Computers and Electronics in Agriculture 226 (2024) 109365

Contents lists available at ScienceDirect

Computers and Electronics in Agriculture

journal homepage: www.elsevier.com/locate/compag

Original papers

The hawk eye scan: *Halyomorpha halys* detection relying on aerial tele photos and neural networks☆

Lorenzo Palazzetti [b,a,*], Aravind Krishnaswamy Rangarajan [d], Alexandru Dinca [c], Bas Boom [d], Dan Popescu [c], Peter Offermans [d], Cristina M. Pinotti [a]

[a] Department of Computer Science and Mathematics, University of Perugia, Italy
[b] Department of Computer Science and Mathematics, University of Florence, Italy
[c] Department of Automatic Control and Computers, University Polytecnica of Bucharest, Romania
[d] One Planet Research Center, Imec, The Netherlands

ARTICLE INFO

Keywords:
Halyomorpha halys detection
Drones
Computer vision algorithm
Technological transfer

ABSTRACT

This paper faces the challenge of monitoring the Brown Marmorated Stink Bug (H.halys) (*Halyomorpha halys*) within orchards, utilizing drones, and computer vision. H.halys is an invasive species originating from East Asia, that is extremely polyphagous and poses a significant threat to various crops. Our first contribution is a drone navigation protocol, which ensures risk-free drone flights in cluttered orchard environments, preserving image quality and avoiding obstacles. We then create a pioneering H.halys dataset consisting of aerial telephotos captured in the field autonomously by the drone. The dataset allows the development and evaluation for the first time of multiple ML models for H.halys detection in the field. We trained YOLOv5, YOLOv8, RetinaNet, and Faster-RCNN models using different learning methodologies, exploiting different percentages of images without the bug, and using different slicing procedures for the images. The Medium YOLOv5 model trained with all images containing a bug detects the largest number of H.halys on the testing set and overall performs the best, while RetinaNet and Faster-RCNN provide the best trade-off between precision and recall. Models vary in their ability to handle occluded H.halys and bug-free images, which are common since the presence of the bug cannot be predicted before capturing a photo. These results show promising potential for automating H.halys monitoring, despite the image complexity and the early dataset stage. Our work marks a significant step towards enhancing smart agriculture practices due to the simplicity of the data acquisition process and the off-the-shelf hardware selection.

1. Introduction

The landscape of agriculture is experiencing a revolutionary transformation propelled by the integration of cutting-edge technologies such as drones and artificial intelligence (Jha et al., 2019). Drones, once primarily associated with recreational use, have swiftly evolved to play a pivotal role in modern agriculture, offering new data with great insights to optimize crop yield, reduce resource consumption, and enhance overall operational efficiency (Rejeb et al., 2022). Notably, the introduction of Machine Learning (ML) algorithms, such as YOLO (You Only Look Once) (Jocher et al., 2022) and other computer vision algorithms has significantly bolstered the use of drones for a plethora of civil applications. These algorithms empower drones to process large datasets in real-time, enabling them to make critical decisions autonomously (Lin et al., 2020; Ren et al., 2017).

This technological convergence marks a breakthrough in orchard monitoring, particularly for insect detection. Current pest monitoring methods are known to be time-consuming and labor-intensive, refraining growers from repeating monitoring and letting growers prefer to make less informed decisions. Indeed, monitoring practices predominantly rely on traps, which exhibit reliability issues and can exacerbate overall damage due to the use of aggregation pheromones. Alternatively, labor-intensive methods, like *frappage*, are used for active

☆ This work was supported in part by the "GNCS – INdAM", by "HALY.ID" project funded by the European Union's Horizon 2020 under grant agreement ICT-AGRI-FOOD no. 862665, no. 862671, by MIPAAF, Italy, by RESIDUAL, and by RB_DMI_2019. L. Palazzetti and C. M. Pinotti are member of the Gruppo Nazionale Calcolo Scientifico-Istituto Nazionale di Alta Matematica (GNCS-INdAM).

* Corresponding author at: Department of Computer Science and Mathematics, University of Perugia, Italy.
E-mail addresses: lorenzo.palazzetti@unifi.it, lorenzo.palazzetti@collaboratori.unipg.it (L. Palazzetti), Aravind.KrishnaswamyRangarajan@imec.nl (A.K. Rangarajan), alex.mdinca@yahoo.com (A. Dinca), Bas.Boom@imec.nl (B. Boom), dan.popescu@upb.ro (D. Popescu), peter.offermans@imec.nl (P. Offermans), cristina.pinotti@unipg.it (C.M. Pinotti).

https://doi.org/10.1016/j.compag.2024.109365
Received 30 March 2024; Received in revised form 4 July 2024; Accepted 17 August 2024
Available online 26 August 2024

monitoring. The adoption of off-the-shelf drones coupled with machine vision algorithms is a key enabling technology for a more informed decision process. Specifically, this paper evaluates the feasibility of monitoring the Brown Marmorated Stink Bug (H.halys) (*Halyomorpha halys*) using computer vision algorithms and acquiring photos with an aerial vehicle.

H.halys is an invasive stink bug that poses a significant threat to various crops, such as, fruit trees, including pears, peaches, and apricots. The economic impact of H.halys has been particularly severe in Italy, especially in northern regions, where millions of euros in losses have been recorded in the primary fruit production sector. H.halys, in addition to cause severe economic damages, led to the substantial interruption of the Integrated Pest Management (IPM) in favor of a strong usage of pesticides that cracked the confidence of the consumers. Therefore, H.halys resulted in both direct crop losses and broader social impacts. Current monitoring systems of H.halys are notably time-consuming and require full human engagement, making them inefficient and labor-intensive. Additionally, the traps currently in use for H.halys have proven to be unreliable because they require the use of aggregation pheromones which attract many specimens and locally increase the damages. For these reasons, such traps are also no effective for developing statistical models to estimate H.halys populations accurately. Our research addresses these challenges by introducing an autonomous monitoring system utilizing drones equipped with advanced Machine Learning (ML) algorithms, specifically refined for aerial data. In this way, the presence of H.halys without using pheromones can be measured.

First goal of this paper is to define a robust image acquisition protocol. Therefore, we will consider only aerial images because challenges in drone maneuvers within orchards, attributed to inaccurate GPS tracing, irregular tree shapes, and the sensitivity of collision avoidance systems were reported in Betti Sorbelli et al. (2023), Betti Sorbelli et al. (2023) and Almstedt et al. (2023). To the best of our knowledge, we proposed the first monitoring system protocol to ensure drone risk-free flights in cluttered environments while guaranteeing satisfactory accuracy for the H.halys detection through aerial imaging. The protocol involves designing a specific mission path above the orchard, eliminating obstacles, and improving image quality based on optical laws. To assess the quality of the acquired images, a thorough analysis of features such as blurriness and H.halys size is carried out, emphasizing their utility for ML tasks. Subsequently, various computer vision detectors, precisely YOLOv5, YOLOv8, RetinaNet, and Faster-RCNN, are trained employing two different image slicing methods, and training sets with different controlled percentages of images without bugs. We use slicing methods to divide the images into patches. We relied on this approach because the images are very cluttered and, also for the expert human eye, it is difficult to detect the H.halys when a full image is considered. The slicing approach significantly improves the precision metric of our models concerning the results obtained in Sorbelli et al. (2023). Dividing the image in patches, many *negative patches*, i.e., patches without bugs, are created. Namely, considering the size of the bug concerning the field-of-view of an image, it is highly expected that there would be a large number of patches that do not contain bugs. To learn to deal with the bug absence, we train our models using different percentages of negative patches.

This work builds upon our earlier conference paper (Sorbelli et al., 2023). Differently from Sorbelli et al. (2023) where a YOLOv5 detector was used trained with full images, in this work, new training strategies are proposed which significantly enhance the detection performance of YOLOv5 algorithms. We also extend our investigation to three other state-of-art computer vision detectors, i.e., RetinaNet, Faster-RCNN, and YOLOv8. It should be noted that, from now on, when we refer to H.halys, we will refer exclusively to H.halys adults.

Contributions and paper organization

Our contributions are:

- Creation of first dataset for scouting the invasive H.halys, relying on autonomously captured aerial telephotos; The suitability for ML purposes is assessed under various perspectives, e.g., H.halys size distribution, or blurriness and brightness levels within the images[1];
- Training of detectors directly on drone captured images relying on state-of-art models, i.e, YOLOv5, YOLOv8, RetinaNet, and Faster-RCNN, able to scout the H.halys on aerial images; Those models are trained employing novel learning techniques with different percentages of negative patches in order to strengthen the ability of the models to deal with the absence of the bug in patches; Simultaneously, two different slicing procedures are employed to avoid loss of information during the training iterations;
- A novel evaluation of the models' robustness against occluded H.halys by directing attention to the bugs that are cut during the slicing process, and finally
- Evaluate the capabilities of the detectors in detect H.halys in full images, proposing novel metrics to gain awareness on never considered TN.

The remainder of the paper is organized as follows: Section 2 presents relevant related work, Sections 3 and 4 detail the creation and composition of the dataset, Section 5 covers the training and validation of detectors, and Section 6 analyzes the conducted ML experiments, including the error rate respect to the negative patches, the occluded H.halys, and the metrics for the patches grouped by images. Finally, conclusions are drawn in Section 7.

2. Related work

Recent years have witnessed significant advancements in drone technology, particularly in the context of agriculture, where they have transformed farming practices, yielding benefits such as cost savings, operational efficiency, and increased profitability. A review conducted recently in Rejeb et al. (2022) on agricultural drones brings to our attention key research trends in areas such as remote sensing, precision agriculture, deep learning, machine learning (ML), and the Internet of Things.

Drones' applications in agriculture are diverse, ranging from creating vegetation indicators in conjunction with satellites (Guo et al., 2019; Murugan et al., 2016; Moreno et al., 2018) to monitoring wildlife and livestock (Tansuriyavong et al., 2018; Mitra et al., 2021). Notably, cost-effective solutions like AgriQ (de Oca and Flores, 2021) have been proposed, featuring a drone, a multispectral imaging system, vision algorithms, autonomous trajectory planning, and open-source software. AgriQ demonstrates superior cost-effectiveness and performance compared to commercial systems, making it a valuable asset for precision agriculture.

In the realm of ML, there is a growing emphasis on utilizing techniques for monitoring insect species. Traditional methods such as support vector machines, adaptive boosting, artificial neural networks (Xie et al., 2015; Espinoza et al., 2016; Valan et al., 2019; Wen and Guyer, 2012), and deep learning techniques based on convolutional neural networks (CNNs) (Chen et al., 2020; He et al., 2020; Li et al., 2021) have shown promising results in insect monitoring. For example, an innovative approach for early detection and continuous monitoring of whiteflies (*Bemisia tabaci*) and thrips (*Frankliniella occidentalis*) in greenhouses employs an image-processing algorithm and artificial neural networks (Espinoza et al., 2016). This approach, with its potential

[1] The data set will be made available at the end of the project to the scientific community under a Open Data Commons.

to enhance Integrated Pest Management (IPM) strategies and reduce chemical usage, underscores the impact of ML in agriculture.

Our paper advances this technological integration by combining drones with ML techniques to detect one of the most damaging insects (H.halys), which causes extensive crop destruction and significant financial losses, providing solution with high detection reliability. Limited studies have demonstrated the direct detection of insects through aerial surveys conducted with drones in open fields. One such study (Park et al., 2021) explores the effectiveness of drones in detecting the immobile stage of *Monema flavescens*, revealing that aerial surveys outperform ground surveys in identifying butterfly cocoons. The use of specialized cameras on drones, coupled with GPS technology, proves instrumental in achieving efficient positioning and accurate insect detection.

Focusing on H.halys scouting, diverse imaging technologies have been explored in literature. The authors in Ferrari et al. (2023) evaluate the potential of Short Wavelength Infra-Red (SWIR), combining chemometric analysis and CNNs, to accurately detect H.halys on various backgrounds mimicking field conditions. While promising, there are no yet devices with a suitable resolution usable in the field.

