to J. Imaging

Abstract: Modern agriculture is facing unique challenges in building a sustainable future for food production, in which the reliable detection of plantations' threats is of critical importance. The breadth of existing information sources, and their equivalent sensors, can provide a wealth of data which, to be useful, must be transformed into actionable knowledge. Approaches based on Information Communication Technologies (ICT) have been shown to be able to help farmers, and related stakeholders, make decisions on problems by examining large volumes of data while assessing multiple criteria. In this paper we address the automated identification (and counting of instances) of the major threat of olive trees and their fruit, the Bactrocera Oleae (a.k.a. Dacus) based on images of the commonly used McPhail trap's contents. Accordingly, we introduce the "Dacus Image Recognition Toolkit" (*DIRT*), a collection of publicly available data, programming code samples and web-services focused at supporting research aiming at the management the Dacus as well as extensive experimentation on the capability of the proposed dataset in identifying Dacuses using Deep Learning methods. Experimental results indicated performance accuracy (mAP) of 91.52% in identifying Dacuses in traps' images featuring various pests. Moreover, the results also indicated a trade-off between images' attributes affecting detail, file size & complexity of approaches and mAP performance that can be selectively used to better tackle the needs of each usage scenario.

Keywords: object recognition; deep learning; Bactrocera Oleae; Dacus; olive-fruit fly; smart-traps; public dataset; public API/web-service; IPM DSS; olive cultivation

1. Introduction

Modern agriculture is facing unique challenges in building a sustainable future [1,2] in a way that empowers the agricultural sector to meet the world's food needs. Reliable detection of plantation's threats by pests/diseases as well as proper quantification of induced damages are of critical importance [3,4]. Moreover, early detection of these phenomena is crucial for managing and reducing their spread, maintaining production's quality and quantity as well as reducing costs, trade disruptions and sometimes even mitigate human health risks.

Pests' and diseases' detection is done on information aggregated from various sources such as plant examination, arrays of plantations' sensors, diagnostic images of plants, weather stations, etc [5,6]. Such wealth of information, to be useful in transforming raw data to actionable knowledge, requires advanced Information Communication Technologies (ICT) approaches that will help farmers, and related stakeholders, make decisions on problems requiring large volumes of data and dependent on multiple criteria [7]. It is thus evident that modern agriculture requires adoption of production

processes, technologies and tools derived from scientific advances, results from research and innovation activities in different fields (ICT, agronomic, entomologic, weather analysis, etc) [8].

As olive trees are the most dominant permanent crop within EU in terms of occupied areas (40% of permanent crops' total area [9]) with more than 1500 cultivars [10] just in the Mediterranean, our work focuses on one of its major threats [11], the olive fruit fly (Bactrocera Oleae, Dacus). Measurements of the fly's infestation in olive groves are predominantly done with manual methods involving traps, while one of the key requirements in verifying an outbreak lies in measuring the pests collected in a trap over a time-span [12]. This process necessitates frequent and time consuming manual checks similar to no other parameter/requirement of the trap.

Advanced traps, or smart-traps, feature a camera taking pictures of the pests collected by the trap that are then examined by interested parties [13–17]. Based on the images of the pests collected in the traps, stakeholders such as farmers, entomologists, agronomists, etc. can identify and measure the collected pests, customise the frequency of trap's examination while also minimise the examination time and its associated difficulty.

1.1. Motivation and Contribution

Despite the aforementioned existing advances of smart-traps, in order to be able to extract knowledge from the aforementioned smart-traps' collected data, existing methodologies must be able to compare results. Thus, use of a common set of traps' observation data is necessary for the testing of the efficiency and effectiveness of the methods, while also providing reference for comparison of new and existing methods in order to show progress, as is the case with most scientific datasets [18,19]. To the best of our knowledge existing research works in automated Dacus identification are not operating on the same data and thus the results presented therein are not easily comparable.

Data collection from Dacus traps is a lengthy process that requires multiple locations of olive groves, frequent physical attention to traps, minor entomological knowledge, appropriate hardware, while it is only possible for the chronological period that Dacuses are active [20]. All of these factors make the collection of Dacus traps data rather difficult. Thus, the evaluation of new methods is hampered by the lack of easily accessible data to test the methods on.

To address the aforementioned requirements, we introduce the Dacus Image Recognition Toolkit (*DIRT*), a collection of publicly available data, programming code samples and web-services focused at supporting research aiming at the management the olive fruit-fly. *DIRT* offers:

- a dataset of images depicting McPhail traps' contents,
- manually annotated spatial identification of Dacuses in the dataset,
- programming code samples in matlab that allow fast initial experimentation on the dataset or any other Dacus image set,
- a public rest https API and web-interface that reply to queries for Dacus identification in user provided images,
- extensive experimentation on the use of deep learning for Dacus identification on the dataset's contents.

The rest of the paper is organised as follows: Section 2 presents the related work on automated Dacus identification, and smart-trap works, while Section 3 discusses a general architecture for a smart-trap. Section 4 details the toolkit's components: the creation processes of the dataset and a complete analysis of its contents, the programming code samples provided and the Dacus identification API. Next, Section 5 explores the use of deep learning for the identification of Dacuses and presents extensive experimental results obtained using the dataset. Finally, the paper is concluded in Section 6 including details on future directions concerning the toolkit that could ameliorate its usability and further support pest management research.

2. Related Research

This Section details related work on automated Dacus identification, and smart-trap works as presented in the literature. Most of the existing research on pests' identification from images utilises some form of image processing in order to discard extraneous information from the images and highlight the features related to the pests to be identified. Moreover, in most cases, an assumption on the size of the pests to be identified has also been made, either based on training or with hard-coded thresholding, rendering thus pixel-sizes outside a range either noise or alternative to the intended pests. This, spatial feature-set, allows only for predefined aimed pests' size variability, while also it makes no distinction of entirely different pests of the same size.

In [17], Tirelli et al. presented an automatic monitoring of pest insects low consumption wireless networking image sensors. Therein, open-air examination areas were placed near plats aiming at recording the plants' pests. Subsequently, images of the examination area during sampling for pests were compared to reference images of the same examination area without pests in order to calculate their difference. Hard-coded thresholding was used in order to trim out very small or very large pixel-sized differences, and the remaining were assumed to be pests. The image processing also included noise reduction and conversion to binary image, all done at a central server. Experimentation showed correlation of the pests detected with the ground-truth of pests.

Another smart-trap is also presented in [16]. Therein, Philimis et al. describe a wirelessly connected monitoring system that consists of automated traps with optical and motion detection modules for capturing the pests as well as a central station that collects, stores and makes available processed results, such as insect population. This system allows for real-time analysis of pests' images leading to determination of its species (Medfly or Dacus) and as well as its sex. Sadly, the work does not elaborate further as to the either the methodology for the real-time identification or its accuracy. Nevertheless, the work is part of the EU Project "e-FlyWatch"[1] wherein little provided explanations indicate that the methodology is based on a spatial features that compare the identified pest with templates in addition to pattern-based identification for "unique insect features, as for example the abdomen, wings and thorax"[2].

