The Pennsylvania State University

The Graduate School

DEVELOPMENT OF A UAV-BASED MULTI-DIMENSIONAL MAPPING FRAMEWORK FOR PRECISE FROST MANAGEMENT IN APPLE ORCHARDS

A Dissertation in

Agricultural and Biological Engineering

by

Wenan Yuan

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

May 2022

The dissertation of Wenan Yuan was reviewed and approved by the following:

Daeun Choi
Assistant Professor of Agricultural and Biological Engineering
Dissertation Adviser
Chair of Committee

Paul H. Heinemann
Professor of Agricultural and Biological Engineering

Long He
Assistant Professor of Agricultural and Biological Engineering

Dimitrios Bolkas
Associate Professor of Surveying Engineering

Suat Irmak
Professor of Agricultural and Biological Engineering
Head of the Department of Agricultural and Biological Engineering

ABSTRACT

As one of the main causes of weather-related damages in agriculture, frost leads to significant economic losses for farmers worldwide. Yet, traditional frost protection and temperature assessment methods in orchards remain rudimentary. Unmanned aerial vehicles (UAVs) and airborne sensing instruments emerged in recent years as promising tools for assisting efficient and convenient crop monitoring and management in precision agriculture, which creates opportunities in revolutionizing orchard frost protection approaches. With an overarching goal of building an autonomous cyber-physical system (CPS) consisting of UAV-based sensing and unmanned ground vehicle (UGV)-based heating for precise frost management, in this dissertation, a multi-dimensional mapping framework was proposed to process UAV-based thermal imagery, RGB imagery, and light detection and ranging (LiDAR) point cloud data to extract growth stage, canopy temperature, and tree structural information of an apple orchard. A thermal image stitching algorithm was developed to create high-resolution orchard temperature maps. A convolutional neural network (CNN)-based classifier was developed for detecting apple flower buds in RGB images, whose robustness against artificial image distortions and training dataset attributes were also investigated in depth. A UAV-LiDAR system was developed for identifying orchard regions that were unsafe for UGV travelling. The final output of the mapping framework, georeferenced orchard navigation maps, indicated both orchard heating requirements and orchard open space regions, which can potentially serve as a guide for UGV path planning and heat treatment application during frost events in future studies.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

This dissertation was not possible without Dr. Daeun Choi, Dr. Paul H. Heinemann, Dr. Long He, Dr. Dimitrios Bolkas, Omeed Mirbod, Tyler J. Reimer, Dr. Xiaomeng Shi, the undergraduate workers at the Smart Agriculture Laboratory, the farm crews at the Penn State Horticulture Research Farm, the ABE Department, Penn State, and State College.

This work was supported by the National Science Foundation and the USDA National Institute of Food and Agriculture under Accession #1018998 and Award #2019-67021-29224, and the USDA National Institute of Food and Agriculture Multistate Research under Project #PEN04653 and Accession #1016510. Findings and conclusions presented here do not necessarily reflect the views of these funding agencies.

CHAPTER 1
INTRODUCTION

Extreme weather conditions have always been a major reason for agricultural production loss [1]. With the consideration of global climate changes, it is expected that crop production will be more vulnerable due to the anticipated rise in the extent and frequency of weather extremes [2]. As one of the agroclimatic risks that are detrimental to crops, frost represents the main cause for weather related damage to crops [3]. Visible frost can form when water vapor in the atmosphere touches surfaces with temperatures below the dew point and freezing point, which often occurs in early spring or late fall because of a sudden cold air invasion or the radiative cooling of the earth's surface [4]. Crops can be damaged through freeze-induced dehydration and physical cell structure damage from ice crystals formed during frost events. In the United States, frost has caused more economic losses than any other weather-related hazards and often devastates local economies [3].

The degree of frost damage is dependent on many factors including rate of temperature drop, cloud and wind conditions, frosting duration, and crop growth stage [5]. Generally speaking, certain crop development stages are more vulnerable to cold, such as flowering or grain filling, and farmers should try to avoid the coincidence of such stages and frost incidences if possible. For fruit trees, even light frosts can greatly influence the fruit quality, while in extreme cases harvesting can be threatened by severe frost events. For example, during apple blossoming periods, temperatures that are a few degrees below freezing point are adequate to harm or kill apple flower buds [6], and damaged blossoms typically do not develop into fruits.

Frost damage in orchards can be avoided or minimized by active or passive protection methods. Examples of passive methods include site and plant selection, cold air drainage, irrigation, soil and plant covers, etc. [3]. They are typically less costly than active methods, yet often insufficient to protect crops under severe frost conditions. Active methods can be implemented right before or during a frost event to prevent formation of ice within plant cells. Sprinklers, wind machines, and heaters are commonly used equipment for active frost protection [7]. Overhead or under-tree irrigation protects trees from frost effectively through the heat of solidification of water, however it has disadvantages such as high installation cost, large water consumption, and causing broken branches due to ice overloading. Wind machines increase orchard temperature simply by bringing warm air down to ground level to prevent thermal stratification, and they typically have low operational costs. Yet wind machines do not work under cold, windy conditions. As one of the oldest methods, directly adding heat to air has the potential to replace substantial energy losses during frost events. However, a single heater cannot cover a large area, and the added heat will rise and dissipate to the sky. Fuel consumption, CO_2 emission, and overall expense can also be some of the additional downsides of the heating method. Distributing many small heaters throughout an orchard is a strategy for improving heating efficiency, and heat dissipation is less of a problem when a low-ceiling inversion layer exists since less volume of air needs to be heated.

Traditionally, thermometers were used for orchard temperature monitoring during frost events, which allows orchardists to adjust burning rates of orchard heaters, or turn on or off wind machines or sprinklers accordingly to keep a dry or wet-bulb temperature above the critical damage temperature depending on the frost protection method. However, such a technique of determining orchard frost treatment needs has a few noteworthy shortcomings. First, the amount of thermometers being used in an orchard is a limiting factor for how precise the orchard temperature can be managed spatially. Second, humans are required for checking thermometers, which can be laborious and untimely in large orchards. Third, plant tissue temperature may differ from air temperature, and readings of thermometers do not necessarily reflect true orchard frost treatment demands. Lastly, spatial plant growth stage variation in an orchard is typically ignored. It is wasteful to maintain the same temperature throughout an orchard, as critical temperatures vary with plant developmental stages [8]. Following the trend of precision agriculture [13], a modernized methodology of orchard temperature deficit or heating requirement determination for frost management that can assess localized, actual temperature increase needs while considering spatial plant growth stage variability and keeping human involvement to the minimum would be desirable.

Research regarding applications of unmanned aircraft systems (UASs) in agriculture exploded in recent years. A UAS consists of an unmanned aerial vehicle (UAV), ground control stations, data links, and other equipment such as sensors. With the rapid development of microelectronic, intelligent and radio technologies [9], it is believed that UASs will be an essential part of precision agriculture in the future [10]. A UAV can be made on a miniature scale, therefore it is significantly cheaper and more compact than manned aircrafts. The adaptability to topography and high maneuverability of a UAV allows low-flying altitudes and high spatial resolutions of sensor measurements. The aerial nature of UAVs, on the other hand, not only enables higher efficiency and flexibility in data acquisition, meaning an improved data temporal resolution, but also reduces the potentially destructive impacts on plants to the minimum. Such advantages make UASs an exceptionally good fit in large scale agricultural studies, and UASs have been successfully applied for various purposes such as field mapping, plant stress detection, plant biomass and nutrient estimation, weed management, etc. [11].

Generally, UAV-based sensing techniques collect information by remotely measuring reflected and emitted radiation from objects in different spectral regions, and commonly employed sensor types include red-green-blue (RGB) camera, thermal camera, and light detection and ranging (LiDAR) scanner. RGB imagers are sensitive to lights or electromagnetic waves in the visible spectrum (380 nm to 700 nm) and record information in the red, green and blue bands to reproduce colors, mimicking the mechanism of human visual perception. Because of its accessibility and affordability, RGB cameras have been long and widely deployed in various agricultural studies, such as analysis of seed morphology, germination rates, seedling vigor, leaf morphology, panicle traits, shoot biomass, root architecture, yield traits, etc. [12]. Thermal cameras, on the other hand, have the ability to gauge infrared radiation (700 nm to 1 mm) to estimate temperature and take area measurements instead of point measurements, which makes them a powerful tool for aerial temperature inspection. Although being generally expensive, in agriculture thermal cameras are often applied for assessing crop water status based on canopy temperature measurements [13]. LiDARs are a type of instrument that measures distances typically based on the time of flight (ToF) principle, commonly utilizing lasers with wavelengths in the near-infrared spectrum (750 nm to 1400 nm). Due to their capability of obtaining point clouds in real time, LiDARs are being widely adapted in agricultural studies for plant 3D structural reconstruction and considered as the next-generation plant phenotyping technology [14].

As the employment of computational algorithms, sensors, actuators and the integration of them, also known as cyber-physical systems (CPSs) [15] in agriculture is booming, frost protection in orchards can be much benefited from modern technologies in terms of monitoring and managing plants with ease, accuracy, precision, and efficiency. In combination with unmanned ground vehicle (UGV)-based actuating, such as heat application, UAV-based remote sensing demonstrates strong potential in being a critical component of a CPS that can assess crop statuses and complete management actions autonomously to minimize plant low temperature damages during frost events. In this dissertation, an effort was made towards establishing a UAV-based multi-dimensional data processing framework that supports such a CPS for frost management in an apple orchard. In Chapter 2, UAV-borne RGB camera and a convolutional neural network (CNN)-based object detector were utilized to evaluate flower bud growth stage variation and generate orchard growth stage maps. UAV-borne thermal camera was deployed to measure canopy temperature, and a thermal image stitching algorithm was developed to generate high-resolution orchard temperature maps. Orchard heating requirement maps were simulated by merging the growth stage maps and temperature maps. In Chapter 3, to gauge the robustness of the CNN algorithm used in Chapter 2 for apple flower bud classification under complex real life scenarios, the sensitivity of the algorithm regrading test image quality and training dataset attribute was investigated. In Chapter 4, a low-cost UAV-LiDAR system was developed and employed for collecting orchard point cloud data. Orchard obstacle and open space maps containing binary orchard regions that were either dangerous or safe for UGV navigation were further generated. The final orchard navigation maps were synthesized by fusing orchard heating requirement maps, obstacle maps, and open space maps, which can potentially guide autonomous UGVs in terms of path planning and heat application during frost events in apple orchards in future studies.

CHAPTER 2
ORCHARD HEATING REQUIREMENT MAP SYNTHESIS

2.1. Introduction

The fast development of UAVs and relevant remote sensing technologies in recent years brings new opportunities for their applications in agricultural management tasks [14]. High-resolution, remotely sensed crop data can be efficiently and conveniently captured through UAVs, which can be valuable for both farm management (e.g. crop monitoring, weed detection) [16] and agronomical research (e.g. plant phenotyping, plant trait modeling) [17]. Potentially, missions such as plant growth stage classification and plant temperature monitoring could be accomplished by UAV-based sensors such as cameras with ease and efficiency.

Aerial thermal cameras, which measure heat through infrared radiation, have been applied in research for a wide range of crops such as citrus [16], grape [17,18], almond [19], maize [20], soybean [21], and sugar beet [22]. Canopy temperature information is typically used for crop water status assessment as it is a function of available soil water and plant transpiration rate [23]; moreover, it has been reported to be correlated with other crop traits such as lodging [24], biomass [25], and yield [26]. Due to hardware limitations, e.g., field of view (FOV), monitoring large orchard temperatures with an airborne thermal camera may require a high flight altitude, resulting in a decrease in image spatial resolution and compromised image details [27,28]. Such an issue can be solved by transforming multiple overlapping images into one mosaic with well-established image stitching techniques [29], although thermal image stitching is a less popular topic than red, green, and blue (RGB) image stitching. For example, Teza [82] developed a MATLAB toolbox that was capable of registering thermal images for wall damage recognition using Harris corner detector. Wang et al. [51] reported a framework for stitching video frames from aerial thermal cameras based on SIFT. Sartinas et al. [83] proposed a robust video stitching method for a UAV forest fire monitoring utilizing ECC algorithm [84]. Besides using SIFT [85], Semenishchev et al. [86] also employed contour centers as keypoints for stitching quantized thermal images. In the agriculture domain, many researchers chose commercial software such as Pix4D (Switzerland) [48,74] or Agisoft Metashape (Russia) [75,79] for UAV thermal image mosaicking, which is convenient for inexperienced programmers and generally can produce robust results.

RGB cameras, on the other hand, play a crucial role in remote sensing since images in the visible spectrum represent agricultural environments well, which makes them ideal instruments for helping identify crop reproductive organs at different growth stages. Object detection is a common computer vision problem that deals with the localization and classification of target objects in images, while newly emerged CNN-based algorithms such as the region-based CNN (R-CNN) series (R-CNN [30], Fast R-CNN [31], Faster R-CNN [32], Mask R-CNN [33], Libra R-CNN [34], etc.) and you only look once (YOLO) series (YOLO [35], YOLOv2 [36], YOLOv3 [37], YOLOv4 [38], etc.) have achieved successes even in the agriculture domain. For instance, Grimm et al. [39] detected shoots, flower buds, pedicels, and berries of grape with a semantic segmentation framework; Chen et al. [40] adopted Faster R-CNN for identifying flowers and immature and mature fruits of strawberry; Koirala et al. [41] compared five CNN-based algorithms on classifying three mango panicle stages; Milicevic et al. [42] distinguished open and non-open flowers of olive based on a custom CNN; Ärje et al. [43] classified buds, flowers, wilted flowers, and seed pods of *Dryas integrifolia* using three CNN models; and Davis et al. [44] counted buds, flowers, and fruits of six wildflower species with Mask R-CNN. Despite the fact that several studies have utilized the deep learning approach for classifying plant developmental stages, it is still an underexplored topic as the difficulty of object detection tasks can vary greatly for different plants and environments depending on factors such as scene disorderliness, object positioning, and object occlusion [45], which remain unknown for many species such as apple.