Focusing on H.halys and RGB-based detection methods, several studies in the last 5 years contribute valuable insights into their feasibility and efficiency. The authors in Trufelea et al. (2021) employ deep learning models, specifically CNNs, to classify pests belonging to the Pentatomidae family, including H.halys. The authors in Ichim et al. (2022) investigate the identification of H.halys using four different CNNs, demonstrating good performance, particularly on data similar to public datasets. A recent development involves the utilization of the "You Only Look Once" (YOLO) model (Jocher et al., 2022) for detecting harmful insects in ecological orchards (Sava et al., 2022). The YOLO model, evaluated against region-based CNNs (R-CNN), exhibits promising results, with the YOLO-m model achieving metrics above 95% for precision, recall, and mean average precision on lab images. Though the achievements are promising, the previously listed studies relied solely on high-resolution images with a macro perspective on the H.halys specimen relevant exclusively in laboratory conditions. Differently, we delve into the development of detectors able to recognize H.halys directly on images acquired in the field, and at the same time ensuring comparable, or even better performance.

Furthermore, in Almstedt et al. (2023) the authors provided some preliminary results on detecting directly on in-field acquired images. A synthetic dataset relying on H.halys silhouette, and several backgrounds is created for this scope. Then, the showcased performance of the trained model tested on an in-field dataset highlighted some criticality with respect to foliage defections and blurriness. The latter issues where investigate in Betti Sorbelli et al. (2023), and further extended in Betti Sorbelli et al. (2023). Indeed, the authors provided a careful evaluation of both image blurriness and brightness impact with respect to a H.halys detection task. Some models based on the YOLO framework are trained and tested on in-field images taken in First Person View (FPV), obtaining promising performance. However, the FPV acquisition protocol proposed is extremely subordinated to the orchard sizes, and especially to the aisle width. Despite the authors claimed that their protocol is autonomous, in presence of cluttered crops, like ours, manual guidance of the drone is mandatory due the limited space for maneuvers.

In this paper, we overcome the FPV issues by developing a completely risk-free acquisition protocol relying on telephotos collected by an autonomous drone, even for extremely cluttered crops. In other words, we are proposing the first in-field autonomous monitoring system able to detect the H.halys with high reliability, potentially without the surveillance of an operator. In addition, we thoroughly investigate the impact of negative patches during the training of our models in an active learning fashion, demonstrating the reliability of our approach employing several metrics.

In the following section, we explain how we collect the RGB images.

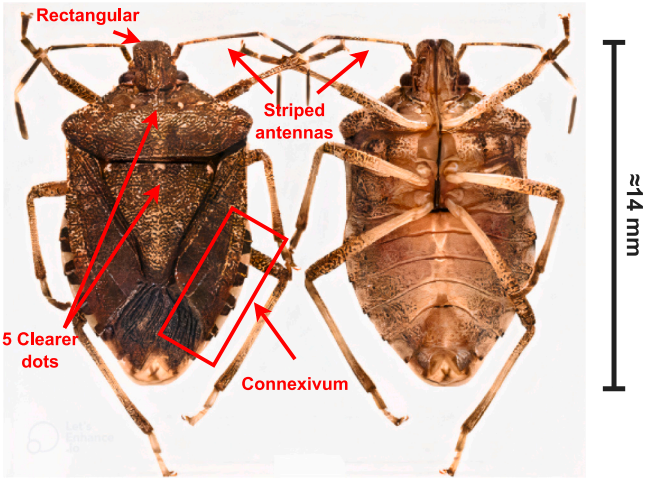

Fig. 1. H.halys adult and its distinctive features.

3. Data acquisition process

In this work, the hardware selection was constrained by the request to propose a solution fully reproducible, without engineering expertise in hardware design. Our choice was among DJI Phantom 4, DJI Mini 2, both with an embedded camera and DJI Matrice 300. We selected DJI Matrice 300 and DJI Zenmuse H20 for the following reasons. We aim to monitor the H.halys adults, whose size is $1\,\text{cm} \times 1.4\,\text{cm}$, by capturing orchard pictures and applying computer vision algorithms to detect the H.halys. Fig. 1 reports the main characteristics of H.halys. It has striped bands on the antennae, the head is rectangular, there are some clearer dots, aligned and well marked, both on the pronotum disk and at the base of the scutellum (the triangular shaped structure between the wings), and a typical pattern of the connexivum (the flattened lateral edge of the abdomen) with checkered medial triangular shaped spots. It is essential that the vision chip in the orchard scouting process has sufficient resolution to distinguish the unique H.halys features.

According to optics theory, the *resolution distance* σ is the minimum distance of two distinguishable points in the captured view. Since the smallest feature of the H.halys is $\approx 0.2\,\text{mm}$ (i.e., the antennae or the clearer dots), the value of σ must be adjusted to $0.2\,\text{mm}$ to identify every single feature. Fixed σ, we find the maximum *Diagonal Field of View* F_D (DFoV) suitable for our goal as the product of σ by the number n_d of pixels on the diagonal of the sensor chip. Clearly, the higher the number of pixels, the larger the DFoV that preserves the suitable resolution. Using the chip *Diagonal Angle of View* (DAoV) α_d, i.e., the diagonal dimension of the solid angle at the lens opposite to DFoV, we can correlate the DFoV with the distance from which the subject is captured (see Fig. 2). Fixed the α_d and F_D, one can find the distance $\mu = \frac{F_D}{2\cdot\tan(\alpha_d/2)} = \frac{\sigma n_d}{2\cdot\tan(\alpha_d/2)}$ to be used to capture the image complying with the desired F_D. Note that if α_d is fixed, there is a unique distance μ between the camera and the subject that complies with F_D. Instead, the same F_D can be obtained at different distances μ if α_d changes. For example, in Fig. 2(b) the blue and black triangles in front of the lens are two images with the same F_D, but diverse distances (μ' and μ).

The two cameras mounted on the DJI Phantom 4 and DJI Mini 2 had a chip of 12 MP with $n_d = 5000$ and, respectively, $\alpha_d = 94°$ and $\alpha_d = 83°$. Hence, the DJI Phantom 4 and DJI Mini 2 drones achieved the suitable $F_D = 0.2 \cdot n_d = 1000\,\text{mm}$ at the distance 466 mm and 565 mm, respectively. The drones' obstacle avoidance system of the DJI Phantom 4 and DJI Mini 2 would have blocked them when they tried to fly closer than 1 m to the trees. Hence, the DJI Phantom 4 and DJI Mini 2 drones were discarded.

Since the aforementioned embedded cameras are not suitable for our goal, we then consider the CMOS chip of the DJI Zenmuse H20

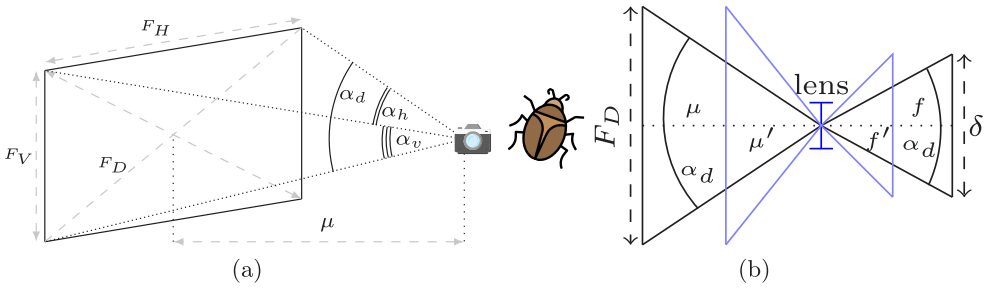

Fig. 2. FoVs and AoVs from a camera (a); and two configurations (one black, one blue) with the same F_D (b).

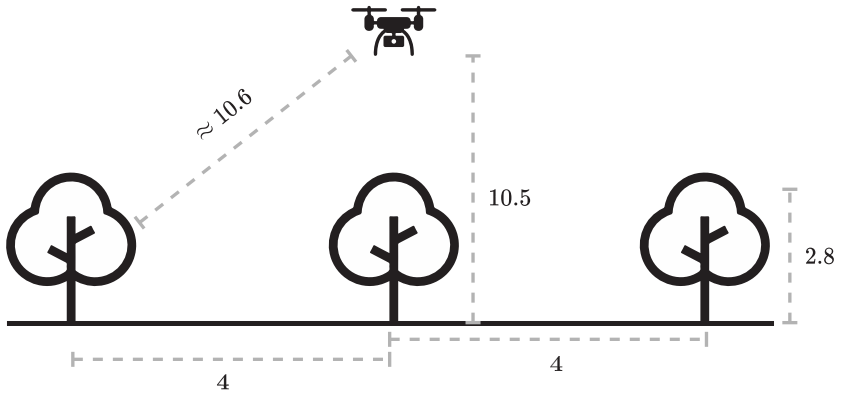

Fig. 3. Illustration on the protocol used by the drone for taking pictures inside the orchard (numbers in meters).

camera. This is an RGB camera mounted as an external payload on the DJI Matrice 300. The chip has effective 20 MP with $n_d = 6483$ px. The DJI Zenmuse H20 camera is a zoom camera whose DAoV α_d varies from 4° to 66.6°. Although the F_D of the DJI Zenmuse H20 does not change too much from those of DJI Phantom 4 and DJI Mini 2, $F_D = 0.2 \cdot 6483 = 1296$ mm, different distances can be suitable because one can rely on a different angle of views. Varying α_d from 4° to 66.6°, the same F_D can be achieved at different distances from the subject. The $F_D = 1296$ mm complying with the desired resolution can be achieved with the DJI Zenmuse H20 by selecting any distance between 18.5 m and 0.98 m. Hence, we chose the DJI Matrice 300 drone (Betti Sorbelli et al., 2022) (6.3 kg) to carry the DJI Zenmuse H20 payload (≈0.7 kg). The drone has open propellers with a diameter of ≈1 m and achieves accurate global positions (1.5 cm displacement) when it operates using the real-time kinematic positioning (RTK).

Given that, the width of the aisles is about 4 m, the DJI Zenmuse H20 position during the flight must be exactly in the center of the aisle to take photos with the desired DFoV. Although the DJI Matrice 300 flight accuracy is in the order of centimeters thanks to RTK, its movements are severely restricted, and due to the presence of out-of-gauge branches the drone experiences undesired stalls. Thus, to capture images from the aisle, it is necessary to guide the drone in First Person View to avoid the obstacles, and in case of stall, to reactivate its flight. This is the kind of mission proposed in Almstedt et al. (2023) and Betti Sorbelli et al. (2023). Differently, in this work, we look for a completely autonomous drone flight.

In order to obtain an autonomous flight, the drone is forced to fly above the orchard. Since the distance among two consecutive rows is 4 m, we set up the drone altitude to 10.5 m, ensuring a minimum distance of approximately 10.6 m between the drone and the canopy of the trees (see Fig. 3). As explained before, at that distance, DJI Zenmuse H20 has an angle of view that guarantees to detect the H.halys when $F_D \approx 1.3$ m. Moreover, placing the drone above the 8 m, according to DJI specifications, the focus troubles raised by telephotos, like ours, are solved.

3.1. Android application overview

To monitor the presence of the H.halys inside the orchards, we design a custom Android application with twofold functionalities. First of all, the application can submit an autonomous inspection of the orchard leveraging the DJI Matrice 300 equipped with the DJI Zenmuse H20 to collect images useful for tuning the detectors. Secondly, the application allows to elaborate any RGB image by invoking one of our trained models hosted on external server through an API call.

The application is composed of two principal components: a *server* and a *client*. The server is responsible for hosting the trained models referenced in Section 5 and processes *requests* submitted by one or more clients. The client is in charge of launching a mission for acquiring images and also of transmitting the captured images to the server to obtain predictions, upon a user request. The architecture of the client–server application is illustrated in Fig. 4.

The server component provides H.halys detection to clients upon request, remaining passive until a client submits captured images and specifies a model for querying. Utilizing the lightweight Python web framework Flask (Grinberg, 2018), the server efficiently handles HTTP POST requests, allowing clients to send images via a specific URL format (`url/model`). Clients can refer to any models discussed in Section 5, such as `yolov5-x` or `retinanet`. Upon receiving a request, the server employs PyTorch-implemented models to perform image recognition. If H.halys are detected, the server returns bounding box coordinates and confidence values to the client.