In [15], Wang et al. presented the design of an image identification system using computer vision methods for a wide variety of fruit flies that negatively impact international fruit trade. The system proposed therein tackles 74 species that constitute the majority of pests in the Tephritidae. Their dataset includes three body parts, the wing, abdomen, and thorax of fruit flies based on which identification is performed either individually or in any combination. The identification process includes steps such as gamma correction for varying illumination issues, multi-orientation and multi-scale features based on Gabor filters and k-NN classification. Experimentation indicated an overall classification success rate of 87% at the species level.

Identification of insects that are submerged in liquids are not only interesting for identifying pests in collections of unsorted material but to the theme if this work as well, as traps that use liquid pheromones to attract pests usually lead to "soup images" where the pests to be identified are more often than not submerged in the liquid. Sun et al. [21] proposed an image analysis approach for analysing insect soup images by identifying dark bodies in bright background leading to shapes of pests. Subsequently, measurements such as size/area, length, width, colour as well as feature extraction are done, based on which insect sub-region sorting based on size is finally made.

Doitsidis et al. [13] presented an automated McPhail trap for the monitoring of Dacus' population. Their smart-trap design allows for image capturing of the contents of the trap that are subsequently transmitted to a server and processed in order identify Dacuses. The methodology for the Dacuses identification in the images is based on image processing procedures such auto brightness correction,

[1] https://cordis.europa.eu/project/rcn/96182_en.html
[2] https://cordis.europa.eu/result/rcn/141151_en.html

edge detection (CLF), conversion to binary (Otsu), bounds detection (Circle Hough Transform) and noise reduction. Training was done using 100 manually annotated images that lead to a computed average size (in terms of percentage of black area - black pixels) of a single fruit fly as a percentage of the area of interest, i.e. the trap, making thus requiring use of a specific trap or knowledge of trap's dimensions. Extensive testing done therein indicated accuracy to reach 75%.

In [22], Potamitis et al. proposed the modification of typical, low-cost plastic fruit flies' trap (McPhail type) with the addition of optoelectronic sensors monitoring the entrance of the trap in order to detect and identify the species of incoming insects, from the optoacoustic spectrum analysis of their wing-beat, leading thus automated streaming of insect count. The identification is done based on comparison of the amplitude of the time domain recording of an insect entering the trap with the ground truth range derived from a large number of same insect recordings, thus not discriminating pests with high overlapping spectra. Experimentation indicated a 0.93 and 0.95 average F1-score for all fruit flies tested and Dacus in the lab, respectively.

Shaked et al. [12] presented the design of two smart-traps for four fruit fly species featuring a variety of configurations, all of which utilise wireless communications to propagate captured of images of trapped insects on sticky surfaces. One rather interesting configuration of this work pertains to the network topology that, in addition to the usually utilised star network, also proposes a mesh network which provides robustness since multiple routes for for data to travel to exist if one node fails. The work did not feature an automated fruit fly identification methodology, while it seems to be an extension of the the earlier work of Alorda et al. [14] (under the auspices of the same project, "FruitFlyNet") presenting a energy efficient and low cost trap for Olive fly monitoring using a ZigBee-based Wireless Sensor Network.

Finally, a number of works address other notorious pests of commercially important fruits and vegetables such as the Tephritid Fruit Flies by use of trapping & detection, control, and regulation mostly at the entomological and agronomical levels [23], codling moth using a convolutional neural network-based detection pipeline on an unamed commercial dataset [24], and wood-boring beetles arriving at high-risk sites [25].

3. Smart Trap

As indicated in Section 1, a large amount of various reliable data are to be collected in a frequent basis with minimum cost in order to be able to address Integrated Pest Management (IPM) requirements. To do so, a generic way of collecting, digitising and transmitting the necessary data at a processing center is developed in this study. This general architecture for a smart-trap, the Electronic Olive Fruit fly trap, utilises an electronic version of the classical McPhail trap and comprises of the following parts:

McPhail-type trap A McPhail type trap with enlarged upper part completely equivalent in terms of size (inner trap volume), environmental conditions (temperature, humidity, etc), and parameters effecting the attraction of Dacuses, the entrance type for the pests, and the difficulty of exit. The extra height of the upper part is important in order to accommodate all the necessary electronic parts described in the sequel as well as to allow space for proper focus of the camera. The electronics compartment is to be completely isolated from the rest part of the trap (e.g by means of a transparent PVC plate or equivalent methods).

Wi-fi equipped microcomputer A microcomputer for the task of orchestrating all the necessary actions in order to record the data and dispatch these to a networking module (e.g. a GSM modem), thus reaching finally to a server/processing center. The microcomputer is to be selected based on the following criteria:

- low cost,
- computational resources,
- number of open-source programs available for it,
- operational stability,

- availability of integrated Wi-Fi (and/or other protocols') transceiver, and
- capability for integration of camera with fast interface and adequate resolution.

The key disadvantage of including a microcomputer is its relative high-power consumption, despite the numerous techniques existing for the minimisation of stand-by consumption. The main alternative, micro-controllers, can also be considered for the task given that preliminary tests indicate that the computational load is not too big for their limited resources (such as RAM & CPU speed). The microcomputer proposed features a Unix-type operating system for openness while with the use of scripts (e.g. python) will collect data from the sensors (mentioned in the sequel) at explicitly defined time instances of the day. Then, the data will be transmitted through the networking to a server/processing center. Both collection and transmission may be synchronised with scripts (e.g. Unix bash).

Real time clock An accurate battery equipped Real-time clock module.

Camera An adequate resolution camera with adaptable lenses system in order to achieve focusing and zooming.

Sensors A high accuracy humidity and temperature sensor set within (and additionally possibly outside) the enclosure of the trap. Similar remote sensors may also be used in order to collect ambient readings.

Power supply A grid power supply system based a battery with adequate capacity in order to supply the necessary electrical power to the smart-trap for a few days. In order for the smart-trap to be an autonomous and a maintenance-free device, a solar panel and a charger system are to be included and accommodated to a waterproof box nearby the trap.

Networking Despite the abundance of alternative networking configurations (e.g star, mesh, ad-hoc, hybrid, etc.) herein we propose the use of a GSM modem that can serve up to 50 smart-traps, leading thus to the star topology. The modem should features external antennas that can be replaced with higher gain antennas should it be deemed necessary. The GSM modem is to be supplied with power by the solar panel – battery system that supplies the smart-trap.

Local data storage Use of local data storage (e.g. Secure Digital), in addition to the aforementioned operating system, for the temporary storage (and recycling) of collected data will allow to ensure the collected data are note lost in case of communication errors or errors of the server/processing center, at least up to the point of the next recycling. Accordingly, attention should be paid on the expected data volume per sampling of the sensors in addition to the frequency of sampling in order to select the required retention level.

Server/processing center The server/processing center is to be accessed through secure protocols (e.g. SSH) and synchronise data directories with the data directories of the smart-traps at explicitly defined time instances every day in order to deal with communication costs. To ensure the collected data are note lost, should a GSM modem failure occur and do not reach the server/processing center, data are also to be stored in the smart-trap's local storage.

Following the aforementioned specifications, a prototype (Figures 1a & 1b) of the Electronic Olive Fruit fly trap has been created at the Dept. of Informatics, Ionian University, Greece. A demonstration network of such Electronic Olive Fruit fly traps has been setup and placed in olive groves in NW Corfu, Greece, as shown in Figure 2, and is currently under rigorous testing in order to verify both its effectiveness and efficiency in collecting data. Pending its evaluation, the network will be significantly expanded and its data will be directly used to further expand *DIRT*'s dataset as well as its trained models.