In this study, the concept of UAV-based apple orchard temperature deficit determination for frost protection in a high resolution, low time cost fashion was demonstrated. Specifically, the objectives of the study were: (1) developing an algorithm that generates georeferenced orchard temperature maps based on thermal image stitching; (2) building a CNN-based classifier for apple flower bud growth stages and implementing the classifier to create orchard flower bud growth stage maps; (3) simulating orchard heating requirement maps in terms of how many degrees tree temperatures must rise for flower buds to not have damages during frost events based on the aforementioned maps.

2.2. Study Site, Equipment, and Data Collection

Field experiments were conducted in the Russell E. Larson Agricultural Research Center, Pennsylvania Furnace, Pennsylvania, USA. The apple orchard block was roughly 25 m long and

15 m wide (40.707918° N, 77.954370° W in WGS 84 datum). It was originally planted in 2008 and consisted of four rows and two cultivars including Jonagold and Daybreak Fuji, with 16 trees in each row and two rows for each cultivar. Tree spacing inside each row was approximately 1.7 m and row spacing was approximately 4 m. The trees were mechanically pruned into hedgerow and the average tree height was around 2.7 m.

The following considerations were kept in mind when deciding hardware suitable for the study and practical usage. First, the UAV needed to have a high payload to carry heavy instruments such as cameras while maintaining a reasonable hovering time. Second, aside from requiring that the thermal camera must have radiometric functionality, a high thermal image resolution could help create detailed orchard temperature maps. Last, the RGB camera should have an exceptional zooming capability to capture high definition images of apple flower buds, as the UAV must fly above a certain altitude to not cause turbulence near trees and prevent buds from being blurry in images. The final selection for the equipment included Matrice 600 Pro, Zenmuse Z30, and Zenmuse XT2 from DJI (China) (Figure 1), and their key specifications are listed in Table 1.

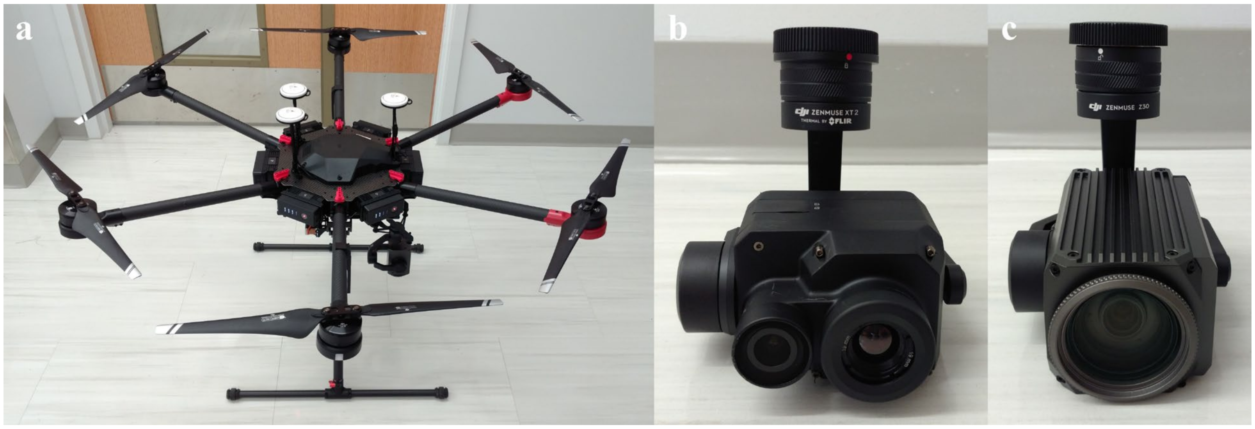

Figure 1. Hardware components employed in the study: (**a**) unmanned aerial vehicle (UAV), (**b**) thermal camera, and (**c**) red green blue (RGB) camera.

Table 1. Key specifications of hardware components employed in the study.

Hardware	Model	Specification
UAV	DJI Matrice 600 Pro (with TB47S batteries)	6 kg payload, 16 to 32 min hovering time, ±0.5 m vertical and ±1.5 m horizontal hovering accuracy, 8 mm focal length
Thermal camera	DJI Zenmuse XT2 (with a 19 mm lens)	−25 to 135 °C scene range, 7.5 to 13.5 μm spectral range, 32°×26° FOV, 640×512 resolution
RGB camera	DJI Zenmuse Z30	30× optical zoom, 63.7°×38.52° wide-end FOV, 2.3°×1.29° tele-end FOV, 1920×1080 resolution

Two types of data were collected in the growing season of 2020, including sets of thermal images covering the whole orchard for generating temperature maps and apple flower bud RGB images at various growth stages for the training and application of a CNN-based object detector. For thermal missions, flight altitude was set at 20 m above ground level (AGL) to mitigate parallax error [46], and the UAV flew at 1 m/s while the thermal camera captured images every 2 s with a 90° pitch. Approximately 100 thermal images were captured in 4 min during each fully autonomous flight and saved in tiff format, and the images had a 75% front overlap and a 70% side overlap. The thermal datasets used for demonstration were collected on June 2, 2020. The RGB images were captured with the RGB camera's full optical zooming capability, a fixed focus to tree canopy top, and a 90° camera pitch. Except for the first data collection date when a 10 m AGL flight altitude was adopted, the UAV flew at 15 m AGL with a manual speed no higher than 0.1 m/s to ensure image sharpness. A semi-autonomous RGB mission required 40 to 90 min to complete, when 600 to 1100 images in jpg format were captured, and the UAV's flying altitudes were maintained automatically by DJI's A3 Pro flight control system. RGB data collected on the 19th, 23rd, 28th of April, the 2nd, 7th, 13th, 16th, 21st of May, and the 24th of September, 2020 were used for model training. In 2021, 400 to 600 RGB images were captured at 15 m AGL in the same fashion as in 2020 on the 13th and 24th of April, and the 1st and 7th of May. The second-year RGB data were used to test the ability of the CNN-based classifier on recognizing apple flower buds from different growing seasons. RGB and thermal flight missions shared a similar flight path as shown in Figure 2.

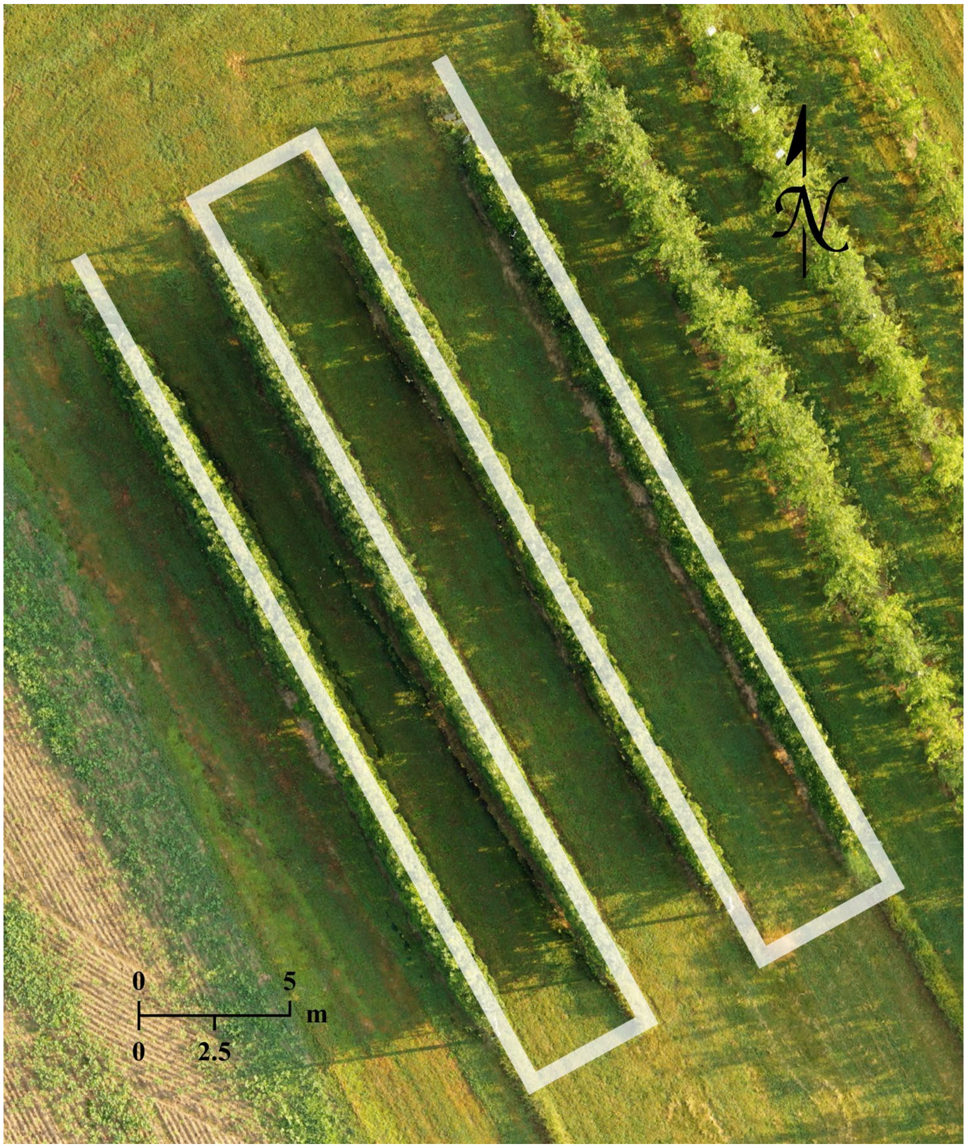

Figure 2. Flight path of the thermal and RGB flight missions conducted over the apple orchard.

2.3. Methods

2.3.1. Thermal Camera Radiometric Calibration

An aerial thermal camera needs to be calibrated before data analysis as its accuracy could be affected by factors such as target object properties and flight altitude [47]. Atmosphere-induced radiometric distortion can be considered negligible under low flight altitude changes [48]; therefore, object surface emissivity was the major variable that needed to be calibrated for in the study. As typical green leaves, such as apple leaves, have a thermal emissivity of 0.95 [49], and pure water has a similar emissivity of 0.97 to 0.99 in the spectral range of 8 to 13 μm with observing zenith angle in between 0° to 30° [50], a plastic container filled with tap water at various temperatures was used as the calibration target. The water temperatures were measured by a digital thermometer (9329H03, Thomas Scientific, USA) with an accuracy of ±0.2 °C. Thermometer readings of water were recorded once they stabilized, and thermal pictures were also taken by the UAV at a 20 m altitude with a 90° camera pitch mimicking a thermal flight mission (Figure 3). In total, 40 pairs of thermometer readings and thermal images were collected. Non-water pixels in each thermal image were first cropped out, and the mean of water pixel values was then computed and used for calibration. On average, each cropped image had approximately 200 water pixels. The relationship between pixel intensity and water temperature was estimated using robust least-squares regression with bisquare weights.

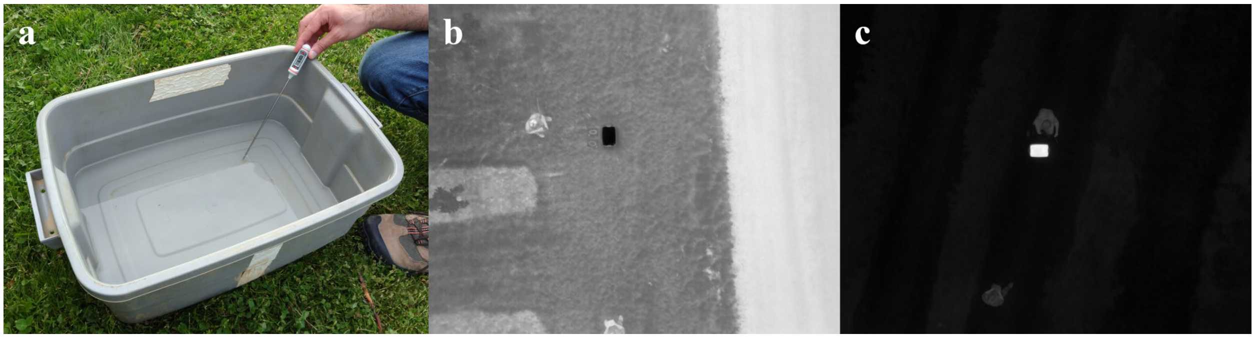

Figure 3. Illustration of the thermal camera calibration: (**a**) temperature measurement of the thermal camera calibration target with a digital thermometer; (**b**,**c**) example thermal images containing the calibration target before cropping taken at 10 °C and 75 °C water temperatures.

2.3.2. Thermal Image Stitching

The generation of an orchard temperature map requires individual thermal images to be first stitched into a mosaic. However, typically thermal images are more difficult to register than RGB images due to their lower resolutions and fewer image details [51]. A feature-based approach was adopted to address this issue. Instead of choosing commonly used algorithms such as SIFT [52] or SURF [53], BRISK [54] was implemented as the method for image feature detection and description. BRISK identifies image keypoints through the FAST detector [55] on octaves in an image pyramid and a non-maxima suppression by comparing a point of interest to its neighboring points on the same octave as well as corresponding points on the neighboring octaves. BRISK describes a circular sampling area of a keypoint in a binary manner with scale and rotation normalized. It has been reported that BRISK had a better computational efficiency and a comparable accuracy than SIFT for image registration [56].

The image stitching algorithm was developed in Python with OpenCV [57] and scikit-image [58]. Its general workflow is described as follows. For any pair of images taken consecutively, the BRISK keypoints and descriptors of the two images are computed. Pairwise Hamming distances [59] between descriptors are calculated exhaustively, and each query descriptor is considered to have a match when the distance ratio between the closest and the second closest neighbor is smaller than 0.8 as suggested by Lowe [60]. The transformation between two image planes is assumed to be 2D Euclidean considering the images were taken with fixed UAV flight altitude and camera pitch. All pairwise homographies are estimated through RANSAC [61] within 100 iterations, where each random data subset contains four pairs of matches, and the maximum distance for a keypoint to be classified as an inlier is set as two pixels. Anchor homographies are calculated by pairwise homography multiplication. Image stitching is achieved by warping non-anchor images based on their anchor homographies and overlaying them to the anchor image (Figure 4). The intensities of all overlapping pixels are averaged in the stitched mosaic.