On the other hand, the client is responsible for capturing images and sending requests to the server. Given the decoupled nature of the server component in this architecture, any device capable of supporting HTTP POST requests can run the client and thus interact with the server (see Fig. 5(a)). This includes smartphones, tablets, laptops, computers, and drones. Our client is an ad hoc Android application which has been developed using Java and the DJI Mobile Software Development Kit (MSDK) V4 for running a drone mission (see Fig. 5(b)). The MSDK facilitates the creation of bespoke Android or iOS client-applications designed for installation on mobile devices that interface directly with

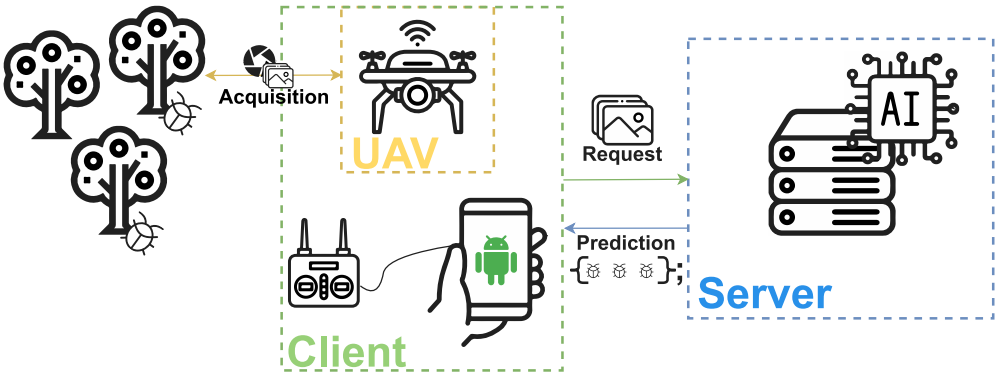

Fig. 4. Android application illustration.

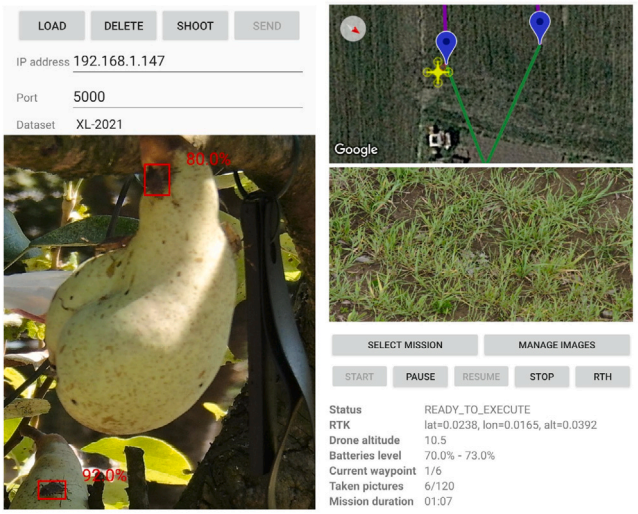

Fig. 5. Example of Android app interfaces. (a) Capturing new images or requesting a detection; (b) Status visualization of the ongoing autonomous mission.

the drone's remote controller. The ad hoc Android application enables the drone to autonomously execute flights along a pre-loaded set of waypoints, adjust the drone's gimbal, and capture images with specified camera parameters, e.g., focal length, or aperture. In order to execute a mission the device must be connected to the remote controller. The communication between the application on the mobile device and the remote controller is established through USB allowing a direct interface with the DJI Matrice 300 and the DJI Zenmuse H20.

In order to start the mission, the application requires a set of geotagged waypoints. A mission for the drone is articulated as a path connecting an initial point (Home) and various geotagged locations (waypoints) within the orchard. Hence, the application prompts the user to define waypoints within a monitoring area where the drone will capture images. Fig. 6(a) details the sequence of waypoints, in our specific scenario. The waypoints have been defined to meet the constraints of our setting. In particular, we limit the number waypoints to 9 due to the battery capacity of the DJI Matrice 300. Notice that, a complete tour of the 9 positions requests ≈45 min of flight, including take-off and landing in a safe zone apart from obstacles. Moreover, the spatial distribution of the waypoints was driven by the position of pheromones, placed to improve the chance to capture an H.halys inside the images. At each waypoint, the drone stops to perform the image acquisition, virtually drawing a *smart grid* of about $6{\times}3 = 18$ sqm, consisting of 4×5 cells (20 cells). To precisely meet the waypoints, the DJI Matrice 300 exploits the RTK technology. Each cell respects the desired $F_D \approx 1.3$ m. Since the slope distance between the DJI Zenmuse H20 and the cells varies with the relative position of the cell in the smart grid, for each cell a slightly different angle of view is selected so that F_D is preserved (Fig. 6(b)). Hence, at each cell, the drone adjusts its yaw and camera angles, along with the angle of view. The 3-axis gimbal stabilization system of the DJI Matrice 300 ensures a stable camera configuration during picture capturing.

The images obtained from this process are subsequently utilized to construct the dataset, paving the basis for training computer vision algorithms.

Finally, while the primary function of this application is to facilitate autonomous flights and image capturing, it also provides the capability for users to manually select and transmit stored images to our external server.

4. The dataset

The dataset is composed of images autonomously captured by a drone as described in Section 3. The dataset consists of images containing only a category of stink bug, the *Halyomorpha halys* (H.halys). The dataset comprises 402 annotated images, each containing instances of the H.halys class. However, when considering all the images, there are actually 546 distinct occurrences of H.halys specimen. Notice that, only images containing H.halys adults were included inside the dataset. An insight into the complexity of the detection challenge at hand is provided in Fig. 7. In principle, the images are characterized by a cluttered background composed mostly of foliage. Despite the background complexity, the good resolution enables clear visualization of all the distinctive traits of the H.halys (Fig. 7).

We validate the decisions made for the drone protocol by surveying the features of the H.halys captured, and assessing the overall quality of the images (Fig. 8). Specifically, for each bounding box, Fig. 8(a) shows on the x-axis (y-axis) its width (height) expressed in pixels (px), whereas Fig. 8(b) shows its relative coordinates, i.e., x and y components of bounding boxes' center, defined in percentage with respect to the image size.

In principle, we can observe that the bounding boxes are small-sized, and with limited variations (Fig. 8(a)). Indeed, their size varies between a minimum of 33×45 px up to a maximum of 183×132 px. The previous observation becomes more evident when the bounding boxes' size is evaluated with respect to the image size, i.e., 5184×3888 px. Moreover, paws and antennae have been included in the annotations, thus the quantity of pixels ascribable to the H.halys diminishes further. Notice that the labeling process has been performed by an expert human operator using the well-known open-source cloud service `makesense.ai` (Skalski, 2019). As a result of the constrained nature of the acquisition process, the variability in the bounding box sizes is minimized. Indeed, the specimen photographed in the same pose will be almost the same size since the distance from the trees is fixed.

Fig. 6. Drone mission path (a); smart grid imaging technique (b).

Fig. 7. Example of drone captured images and their resolution.

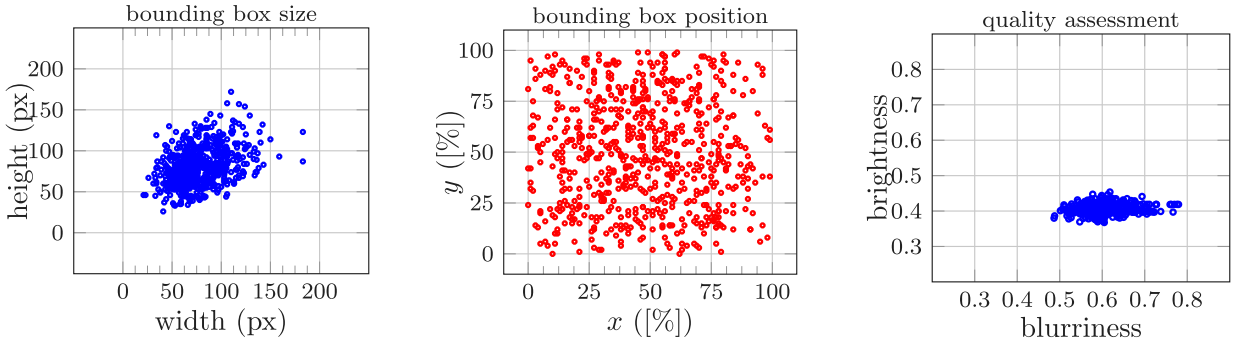

Fig. 8. Bounding box size distribution (a); bounding boxes relative position distribution (b); (c) Dataset evaluation on blurriness, and perceived brightness.

Moreover, specimens captured with different poses will experience limited variation in their size. Notice that the majority of the H.halys annotated in the dataset is captured with a back perspective (see Fig. 7). This is evident from the clustered nature of the plot in Fig. 8(a).

The position of the bounding box's centers varies a lot as reported in Fig. 8(b): the centers span practically throughout the entire plot surface. This is obvious because the bugs' positions cannot be predetermined a priori during the flight and they move seeking warm temperature, or food.

Moreover, since the image acquisition process may fail for several reasons, e.g., failures in the auto-focus protocol or drone flickering during image capturing, we introduce no-reference estimators for assessing the dataset quality. Specifically, we rely on Narvekar and Karam (2011) for blurriness, and on Ware (2021) for perceived brightness estimation, respectively. Fig. 8(c) depicts the perceived blurriness, and brightness observed. For each image, the x-axis displays the blurriness score, while the y-axis the brightness score. The blurriness score falls in the range $[0, 1]$, where 0 indicates a totally blurred image and 1 represents an extremely sharp one. Similarly, the brightness metric ranges from 0 to 1, where 0 states absent lights, and 1 denotes dazzling light. Consequently, images with favorable metrics should have a blurriness score close to 1 and a brightness score around 0.5. From an analysis of blurriness levels (Fig. 8(c)), we can notice that all the images achieve satisfactory results, with a stable score around to 0.6. These results have been quite predictable because the drone acquisition protocol is strictly configured taking into account optics law and employing both the RTK and the gimbal stabilization. Indeed, these two latter technologies mitigate the out-of-focus occurrences at unexpected shacking. Moreover, exploiting the well-known *hyperfocal focus* technique, we are able to guarantee limited blurriness spots. This distance depends on both the focal length and the aperture settled, but once a lens is set at the hyperfocal distance, the depth of field extends from half this distance to infinity. Therefore, the blurred spots shrink to the objects that stand before the hyperfocal distance, e.g., protruding branches.

A very similar behavior is shown for brightness (Fig. 8(c)). However, since the brightness depends on the environmental light condition, the

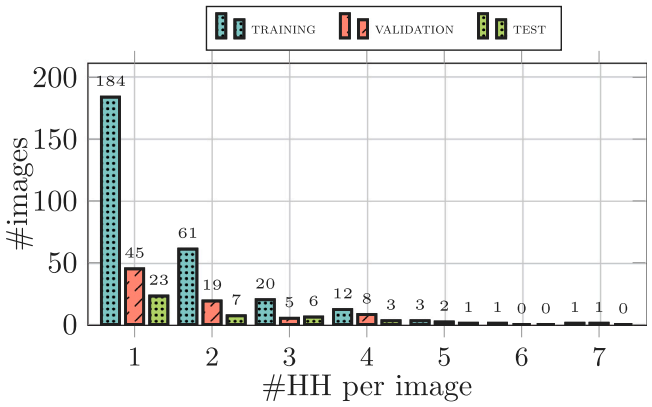

Fig. 9. Distribution of the number of H.halys per image.

trend depicted in the plot is almost constant. Indeed, the points trend settles stable to 0.4 very close to the optimal value, i.e., 0.5. This is due to the scarce variability of light conditions experienced during a single mission. Considering that a drone flight is at most 50 minutes, the light levels may not vary consistently in this short period of time. Hence, camera settings settled at the beginning remain optimal for almost the total of the mission.