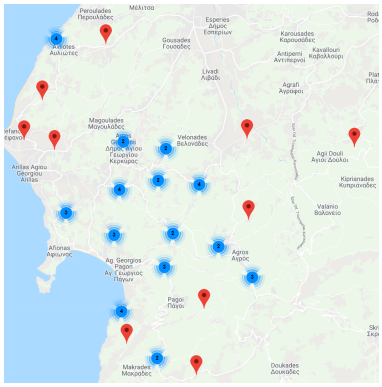
4. The Toolkit

This Section presents the toolkit's components: the creation processes of the dataset and a detailed analysis of its contents, the programming code samples provided and the Dacus identification API. Figure 3 presents the flowchart of *DIRT*'s creation and experimentation process: Data collection, splicing, filtering and annotation are described in Section 4.1, while data splitting, CNN training and

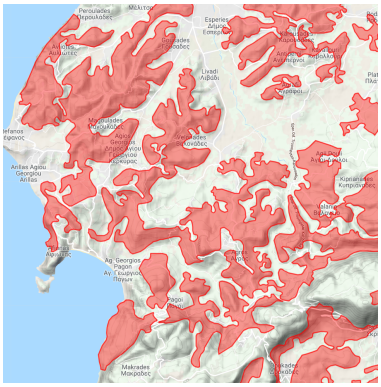
(a) Prototype of Electronic Olive Fruit fly trap, view 1.

(b) Prototype of Electronic Olive Fruit fly trap, view 2.

Figure 1. Electronic Olive Fruit fly trap.



(a) Demonstration network of Electronic Olive Fruit fly traps.



(b) Olive groves in NW Corfu, Greece.

Figure 2. Electronic Olive Fruit fly trap network.

evaluation are presented in Section 5. The complete *Dacus Image Recognition Toolkit* is available at http://lefkimi.ionio.gr/~avlon/dirt-dacus-image-recognition-toolkit/.

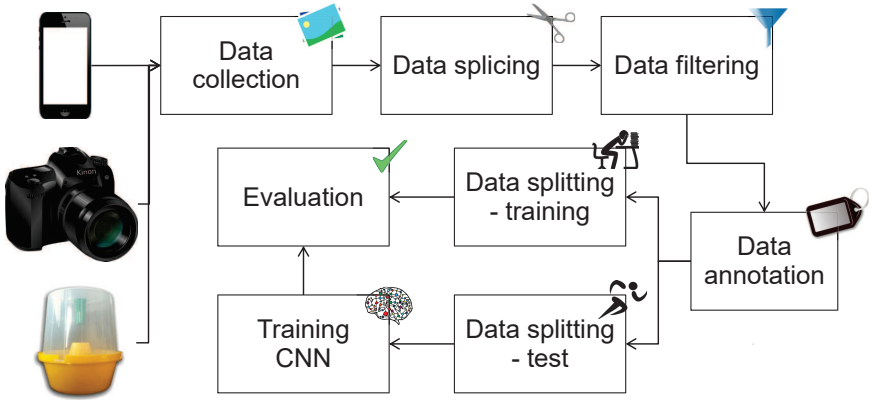


Figure 3. Flowchart of *DIRT*'s creation and experimentation.

4.1. Dataset

DIRT's data consists of images, the majority of which depict olive fruit fly captures in McPhail traps, collected from year 2015 to 2017 in various locations of Corfu, Greece. The images were acquired mainly via the e-Olive[3] smart-phone application, which allows users to submit reports about olive fruit fly captures, and upload images captured from the trap in the field. As the collection of images has been done using a variety of hardware (smart-phones & tablets running the e-Olive app, photo-cameras available at the field during trap inspection, etc.) the images of the dataset not standardised. Figure 4 shows the distribution of images as far as their dimensions are concerned.

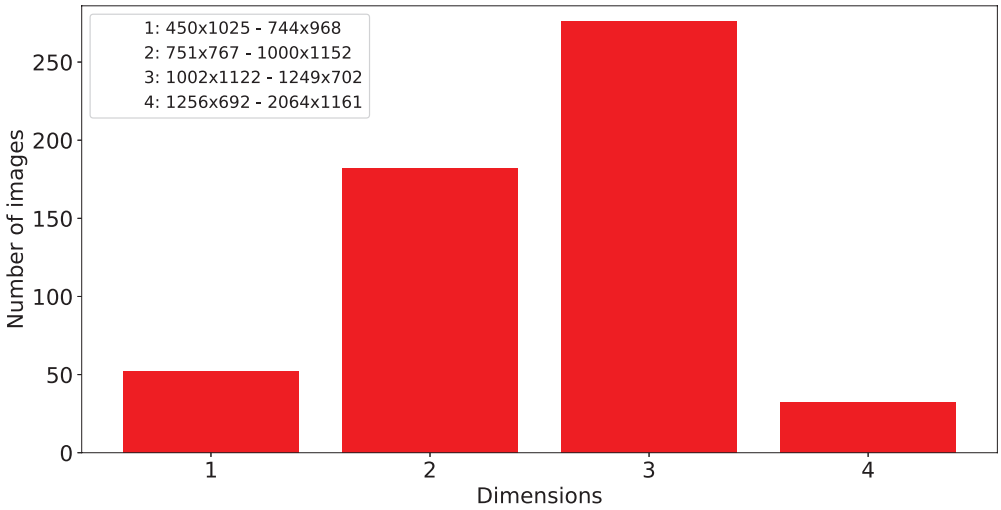


Figure 4. Distribution of *DIRT*'s images dimensions.

The original dataset consisted of 336 images, but after discarding images that either were too blurry in order to distinguish olive fruit flies from other insects or no olive fruit flies were present, the size of the dataset was reduced to 202 images. Moreover, due to the fact that training on this dataset for 50.000 steps did require superior than commonly available hardware, due to memory requirements associated with the large size of the available photographs, we decided to slice each image into four parts. Thus, after discarding image parts that didn't depict any olive fruit fly, the final dataset includes 542 images, a sample of which is shown in Figure 5. From those images 486 were randomly selected for training, while the remaining, also randomly selected, 56 images were used for evaluation in our experiments. Figure 6 presents the histogram of manually annotated olive fruit flies in images, before and after the slicing process of images, as aforementioned.

[3] https://play.google.com/store/apps/details?id=gr.cwa.eolive

Figure 5. Sample images from the dataset.