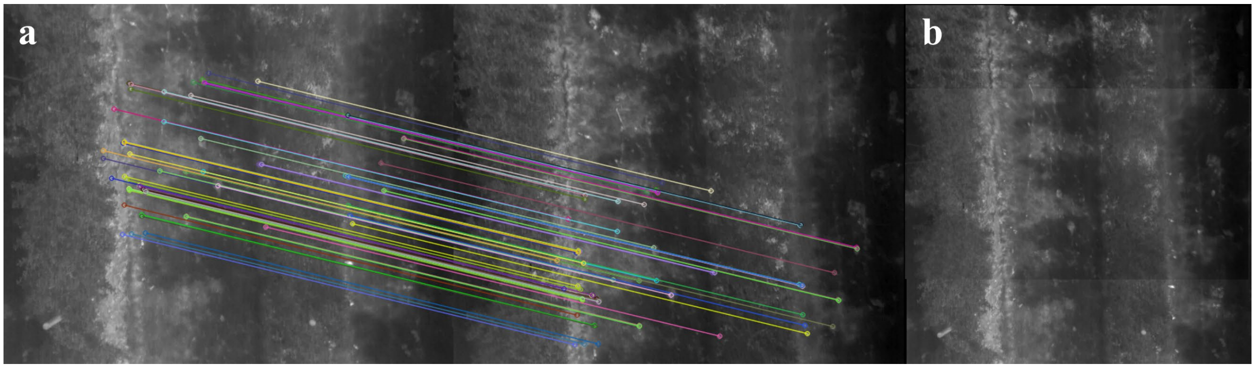

Figure 4. An example of stitching two consecutive thermal images taken over the orchard showing a section of two rows of apple trees: (**a**) inliers of the matched BRISK keypoints between two images and (**b**) the stitching result.

2.3.3. Thermal Mosaic Georeferencing

Three plastic saucers filled with room temperature water were used as ground control points (GCPs) for georeferencing thermal mosaics, which appeared as circles with relatively uniform pixel intensities in the thermal images hence were easy to detect. The saucer diameters varied from 0.46 to 0.63 m, and the size difference allowed them to be later identified uniquely during image processing. The saucers were placed around the orchard edge, and their GPS coordinates were measured by a pair of global navigation satellite system (GNSS) modules (Reach Module, Emlid, China) with single-band antennas (TW4721, Tallysman, Canada). As the outputs of GNSS modules always carry a certain level of noise, to achieve better location estimations, for each saucer 900 to 1000 latitude and longitude readings of the saucer center were recorded at a 5 Hz GPS update rate, and the median values were taken as the saucer center coordinates to avoid potential outliers. Preliminary tests showed that the saucer GPS measurements attained an accuracy of ±0.13 m. 2–1 Hough transform [62] was implemented in Python as the method for saucer detection. During georeferencing, pre-measured GPS coordinates were assigned to the center pixels of detected saucers (Figure 5), and least-squares fitting-based first-order polynomial transformation was utilized for converting rows and columns of raster thermal mosaics to coordinate maps.

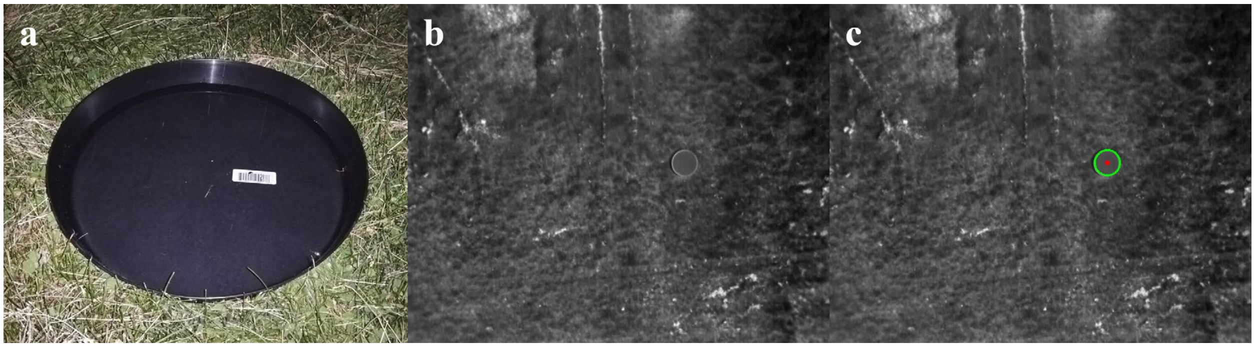

Figure 5. A ground control point (GCP) for georeferencing thermal mosaics: (**a**) in the orchard, (**b**) its appearance in a thermal image, and (**c**) being detected by the algorithm.

2.3.4. Flower Bud Growth Stage Classifier Development

Apple flower buds are usually considered to have nine growth stages including silver tip, green tip, half-inch green, tight cluster, first pink, full pink, first bloom, full bloom, and petal fall [63]. Since the transition of buds from one growth stage to another is a gradual process, it could be very difficult to distinguish certain growth stages from one another in the RGB images. As the neighboring stages have either identical or close critical temperatures [63], a six-class system was created for the apple flower bud classification that merged certain growth stages (Table 2). Since the critical temperatures of Jonagold and Daybreak Fuji remain unknown in current literature, Red Delicious' critical temperatures were adopted for both apple varieties as a reference [63]. Note that, aside from growth stage, plant variety also has an influence on plant cold tolerance, and one should consider vegetation heterogeneity for the most precise orchard temperature management. As a clarification, in this study a flower "bud" refers to the complete structure of what a flowering tip develops into, such as a flower cluster with several flowers and numerous leaves; a "flower" refers to an unbloomed pink flower in pink stage, a bloomed white flower in bloom stage, or a petal-less flower in petal fall stage.

Table 2. Critical temperatures (10% kill) in Celsius of the six apple flower bud growth stages adopted in this study, which are defined as the temperatures that buds will endure for 30 minutes or less without injury [12].

Growth stage	Tip	Half-inch green	Tight cluster	Pink	Bloom	Petal fall
BBCH-identification code [64]	01–09	10–11	15–19	51–59	60–67	69
Critical temperature	−8.89	−5.00	−2.78	−2.22	−2.22	−1.67

Typically, CNN-based models rely on supervised learning, meaning "ground truths" need to be prepared by experts such as labels of target objects in an image. However, there were several factors that posed challenges to the image annotation process. First, as mentioned above, the development of buds happens in a gradual manner, and buds oftentimes do not possess enough distinct characteristics to be classified in a certain growth stage, which adds difficulties to the decision-making during bud labeling. Second, because of the training systems adopted for the orchard, apple trees can have a dense 3D distribution of buds and flowers that grow closely to each other, which creates confusing scenes for humans to identify a complete flower bud in 2D images. Third, vegetative buds can be mistaken as flower buds [65], especially the ones before tight cluster stage. Fourth, an apple flower bud grows from a tiny tip into a flower cluster containing multiple flowers and leaves, which involves a significant size change. Identifying flowers and leaves that belong to the same bud could be demanding and impractical. Last, as the RGB images were taken by an aerial camera observing a small area, any slight movements of the trees or the UAV could lead to reduced image quality. To address these issues, several criteria were determined as the guidance for bud labeling in various scenarios:

- For stages at or before tight cluster, include the whole bud in a bounding box. For stages at or after pink, exclude leaves from a bounding box (Figure 6).
- For stages at or before tight cluster, each bounding box should contain only one bud (Figure 6a–c). For pink or petal fall stage, if the flowers that belong to the same bud are recognizable and relatively close to each other, label all flowers in the same bounding box (Figure 6d,g); otherwise, label each flower with a bounding box (Figure 6e,h). For bloom stage, each bounding box should contain only one flower (Figure 6f).
- When knowing a bud or flower exists but the complete shape of it cannot be identified, due to image blurriness or dense flower bud distribution, do not label.
- When having doubts whether a bud is a leaf or flower bud, do not label.

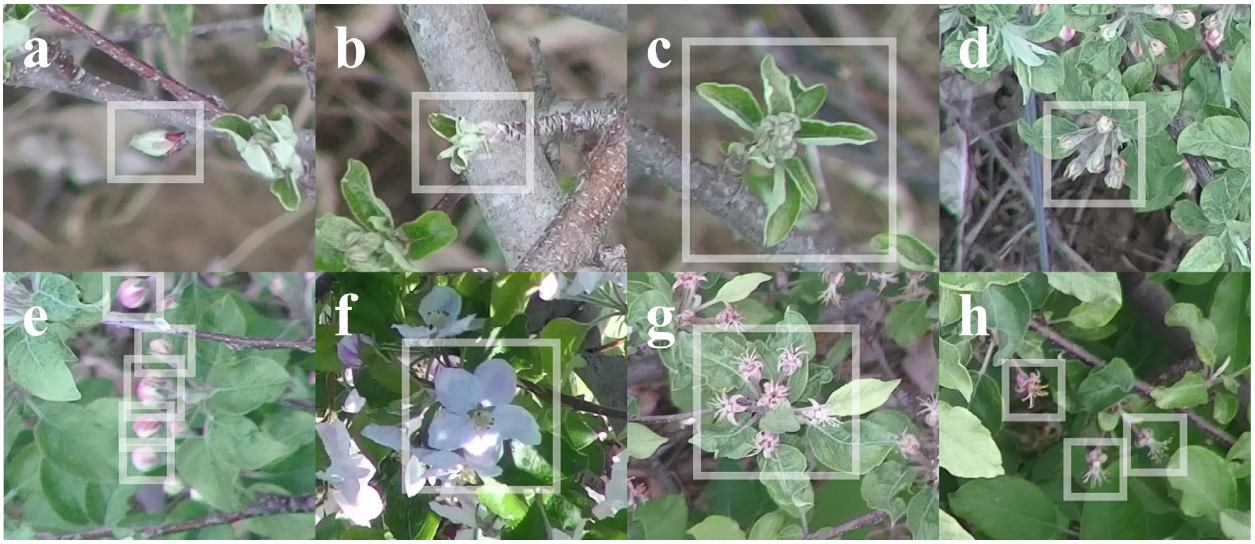

Figure 6. Examples of apple flower bud annotations at various growth stages: (**a**) tip, (**b**) half-inch green, (**c**) tight cluster, (**d**,**e**) pink, (**f**) bloom, and (**g**,**h**) petal fall.

YOLOv4 [38], a state-of-the-art CNN-based object detector, was deployed for the apple flower bud growth stage classification task. It is capable of operating in real-time on a conventional GPU and has been demonstrated to have a high performance on the Microsoft COCO dataset [66]. The model architecture consists of a CSPDarkNet53 [67] backbone with Mish activation [68], a neck of SPP [69] and PANet [70] with Leaky activation, and a YOLOv3 head. To improve class balance, instead of using all the collected images, 450 positive samples containing at least one bud or flower were arbitrarily selected from each of the eight datasets collected in April and May, 2020 for labeling. The image annotations were prepared in Darknet format using an open source tool YoloLabel [71] by a two-stage process, where the images were first labeled by five trained image annotators, and then double-checked and relabeled by only one annotator for unlabeled buds, mistaken labels, and labeling style inconsistency. Annotated images of each date were split into 70%, 20%, and 10% segments for model training, validation, and test, respectively. An equal amount of negative samples containing no bud or flower, which were captured mostly in September, 2020, were also added to the training dataset to improve model robustness. In total, the training, validation, and test datasets consisted 5040, 720, and 360 images, respectively.

The classifier trainings were executed on a computer with an Intel® Core™ i9–9900 (USA) CPU, an NVIDIA GeForce RTX 2060 (USA) GPU, and a 16 GB RAM. Three network sizes were experimented with: 320, 480, and 640. Except for learning rate, which was manually tuned between 0.002 and 0.00001 during training to prevent overfitting, all other hyperparameters were adopted as default. Average precisions (APs) of a class were calculated based on 11-point interpolated precision–recall curves [72], while the trainings were stopped when the mean average precisions (mAPs) of the validation dataset no longer improved. Model weights with the highest validation mAPs were selected for further evaluation on the test dataset.

2.3.5. Flower Bud Classifier Second-Year Test

To test the generalizability of the developed apple flower bud classifier on multi-year data, the RGB images collected in 2021 were used as test datasets to evaluate the performance of the best classifier trained with the 2020 datasets. However, manually annotated ground truths were no longer adopted as the evaluation method for two reasons. First, based on the experience from 2020, manual image labeling is extremely time-consuming, inefficient and prone to error. Second, for the purpose of determining orchard regional heating requirements, building a perfect model that can detect every single flower bud in an image is unnecessary, since each RGB image only looks at a small area (e.g. 0.5 m × 0.3 m) and flower buds that are close to each other often have similar growth stages.

A random sampling-based evaluation method was proposed as an alternative but more efficient way of assessing the classification precisions of the model for the second-year RGB datasets. The method was designed to quickly check two aspects of the flower bud detection results: whether at least one flower bud was detected in each image by the classifier, and whether a randomly sampled detection result in an image was correct based on manual inspection. After applying the classifier and checking the detected flower numbers in all images collected in 2021, 100 images where randomly selected from each of the four data collection dates, and one detected flower bud was randomly sampled in each of the selected images for further manual inspection (Figure 7). Both flower bud detection number checking and random image and flower bud sampling were completed using programs written in MATLAB 2021a (USA).

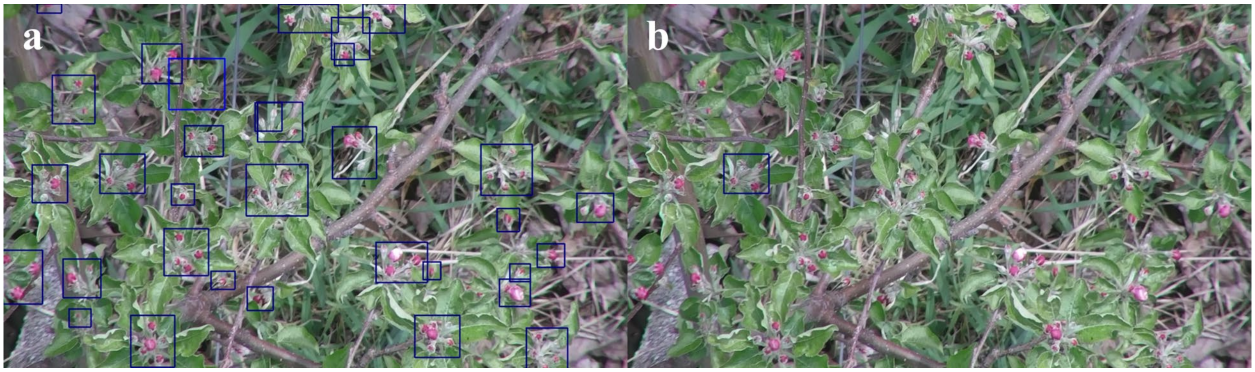

Figure 7. An example of randomly sampling a second-year RGB image for manual inspection on the growth stage classification accuracy by the best apple flower bud detector trained with the first-year data: (**a**) all detected flower buds by the classifier; (**b**) a randomly sampled flower bud.