In conclusion, the dataset exhibits satisfactory quality in terms of blurriness and brightness supporting the effectiveness of the parameters adopted for the DJI Matrice 300 and DJI Zenmuse H20 during the navigation protocol. Additionally, despite the constrained acquisition protocol, the dataset ensures a notable variance in bug position, resulting in a suitable set of images for training computer vision models.

We partition the dataset in the following way, 70% is allocated for training, 20% for validation, and 10% for testing. Hence, the original 402 images are grouped in 282 images for training, 80 for validation, and 40 for testing.

Fig. 9 provides a good insight into the number of H.halys occurrences per image. The distribution is aggregated according to the split applied. The bar plot shows that the occurrence of H.halys is relatively rare in our dataset, with most images containing only one instance. In principle, the more H.halys in an image, the rarer the occurrence of such an image. Moreover, the percentage of images aggregated per number of H.halys inside remains stable with limited variations across all three sets. The analysis made stresses the predominance of background pixels against H.halys ones.

Given that, the H.halys area occupies only a minuscule fraction of the entire image, it becomes evident that directly inputting the raw image into the neural network will yield unpromising results. Moreover, as emphasized by the authors in Sorbelli et al. (2023), the primary cause of performance degradation is the loss of information resulting from image scaling. Specifically, the models proposed in Sorbelli et al. (2023) achieved an average Precision of 56% and an average Recall of 33% extremely unsatisfactorily for an in-field solution. Indeed, neural networks grapple with high-dimensional data, making their training on large and intricate datasets computationally demanding. Since the image is 5184×3888 px, shrink it to 640×640 px makes practically invisible tiny objects as the H.halys. Hence, the scaling operation causes image compression reducing the width and the height by approximately 8 and 6 times, respectively. As a consequence, due to the scaling operation, the surface of the H.halys is reduced $\approx$64 times in terms of pixels. Given that, the average size of the H.halys is $\approx 108 \times 89$ px, the scaling operation would compress the H.halys into a 13×14 px instance. Such compression would cause a huge loss of information inside a scene where the H.halys was already challenged to recognize due to its limited dimension. To mitigate the risk of losing crucial features due to image scaling, we have chosen to implement a slicing procedure. Therefore, we deal with high-resolution images by cropping them in equally sized patches. This approach enables us to reduce the training complexity and put the network's attention on specific sub-portions of the image. In view of this, we decide to extract from the images patches of size 640×640 px. Notice that the chosen patch size represents a good trade-off between prediction time for the entire image and network performance. Indeed, patches of 640×640 px represent the input setting optimized by YOLOv5 developers, and the size settled for the network weights pre-trained on MS-COCO dataset (Lin et al., 2014). So, from here on out we will use the word *patch* for referring to an image slice of size 640×640 px.

Concerning the patch extraction procedure, we decide to rely on two approaches, i.e., *random sampling*, and *regular sampling*. The former technique is a randomized procedure, and the basic idea behind is to randomly extract patches until the desired number of patches is reached. In detail, random patches are extracted from original images looking for H.halys instances (Fig. 10(a)). In case a H.halys is surrounded, the annotation file is adjusted according to the relative position of the H.halys inside the patch. On the other hand, the *regular sampling* consists in extracting patches from full-size images by applying a regular grid of cell 640×640 px. More in detail, the very first step is applying black padding on both sides since both the images' side sizes are not divisible by 640. Then, the slicing procedure is performed with an annotation adaptation to the new dimensions (Fig. 10(b)). The latter procedure returns a number of patches equal to $\left\lceil \frac{W}{640} \right\rceil \times \left\lceil \frac{H}{640} \right\rceil$, where W and H represent the width and height of the image, respectively. Hence, we obtain $\lceil \frac{5184}{640} \rceil \times \lceil \frac{3388}{640} \rceil = 7 \times 9$ patches from each image. Notice that, the image is padded so that W and H are multiple of 640.

From now on, recalling that a patch is said *negative* if it does not contain any H.halys, we call *positive* a patch with the bug. Finally, we observe that some patches contain a fraction of the pixels of a H.halys because the bug falls on the border of a patch. To ensure that patches with a fraction of a H.halys contain a significant portion of the H.halys pixels, a thresholding principle is applied by comparing the current instance with the original pixel mask. Only the patches with more than 50% of the original H.halys pixels are considered positives.

5. The proposed HH detectors

In this section, we describe the four detectors employed to tackle the problem of H.halys detection. The selected detectors are among the most widely used neural networks for object detection. Specifically, we utilize YOLOv5, YOLOv8, RetinaNet, and Faster-RCNN, all of which are capable to achieve state-of-art performance on the most challenging dataset recognized within the computer vision community (Zou et al., 2023). Although CenterNet's state-of-art performance, we decided to exclude it from our evaluation due to its poor results in previous research on H.halys detection (Almstedt et al., 2023). We preserved the same training set for every model, although for each model the training process may be different. We start by analyzing the training configuration settled for YOLOv5 in Section 5.1, and YOLOv8 in Section 5.2. Then, we describe the configurations for RetinaNet (Section 5.3), and Faster-RCNN (Section 5.4). We conclude by evaluating training and validation performance (Section 5.5).

5.1. YOLOv5 *Training specs*

We employ for our experiments YOLOv5 (v5) algorithm, one of the latest iteration of the YOLO (You Only Look Once) series introduced initially by Redmon et al. (2015) and more recently enhanced by Jocher et al. (2022). The peculiarity of this family of algorithms is that the object detection task is reframed as a regression problem rather than a classification task. This is achieved by spatially separating bounding boxes and assigning probabilities to each detected image using a single

Fig. 10. Illustration of patches extraction techniques: (a) *random sampling*, and (b) *regular sampling*.

CNN. The main feature of Y5 is its efficiency since it is less computationally demanding than other state-of-the-art models (Jocher et al., 2022).

For this computer vision algorithm, we create a custom training configuration. In particular, by applying the *random sampling* procedure, we extract 8000 patches for the training and 800 for the validation set. Given the elevated sparsity of the H.halys instances within the images, it follows a high probability of cropping patches that do not contain the bug (negative patches). This suggests that the models should get used to negative patches. Due to this fact, we create three different training sets, each of size 8000, which differ from the percentage of negative patches. The training set T_0 contains no negative patches and thus 100% of positive patches, whereas T_{50}, and T_{75} consist of 50% (i.e., 4000), and 75% (i.e., 6000) negative patches, respectively. We kept the number of patches stable at 8000 during training, despite varying negative patches across T_0, T_{50}, and T_{75}. This resulted in a decrease in the number of positive patches. Regarding the validation set, instead, we decide to create a set with always the 100% of positive patches. Finally, we augment the number of training samples by generating three additional images for each training image through random transformations. Every transformation is computed at each epoch on the fly exploiting the Python library `Albumentation` (Buslaev et al., 2018). These transformations involve Hue, Saturation, and Value (HSV) variation, translation, scaling, flipping, and mosaic techniques.

All the YOLOv5 model sizes, i.e., Nano ($\mathcal{N}$), Small ($\mathcal{S}$), Medium ($\mathcal{M}$), Large ($\mathcal{L}$), and eXtra-large ($\mathcal{X}$) have been trained.

In conclusion, we trained 5 × 3 models. We refer to YOLOv5 models using as superscript the percentage of negative patches in the training set, and as subscript the model size, e.g., $\text{Y5}_{\mathcal{S}}^{50}$ it refers to the Small YOLOv5 model trained using the training set with 50% of negative patches. For each model, we adopt the Transfer Learning (TL) paradigm: the pre-trained weights are obtained from intense training over the MS-COCO dataset (Lin et al., 2014).

We rely on an NVIDIA RTX 3060 with 12 GB of VRAM for training YOLOv5 detectors.

Each model undergoes training of all the patches in batches of 32 for 1000 epochs, using a learning rate of 10^{-2}, and the Stochastic Gradient Descent optimizer (Bottou, 2010). Notice that, we configure and permit the early stopping for the training process.

5.2. YOLOv8 *Training specs*

YOLOv8 (Y8), released in 2023 by Ultralytics (Jocher et al., 2022), is a deep learning model for real-time object detection. This version is the direct successor of YOLOv5 (Section 5.1), and although some improvements, it inherits the majority of its features. A code analysis reveals YOLOv8's composition, featuring a decoupled head structure for classification and a backbone for bounding box regression. Differently from YOLOv5, it is an anchor-free detector, predicting object centers directly. Similarly to YOLOv5, YOLOv8 comes in various variants. We adopted the YOLOv8 versions $\mathcal{N}$, $\mathcal{M}$, and $\mathcal{L}$. Regarding the training and the validation set, we decided to apply the *regular sampling* procedure to differentiate from YOLOv5. As a result, we extract 17,766 patches for the training (out of which only 423 positive patches), and 5040 (out of which 136 positive patches) for the validation set. Notice that, differently from the random sampling, since a single grid is applied to crop the image, every image pixel is extracted just once. Hence, the training set contains 423 positive patches, and the validation set 136. As a result, the percentage of positive patches significantly decreases to ≈2%, or equivalently, 98% of negative patches, for both sets.

YOLOv8 is trained on a NVIDIA 2080Ti with 11 GB of RAM employing TL, such as MS-COCO weights (Lin et al., 2014), and settling 1000 epochs. The rest of the parameters are set up equally to YOLOv5.

5.3. RetinaNet *specs*

In our experiments, we adopt RetinaNet (RE). The RetinaNet is a one-stage detector with two sub-networks for classification and regression. Moreover, RetinaNet employs a Feature Pyramid Network, specially tailored to address challenges related to small objects and varying scales. This innovative loss function prioritizes challenging patches, enhancing the model's effectiveness (Lin et al., 2020).

For this model, we rely again on the powerful TL paradigm (RE_T). However, we decided to explore the use of a different weight initialization, namely the random initialization (RE_S). Considering the data on which the RetinaNet bases the learning procedure, we employ the promising *random sampling* extracting procedure. However, to intercept the actual behavior of the learning mechanism, we adjust the number of training and validation patches. In particular, we extract 5900 patches, and 1640 patches, for training and validation sets, respectively. Similarly to YOLOv5, we create different training sets to understand the impact of negative patches. Indeed, we create a first training set with 100% positive patches (RE^0), and a second one with 50% of negative patches (RE^{50}). As a consequence, four models are obtained.

The training of RetinaNet is conducted using an NVIDIA Tesla K80 with 12 GB of RAM, spanning 1000 epochs with a learning rate of 2.5×10^{-3} and an Intersection over Union (IoU) of 0.5. Early stopping is employed to prevent overfitting (on the validation dataset). Moreover, for enhanced model generalization, augmentation methods such as random rotation, and horizontal and vertical flipping are applied to the training patches.

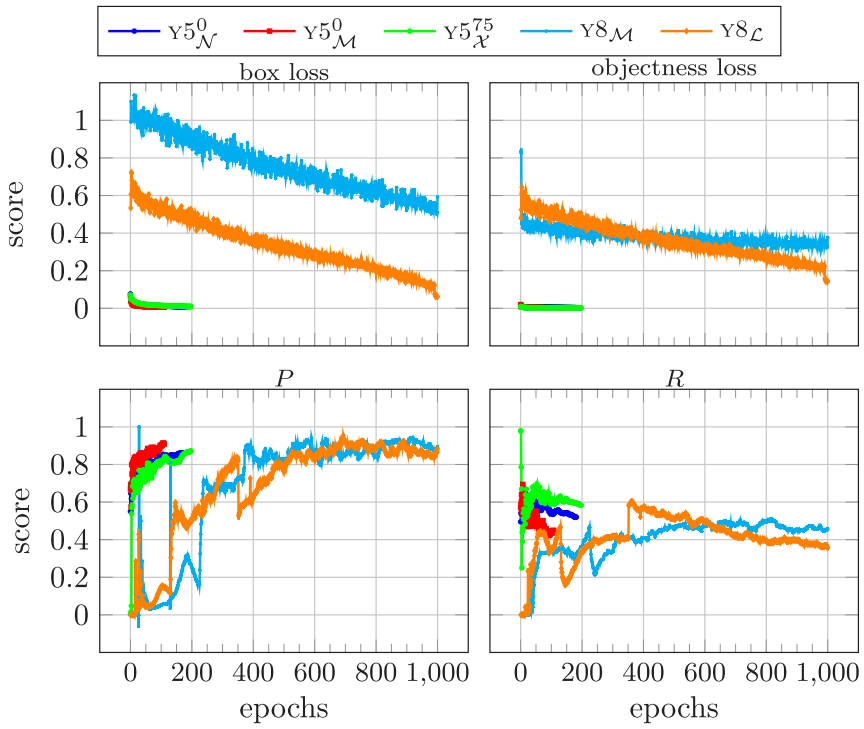

Fig. 11. YOLOv5 and YOLOv8 training and validation results.