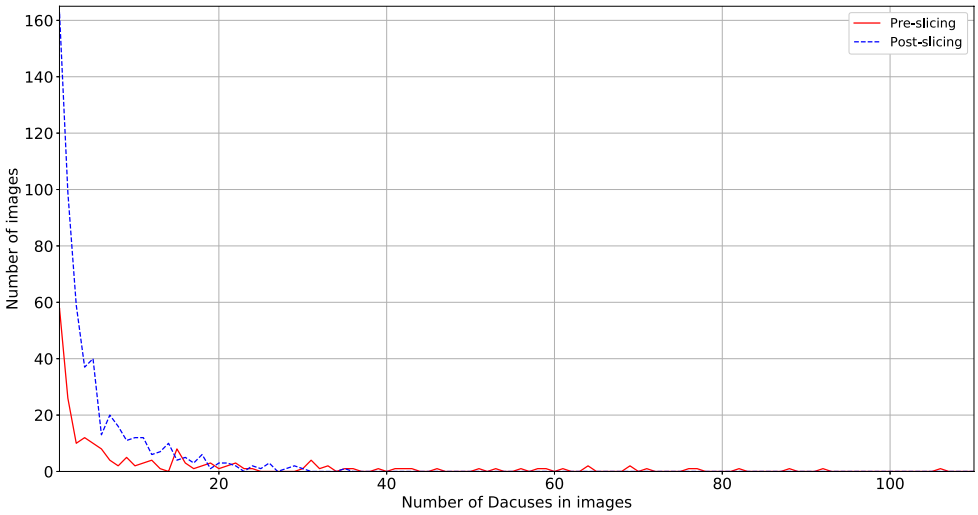


Figure 6. Distribution of images per counted Dacuses.

Label annotation upon our dataset was done with the LabelImg[4] annotation tool which allows the user to manually draw bounding boxes around objects in images and associate each box to a label, as shown in Figure 7. It is important to note, that for images depicting McPhail traps, we annotated mostly olive fruit flies that were floating in the liquid solution in addition to the distinguishable submerged olive fruit flies, since when submerged, Dacuses tend to create clusters making the task to identification very difficult. Bound to this restriction we annotated 2672 olive fruit flies.

4.2. Programming Code Samples

In order to stimulate further the research on pests' management as well as Dacuses image recognition, *DIRT* also includes a set of programming code samples that will allow interested researchers to fast-track their use of *DIRT*'s contents as well as guide them into some rudimentary experimentation.

[4] https://github.com/tzutalin/labelImg

Figure 7. Sample images from the dataset with label annotation.

The programming code samples are in Matlab and include the following generic/handling functions:

Load dataset Function *load_DIRT_dataset* parses the files of the dataset as provided in the archive of *DIRT* and produces a struct array with the associated filenames and folders of both images and (xml) annotations using local relative paths. No input arguments are required and the function returns the resulting struct array as well as saves it as a file titled *DIRT_dataset.mat* for future use. The definition of the images' and annotations' local paths, for content discovery, is clearly noted in lines 5 and 6 of the function.

Preview image with annotation Function *preview_img_DIRT_dataset* shows a random image of the dataset with overlayed annotation(s) of its Dacuses. The first argument is required and refers to the dataset as produced by *load_DIRT_dataset* function. The second argument is the array id of the image to preview (optional). If no second argument is provided or its value is *false*, then a random image of the dataset is shown. The third argument is a switch whether to show the image or not: if set to *true* (default) then it displays the image while if set to *false* then it does not display the image. The function returns the image including the overlay of the annotation(s).

Parse annotation Function *parse_DIRT_annotation* parses an annotation's xml file of DIRT dataset and returns its contents. The input argument is the char array or string containing the full path to the annotation file, including the annotation file's name and extension. The returned value is a struct containing (a) an aggregated array, with size $k \times 4$ where k is the number of Dacuses annotated in the image, while the four numbers per annotation describe the lower left x and y coordinates of the annotation's bounding box and the box's width and height, as well as (b) an array of the x_{min}, x_{max}, y_{min}, and y_{max} coordinates of each annotation's bounding box.

Moreover, the programming code samples in Matlab also include a simple but yet complete R-CNN [26] training and testing scenario with the following functions:

Driver Function *startFromHere* is a driver function for training and testing an R-CNN with the *DIRT* dataset. No input arguments are required and the function does not return any information as the last called function therein, *test_r_CNN* function, presents a graphical comparison of the identified by the R-CNN Dacus and the manually annotated ground-truth equivalent. The function slices the DIRT_dataset in order to train it with all but one, randomly selected from the available, image that will be subsequently used in the *test_r_CNN* function performing the test of the R-CNN.

Training the R-CNN Function *train_r_CNN* trains an R-CNN object detector for the identification of Dacuses in images. The input argument is the dataset in the format produced by *load_DIRT_dataset* function, while the output of the function is the trained R-CNN object detector.

The network utilised herein is not based on a pre-trained network but it is simplistically trained from scratch as a demo of the complete process. The resulting network is saved as a file titled *rcnn_DIRT_network.mat* for future use. The function also includes a switch that allows the training to be completely avoided and a pre-trained network loaded from the file titled *rcnn_DIRT_network.mat* to be used/returned instead.

Testing the R-CNN Function *test_r_CNN* presents a graphical comparison of the identified by the R-CNN Dacus, based on the trained network, and the manually annotated ground-truth equivalent, as prepared by the *preview_img_DIRT_dataset* function, side-by-side. The function's first input is the trained R-CNN network, as provided by the *train_r_CNN* function, the second argument is the dataset in the format produced by *load_DIRT_dataset* function, and the third argument is the test image's id from the array of *load_DIRT_dataset*, as selected by the *startFromHere* function for the testing procedure.

4.3. Dacus Identification API

The capability to identify Dacuses is of paramount importance to olive fruit cultivation and oil production, as described in Section 1. In order to further support this necessity, *DIRT* also includes a publicly available rest https API that replies to queries for Dacuses' identification in user provided images. In that manner, the complexity of Dacuses' identification in images is alleviated from users that only need to manage the interaction with the *DIRT*'s API, also addressed by simplistic web-based interface provided by *DIRT*.

It should be clearly noted that the proposed API is experimental and under permanent upgrade/development as new methods are implemented and more and more data are collected from our traps, annotated by experts, submitted to the system and the R-CNN is trained on. Accordingly, under no circumstances should the proposed API be used as a sole point of information for any related to Dacus infestation or olive-tree pest management decision making.

The API does not currently require any form of authentication and it features a single endpoint that, using the post method, allows the user to upload a single jpeg file-type image of maximum 2 Mebibytes. On success, the HTTP status code in the response header is 200 OK and the response body contains the JSON formatted file that describes the spatial coordinates of the bounding box of each of the Dacus(es) identified in the submitted image. On error, the header status code is an error code and the response body contains an error object detailing the error that occurred and possible method(s) to mitigate it.

As the process of identification of Dacuses in an image is quite heavy, both in terms of CPU and RAM of the server that provides this service, users are informed that our experimentation with moderate concurrent load showed that a possible lack of real-timeness in reply reaching at most 30 seconds, is a strong possibility. Accordingly, both the API and the web interface feature access limit: Only 1 access of these services is allowed per IP per 60 seconds, while on transgression the header status code is an error code and the response body contains an error object detailing the error that occurred and possible method(s) to mitigate it.

5. Experimental Evaluation

5.1. Experimental Setup

For our experiments we choose the Tensorflow Object Detection API[5] which is an open source framework built upon Tensorflow[6], an open source machine learning framework. The Tensorflow

[5] https://github.com/tensorflow/models/tree/master/research/object_detection
[6] https://www.tensorflow.org/

Object Detection API provides a number of pre-trained models[7] for the user to use in his experiments. The detection models provided were pre-trained on the COCO[8], KITI[9] and Open Image[10] datasets.