2.3.6. Flower Bud Location Calculation

Every RGB image collected in the study was geotagged by DJI's A3 Pro flight control system. For mapping purposes, the GPS coordinates of each image were utilized for georeferencing the flower buds in the image detected by the classifier. As the camera was fixed at a front position relative to the UAV's GNSS units along the drone heading direction, the GPS coordinates of an image represented a point at a rear position relative to the image center. Considering the small offset between the GNSS units and the RGB camera (approximately 4-27 cm) as well as the narrow tele-end FOV of the camera, to simplify flower bud location calculation, the GPS coordinates of an image were assumed to represent the location of the image bottom center point (Figure 8). Based on the relative position of a bud in an image, the latitude and longitude of the bud were calculated in Python using the following formulas [73] ignoring Earth's ellipsoidal effects:

$$\varphi_2 = \mathrm{asin}(\sin(\varphi_1) \cdot \cos(\delta) + \cos(\varphi_1) \cdot \sin(\delta) \cdot \cos(\theta)), \tag{1}$$

$$\lambda_2 = \lambda_1 + \mathrm{atan2}(\sin(\theta) \cdot \sin(\delta) \cdot \cos(\varphi_1), \cos(\delta) – \sin(\varphi_1) \cdot \sin(\varphi_2)), \tag{2}$$

where φ_1 is the latitude of an image bottom center, λ_1 is the longitude of an image bottom center, φ_2 is the latitude of a bounding box center, λ_2 is the longitude of a bounding box center, θ is the bearing from image bottom center to bounding box center, and δ is the angular distance between image bottom center and bounding box center. θ is determined by subtracting the "zenith" angle α from the drone heading. If θ is larger than 180°, an additional 360° needs to be subtracted to keep θ in the −180° to 180° range. On average, the drone heading was either 151.33° or −28.67° during RGB flight missions depending on the tree row. α ranges from −90° to 90°. δ is calculated as d/R, where d is the distance between image bottom center and bounding box center, and R is Earth's average radius being 6,371,000 m. Given UAV flight altitude, average tree height, image resolution, and camera FOV, the area of each RGB pixel was computed to be approximately 6.60×10^{-8} m^2, and d is further derived from the pixel numbers in between image bottom center and bounding box center (Figure 8).

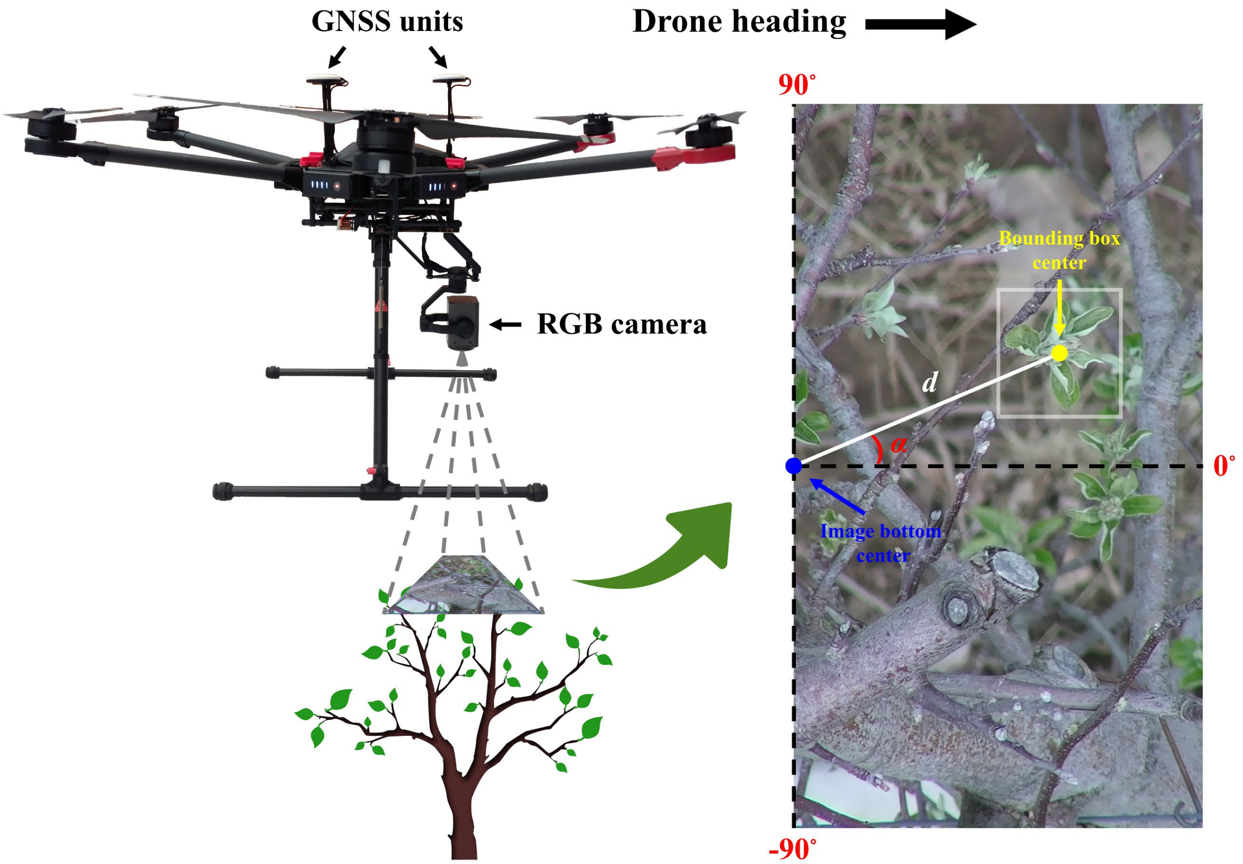

Figure 8. Schematic diagram illustrating GPS coordinate calculation of a detected bud in an RGB image.

2.3.7. Regional Heating Requirement Determination

Given detected bud coordinates and a georeferenced thermal mosaic, apple flower buds at various growth stages can be mapped as pixels with unique colors onto the coordinate map, representing different critical temperatures. Orchard heating requirement maps could be calculated by simply subtracting temperature maps from critical temperature maps, where pixels indicate the heating needs at detected bud locations. Yet, such maps will not be viewing-friendly and practical for usage, as heating requirements of individual buds are not directly helpful for orchardists to determine heating treatments of a region. To solve this issue, heating requirement maps were simplified by dilating heating requirement pixels and "prioritizing" pixels with higher intensities representing higher heating demands (Figure 9), which allowed humans to conveniently see the highest possible heating needs of a region. Each detected bud was initially mapped as one pixel before dilation. As during the study a thermal dataset during a frost event when tree temperatures were below critical temperatures was never collected, heating requirement maps were simulated by employing a temperature map and a flower bud growth stage map from two dates, and artificial critical temperatures to demonstrate the concept.

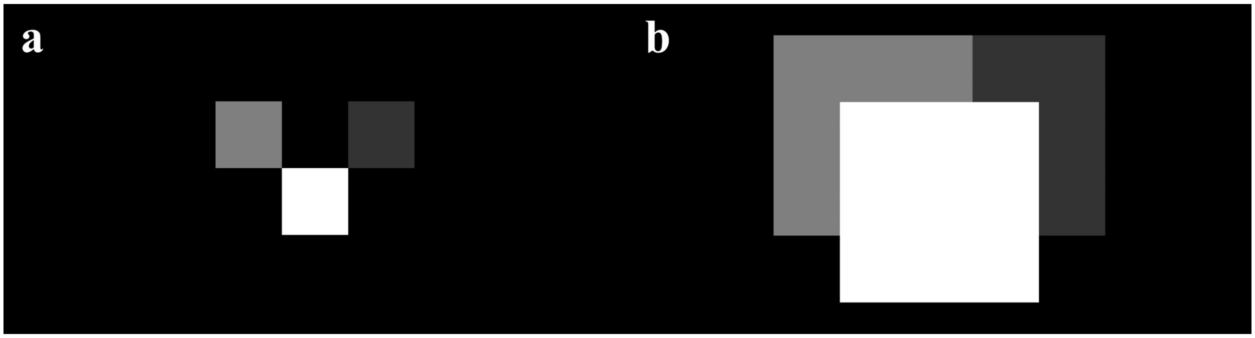

Figure 9. A simplified heating requirement map containing three heating requirement pixels of three different apple flower buds, where brighter pixels represent higher heating demands: (**a**) before dilation and (**b**) buds with higher heating requirements are prioritized after dilation by overwriting small pixel values with larger pixel values in the overlapping area.

2.4. Results and Discussion

2.4.1. Thermal Camera Calibration Results

Regression analysis of thermal pixel intensity and ground truth temperature resulted a strict linear relationship with an R^2 of 0.99 and an RMSE of 1 °C (Figure 10), which is similar to what many relevant studies obtained for airborne thermal imagers, such as R^2 of 0.98 and RMSE of 1.53 °C for ICI 8640 P-Series (USA) [74], R^2 of 0.98 and RMSE of 1.9 °C for ICI 9640 P-Series [75], R^2 of 0.93 and RMSE of 0.53 °C for senseFly thermoMAP (Switzerland) [76], and R^2 of 0.96 [48] and 0.937 [47] for FLIR Vue Pro R 640 (USA).

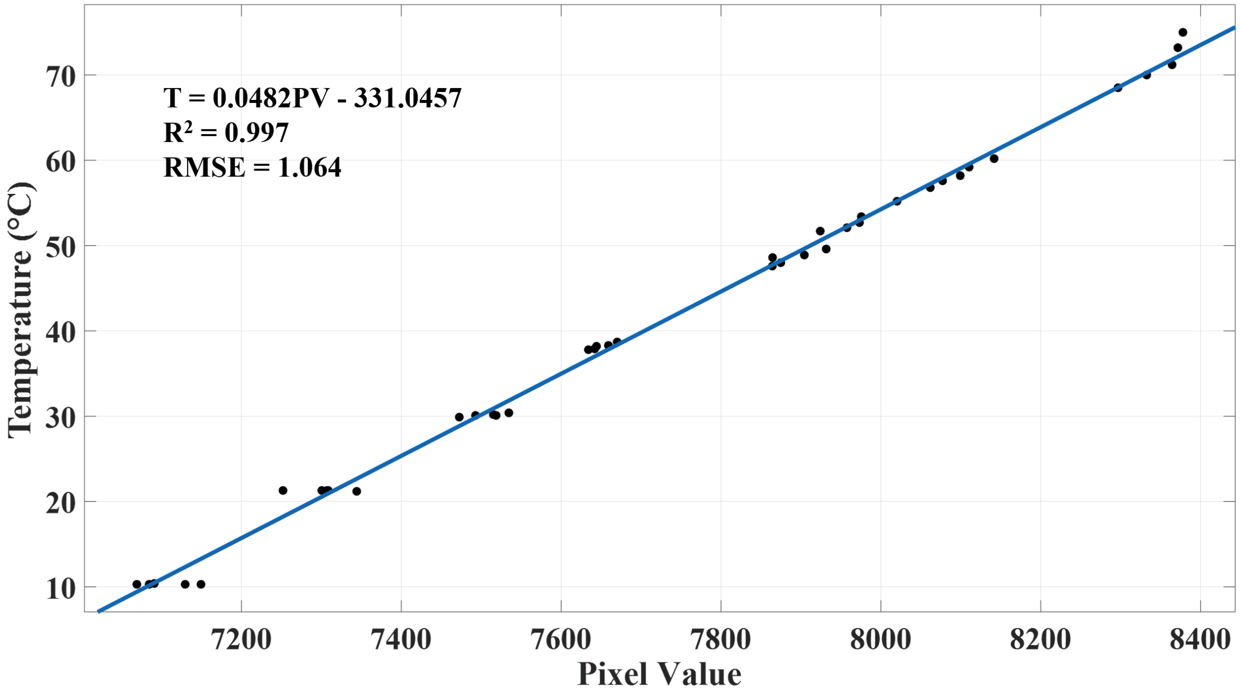

Figure 10. Calibrated relationship between thermal pixel intensity and temperature based on least-squares regression.

Generally, thermal cameras can be divided into two categories: cooled and uncooled [77]. Modern drone thermal cameras such as the ones mentioned above are typically equipped with uncooled detectors such as microbolometers [78]. Uncooled cameras have advantages such as light weights, small sizes, and low prices, which make them an ideal choice for many UAV applications [79]. However, they have higher thermal time constants and are known to suffer from thermal noise induced by changes in camera body temperature [80]. Although being expensive, cooled thermometers usually have greater magnification capabilities, better sensitivities, and are able to capture sharp images under short integration time. As a thermal camera can be interfered with by its surroundings, thermal calibration results should be utilized as mission-specific when significant environmental changes are involved, such as air temperature, humidity, and flight altitude [81]. Nevertheless, the capacity and accuracy of an uncooled thermal camera are sufficient for the application of orchard temperature monitoring, while cooled thermal cameras seemingly have the potential to be a superior option when future technology matures.