5.4. Faster-RCNN *specs*

In contrast to YOLOv5 and YOLOv8 single-stage architecture, Faster-RCNN (fr) adopts a sophisticated two-stage design. This architecture comprises a Region Proposal Network, responsible for generating detection candidates. Then, both the proposals and the feature maps are passed to Faster-RCNN, where the object is classified and bounding box values are decided (Ren et al., 2017). The chosen backbone architecture for Faster-RCNN is ResNet-50, known for its robust performance.

For both the training and validation setting, we reuse the same applied for in Section 5.3.

5.5. *Training and validation results*

In this section, we conduct an extensive evaluation of the models presented in the previous sections. We evaluate YOLOv5 and YOLOv8 performance on both the training and validation set employing the most common scores, i.e., box-loss, objectness loss, precision, and recall (Padilla et al., 2021). In Fig. 11, we report the training performance of the resulting YOLO models. Each plot displays on the y-axis the score, and on the x-axis the epochs. It is worth noting that, the curves for YOLOv5 span up to 200 epochs, while those for YOLOv8 up to 1000 epochs. Although YOLOv5 is set to run up to 1000 epochs, the early stopping option breaks the computation after at most 200 epochs because the curves do not improve hereafter. Furthermore, we limit the performance evaluation to the most representative subset of models for YOLOv5. In particular, we rely on model $\mathcal{N}$ trained on T_0 ($\text{Y5}^0_{\mathcal{N}}$), $\mathcal{M}$ trained on T_0 ($\text{Y5}^0_{\mathcal{M}}$), and on $\mathcal{X}$ trained on T_{75} ($\text{Y5}^{75}_{\mathcal{X}}$). Concerning YOLOv8, we focus only on models $\mathcal{M}$ ($\text{Y8}_{\mathcal{M}}$) and $\mathcal{L}$ ($\text{Y8}_{\mathcal{L}}$).

In Fig. 11 (top-left), we observe the *box loss* representing the model's ability to predict bounding box coordinates. The YOLOv5 models reach a low box loss rapidly, although the $\text{Y5}^{75}_{\mathcal{X}}$ converges faster. Different behaviors for $\text{Y8}_{\mathcal{M}}$ and $\text{Y8}_{\mathcal{L}}$, which converge slower, and reach a loss value much far from the YOLOv5 one. In Fig. 11 (top-right) is reported the *objectness loss*, which is roughly the confidence that an object exists in a given box. The behavior is similar to that of the *box loss*.

In the second row of Fig. 11 are reported the metrics during the validation phase. All the models achieve good performance and tend to stabilize after a few epochs. As regards the precision P (Fig. 11, bottom-left), the YOLOv5 models reach a peak immediately, maintaining it up to the last epoch. Conversely, YOLOv8 behaves in a complementary way, growing slowly reaching the peak at the very last epochs. Fig. 11 (bottom-right) reports R trends observed during the validation. The YOLOv5 models reach a peak, detecting the maximum number of objects, during the first epochs, stabilizing, then, at ≈60%. The trend just observed for YOLOv5 models can be found extended on 1000 epochs for YOLOv8. Both P and R stabilize within the last epoch.

In the following, we consider RetinaNet, and Faster-RCNN models, and report the P, the R, and the mean Average Precision Pascal ($mAP_{0.5}$) (Everingham et al., 2010) metrics (see Fig. 12). As regards RetinaNet, we consider its performance under 0% of negative patches, and employing both the random weights initialization (re^0_S), and the transfer learning (TL) (re^0_T). Whereas, the behavior of Faster-RCNN under transfer learning and two different values of negative patches, 0% (fr^0_T), and 50% (fr^{50}_T). Notice that for each model, we vary one parameter at a time to comprehend their impact on the models trained.

We report the results only for the first 50 epochs because, although the number of epochs is set to 1000, the early stopping activation occurred after 50 epochs. Among the Faster-RCNN models, the best performing model is fr^{50}_T, which overcomes the metrics of the others. Concerning RetinaNet performance, we observe a slow ascending growth in both the models' P curves, whereas a stable and solid behavior for R, and $mAP_{0.5}$. Although re^0_T, and re^0_S work with almost the same trend, re^0_T marks better metrics. This is due to the usage of the well-known TL technique, which boosts model detection by transferring useful knowledge from pre-trained weights.

Summarizing, the usage of TL has a positive impact on the training procedure, and the usage of the random sampling extraction technique leads to more stable trends on both losses and metrics. Furthermore, there is not any significant difference between RetinaNet and Faster-RCNN models regard the usage of TL and the injection of negative patches during the training.

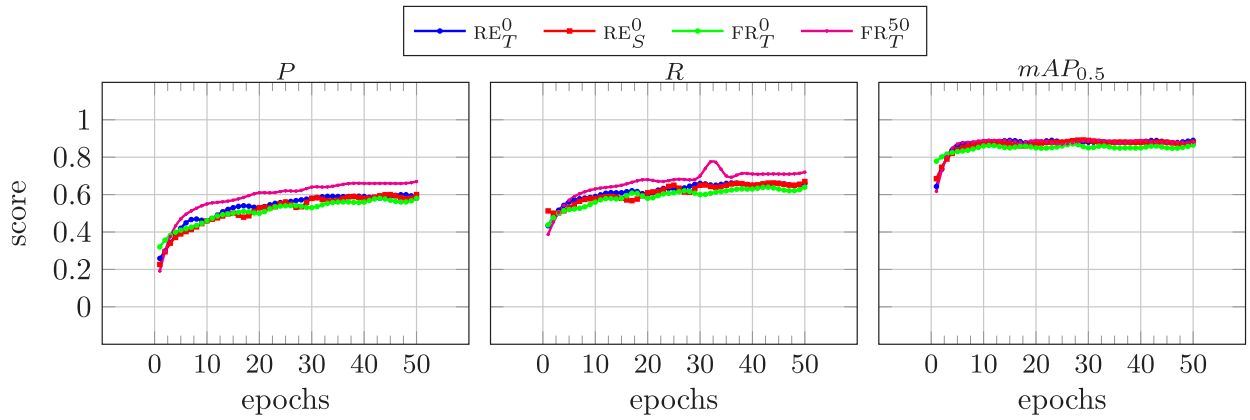

Fig. 12. RetinaNet and Faster-RCNN validation results.

6. Testing results

In the following section, we evaluate the results of the proposed H.halys detectors, each trained using different methods, on the same testing set. For evaluating the detection ability of the models we use the remaining 10% of the dataset, i.e., 40 images. Since in the test set, we do not have any clue on the target object position, we extract the patches from the test set by applying a regular grid of cell 640×640 px, i.e., *regular sampling* (Fig. 10(b)). Given that, we observed that the majority of the patches belonging to the last row (column) are mostly padding pixels. Since padding pixels are useless for detection purposes, we decided to discard these patches. Specifically, we exclude both the last row and column patches, i.e., 15 patches per image. In this way, employing *regular sampling* a new test set of 1920 patches is obtained. We know that there are overall 71 H.halys instances in the testing set. In the next sections, we are going to estimate model performance under different degrees of evaluation. We would like to remark that, the adults of H.halys are the only focus of this work. Since we aimed to prove the feasibility of discovering H.halys specimens via aerial photos, we decided to focus only on adults, which are more likely to be found flying and are commonly present on the tree canopy.

Nonetheless, our models have demonstrated the ability to recognize other bugs with similar characteristics along their predicted false positive. Therefore, while we cannot claim that our detectors are capable of recognizing other stages of H.halys with the same metrics presented in this manuscript, we are optimistic about the potential for tuning the neural network weights. With a significant number of images representing juvenile stages, we believe our system could establish notable detection capabilities also for these stages with accuracy in line with what already observed.

In Section 6.1, we are going to measure the pure ability to detect the H.halys, by considering the positive patches in the test set. Then, in Section 6.2 we will benchmark the models against negative patches, evaluating the models on the 1920 test patches extracted by the regular sampling from our testing set. Finally, the sensibility against occluded instances is handled in Section 6.3.

6.1. Detectors performance

In this section, we evaluate the performance of the models in detecting H.halys. For this purpose, we run the detection on the 71 unseen positive patches within the test set, without applying any transformations. Hence, every model will be evaluated on patches that contain at least one H.halys.

Table 1 depicts the detectors' performance, and in bold the metrics of $\text{Y5}_{\mathcal{M}}^0$ which appears to be overall the model with the best metrics. In detail, we compute the number of True Positives (TP), False Positives (FP), and False Negatives (FN), which are primitive estimators for counting the number of correct, incorrect, and missed detection, respectively. Then, we leverage the metrics used during the validation phase, namely precision (P), recall (R), and mean Average Precision Pascal ($mAP_{0.5}$) (Everingham et al., 2010). We extend the analysis to the $mAP_{0.5:0.95}$ (MS COCO challenge (Lin et al., 2014)), also known as mAP[0.5:0.05:0.95], which is the area under the precision–recall curve for the values of IoU in the interval $[0.5, 0.95]$ with step 0.05. The $mAP_{0.5:0.95}$ measures the robustness of the precision and recall score according to the confidence, and several values of IoU. If $mAP_{0.5:0.95} \approx mAP_{0.5}$, it means that the precision–recall area is still the same though IoU increases, or equivalently all the predictions have a large IoU. In addition, we mint a novel further measure, the $mAP_{0.25:0.95}$, also denoted as mAP[0.25:0.05:0.95], which extends the metric MS COCO challenge (Lin et al., 2014) to the interval $[0.25, 0.95]$. This metric intends to valuate if the model benefits of small values of IoU. If $mAP_{0.25:0.95}$ is above $mAP_{0.5:0.95}$, it implies that the precision–recall curve area is larger for IoU values in the range $[0.25, 0.5]$ compared to the range $[0.5, 0.95]$. We came up with this measure since the H.halys size is limited and a large value of the IoU is not crucial for an IPM system.

In the following, we discuss the results for P, R, $mAP_{0.5}$, $mAP_{0.25:0.95}$, $mAP_{0.5:0.95}$, and the effect of including negative patches in the training set for the models. In principle, considering the complexity of the images under consideration (see Fig. 7), we can observe that all the detectors obtain satisfactory results. Starting from the P, we can notice that every model performs achieving at least a score of 60%, and the majority of them compute with levels larger than 70%. The peak in P is reached by $\text{Y5}_{\mathcal{M}}^{50}$, and $\text{Y5}_{\mathcal{L}}^{50}$ with a notable 97%. Notice that, both Y5 and Y8 perform overall the best concerning P, recording a particular low number of FP. This can be attributed to YOLO's nature since it is prone to notably distinguish the background. As a result, YOLO models tend to be more conservative in drawing bounding boxes than Faster-RCNN, and RetinaNet.

Concerning the R, its value stabilizes around the 50% excluding a bunch of models that perform quite poorly. The best performing model according to R is $\text{Y5}_{\mathcal{M}}^0$ with a 73% of H.halys detected. The $\text{Y5}_{\mathcal{M}}^0$ model obtains also a particularly good P of 88% which candidates itself as the best overall model in detecting the H.halys. The merit of this consistent performance may be a consequence of the model size. Indeed, we observe that YOLOv5 $\mathcal{N}$, and $\mathcal{X}$ perform almost the same. This behavior suggests that $\mathcal{N}$ intercepted a too simple rule for detecting, whereas $\mathcal{X}$ a complex rule too tailored to data. So, $\mathcal{M}$ may have generalized the most the detection rule due to its intermediate dimension. On the other hand, $\text{Y8}_{\mathcal{N}}$ achieved the worst R, i.e., 28%. However, this result is predictable since the unpromising performance during the training phase.