All training sessions run for 100000 steps, with a batch size of one. Thus, training run for 184 epochs. The hardware configuration where all experiments were conducted can be seen in Table 1.

Finally, the performance measurement used throughout the experimentation is the Mean Average Precision (mAP)[27], a common metric to compare model performance in object detection and it is the average maximum accuracy for different recall values. Essentially, mAP combines all individual (per test query) average precision into one number. mAP is formally defined in Equation 1, where the set of relevant documents for an information need $q_j \in Q$ is $d_1, \ldots, d_{m_j}$ and R_{jk} is the set of ranked retrieval results from the top result until document d_k. Among evaluation measures, mAP has been shown to "have especially good discrimination and stability" [28].

$$mAP(Q) = \frac{1}{|Q|} \sum_{j=1}^{|Q|} \frac{1}{m_j} \sum_{k=1}^{m_j} Precision(R_{jk}) \tag{1}$$

Table 1. Hardware configuration

CPU	Intel Core i7 920 @ 2.67GHz
GPU	NVIDIA TITAN Xp 11GB
RAM	16GB

5.2. Experimental Results

In order to verify the usefulness of the proposed dataset, a variety of experiments were conducted, pertaining mostly at the ability of automatically identifying Dacuses in the images using as ground-truth manually annotated spatial identification of Dacuses.

Firstly, training was performed on the following pre-trained models and the one with the best performance was selected in order to conduct further experiments.

- faster_rcnn_inception_v2_coco
- faster_rcnn_resnet50_coco
- faster_rcnn_resnet50_lowproposals_coco
- rfcn_resnet101_coco
- faster_rcnn_resnet101_coco
- faster_rcnn_inception_resnet_v2_atrous_coco

Table 2 presents the total loss and total time to complete training in the specified steps for each of the aforementioned detection models, after training on the *DIRT* dataset (and thus the removal of the "_coco" postfix).

Figure 8 presents the performance of all models trained on our dataset. The detection model that performed the worst, aside from the aforementioned model, was model 4 with a mAP of 57.42%. Detection models 3, 5 and 6 performed relatively well, ranging between 68% and 80%. However, models 1 and 2 outperformed all the rest with a mAP value of 91.52% and 90.03%, respectively. Although, the difference in performance is small (1.49%) between the two, we selected model 1 for the rest of our experiments since its training time is nearly four times faster than model 2 (see Table 2).

[7] https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/detection_model_zoo.md
[8] http://cocodataset.org/#home
[9] http://www.cvlibs.net/datasets/kitti/
[10] https://github.com/openimages/dataset

Table 2. Models' training details

Index	Model	Total Loss	Training Time
1	faster_rcnn_inception_v2	0.04886	5h 35m 51s
2	faster_rcnn_inception_resnet_v2_atrous	0.1883	23h 21m 39s
3	faster_rcnn_resnet50	0.1029	8h 35m 53s
4	faster_rcnn_resnet50_low_proposals	0.5773	8h 29m 54s
5	faster_rcnn_resnet101	0.1608	12h 57m 49s
6	rfcn_resnet101	0.1349	13h 45m 51s

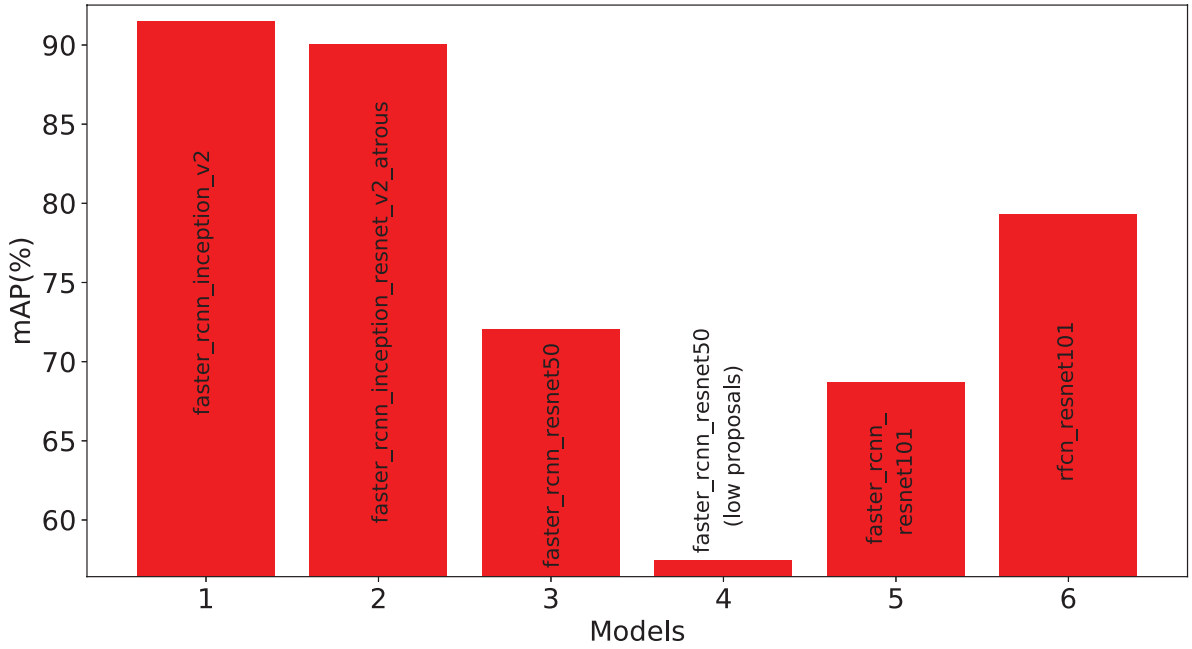


Figure 8. Detection models performance comparison after training on the olive fruit fly image dataset.

After selecting the best performing model, we investigated how the performance is affected by images' detail conducting thus training on resized images from the initial dataset. In detail, we trained our model on 10%, of the original size, to 100% with a 10% increase step. Furthermore, the same experiment was repeated, only this time all images were converted to gray-scale in order to verify the effect of color in the results obtained.

Figure 9 shows how the performance of detection model 1 changes when trained upon different size scales of the images from the original olive fruit fly dataset. For 10% of the original size the model performs rather poorly, with a mAP of 63.15%, in regard to the subsequent resized datasets. For 20% and greater, detection precision ranges between 85% and 91%. Interestingly enough, values of mAP for scaling between 50% and 100% are nearly stable with small fluctuations, while the difference between the full sized images and the images resized in half is 1.13%.