2.4.2. Thermal Mosaic and Orchard Temperature Map

Running on the same computer for the classifier training, the stitching algorithm was able to generate a mosaic from a dataset of 100 thermal images within 30 s on average. The stitching results were satisfactory as all apple trees could be clearly distinguished from background, although certain levels of blurriness and misalignment existed due to parallax effect and lens distortion. For demonstration purposes, only a minimum amount of GCPs was employed, while higher overall accuracy of coordinate maps can be achieved with more GCPs, allowing complex image distortion to be corrected with higher order transformations. Generally, sunlight helps robust stitching by creating shadows and improving image contrast, as the temperature difference between plants and soil increases under sun exposure, and more distinctive image features can be detected and matched. Conversely, rapidly changing weather conditions can bring challenges to the stitching process when temperatures of the same objects differ significantly in two images. Although sunny weather is a favorable condition, the absence of direct sunlight did not cause difficulty to image stitching, and thermal images taken at night-time and cloudy daytime were expected to share similar appearances. On average, 976 BRISK keypoints were detected in each image from the cloudy dataset (Figure 11c) and 341 were matched in an image pair. In contrast, the two numbers were 1162 and 428 for images from the sunny dataset (Figure 11a). The customized algorithm has the advantage of being flexible in modifying its components. For example, the BRISK feature can be easily changed into newer state-of-the-art image feature detectors and descriptors. It is possible to improve the final stitching results by applying techniques such as lens distortion correction [87], bundle adjustment [29], and blending [51]. These steps were excluded to keep the algorithm as simple and fast as possible, and not to make unnecessary pixel value manipulations hence preserving original temperature measurements.

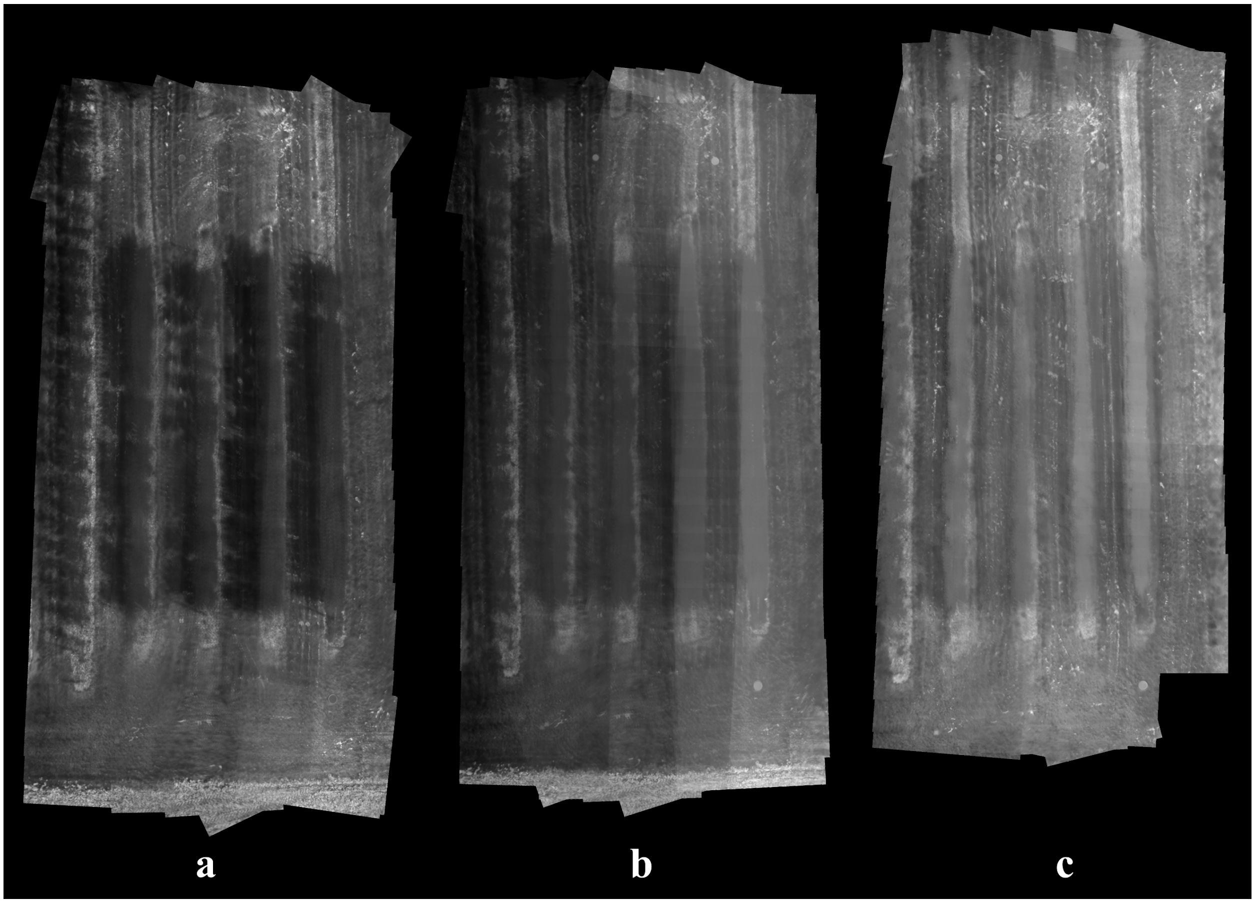

Figure 11. Stitched thermal mosaic examples of the apple orchard under different weather conditions: (**a**) mostly sunny, (**b**) partially sunny, and (**c**) mostly cloudy.

Utilizing the calibrated pixel value-temperature relationship, viewing-friendly maps containing orchard temperature information were generated from thermal mosaics at a negligible time cost as shown in Figure 12. Each pixel in the maps covers an area of 3.23×10^{-4} m^2 approximately. Note the maps reflect orchard temperature changes in a 4-minute window. Under a partially sunny condition, certain regions of a map can appear substantially darker or brighter than the rest (Figure 11b).

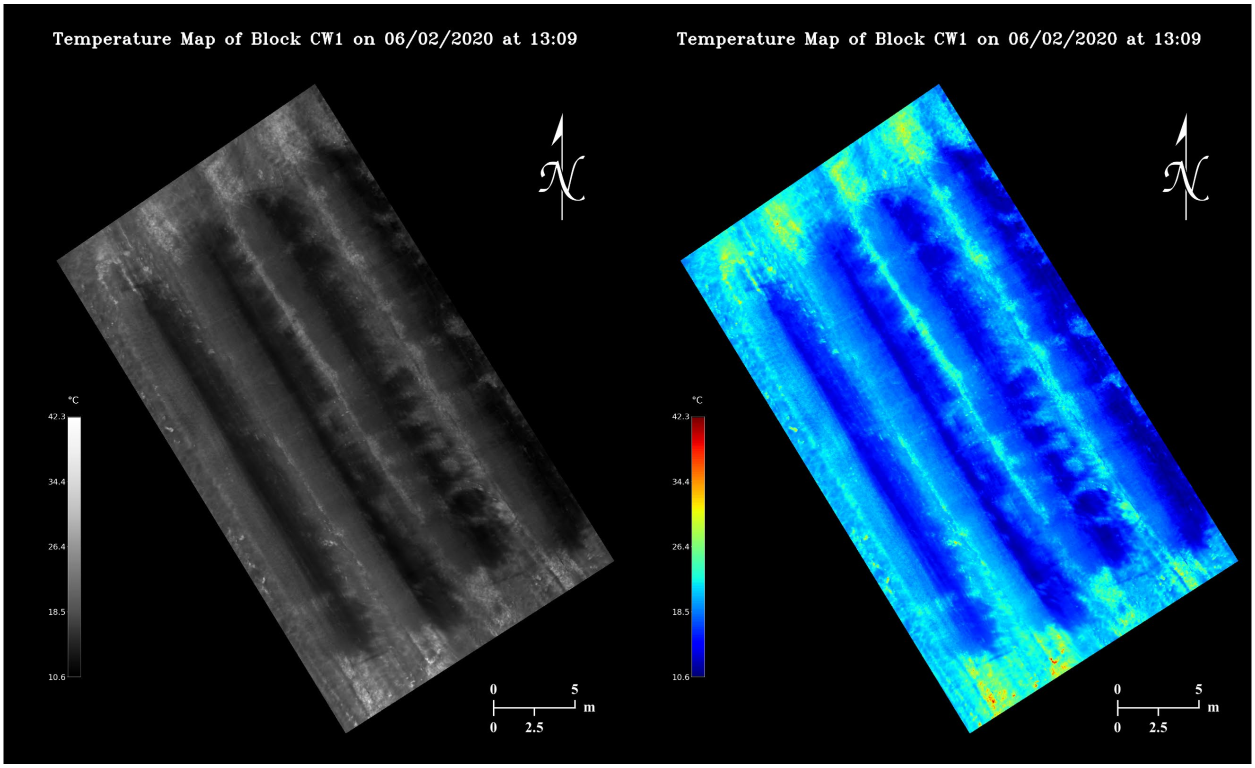
Temperature Map of Block CW1 on 06/02/2020 at 13:09
N
°C
42.3
34.4
26.4
18.5
10.6
0
5
m
0
2.5
Temperature Map of Block CW1 on 06/02/2020 at 13:09
N
°C
42.3
34.4
26.4
18.5
10.6
0
5
m
0
2.5

Figure 12. Examples of grayscale and false-color apple orchard temperature maps generated from the same thermal mosaic.

Flight time and stitching time are two important factors that govern how timely orchard temperature can be managed using the proposed methodology. Currently image collection and algorithm execution together could cause a minimum 5-minute delay from real time temperature monitoring. As the computation simplicity was already prioritized when designing the stitching algorithm, optimizing UAV flight missions has more potential in improving map generation speed, which essentially is a trade-off between orchard temperature monitoring resolution and timeliness. At a higher altitude, faster UAV flying speeds can be adopted as a larger area will be covered in each image, and motion blur would be less problematic. The maximum speed of the UAV is up to 15 m/s, with which a thermal flight mission could be completed within 30 s. As trees growing next to each other tended to have similar temperatures (Figure 12), much lower temperature map resolutions, for instance, even if each thermal pixel covers a whole tree, could still be potentially sufficient for determining orchard regional heating needs, in which case only a few images need to be taken at a high altitude to minimize flight time in the future.

2.4.3. Flower Bud Growth Stage Classifier Performance

For each of the three experimented network sizes, APs and mAPs of the best model at 50% intersection over union (IoU) on the validation and test datasets are listed in Table 3. Out of the three network sizes, 320 had the lowest mAP, while 480 and 640 had comparable mAPs, indicating image input resolution higher than 480 might not bring additional useful information to the network training. AP seemed to have a direct correlation with object size. Out of the six stages, tip and half-inch green consistently had the lowest APs, when apple flower buds were the most underdeveloped and had the smallest sizes; pink had considerably lower APs than tight cluster and bloom, while generally it also had smaller bounding boxes as leaves were excluded in pink but included in tight cluster, and flowers were closed in pink but open in bloom; from bloom to petal fall, APs decreased significantly as the absence of petals led to smaller bounding boxes of petal falls than bloom. Due to a smaller sample size, the test dataset showed substantially larger APs of tip and half-inch green than the validation dataset, indicating larger sampling biases.

Table 3. Performance of three apple flower bud growth stage classifiers with different network sizes at 50% intersection over union (IoU) on the validation and test datasets.

Statistics		**Network size**					
		320×320		480×480		640×640	
		Validation	Test	Validation	Test	Validation	Test
AP	Tip	31.74%	51.65%	48.34%	65.72%	50.08%	61.72%
	Half-inch green	39.48%	50.46%	48.25%	57.08%	45.95%	56.68%
	Tight cluster	85.07%	86.98%	85.50%	87.65%	82.83%	85.48%
	Pink	71.41%	69.29%	72.83%	71.79%	71.50%	70.18%
	Bloom	81.54%	81.33%	84.58%	84.52%	84.12%	83.49%
	Petal fall	57.34%	56.75%	63.46%	62.68%	63.69%	62.88%
mAP		61.09%	66.08%	67.16%	71.57%	66.36%	70.07%

Deep learning-based apple flower bud growth stage classification has never been studied in existing literature; however, attempts for CNN-based apple flower detection have been made previously. Tian et al. [88] proposed a modified Mask Scoring R-CNN model for classifying three types of apple flowers based on their morphological shape, including bud, semi-open, and fully open. Trained on a dataset of 600 images, out of which 400 were manually collected and 200 were generated through image augmentation, their model achieved a 59.4% mAP on a test dataset of 100 images. Wu et al. [89] identified flowers, or rather stamens, of three apple varieties with a simplified YOLOv4 network. With manually collected 1561 images for training and 669 images for testing, they obtained a 97.31% AP for flower detection.

Despite the similarities in nature, the current study possess a higher complexity than the two studies reviewed above because of two important factors: data temporal coverage and flower distribution density. The two studies collected data only at a narrow time window, which corresponded to the bloom stage in this study. On the contrary, the datasets in this study captured the full range of apple flower bud developmental changes. During image labeling judgment calls had to be constantly made on which stage a flower bud belonged to, when facing scenarios such as a half-inch green slightly opened up its leaves, a tight cluster started to show light pink color, a pink transitioned from being pinkish to whitish, a bloom wilted or dropped a few petals, etc. The inevitable inconsistency of bud labels, arguably, is one of the crucial reasons for the imperfect results of the classifiers. Furthermore, it was observed that the sample images provided in the two studies had much lower floral density compared to the images in this study. Using the 450 annotated images collected on May 7, 2020 when buds were mostly blooms as an example, the minimum, medium, and maximum number of bounding box per image were 5, 58, and 112, respectively. The dense bud distribution contributed much difficulty to labeling every single flower bud in an image and recognizing all the deformed, occluded flower buds at various depth correctly during image annotation (Figure 13), which could be another key reason that prevented the classifiers from achieving high performances. While building new classifiers with larger image input sizes than 640, or training current classifiers with new images might improve APs for tip, half-inch green, pink, and petal fall, reexamining current image labels for annotation accuracy, consistency, and completeness has a greater potential in helping increase model sophistication, although the process can be very labor-intensive. Considering the average apple flower bud distribution density, suboptimal classifier detection results, such as missing several buds, should have little to no influence on orchard heating requirement determination.