Focusing on $mAP_{0.5}$, we observe a quite stable behavior of every model which stabilizes at ≈50%. As a result, every model predicts

Table 1
Performance of the detectors for 8 different metrics evaluated using a test set with only positive patches.

Model	TP	FP	FN	P	R	$mAP_{0.5}$	$mAP_{0.25:0.95}$	$mAP_{0.5:0.95}$
$\text{Y5}^{0}_{\mathcal{N}}$	38	4	33	0.90	0.54	0.53	0.35	0.26
$\text{Y5}^{50}_{\mathcal{N}}$	46	5	25	0.90	0.65	0.62	0.43	0.31
$\text{Y5}^{75}_{\mathcal{N}}$	43	6	28	0.88	0.61	0.57	0.39	0.27
$\text{Y5}^{0}_{\mathcal{S}}$	40	11	31	0.78	0.56	0.54	0.46	0.26
$\text{Y5}^{50}_{\mathcal{S}}$	36	6	35	0.86	0.51	0.46	0.34	0.24
$\text{Y5}^{75}_{\mathcal{S}}$	36	4	35	0.90	0.51	0.49	0.33	0.24
$\text{Y5}^{0}_{\mathcal{M}}$	**52**	**7**	**19**	**0.88**	**0.73**	**0.71**	**0.46**	**0.32**
$\text{Y5}^{50}_{\mathcal{M}}$	28	1	43	0.97	0.39	0.39	0.26	0.19
$\text{Y5}^{75}_{\mathcal{M}}$	40	2	31	0.95	0.56	0.55	0.37	0.27
$\text{Y5}^{0}_{\mathcal{L}}$	39	5	32	0.89	0.55	0.54	0.35	0.25
$\text{Y5}^{50}_{\mathcal{L}}$	29	1	42	0.97	0.41	0.40	0.27	0.20
$\text{Y5}^{75}_{\mathcal{L}}$	35	4	36	0.90	0.49	0.46	0.30	0.21
$\text{Y5}^{0}_{\mathcal{X}}$	33	5	38	0.87	0.47	0.45	0.3	0.21
$\text{Y5}^{50}_{\mathcal{X}}$	34	2	37	0.94	0.48	0.47	0.31	0.22
$\text{Y5}^{75}_{\mathcal{X}}$	38	2	33	0.95	0.54	0.53	0.34	0.24
FR^{0}_{T}	44	26	27	0.62	0.61	0.46	0.31	0.19
FR^{50}_{T}	45	14	26	0.76	0.63	0.47	0.33	0.19
FR^{0}_{S}	39	26	32	0.60	0.54	0.43	0.28	0.19
FR^{50}_{S}	46	20	25	0.69	0.64	0.46	0.32	0.18
RE^{0}_{T}	49	13	22	0.79	0.69	0.56	0.37	0.23
RE^{50}_{T}	49	32	22	0.60	0.69	0.51	0.37	0.24
RE^{0}_{S}	49	10	22	0.83	0.69	0.55	0.38	0.25
RE^{50}_{S}	44	10	27	0.81	0.61	0.46	0.33	0.21
$\text{Y8}_{\mathcal{N}}$	20	5	51	0.80	0.28	0.52	0.31	0.29
$\text{Y8}_{\mathcal{M}}$	27	1	44	0.96	0.38	0.59	0.33	0.31
$\text{Y8}_{\mathcal{L}}$	20	1	51	0.95	0.28	0.57	0.36	0.34

with acceptable levels of confidence. Again, $\text{Y5}^{0}_{\mathcal{M}}$ shines with a 71% score, supporting a notable detection generalization. On the other hand, excepting $\text{Y5}^{0}_{\mathcal{M}}$, the models are not IoU-robust because the $mAP_{0.5:0.95}$ and $mAP_{0.5}$ scores significantly differ. Indeed, $mAP_{0.5:0.95}$ values are always ≈15% lower than $mAP_{0.5}$. However, this behavior was quite predictable due to the small size of the H.halys. Despite the poor results, this does not represent a big pitfall in training our models. Comparing the $mAP_{0.5:0.95}$ with its customized version $mAP_{0.25:0.95}$, we note that the latter is ≈10% larger than the former. This behavior implies that the number of correct detections drastically increases due to the lower values of IoU. Given that, detecting the H.halys is the most crucial task, rather than tightly contouring it, we can conclude that the model benefits from low values of IoU.

Looking at the percentage of negative patches injected during the training, we observe a sharp impact on YOLOv5 models. In particular, Table 1 showcases a diminishing of FP with an increasing percentage of negative patches. We notice a descendant trend of FP and in the majority of models a first reduction of TP in correspondence of $\mathcal{T}_{50}$ followed by a consequent improvement at $\mathcal{T}_{75}$. Hence, we achieve an overall reduction of erroneous detection, i.e., false positives, by injecting patches with no H.halys for YOLOv5. The motivation behind this behavior can be explained as a direct consequence of the learning procedure. Though background pixels are the most frequent, these pixels are particularly challenging to differentiate from H.halys ones. Therefore, injecting negative patches in the training set, boosted the network FP awareness.

Faster-RCNN benefits from negative patches too. In this case, regardless of the weight initialization employed, the models improve successful detections (TP) and decrease errors (FP). The metrics gap between $\mathcal{T}_0$ and $\mathcal{T}_{50}$ becomes more evident on the model trained from scratch. A possible reason for the improved detection of negative patches is that random initialization enables models to converge more tightly on data. Conversely, RetinaNet depicts a sharp deterioration after the injection of negative patches, though a smoother drop is observed using random weights. This behavior is a consequence of the RetinaNet focal loss (Lin et al., 2020). Indeed, one of the main RetinaNet features is the capacity to handle unbalanced class distribution giving relevance to the rare class. Since the background is the most frequent class, introducing more patches from it "distracts" the network from learning salient H.halys features. Notice that both RetinaNet and Faster-RCNN reach balanced and solid performance though without showcasing any peak.

Finally, since YOLOv8 is trained using only the regular sampling, the performance evaluation brings out that pushing on the percentage of negative patches determines a hazy knowledge. Indeed, as a result of the large number of negative patches, i.e., 98%, the models become very conservative and only find objects when they are extremely confident, leading to fewer false positives (we can see this in high P) but also missing many true positives (the low R values). This behavior was predictable since neither YOLOv5 benefits from the increment of negative patches percentage, i.e., from 50% to 75%. Due to the limited number of true positive detected by YOLOv8 with respect to the other models, we decide do not include YOLOv8 in the following performance comparisons.

In conclusion, we obtain particularly solid results, strong enough to be compared with those reported in Betti Sorbelli et al. (2023) where the images were collected in First Person of View. We attribute this achievement to the quality of our dataset and the fact that both the testing and training sets comprise images captured under similar conditions. The strongest positive influence comes from the slicing procedure which allows to focus on tiny details of the image. Finally, it is worth noting that, overall the injection of patches without the bug during the training affects positively the performance reducing the number of false positives. However, the impact of this behavior is not always consistent across all models due to the varying learning parameters that affect their performance.

6.2. Negative patches robustness

In this section, we evaluate the models' performance in detecting H.halys on all the $n = 1920$ patches of the test set. We call this test set All Patches, while the one used in Section 6.1 is denoted from now on as Positive Patches. Note that, the All Patches test set becomes extremely unbalanced, with only 71 positive patches compared to 1849 negative ones. Approximately the 94% of patches are background only, envisaging a challenging task for the models.

This experiment can be seen as a real-life performance measurement. Indeed, when the drone starts a mission and collects images, we do not have any prior knowledge of the H.halys position in the frame. Therefore, since the position is unknown, we are forced to examine the entire image, and consequently, many patches are without H.halys.

For estimating detectors robustness against negative patches, we introduce the concept of *error rate* (ER), defined as $\text{ER} = \frac{FP}{n}$, that is, the ratio of the FP over the size of the testing set. Precisely, in All Patches, the total number of patches is $n = 1920$, while in Positive Patches (Section 6.1), is $n = 71$. The introduced ER rate allows us to handle a parameter that is never managed in the context of Object Detection, i.e., the True Negatives (TN). Roughly speaking, a TN is a correct negative detection, in juxtaposition with a FP that is a missed TN. In other words, when a TN occurs, the algorithm correctly states that the area it checked does not contain an object. Since in an image you may have almost an infinite number of TN, this value is usually neglected. Although classical Object Detection ignores TN, its value is crucial in our investigations due to the high percentage of negative patches that could lead our algorithm to increase only the number of FP. It is worth pointing out that in the experiment All Patches the number of TP and FN remains unaltered with respect to those in the

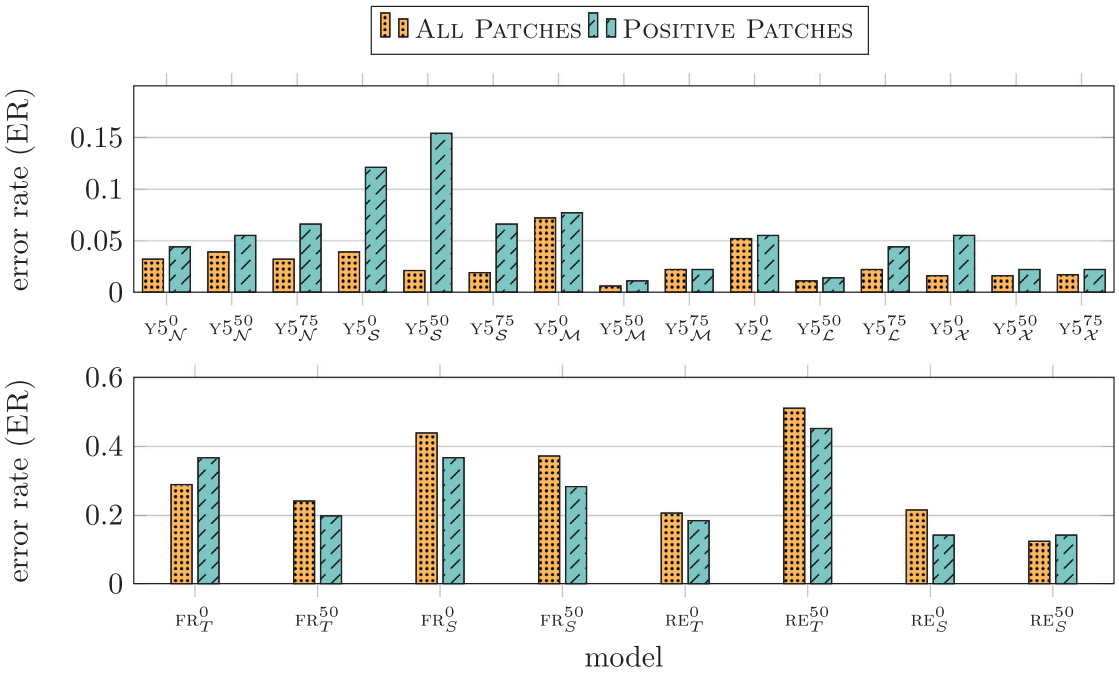

Fig. 13. Error rate comparison between All Patches and Positive Patches.

experiment Positive Patches, since the positive patches remain the same, and consequently the predictions made remain the same.

Fig. 13 compares for each model the ER achieved relying on only positive patches (Positive Patches), and on the entire test set (All Patches).

First of all, the ER ratio for All Patches observed is lower than the ER ratio for Positive Patches for the majority of the models. This means that while the models do not always correctly identify TN, they make fewer mistakes than one might expect when projecting the results of Positive Patches onto the All Patches ones. For YOLOv5, we can notice that the majority of models trained by injecting negative patches reduce a lot the number of FP per patch, even if the behavior does not state clearly that the models trained with 50% or 75% of negative patches are doing definitely better than those trained with 0% of negative patches. Nonetheless, YOLOv5 exhibits resilience towards negative patches with an ER always lower for All Patches. This is a consequence of YOLOv5 architecture, which is inclined to tune itself against the background.