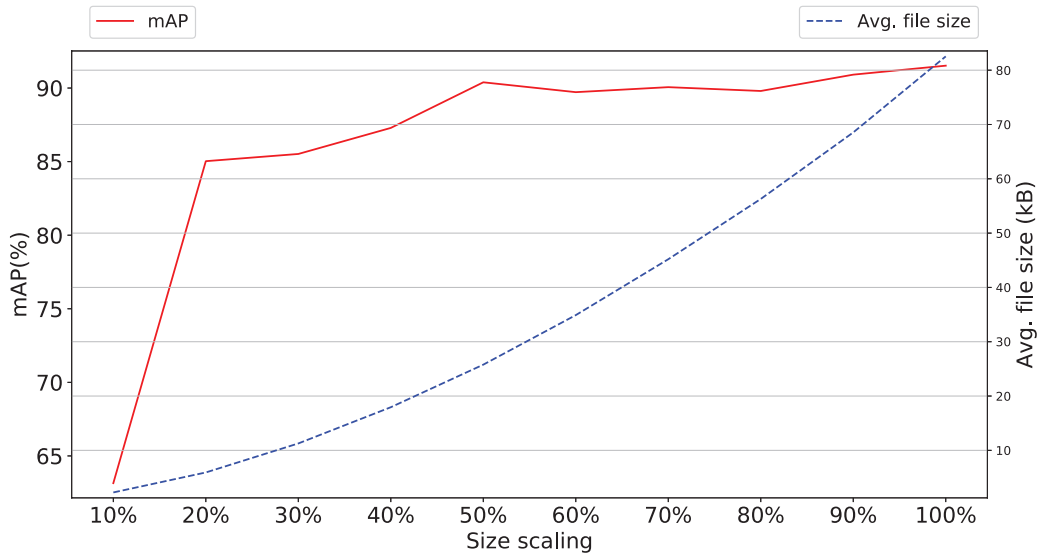


Figure 9. Detection model 1 performance for different size scales of the images in the dataset.

Similarly, Figure 10 presents the change in performance of detection for model 1 for different size scales of the images, but converted to gray-scale. Once more, for 10% resized images the detection precision is low (60.6%) compared to the rest of the sizes. While, between 20% and 100% scaling, mAP ranges approximately between 81% and 91%. Finally, after 60% scaling the detection precision is quite stable with small fluctuations.

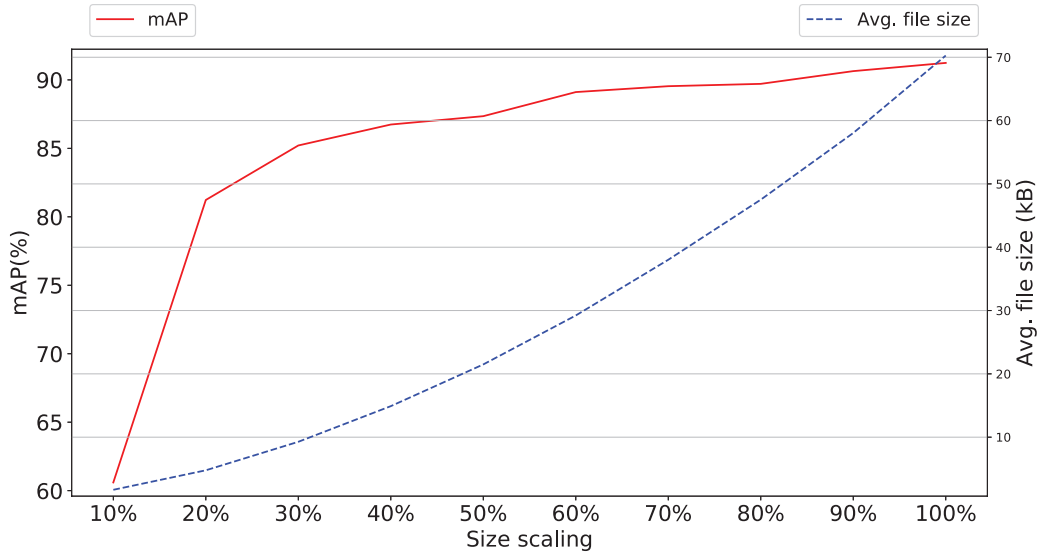


Figure 10. Detection model 1 performance for different size scales of the images in the dataset, after conversion to gray-scale.

In Figure 11, the detection precision between the gray-scale and color (RGB) datasets from the previous two experiments are compared. Both, have a similar trend in increasing precision as the original size of the images is approached. However, the average difference in detection precision between 10% and 50% scaling is 2.05% in favour of the RGB dataset. On the other hand, from 60% and onward the precision between the RGB and gray-scale datasets reaches about the same with an average difference of 0.354% in favour of the RGB dataset.

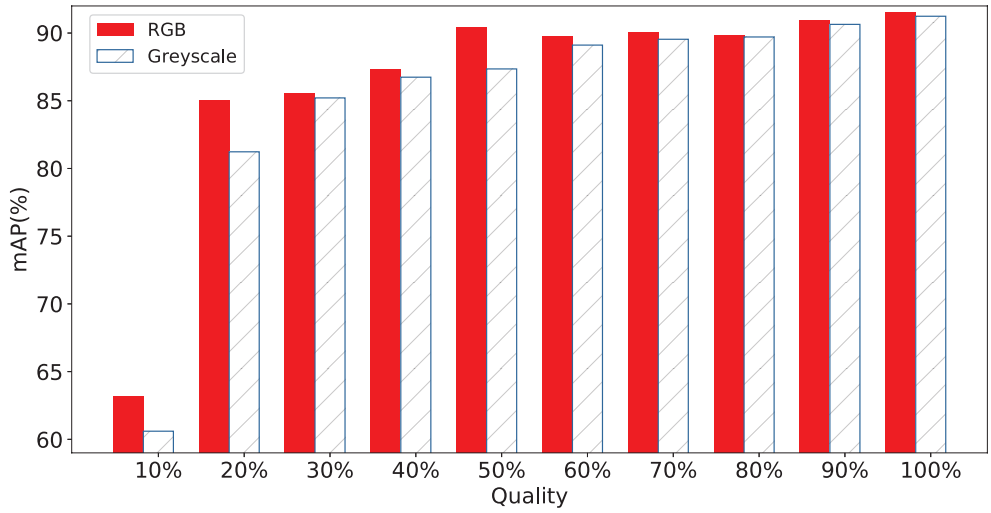


Figure 11. Performance comparison between RGB and gray-scale images for different size scales of the images.

In the next experiment, we investigated how the performance of the selected model is affected in relation to the total number of olive fruit flies in an image. Therefore, we created four new sub-datasets (see Table 3) based on the initial dataset, for both the training and testing sets. Specifically, we tested the performance on images that contained 3, 7, 10 and 14 olive fruit flies in order to verify the ability of the proposed model to retain high performance irrespectively of the density of Dacuses to be identified in an image. The values of fruit flies were selected based on the availability of

Table 3. Trap counts based datasets

Olive fruit fly counts	No. of training images	No. of test images
3	483	59
7	522	20
10	530	12
14	532	10

Finally, the performance of the proposed methodology was experimented with in relation to the number of ground-truth (i.e. manually counted) Dacuses as well as the cumulative number of pests in each image. Figure 12 presents the detection precision for four groups of images, where each group contains solely images with the same number of ground-truth Dacuses. In all tested numbers of ground-truth Dacuses (3, 7, 10, 14) the detection precision doesn't fall below 89%. The limited variation that exists (detecting three olive fruit flies produces the highest mAP value of 96.1% with the lowest detection performance of 89.5% for ten Dacuses) is attributed to the size of the each group's available image content in terms of pests to be examined.