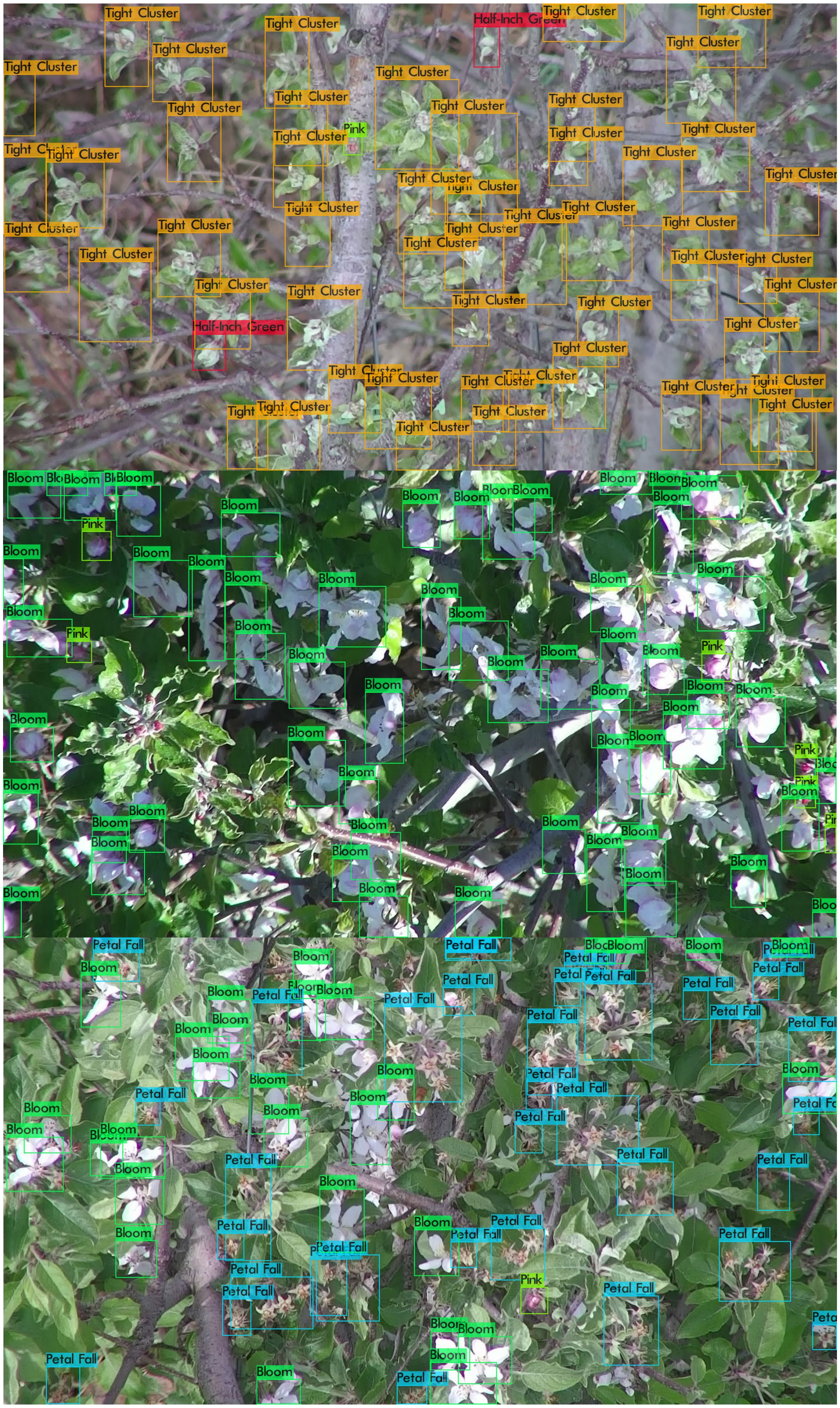

Figure 13. Detection results of the size 480 classifier on sample apple flower bud RGB images with high bud distribution density.

Encouragingly, the random sampling-based evaluation resulted outstanding classification precisions by the size 480 classifier on the second-year data. At least one flower bud was detected by the model in all 1985 images. For the 400 randomly sampled flower buds, 100 from each data collection date, the size 480 classifier achieved a 97.5% overall flower bud classification precision (Table 4). While the first year mAP results suggest that the trained models might not always detect every single flower bud in an image and return bounding boxes that follow manual labeling styles perfectly, the second year results indicate that the size 480 classifier only makes highly accurate predictions, which is likely to be true for the models with different network sizes. Such a model is more than sufficient for guiding heat applications in orchards, as the area within each RGB image is far smaller than the unit area that is covered by various frost protection equipment such as heaters, wind machines, or sprinklers.

Table 4. Classification accuracies of the size 480 classifier on the randomly sampled second-year RGB datasets.

Date	Growth Stage	Precision
4/13/2021	Tight cluster	96%
4/24/2021	Pink	96%
5/1/2021	Bloom	99%
5/7/2021	Petal fall	99%

2.4.4. Orchard Flower Bud Growth Stage Map

Using the Python program that applies a YOLOv4 classifier on RGB images and saves detection results to files, the size 480 classifier had an average image “inference time” of 0.16 s, while mapping detected buds to a thermal coordinate map had a negligible time cost. Figure 14 shows examples of apple flower bud growth stage maps generated with the size 480 classifier, where dots with different colors represent flower buds at various growth stages.

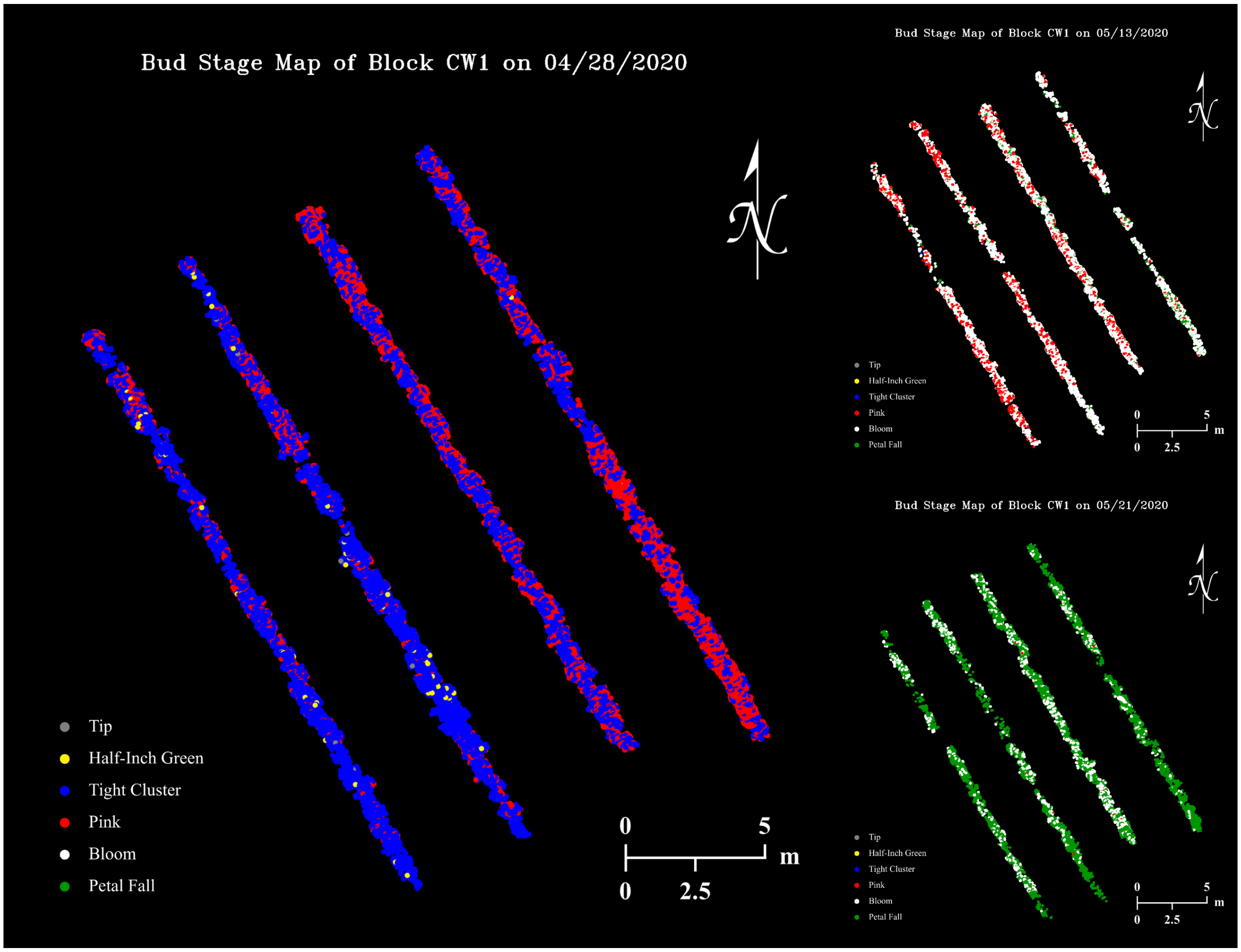

Figure 14. Apple flower bud growth stage map examples of the apple orchard on three data collection dates generated with the size 480 classifier.

UAV GPS accuracy is a major factor for the success of apple flower bud mapping. When a bud appears in multiple images taken without optimal GPS signals, it is likely to be mapped multiple times as a bud cluster on the map. This is generally not an issue for determining orchard regional heating requirement as little to no temperature variation exists for tree thermal pixels with short distances in between. With navigation techniques such as real-time kinematic (RTK) [90], it is possible to achieve centimeter-level accuracy for UAV positioning.

Unexpectedly, the flower bud growth stage maps seemed to have the potential to be an informative tool for horticulturists to visualize both spatial variation (on the same map) and temporal changes (among different maps) of apple tree developments in high resolution (Figure 15). Potentially, growth stage timing and duration of different apple varieties could be studied when RGB datasets collected at shorter time intervals are available. Most RGB images only sampled buds growing in the upper and center parts of tree canopies. The camera was not able to include lateral buds in a frame due to its limited FOV, while bottom buds were largely blocked by tree branches, leaves, and upper buds.

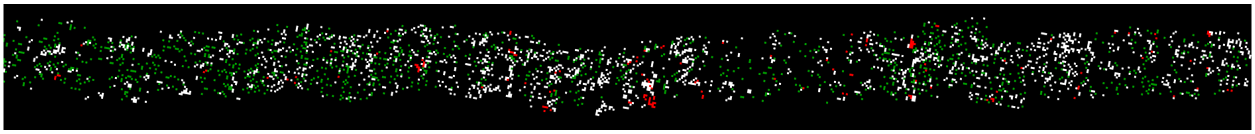

Figure 15. A close-up view of the apple flower bud growth stage map on May 16, 2020, showing actual floral density of a tree row section with each detected flower bud mapped as one pixel.

2.4.5. Orchard Heating Requirement Map

Using a temperature map from June 2, 2020 (Figure 12) and the flower bud growth stage map on April 28, 2020 (Figure 14), orchard heating requirement maps (Figure 16) were simulated assuming the critical temperatures of apple flower buds were 22, 19, and 18 °C higher than the actual values (Table 2). As can be observed in the maps, pixels with brighter colors represent orchard regions with higher heating demands, and sections of tree rows will disappear when it does not have any heating needs. The realization of map appearance in Figure 16 involved several image processing techniques including dilation, Gaussian blurring, and masking. Orchard heating requirement resolution can be increased by dilating pixels with smaller kernels, although practically lower resolution maps would be more suitable for determining regional heating treatments. Upon further dilations, the tree rows can be merged into an area, and heating requirement levels can be quantized to display discrete heating demand levels of the entire orchard.

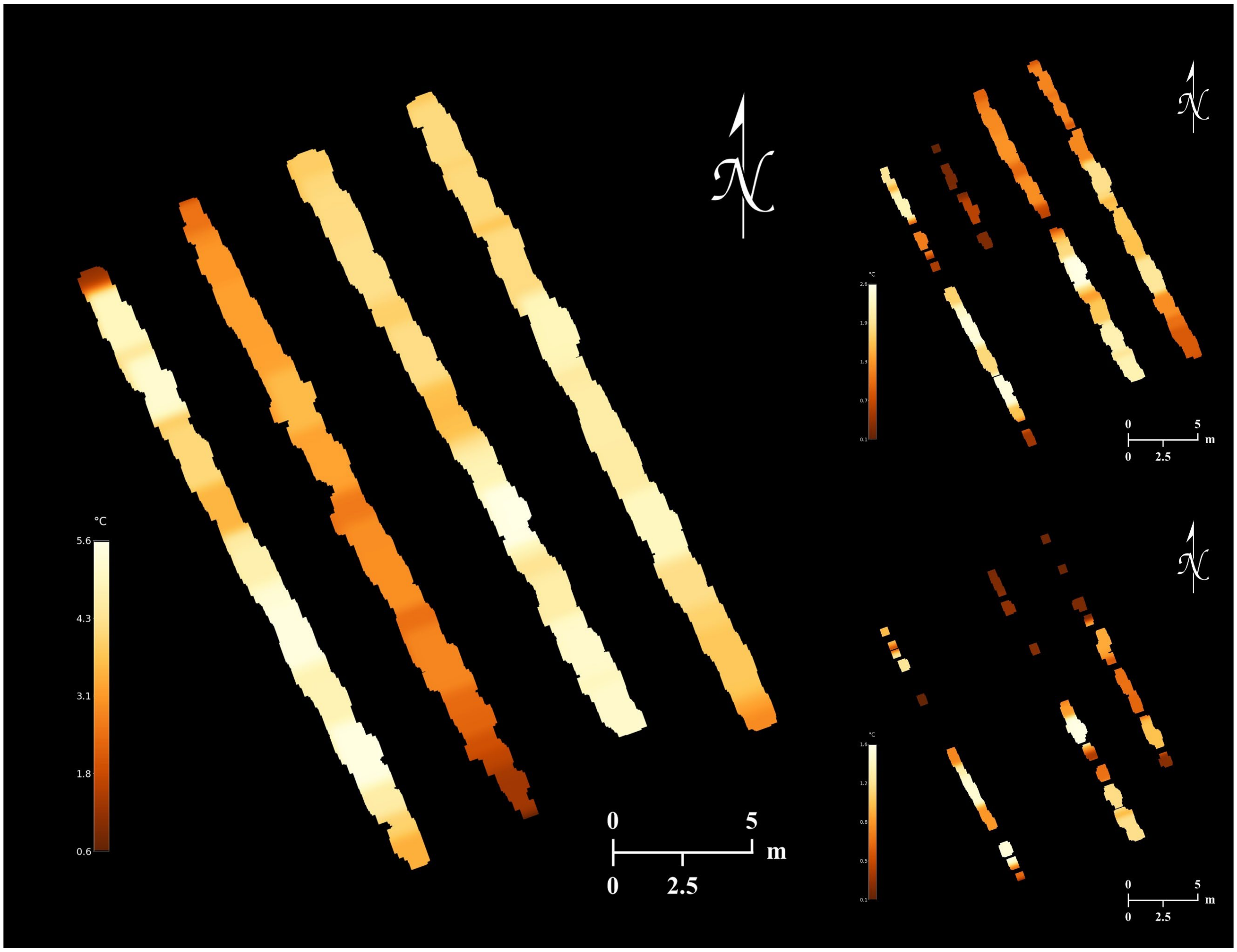

Figure 16. Orchard heating requirement map examples simulated based on the June 2, 2020 temperature map and the April 28, 2020 flower bud growth stage map with three levels of artificial critical temperatures.