The All Patches ER ratio is larger than in Positive Patches for Faster-RCNN and RetinaNet. Precisely, both Faster-RCNN and RetinaNet deteriorate the ER in the All Patches training except for FR^0_T and RE^{50}_S, showing that these models are not so robust to background patches. Again, for those models the presence of negative patches in the training set does not appear decisive in solving the problem.

Last, recalling that our final aim is to detect the H.halys occurrence in images captured by the drone, we evaluate the impact of aggregating the results of the All Patches scenario by image. The experiment runs the detection on 33 full images. Each image consists of $6 \times 8 = 48$ patches because we do not consider the padded patches. Finally, we averaged the 33 values obtained. Notice that, we rely only on the YOLOv5, RetinaNet, and Faster-RCNN models since they have shown the best performance in the previous experiments (Section 6.1).

Figs. 14 and 15 show the average number of TP, FP, FN per image for each YOLOv5, Faster-RCNN, and RetinaNet model. The plots show on the x-axis the model considered, and on the y-axis the three estimators, i.e., TP, FP, FN, stacked.

Looking at Fig. 14, the number of FP stabilizes around 1.2 per image, with some values below this threshold, whereas the average FN sticks around 0.55 per image. Concerning the number of TP, every model computes almost 1 correct detection per image. Since the number of H.halys is particularly sparse (see Fig. 9), and we have more frequently one H.halys per image, we can conclude that the results distillation obtained by aggregating the results by image has boosted significantly the performance. Nonetheless, the models have reduced both their erroneous and missed detections.

Concerning Faster-RCNN and RetinaNet, the number of TP increases, stabilizing itself a 20% higher than YOLOv5 models. The number of FP per image experiences a notable variation within the models, from a minimum of 1.98 achieved by both FR^{50}_T, and RE^{50}_S, up to a maximum of 11.47 obtained by RetinaNet^0_T. Despite the large values in FP, the number returned by our analysis sticks comparable with the non-aggregated ones. The number of FN stabilizes around 0.9, determining a slight improvement concerning non-aggregated evaluation. In addition, it is worth noting that from the aggregated analysis is possible to observe a strong improvement determined by the injection of the 50% of patches without any bug. Indeed, the number of TP sticks stable to the values observed in both plots, whereas the number of FP and FN decreases significantly. This behavior sheds light on the impact of injecting patches without the H.halys during the training. In particular, the substantial drop in errors and misdetections clearly demonstrates the beneficial effect of a more tailored training process. In other words, the presence of negative samples inside the training set represented a step toward a better characterization of future predictions, helping those models to generalize better. In addition, averaging over the images mitigates the masking effect of errors/misdetections on metrics values.

In conclusion, the performance of the models improves in the presence of empty patches, especially when the results are aggregated per image. This finding is noteworthy and enriches the relevance of our achievements, as it aligns with our ultimate objective of identifying H.halys in real-time from the images taken by the drone in the orchard. Moreover, we have proved that the metrics for evaluating the model's performance strictly depend on the task you are facing and its complexity. Indeed, averaging the metrics per image has improved the comprehensibility of the results, providing a better strategy for more meaningful, broader, and more applicable metrics in the presence of a large number of background pixels, i.e., TN.

Next Section 6.3 delves into the relevant topic of occluded instances recognition, highlighting the ability of the models to detect tiny or partially visible H.halys.

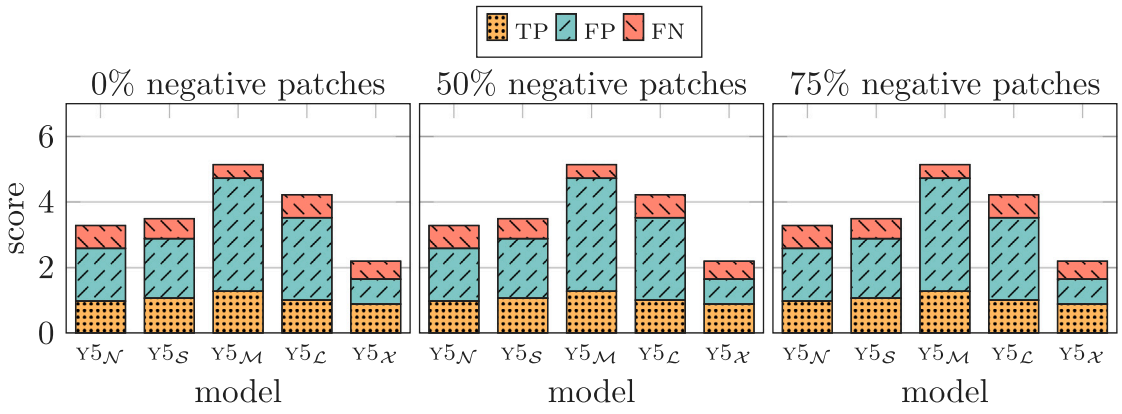

Fig. 14. Average TP, FP, and FN values aggregated by image for YOLOv5.

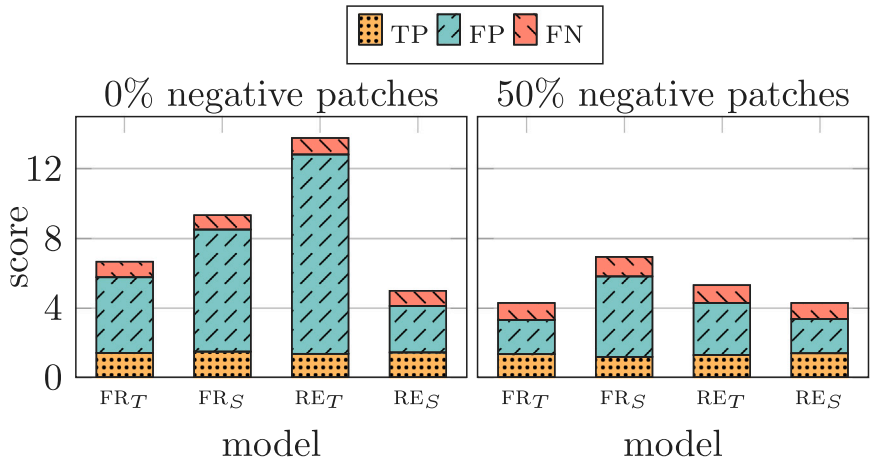

Fig. 15. Average TP, FP, and FN values aggregated by image for RetinaNet and Faster-RCNN.

6.3. Occlusion handling evaluation

In this section, we are investigating the models' sensitivity in detecting H.halys occurrence partially visible. The response against occluded objects commonly embodies a significant experiment in the context of object detection, especially when we move to object detection for insect monitoring. To measure such ability, we leverage the side effect of the image slicing, that is, we test the models on the H.halys cut between two patches during the slicing procedure. Specifically, 20 H.halys out of 71 are split between two patches, namely the 28% of instances. During the previous experiments the cut instances were included, though, we retain only H.halys whose sizes are at least half of their original dimension. Notice that, a random portion would have been selected if an instances were exactly halved between two patches. To measure the actual sensitivity to partially visible instances, we rely on the 20 H.halys discarded, i.e., the H.halys whose sizes were smaller than half of the their original size. Notice that, given that the size of the H.halys is extremely limited, thus the detection of an occluded instance appears challenging also for the human eye.

Fig. 16 shows the number of occluded H.halys correctly detected. In particular, the x-axis reports the model used for the detection, while the y-axis the number of instances detected. Notice that, we avoid the evaluation of YOLOv8 due to its poor performance (Section 6.1).

In principle, the models perform well finding at least the 10% of the challenging instances. When analyzing the behavior of YOLOv5 we observe that every model recognizes at least two instances. The best performing model is $Y5^0_{\mathcal{S}}$ able to detect six H.halys. The results confirm the conservative nature of YOLOv5 which is not prone to draw a lot of bounding boxes. However, it is possible to point out that all the models perform equally well handling occlusion.

Looking at Faster-RCNN and RetinaNet performance we can note that their optimism in predicting more bounding boxes results in more challenging H.halys being detected. Indeed, they stably found 3 out of 20 H.halys with a few exceptions. Both Faster-RCNN and RetinaNet gained the benefit of the TL in correspondence of higher percentage of negative patches, whereas, with a random initialization they tend to predict better with lower percentages of negative patches.

Despite every model differs for its own peculiarities, a motivation for these results can be found in the more complex nature of both the networks which predict with more "optimism" than YOLOv5.

In summary, in the recognition of occluded H.halys RetinaNet and Faster-RCNN showcased stabler performance than YOLOv5. The experiment unveiled a good ability to detect partially visible instances and within the context of this task achieved an appreciable ≈10% of detections.

6.4. Take away lessons: Upsides and downsides

While our trained detectors are the first to achieve notable performance in detecting the H.halys within aerial images, several challenges remain. Specifically, we identify two main limitations strictly related to our solutions. The resources required for training and detecting pose the first significant challenge. The models employed in our experiments are "desktop-oriented" and thus have substantial hardware requirements that made their portability on embedded system arguable. Future research should investigate the implementation of bespoke solutions, such as applying quantization to the current detectors, to ensure

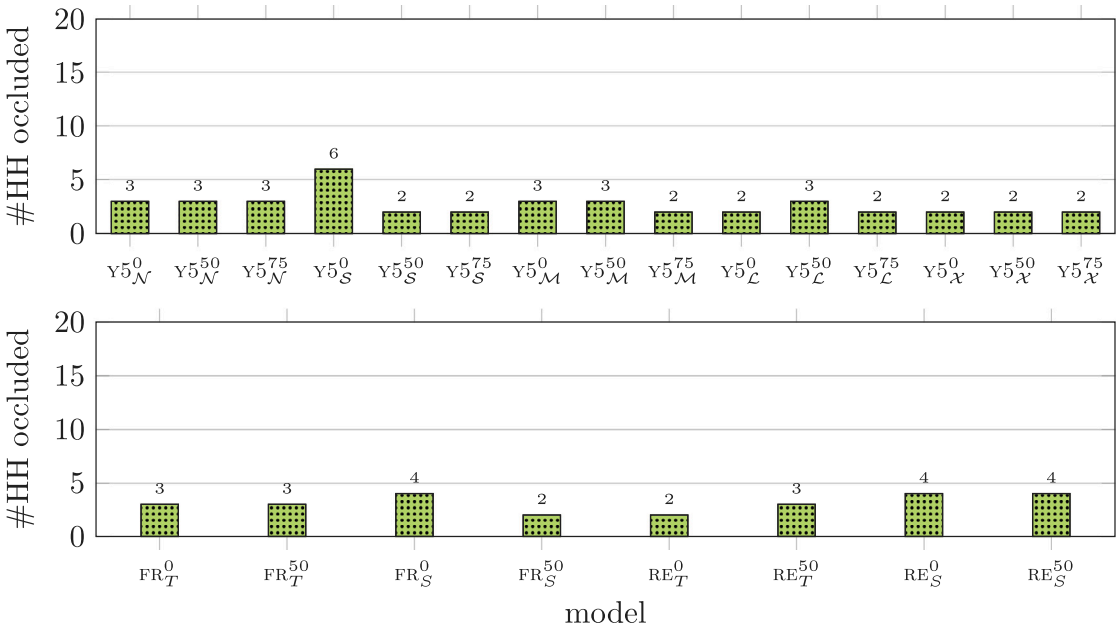

Fig. 16. Models performance evaluation on occluded H.halys.

that their resource footprint is compatible with common embedded system boards. Secondly, our solution exhibits lower performance in recall scores compared to precision. Several factors could contribute to this discrepancy. To address this issue, further improvements in recall scores may be achieved by integrating neural network attention modules (Guo et al., 2022). These modules can enhance the detectors' ability to extract salient features from images, thereby increasing their effectiveness in identifying the H.halys.

Our research is devoted to adults of H.halys. Namely, although the collected images employed for the training comprises different H.halys stages, apart from adults we observed a lack of samples for the rest of stages. As a result, the detectors have been tuned restricting to adult specimens. Despite the detectors exhibited the ability to other bugs that fell inside the scene, it would be interesting to extend the dataset range to other H.halys stages and fine-tuning detectors weights to handle the entire lifespan of the bug.