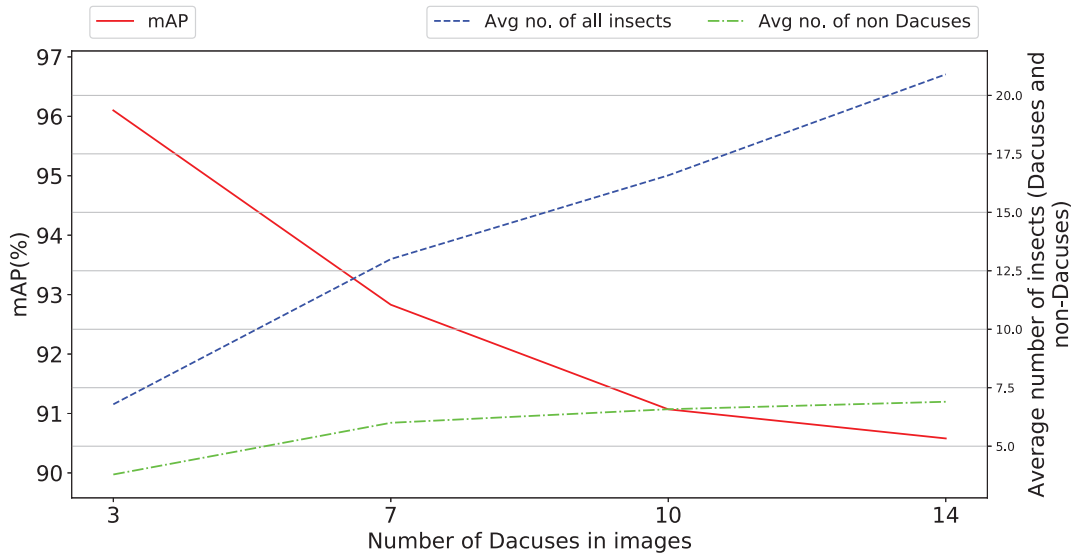


Figure 12. Detection precision comparison for trap count based datasets

5.3. Results' Discussion

Based on the experimental results of Section 5.2, there are three important takeaways:

Size of images The experimental results on the size of the images taken from the smart-trap, as shown in Figure 9, indicate that a high detail provided in photos with increased pixel availability is indeed affecting the performance of the proposed methodology, but the ratio of performance's increase falls sharply after discarding 80% of the original information while the difference between discarding 50%-10% is approx 1% and thus almost negligible, for some applications. Accordingly, the widespread availability of high-pixel cameras, although has been show to increase the effectiveness of the identification of Dacuses, develops to be a trade-off between marginally higher performance and increased volume of data that potentially have to be stored locally to limited persistent storage or transported over either meter connections (e.g. GSM modem) or in ad-hoc networks affecting thus the network's load.

Color information of images Similarly to the previous argument, the color information of the images obtained from the smart-trap, as shown in Figures 10 and 11, is shown to be of secondary importance as, in both scaled gray-scale images as well as in full scale images after conversion to gray-scale, the effect of RGB color on performance is almost negligible. This supports further the previous argument of the diminished role of high-pixel images even when these are gray-scale and thus require approx. one third of the RGB equivalent images. Thus, the selection of RGB or gray-scale cameras reverts to the aforementioned trade-off between minor increase in identification performance versus volume of data.

Number of Dacuses in images The ability of the proposed method to retain high performance irrespectively of the number of collected Dacuses in the trap is very important in order to address a variety of scenarios of traps' designs, geo- & weather- characteristics of the olive grove, varieties of olives etc. The variation shown in Figure 12 is approx. 6.5% and thus requires further examination, despite the fact that for the specific traps used, all values of number of Dacuses equal or greater than seven are similarly considered to be a significant infestation indication requiring action. All in all, a general trend is evident for the parameters tested: the number of insects (and by extension Dacuses) inversely affects the detection precision, an effect attributed to the proportionally higher number of insects in the trap when increased number of Dacuses are measured.

6. Conclusion

This work presents the "Dacus Image Recognition Toolkit" (*DIRT*), a collection of publicly available data, programming code samples and web-services focused at supporting research aiming at the management the olive fruit-fly as well as extensive experimentation on the capability of the proposed dataset in identifying Dacuses using Deep Learning methods. The dataset includes 542 images depicting McPhail traps' contents with experts' manually annotated spatial identification of Dacuses in each image. To further support research on Dacuses identification, the toolkit includes programming code samples in matlab that allow fast initial experimentation on the dataset or any other Dacus image set while in addition, a public rest https API and web-interface has been developed that reply to queries for Dacus identification in user provided images.

It should be noted, that we intend to maintain and enhance the Toolkit and, accordingly, *DIRT*'s dataset enlargement and further training of the web-service is assumed by our team as a perpetual process. Up-to-date information on *DIRT*'s dataset volume and assorted programming code samples as well as further training of the models used in the Dacus Identification API, are available at the Toolkit's website.

Extensive experimentation on the use of deep learning for Dacus identification on the dataset's contents, presented herein, include a variety of experiments pertaining mostly at the ability of automatically identify Dacuses in the images using as ground-truth manually annotated spatial identification of Dacuses. Results indicated that the performance accuracy (mAP) of 91.52% is possible based on publicly available pre-trained and further trained on Dacuses models. Moreover, the experimental results indicated a trade-off in both images' pixel size & color information (both adding to images' detail, file size and complexity of approaches) and mAP performance that can be selectively used to better tackle the needs of each usage scenario.

There exist a number of research directions that *DIRT* could be further ameliorated in future versions. The most obvious one pertains to the enlargement of the toolkit's dataset in both volume of images as well as in variability of traps' type, allowing thus better training and accordingly recognition performance. Moreover, the use McPhail type traps with liquid pheromones to attract pests have indeed lead to some degree to "pest soup images" where a number of the Dacuses to be identified are submerged in the liquid and thus indistinguishable even by *in situ* experts much less by remote observers through images. As the aim of this work is to provide a methodology to identify Dacuses through images, future work on the toolkit could include a more detailed ground-truth identification of clustered and submerged Dacuses for the scenarios supporting scarce image sampling of the trap, wherein such situations may arise. Moreover, the identification of the genre of the Dacuses collected at the traps is of high importance and requires further exploration. As far as the API is concerned, a more time-responsive and with less access restrictions is certainly warranted for wider use and experimentation, both of which will be possible with special hardware and more advanced indexing methods. Moreover, an extension of the API to feature evaluations of, both of it's submitters and experts, users will certainly provide for, at least, an ever expanding dataset and potentially increased performance leading to better user experience. Finally, this research can be further ameliorated by additional use of methods employed in generic image recognition such as combination of deep and handcrafted image features, local learning frameworks to predict images' class as well as use of insect-specific image recognition methods focusing on insects' wing, body and eye features.

Funding: The financial support of the European Union and Greece (Partnership Agreement for the Development Framework 2014-2020) under the Regional Operational Programme Ionian Islands 2014-2020, for the project "Olive Observer" is gratefully acknowledged.