2.5. Implications and Future Work

This study provides a framework for the concept of assisting orchard frost management with modern technologies and computer vision algorithms, for the purpose of minimizing labor and heating costs and maximizing management efficiency. However, any component of this framework can be modified as needed, although the outcome might be less optimal. For example, the UAV can be replaced with an unmanned ground vehicle, the YOLOv4 detector can be switched to other older but more widely implemented algorithms such as Faster R-CNN, and the image stitching can be completed using readily available commercial software.

From the application standpoint, the proposed methodology requires flower bud growth stage maps to be obtained prior to temperature maps. As capturing high-quality RGB images requires good lighting conditions, without artificial lighting, RGB flight missions should only be conducted after sunrise and before sunset. Thermal flight missions, on the other hand, are not restricted by the time of day as objects emit infrared radiations at all times. In the daytime before a forecasted frost event, orchardists can either fly manually to sample buds at preplanned locations that are representative of the whole orchard, or set up autonomous flight missions to capture bud images as demonstrated in this study, and then generate a flower bud growth stage map from the collected data using a pre-trained classifier. Note RGB image collection and flower bud growth stage map generation are not time-sensitive, unlike temperature map generation. Based on the assumption that there will be no significant flower bud growth stage changes overnight or within a day, during a frost event, orchardists can initiate a thermal flight mission whenever orchard temperatures need to be gauged. Depending on UAV flight altitudes and speeds, temperature maps can be created either in low resolution but quickly, or in high resolution but with a longer delay, and heating requirement maps can be generated as soon as the temperature maps are available. While a temperature map already allows orchardists to decide the level of needed heating treatments, a heating requirement map puts plant growth stage variation in an orchard in consideration to minimize unnecessary regional heating. Note that, depending on orchardists' unique situations during frost events such as frost protection methods, orchard layout and topography, and weather conditions, plant temperatures can change at different rates, and orchard heating requirement maps might need to be generated at higher or lower frequencies for either timely orchard heating demand assessment or observable plant temperature changes.

Aside from orchard heaters, the proposed methodology can also be beneficial to other traditional frost protection methods such as irrigation. Although overhead or under-tree irrigation has disadvantages such as high installation cost, large water consumption, and causing broken branches due to ice overloading, it generally can protect trees from frost effectively through the heat of solidification of water. Unlike orchard thermometers that measure air temperatures, thermal cameras can directly assess how irrigation changes plant temperatures. With the information on a heating requirement map, variable rate irrigation (VRI) systems can potentially apply larger amounts of water to orchard regions with higher heating demands. On the other hand, large-scale frost management methods such as wind machine might not be the best match to the proposed methodology. Wind machines increase orchard temperature simply by bringing warm air down from an inversion layer to ground level to break thermal stratification, and they typically have low operational costs. Yet, wind machines do not work under cold, windy conditions, and they cannot make the most use of a heating requirement map with localized orchard heating demands as a single commercial wind machine often covers an area up to 40,000 to 60,000 m^2. The methodology of orchard heating requirement determination proposed in this study does not come without limitations. Weather condition is a major factor that influences UAV performance in the air. Using DJI Matrice 600 Pro as an example, it cannot be operated in rain or snow as it is not a waterproof drone, and it has a lower operating temperature limit of −10 °C and a maximum wind resistance of 8 m/s. In addition to affecting drone stability, strong wind can also significantly reduce image quality by causing tree movements. For large orchards, multiple sets of batteries might be needed for prolonged flight missions. As thermal cameras have limited dynamic ranges, with the presence of heating sources such as orchard heaters, thermal cameras can be blinded and fail to prevent overheating in the heating region. Before trees develop leaves, it can be challenging to map detected flower buds such as tips exactly onto tree branch pixels of a temperature map, as doing so requires navigation systems with very high precision. Luckily, during such growth stages flower buds also tend to have high cold resistance [12] and frost damage is not a prominent issue. The total cost of equipment used in this study, including a UAV, a thermal camera, an RGB camera, an iPad, two extra sets of UAV batteries, and a computer, was close to $25,000. Despite being a big initial investment, this amount could be substantially cheaper when compared to human labor costs for temperature monitoring of large orchards in the long-term.

To fully automate orchard frost management, future studies should focus on the development of heating platforms that are capable of site-specific heating such as self-navigating heating vehicles, as well as the communication and systemic integration of temperature monitoring devices and heating units. For example, a CPS consisting of UAV-based orchard heating requirement assessment, unmanned ground vehicle (UGV)-based heating, and station-based vehicle communication, UAV data processing, and heating treatment determination has the potential to replace current manual frost protection methods. A UAV can maintain a predefined search pattern to monitor orchard temperatures, while a UGV will be dispatched to apply heating treatments as soon as orchard temperatures fall below the critical levels. A orchard heating requirement map generated by a base station using the proposed methodology contains both temperature and location information, which can be further utilized by the base station to decide the order of heating locations and the duration of heating treatments for the UGV. With the advancements of modern object detection algorithms such as YOLOv4, real-time heating demand determination is possible by flying thermal and RGB cameras simultaneously, applying flower bud detector on RGB data, and registering thermal and RGB video frames, although much hardware integration and software development need to be involved to achieve robust results.

2.6. Conclusions

The concept of utilizing a UAV-based thermal camera for temperature monitoring and an RGB camera for plant growth stage spatial variability characterization in apple orchard was demonstrated. The thermal image stitching algorithm produced georeferenced temperature maps with high resolutions sufficient for individual flower bud mapping, whose generation time can be improved by sacrificing thermal mosaic resolutions for more timely orchard temperature assessments. The fully functioning YOLOv4 apple flower bud growth stage classifier achieved satisfactory performance considering the difficulty of the RGB datasets. Training the classifier with new images or larger image input sizes might improve APs of certain growth stages, while refining existing image labels is more likely to fundamentally enhance the model sophistication. Orchard heating demand variations were observed in the simulated heating requirement maps, which proved uniform heating treatments for frost protection in orchards might be wasteful. The proposed methodology of orchard heating requirements determination is concluded to be feasible, and this study can serve as a stepping stone for future research on fully autonomous, site-specific orchard frost management systems.

CHAPTER 3
APPLE FLOWER BUD CLASSIFIER SENSITIVITY ANALYSIS

3.1. Introduction

Ever since the groundbreaking achievement by Krizhevsky et al. [18] in ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2012, CNN-based object detection has been a popular topic for many years. CNN, originally proposed by Fukushima [19] in the name of "neocognitron" in 1979, is a class of artificial neural network (ANN) commonly utilized in image-related tasks involving spatial pattern analysis. Typically, a CNN takes a tensor, or the R, G, and B channels of an image as inputs. It extracts image features through sliding kernels performing the Hadamard product in multiple convolutional layers, with early convolutional layers in the network extracting simple, low-level features such as corners, circles, or edges, and late layers extracting complex, high-level abstract patterns such as car headlights or leaf forms [20]. In between convolutional layers, pooling layers performing max pooling, average pooling, etc. are often utilized for downsampling inputs, and activation functions such as rectified linear units (ReLUs) are deployed to introduce non-linearity into the output of a neuron [21]. The last few layers in a CNN architecture are commonly chosen to be fully-connected feed-forward neural networks in combination with an activation layer such as softmax, which take flattened vectors from previous layers as inputs and output interpretable class probabilities [22]. During model training, parameters of a CNN such as kernel weights and biases are learned through backpropagation [23], with the goal of optimizing a loss function such as cross-entropy loss [22].

Object detection is a fundamental and challenging computer vision problem where the locations and classes of object instances in an image need to be identified. Owing to CNNs' strong abilities of classifying whole images, CNN-based object detection frameworks or object detectors that combine both object localization and classification have made tremendous progress in recent years. Object detectors can be divided into two types, namely two-stage detectors such as the R-CNN series [24–28] where object classification occurs after a region proposal module, and one-stage detectors such as the YOLO series [29–33] where object detection is treated as a regression problem of bounding boxes and class probabilities [29]. Generally, two-stage detectors are more accurate in object localization and classification due to their dedicated networks searching for regions of interests, while one-stage detectors have the advantage in inference speed because of their straightforward architectures [34]. Among the most influential object detectors, YOLOv4 was developed by Bochkovskiy et al. [32] in 2020 based on the previous work by Redmon et al. [29–31]. The architecture of YOLOv4 adopted a CSPDarkNet53 [35] backbone with Mish [36] activation for feature extraction, a neck of SPP [37] and PANet [38] with Leaky-ReLU [39] activation for feature aggregation, and a YOLOv3 [31] head for class prediction. Based on the Microsoft COCO benchmark [40], YOLOv4 was able to achieve comparable accuracies to other state-of-the-art detectors but with double inference speeds [32].

Out of the many areas that have benefited from CNNs, agriculture is one where object detectors demonstrated their great potential in assisting farm management and have been utilized extensively in relevant research. For example, since its release, YOLOv4 has been applied for fruit detection of cherry [41], pear [42], fig [43], apple [44], and citrus [45], flower detection of apple [46], citrus [47], and pumpkin [48], ear counting of wheat [49], plant disease detection such as corn leaf blight [50], citrus canker [51], and citrus greening [51], weed detection in rice [52] and carrot [53] fields, pest detection such as whitefly [54], thrip [54], locust [55], mealybug [56], Coccidae [56], and Diaspididae [56], livestock detection such as pig [57] and cattle [58], and aquaculture [59,60]. Despite the successes accomplished in these studies, unique challenges often reside in detecting objects within complicated agricultural scenes, especially during the image annotation process since object detectors are typically supervised learning algorithms that require ground truth. For example, as reported in the previous study when apple flower bud growth stages in aerial RGB images were classified using YOLOv4, the issues that were encountered during image labeling included: (1) too many instances to label due to high floral density (Figure 17a); (2) identifying complete bloom shapes due to dense floral spatial distribution (Figure 17a); (3) labeling occluded flower buds (Figure 17a); (4) distinguishing vegetative buds from flower buds (Figure 17b, 17c); (5) labeling out-of-focus flower buds (Figure 17d); (6) labeling flower buds possessing ambiguous characteristics (Figure 17e, 17f, 17g). These difficulties can significantly influence the quality of ground truth, that is, instance label completeness and consistency of training, validation, and test image datasets. As a result, it is less likely for an object detector that is trained on such data to achieve high classification accuracies according to traditional metrics such as average precision (AP) [61], since the integrity of ground truth has been compromised. Aside from the challenges in ground truth preparation, CNN models can be highly biased due to the limited scale of training datasets and the failure of including rare events in the datasets, in contrast to real life that involves countless variables, resulting in low object recognition performances of the models during stress tests [62–64]. In an agricultural setting, crop appearances in an image can be altered substantially by factors such as environmental conditions and camera settings, which adds many uncertainties to the performance of an object detector when it is employed for recognizing outdoor plants.

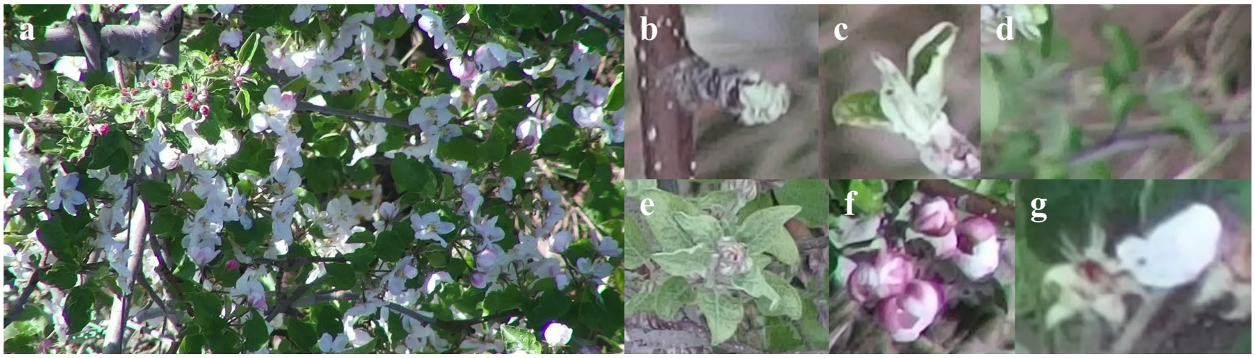

Figure 17. Examples of difficult apple flower bud annotation scenarios: (**a**) a scene of densely distributed apple blooms, many of which are occluded by leaves and branches and clustered with one another; (**b**) a tip that could potentially develop into either a vegetative bud or a flower bud; (**c**) a seeming half-inch green while the presence of its flower structures are unknown because of its non-fully-opened leaves; (**d**) two defocused flower buds in the background that could be either tight clusters or pinks; (**e**) a tight-cluster-shaped flower bud showing slight pinkness; (**f**) flower buds in the transition from pink to bloom; (**g**) a petal fall hanging on to a piece of wilted petal.