Moreover, we recognized that drone monitoring may suffer of "perspective curse". Indeed, we believe that due to the drone monitoring perspective the stream of images may miss H.halys instances that lay under the leaves or juvenile stages that do not fly. It is worthy to note that a similar issue may arise even with a human operator or a rover: they have a bottom perspective and lose the top one. For this reason, it is worth to investigate a statistical model that considers various parameters, such as time of year and temperature, and multiple perspectives.

7. Conclusions

Our research faces the significant challenge of H.halys infestations in orchards. We proposed, to the best of our knowledge, the first drone navigation protocol for creating a pioneering H.halys dataset, leveraging aerial telephotos autonomously captured , as well as guaranteeing a risk-free flight even in extreme cluttered crops. We develop ML models relying on that dataset for H.halys detection, showcasing solid performance. Specifically, the experimental evaluation demonstrates that the detectors' performance are comparable with the ones operating on images with a closer focus on the bug already proposed in the literature. We demonstrate the models' robustness against occlusions and background-only patches, emphasizing the efficacy of diverse training and slicing techniques. In conclusion, this work shows how to overcome the labor-intensive monitoring using the current technology. Performing a regular and constant monitoring, more data will be collected which pave the way for an informed decision-making system that will help reduce pesticide use, improve fruit quality, and improve farmers' bottom line.

As pointed out in Section 6.4, further research is needed to achieve fully autonomous orchard monitoring with potential extension to other invasive and emergent pests. It is worth developing a cloud application integrating bug detectors for real-time detection, to improve the monitoring system's portability. Additionally, integrating bug predictions with microclimate weather observations to construct a H.halys prediction model holds significant promise, offering valuable insights to the farmer for decision-making processes.

CRediT authorship contribution statement

Lorenzo Palazzetti: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Aravind Krishnaswamy Rangarajan:** Writing – review & editing, Writing – original draft, Validation, Software, Investigation, Data curation. **Alexandru Dinca:** Writing – review & editing, Writing – original draft, Software, Investigation, Data curation. **Bas Boom:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Methodology, Formal analysis, Conceptualization. **Dan Popescu:** Supervision, Methodology, Investigation, Formal analysis, Conceptualization. **Peter Offermans:** Writing – review & editing, Writing – original draft, Validation, Methodology. **Cristina M. Pinotti:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

Almstedt, L., et al., 2023. Technological Innovations in Agriculture for Scouting Halyomorpha Halys in Orchards. In: 2023 19th International Conference on Distributed Computing in Smart Systems and the Internet of Things (DCOSS-IoT), Pafos, Cyprus. pp. 702–709. http://dx.doi.org/10.1109/DCOSS-IoT58021.2023.00110.

Betti Sorbelli, F., Corò, F., Das, S.K., Palazzetti, L., Pinotti, C.M., 2022. Drone-based optimal and heuristic orienteering algorithms towards bug detection in orchards. In: 18th Intl. Conf. on Distributed Computing in Sensor Systems. DCOSS, IEEE, pp. 117–124.

Betti Sorbelli, F., Palazzetti, L., Pinotti, C.M., 2023. Preliminary results for *Halyomorpha halys* monitoring relying on a custom dataset. In: 2023 IEEE Workshop on Metrology for Agriculture and Forestry (MetroAgriFor). IEEE, pp. 1–6.

Betti Sorbelli, F., Palazzetti, L., Pinotti, C.M., 2023. YOLO-based detection of halyomorpha halys in orchards using RGB cameras and drones. Comput. Electron. Agric. 213, 108228. http://dx.doi.org/10.1016/j.compag.2023.108228.

Bottou, L., 2010. Large-scale machine learning with stochastic gradient descent. In: 19th Intl. Conference on Computational Statistics. COMPSTAT, Physica-Verlag, Paris, pp. 177–186.

Buslaev, A.V., Alex, P., Eugene, K., Iglovikov, V.I., Kalinin, A.A., 2018. Albumentations: fast and flexible image augmentations. CoRR abs/1809.06839.

Chen, C.-J., Huang, Y.-Y., Li, Y.-S., Chang, C.-Y., Huang, Y.-M., 2020. An IoT based smart agricultural system for pests detection. IEEE Access 8, 180750–180761.

de Oca, A.M., Flores, G., 2021. The AgriQ: A low-cost unmanned aerial system for precision agriculture. Expert Syst. Appl. 182, 115163.

Espinoza, K., Valera, D.L., Torres, J.A., López, A., Molina-Aiz, F.D., 2016. Combination of image processing and artificial neural networks as a novel approach for the identification of *Bemisia tabaci* and *Frankliniella occidentalis* on sticky traps in greenhouse agriculture. Comput. Electron. Agric. 127, 495–505.

Everingham, M., Van Gool, L., Williams, C.K., Winn, J., Zisserman, A., 2010. The pascal visual object classes (voc) challenge. Int. J. Comput. Vis. 88, 303–338.

Ferrari, V., et al., 2023. Evaluation of the potential of near infrared hyperspectral imaging for monitoring the invasive brown marmorated stink bug. Chemometr. Intell. Lab. Syst. 104751.

Grinberg, M., 2018. Flask Web Development: Developing Web Applications with Python. " O'Reilly Media, Inc.".

Guo, Y., Jia, X., Paull, D., Zhang, J., Farooq, A., Chen, X., Islam, M.N., 2019. A drone-based sensing system to support satellite image analysis for rice farm mapping. In: IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium. pp. 9376–9379.

Guo, M.-H., Xu, T.-X., Liu, J.-J., Liu, Z.-N., Jiang, P.-T., Mu, T.-J., Zhang, S.-H., Martin, R.R., Cheng, M.-M., Hu, S.-M., 2022. Attention mechanisms in computer vision: A survey. Comput. Vis. Media 8 (3), 331–368. http://dx.doi.org/10.1007/s41095-022-0271-y.

He, Y., Zhou, Z., Tian, L., Liu, Y., Luo, X., 2020. Brown rice planthopper (*Nilaparvata lugens* (stål)) detection based on deep learning. Precis. Agric. 21 (6), 1385–1402.

Ichim, L., Ciciu, R., Popescu, D., 2022. Using drones and deep neural networks to detect halyomorpha halys in ecological orchards. In: IGARSS 2022-2022 IEEE Intl. Geoscience and Remote Sensing Symposium. IEEE, pp. 437–440.

Jha, K., Doshi, A., Patel, P., Shah, M., 2019. A comprehensive review on automation in agriculture using artificial intelligence. Artif. Intell. Agric. 2, 1–12.

Jocher, G., et al., 2022. Ultralytics/yolov5: v7.0 - YOLOv5 SOTA realtime instance segmentation. http://dx.doi.org/10.5281/zenodo.7347926.

Li, W., Wang, D., Li, M., Gao, Y., Wu, J., Yang, X., 2021. Field detection of tiny pests from sticky trap images using deep learning in agricultural greenhouse. Comput. Electron. Agric. 183, 106048.

Lin, T.-Y., Goyal, P., Girshick, R., He, K., Dollár, P., 2020. Focal loss for dense object detection. IEEE Trans. Pattern Anal. Mach. Intell. 42 (2), 318–327. http://dx.doi.org/10.1109/TPAMI.2018.2858826.

Lin, T., Maire, M., Belongie, S.J., Bourdev, L.D., Girshick, R.B., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L., 2014. Microsoft COCO: common objects in context. CoRR abs/1405.0312. arXiv:1405.0312.

Mitra, A., Bera, B., Das, A.K., 2021. Design and testbed experiments of public blockchain-based security framework for IoT-enabled drone-assisted wildlife monitoring. In: IEEE INFOCOM 2021 - IEEE Conference on Computer Communications Workshops. INFOCOM WKSHPS, pp. 1–6.

Moreno, L., Ramos, V., Pohl, M., Huguet, F., 2018. Comparative study of multispectral satellite images and RGB images taken from drones for vegetation cover estimation. In: 2018 IEEE 38th Central America and Panama Convention. CONCAPAN XXXVIII, pp. 1–8.

Murugan, D., Garg, A., Ahmed, T., Singh, D., 2016. Fusion of drone and satellite data for precision agriculture monitoring. In: 2016 11th International Conference on Industrial and Information Systems. ICIIS, pp. 910–914.

Narvekar, N.D., Karam, L.J., 2011. A no-reference image blur metric based on the cumulative probability of blur detection (CPBD). IEEE Trans. Image Process. 20 (9), 2678–2683.

Padilla, R., Passos, W.L., Dias, T.L.B., Netto, S.L., da Silva, E.A.B., 2021. A comparative analysis of object detection metrics with a companion open-source toolkit. Electronics 10 (3), http://dx.doi.org/10.3390/electronics10030279.

Park, Y.-L., Cho, J.R., Lee, G.-S., Seo, B.Y., 2021. Detection of *Monema flavescens* (Lepidoptera: Limacodidae) cocoons using small unmanned aircraft system. J. Econ. Entomol. 114 (5), 1927–1933.

Redmon, J., Divvala, S.K., Girshick, R.B., Farhadi, A., 2015. You only look once: Unified, real-time object detection. CoRR abs/1506.02640. arXiv:1506.02640. URL http://arxiv.org/abs/1506.02640.

Rejeb, A., Abdollahi, A., Rejeb, K., Treiblmaier, H., 2022. Drones in agriculture: A review and bibliometric analysis. Comput. Electron. Agric. 198, 107017.

Ren, S., He, K., Girshick, R., Sun, J., 2017. Faster R-CNN: Towards real-time object detection with region proposal networks. IEEE Trans. Pattern Anal. Mach. Intell. 39 (6), 1137–1149. http://dx.doi.org/10.1109/TPAMI.2016.2577031.

Sava, A., Ichim, L., Popescu, D., 2022. Detection of halyomorpha halys using neural networks. In: 2022 8th Intl. Conf. on Control, Decision and Information Technologies. CoDIT, Vol. 1, IEEE, pp. 437–442.

Skalski, P., 2019. Make sense. https://github.com/SkalskiP/make-sense/.

Sorbelli, F.B., Palazzetti, L., Pinotti, C.M., 2023. A drone-based automated halyomorpha halys scouting: A case study on orchard monitoring. In: 2023 IEEE International Workshop on Metrology for Agriculture and Forestry. MetroAgriFor, pp. 380–385. http://dx.doi.org/10.1109/MetroAgriFor58484.2023.10424287.

Tansuriyavong, S., Koja, H., Kyan, M., Anezaki, T., 2018. The development of wildlife tracking system using mobile phone communication network and drone. In: 2018 International Conference on Intelligent Informatics and Biomedical Sciences. ICIIBMS, Vol. 3, pp. 351–354.

Trufelea, R., Dimoiu, M., Ichim, L., Popescu, D., 2021. Detection of harmful insects for orchard using convolutional neural networks. UPB Sci. Bull. Ser. C 83 (4), 85–96.

Valan, M., Makonyi, K., Maki, A., Vondráček, D., Ronquist, F., 2019. Automated taxonomic identification of insects with expert-level accuracy using effective feature transfer from convolutional networks. Syst. Biol. 68 (6), 876–895.

Ware, C., 2021. Chapter three - lightness, brightness, contrast, and constancy. In: Ware, C. (Ed.), Information Visualization (Fourth Edition). In: Interactive Technologies, Morgan Kaufmann, pp. 69–94.

Wen, C., Guyer, D., 2012. Image-based orchard insect automated identification and classification method. Comput. Electron. Agric. 89, 110–115.

Xie, C., Zhang, J., Li, R., Li, J., Hong, P., Xia, J., Chen, P., 2015. Automatic classification for field crop insects via multiple-task sparse representation and multiple-kernel learning. Comput. Electron. Agric. 119, 123–132.

Zou, Z., Chen, K., Shi, Z., Guo, Y., Ye, J., 2023. Object detection in 20 years: A survey. Proc. IEEE 111 (3), 257–276. http://dx.doi.org/10.1109/JPROC.2023.3238524.