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

1. Altieri, M.A.; Farrell, J.G.; Hecht, S.B.; Liebman, M.; Magdoff, F.; Murphy, B.; Norgaard, R.B.; Sikor, T.O. Toward sustainable agriculture. In *Agroecology*; CRC Press, 2018; pp. 367–379. doi:10.1201/9780429495465.
2. Van Grinsven, H.J.; Erisman, J.W.; de Vries, W.; Westhoek, H. Potential of extensification of European agriculture for a more sustainable food system, focusing on nitrogen. *Environmental Research Letters* **2015**, *10*, 025002. doi:10.1088/1748-9326/10/2/025002.
3. King, A. The future of agriculture. *Nature* **2017**, *544*, S21–S23. doi:10.1038/544S21a.
4. Donatelli, M.; Magarey, R.D.; Bregaglio, S.; Willocquet, L.; Whish, J.P.; Savary, S. Modelling the impacts of pests and diseases on agricultural systems. *Agricultural systems* **2017**, *155*, 213–224. doi:10.1016/j.agsy.2017.01.019.
5. Bogue, R. Sensors key to advances in precision agriculture. *Sensor Review* **2017**, *37*, 1–6. doi:10.1108/SR-10-2016-0215.
6. Ojha, T.; Misra, S.; Raghuwanshi, N.S. Wireless sensor networks for agriculture: The state-of-the-art in practice and future challenges. *Computers and Electronics in Agriculture* **2015**, *118*, 66–84. doi:10.1016/j.compag.2015.08.011.
7. Wolfert, S.; Ge, L.; Verdouw, C.; Bogaardt, M.J. Big data in smart farming–a review. *Agricultural Systems* **2017**, *153*, 69–80. doi:10.1016/j.agsy.2017.01.023.
8. Lindblom, J.; Lundström, C.; Ljung, M.; Jonsson, A. Promoting sustainable intensification in precision agriculture: review of decision support systems development and strategies. *Precision Agriculture* **2017**, *18*, 309–331. doi:10.1007/s11119-016-9491-4.
9. Eurostat. Agri-environmental indicator - cropping patterns, Data from March 2017, 2017.
10. Fogher, C.; Busconi, M.; Sebastiani, L.; Bracci, T. Chapter 2 - Olive Genomics. In *Olives and Olive Oil in Health and Disease Prevention*; Preedy, V.R.; Watson, R.R., Eds.; Academic Press: San Diego, 2010; pp. 17 – 24. doi:10.1016/B978-0-12-374420-3.00002-4.
11. Haniotakis, G.E. Olive pest control: present status and prospects. *IOBC wprs Bulletin* **2005**, *28*.
12. Shaked, B.; Amore, A.; Ioannou, C.; Valdés, F.; Alorda, B.; Papanastasiou, S.; Goldshtein, E.; Shenderey, C.; Leza, M.; Pontikakos, C.; Perdikis, D.; Tsiligiridis, T.; Tabilio, M.R.; Sciarretta, A.; Barceló, C.; Athanassiou, C.; Miranda, M.A.; Alchanatis, V.; Papadopoulos, N.; Nestel, D. Electronic traps for detection and population monitoring of adult fruit flies (Diptera: Tephritidae). *Journal of Applied Entomology* **2017**, *142*, 43–51. doi:10.1111/jen.12422.
13. Doitsidis, L.; Fouskitakis, G.N.; Varikou, K.N.; Rigakis, I.I.; Chatzichristofis, S.A.; Papafilippaki, A.K.; Birouraki, A.E. Remote monitoring of the Bactrocera oleae (Gmelin) (Diptera: Tephritidae) population using an automated McPhail trap. *Computers and Electronics in Agriculture* **2017**, *137*, 69 – 78. doi:10.1016/j.compag.2017.03.014.
14. Alorda, B.; Valdes, F.; Mas, B.; Leza, M.; Almenar, L.; Feliu, J.; Ruiz, M.; Miranda, M. Design of an energy efficient and low cost trap for Olive fly monitoring using a ZigBee based Wireless Sensor Network. European Conference Precision Agriculture, 2015.
15. Wang, J.n.; Chen, X.l.; Hou, X.w.; Zhou, L.b.; Zhu, C.D.; Ji, L.q. Construction, implementation and testing of an image identification system using computer vision methods for fruit flies with economic importance (Diptera: Tephritidae). *Pest Management Science* **2016**, *73*, 1511–1528. doi:10.1002/ps.4487.
16. Philimis, P.; Psimolophitis, E.; Hadjiyiannis, S.; Giusti, A.; Perello, J.; Serrat, A.; Avila, P. A centralised remote data collection system using automated traps for managing and controlling the population of the Mediterranean (Ceratitis capitata) and olive (Dacus oleae) fruit flies. International Conference on Remote Sensing and Geoinformation of the Environment, 2013, Vol. 8795, pp. 8795 – 8795 – 8. doi:10.1117/12.2028244.
17. Tirelli, P.; Borghese, N.A.; Pedersini, F.; Galassi, G.; Oberti, R. Automatic monitoring of pest insects traps by Zigbee-based wireless networking of image sensors. IEEE International Instrumentation and Measurement Technology Conference, 2011, pp. 1–5. doi:10.1109/IMTC.2011.5944204.
18. Uhlir, P. The Value of Open Data Sharing. *Group on Earth Observations, CODATA* **2015**.
19. Mayernik, M.S.; Phillips, J.; Nienhouse, E. Linking publications and data: Challenges, trends, and opportunities. *D-Lib Magazine* **2016**, *22*, 11. doi:10.1045/may2016-mayernik.

20. Potamitis, I.; Eliopoulos, P.; Rigakis, I. Automated remote insect surveillance at a global scale and the internet of things. *Robotics* **2017**, *6*, 19. doi:10.3390/robotics6030019.
21. Sun, C.; Flemons, P.; Gao, Y.; Wang, D.; Fisher, N.; La Salle, J. Automated Image Analysis on Insect Soups. Digital Image Computing: Techniques and Applications (DICTA), 2016 International Conference on. IEEE, 2016, pp. 1–6. doi:10.1109/DICTA.2016.7797010.
22. Potamitis, I.; Rigakis, I.; Tatlas, N.A. Automated surveillance of fruit flies. *Sensors* **2017**, *17*, 110. doi:10.3390/s17010110.
23. Shelly, T.; Epsky, N.; Jang, E.B.; Reyes-Flores, J.; Vargas, R. *Trapping and the detection, control, and regulation of Tephritid fruit flies: lures, area-wide programs, and trade implications*; Springer, 2014. doi:10.1007/978-94-017-9193-9.
24. Ding, W.; Taylor, G. Automatic moth detection from trap images for pest management. *Computers and Electronics in Agriculture* **2016**, *123*, 17–28. doi:10.1016/j.compag.2016.02.003.
25. Rassati, D.; Faccoli, M.; Chinellato, F.; Hardwick, S.; Suckling, D.; Battisti, A. Web-based automatic traps for early detection of alien wood-boring beetles. *Entomologia Experimentalis et Applicata* **2016**, *160*, 91–95. doi:10.1111/eea.12453.
26. 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. doi:10.1109/TPAMI.2016.2577031.
27. Gros, D.; Habermann, T.; Kirstein, G.; Meschede, C.; Ruhrberg, S.D.; Schmidt, A.; Siebenlist, T. Anaphora Resolution: Analysing the Impact on Mean Average Precision and Detecting Limitations of Automated Approaches. *International Journal of Information Retrieval Research (IJIRR)* **2018**, *8*, 33–45. doi:10.4018/IJIRR.2018070103.
28. Manning, C.D.; Raghavan, P.; Schütze, H. *Introduction to information retrieval*; Cambridge University Press, 2008.