Improving CNN model robustness has always been a core issue from the perspective of reliability in real-world implementation, which starts by first knowing the weakness, or limitations of CNNs. In current literature, several studies have explored the performance of CNN models against artificial image distortions. Dodge and Karam [65] tested four pretrained CNN models, namely Caffe, VGG-CNN-S, VGG-16, and GoogLeNet, with distorted images from the ImageNet [66] dataset in 1,000 categories. The five types of examined image distortions included JPEG compression, JPEG 2000 compression, additive Gaussian noise, Gaussian blur, and contrast reduction. Similar studies were also carried out by Ghosh et al. [67] for VGG-16 and AlexNet, and Geirhos et al. [68] for VGG-19, GoogLeNet, and ResNet-152. Later Dodge and Karam evaluated pretrained and fine-tuned VGG-16, GoogLeNet, and ResNet-50 models using images of ten dog classes with Gaussian blur and additive Gaussian noise [69]. They also studied VGG-16's resilience against Gaussian noise, Gaussian blur, and the mixture between the two distortions [70,71]. Zhou et al. [72] applied two CNN models, LeNet-5 and CIFAR-10-quick, on the MNIST [73], CIFAR-10 [74], and ImageNet datasets with motion blur, defocus blur, Gaussian noise, and combined noises. Vasiljevic et al. [75] investigated the performance of VGG-16 for five types of blurs including defocus, linear horizontal motion blur, linear vertical motion blur, camera shake, and Gaussian blur using the ImageNet dataset. Tadros et al. [76] employed PlacesNet, InOutNet, GIST-LDA, and HSV-LDA four CNN models classifying images from the Places-205 [77] dataset in ten indoor and ten outdoor categories with artificial noises including Gaussian blur, white noise, block scramble, and occluding grid. It is expected and was also confirmed by the aforementioned studies that the performance of a CNN decreases when higher levels of distortions are present in an image. However, the fragility of CNNs was also revealed when they struggled to recognize images corrupted by noises that humans had no trouble with [68,69], and researchers have utilized approaches such as finetuning and retraining to alleviate the effect of such noises on CNN models [72]. Aside from image distortions, the influence of training data size on CNNs has also been studied for axial computed tomography images [78,79], blood cell images [80], benchmark datasets such as CIFAR-10 [81]. However, in terms of investigating the robustness of an object detection framework instead of pure CNN models against image distortions, as well as how different training dataset attributes might influence object detector performance in an agricultural setting, an effort is still missing in existing literature.

Considering ground truth annotation difficulty and potential quality inconsistency of agricultural images, the sensitivity of YOLOv4 was examined using distorted test datasets and manipulated training and validation datasets, with the goal of identifying the strengths and weaknesses of state-of-the-art object detectors as well as deriving practical guidelines for future studies implementing object detectors in agriculture. Specifically, the objectives of the study included: (1) quantifying the performance of YOLOv4 against test images with different types of artificial distortions; (2) evaluating the significance of various training dataset attributes for YOLOv4 based on its classification accuracies on test datasets.

3.2. Materials and Methods

3.2.1. Apple Flower Bud Dataset

For a better focus on apple flower buds that are more susceptive to low temperature damage and faster model trainings, a subset of the 2020 RGB image data from the previous study was selected as the dataset. The dataset covered four apple flower bud growth stages including tight cluster, pink, bloom, and petal fall. Negative samples, or apple tree images containing no flower buds were also collected in the late growing season of 2020 and utilized during model training. A summary of the image dataset is listed in Table 5.

Table 5. Summary of the original apple flower bud image dataset.

Collection date	Corresponding growth stage	Sample type	Image amount
4/23/20	Tight cluster	Positive	450
5/2/20	Pink	Positive	450
5/7/20	Bloom	Positive	450
5/16/20	Petal fall	Positive	450
9/25/20	N/A	Negative	1260

3.2.2. Data Preparation and Model Training Protocol

Image annotations prepared in the previous study were utilized. Following the previous study, positive samples were divided into 70%, 20%, and 10% segments for model training, validation, and test, respectively. In each segment there was an equal number of images from each date. Unless specified, all negative samples were also used for model training. YOLOv4 models were trained on a Windows 10 computer with an Intel® Core™ i9-9900 (USA) CPU, an NVIDIA GeForce RTX 2060 (USA) GPU, and a 16 GB RAM. A network size of 480×480 was adopted for all models based on the findings from the previous study. Except for learning rate, which changed from 0.002 to 0.00001 once training iterations went above 2000, all other hyperparameters were kept default as described in the original publication [32]. Model trainings were stopped when the mean APs (mAPs) of validation datasets no longer improved, which were automatically calculated by YOLOv4 based on 11-point interpolated precision-recall curves [61].

3.2.3. Sensitivity Examination

The sensitivity of YOLOv4 was examined to answer two questions. The first one was at what levels of noises a YOLOv4 model trained and validated on normal images but tested with distorted images would "fail". For this question such a model is considered as the baseline model: a model whose development and usage only involve normal images as in most studies employing object detectors. Simulating possible external influences on image quality in an agricultural field, five types of image noises were explored including white noise, motion blur, hue shift, saturation change, and intensity change.

The second question was how various quantitative and qualitative attributes of training datasets would influence the performance of YOLOv4 on normal or distorted test datasets in reference to the classification accuracies of the baseline model. The following aspects were investigated regarding training dataset attributes which are further explained in the subsections below: training dataset size, training image label quality, negative sample presence, training image order, and image distortion levels. Image processing and data analysis were completed using MATLAB R2021a.

3.2.3.1. Baseline Model

The baseline model was trained, validated, and tested using the original datasets (Table 5), which consisted of 2520, 360, and 180 images respectively in random orders.

3.2.3.2. Test Image Distortion

Distorted test datasets were generated by adding various levels of noises to the original test dataset as described below. Each unique dataset contained 180 images and was used to test the robustness of the baseline model. The mAP of 30% was arbitrarily selected as the "failing point" of the baseline model, where the apple flower bud growth stages could no longer be identified by the model with high accuracies. Generally the baseline model was no longer being further tested with more severely distorted test datasets once a mAP lower than 30% was observed. The relationships between model classification accuracies in terms of mAP and noise levels were interpreted as cubic smoothing splines. A "critical level" was also identified for each type of noise during data analysis that represented the noise level beyond which the baseline model performance would decrease at a substantially higher rate. By normalizing both the mAP values larger than 30% and corresponding noise levels into the range of 0 and 1, the point on a fitted spline curve where the slope of the curve first dropped below -30° or -tan(π/6) was defined as the critical point.

3.2.3.2.1. White Noise

Images captured under poor lighting conditions or with high camera shutter speeds generally look noisy due to their low signal-to-noise ratios (SNRs) [82]. The influence of image sensor noise on the performance of YOLOv4 was examined by simulating images with low SNRs. "Dependent" white noises were added to the original test images, meaning the absolute noise values were dependent on their corresponding signal pixel values. For each original test image, the pixel values within its R, G, and B channels were increased or decreased by random percentages. However, the random percentages for the same image channel had a uniform probability distribution centered at 0 and ranged within a specified absolute level. Four levels of white noises were explored, including 10%, 20%, 30%, and 40% (Figure 18). Post-distortion RGB channel values outside the normal range, for instance 0 to 255 for 8-bit images, were saturated to either the lower or upper limit.

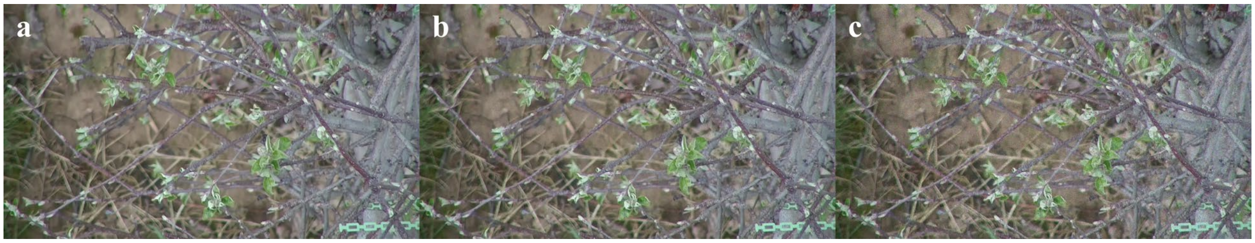

Figure 18. Examples of distorting images with white noise: (**a**) original image; (**b**) 20% noise; (**c**) 40% noise.

3.2.3.2.2. Motion Blur

One of the downsides of using a UAV as the camera-carrying platform is that the captured images can be influenced by motion blur, especially at high flying speeds. For a camera with high zoom capabilities such as the DJI Zenmuse Z30, any relative motion between the camera and plants can lead to significant image quality degradation. To simulate low-quality UAV imagery, linear motion blur was applied to the original test images, which were blurred by 5 pixels, 10 pixels, 15 pixels, 20 pixels, and 25 pixels (Figure 19). The direction of motion blur was randomly assigned to each image while kept the same across datasets.

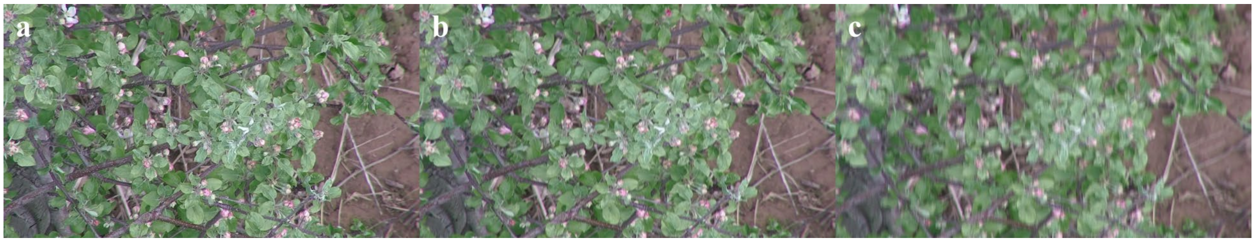

Figure 19. Examples of distorting images with motion blur: (**a**) original image; (**b**) 10-pixel blur; (**c**) 25-pixel blur.

3.2.3.2.3. Hue

Hue is defined as "the attribute of a visual sensation according to which an area appears to be similar to one of the perceived colors, red, yellow, green and blue, or a combination of two of them" [83]. Depending on camera white balance setting, sun position, etc., plant hue can appear differently in an image. The effect of hue change on YOLOv4 was investigated by converting the original test images into hue-saturation-intensity (HSI) color space [84] and shifting their hue channels. Typically, hue values are represented by angular positions around a neutral point in a color space with similar-appearing colors being closer to one another. The original hue values were replaced by new hue values with a certain degree of shift, and 20°, 30°, 40°, 50°, 60°, 120°, 180°, 240°, 300°, 310°, 320°, 330°, and 340° clockwise or counterclockwise shifts were tested (Figure 20).

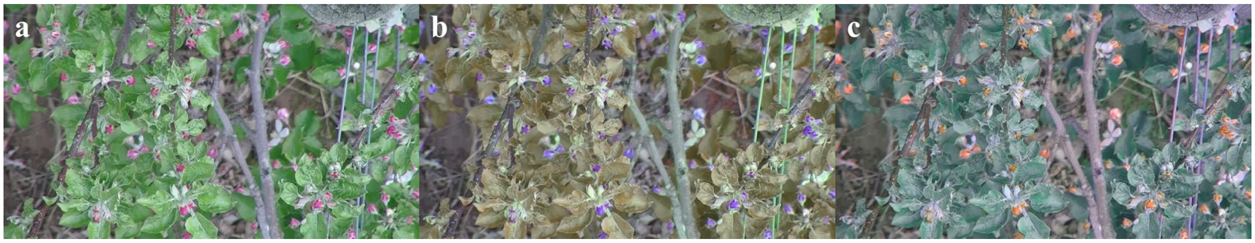

Figure 20. Examples of distorting images with hue shift: (**a**) original image; (**b**) 60° clockwise or 300° counterclockwise; (**c**) 60° counterclockwise or 300° clockwise.

3.2.3.2.4. Saturation

The degree of paleness of a color, or saturation, can be simply interpreted as the ratio between the amount of spectral light at the color's dominant wavelength and the amount of white light [85]. Factors such as sunlight, camera settings, image sensor, etc. can affect image saturation. The significance of saturation was studied through increasing or decreasing the S channel values of the original test images in HSI color space by different percentages, including 10%, 30%, 50%, 70%, 80%, and 90% decreases as well as 10%, 30%, 50%, 80%, 100%, 150%, 200%, 300%, 500%, 700%, 800%, and 900% increases (Figure 21). Post-distortion S channel values outside the normal range were saturated to either the lower or upper limit.

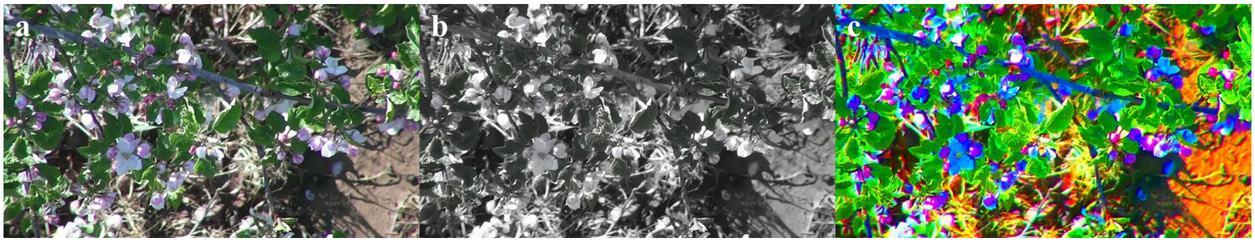

Figure 21. Examples of distorting images with saturation change: (**a**) original image; (**b**) 90% decrease; (**c**) 900% increase.

3.2.3.2.5. Intensity

The brightness, lightness, or intensity of an image can be influenced by camera aperture, shutter speed, and ISO settings as well as field lighting conditions. The intensities of the original test images in HSI color space were changed through increasing or decreasing their I channel values by different percentages, including 10%, 30%, 50%, 60%, 70%, and 80% decreases as well as 10%, 30%, 50%, 80%, 90%, 100%, 110%, 120%, 130%, 140%, 150%, 160%, and 180% increases (Figure 22). Post-distortion I channel values outside the normal range were saturated to either the lower or upper limit.

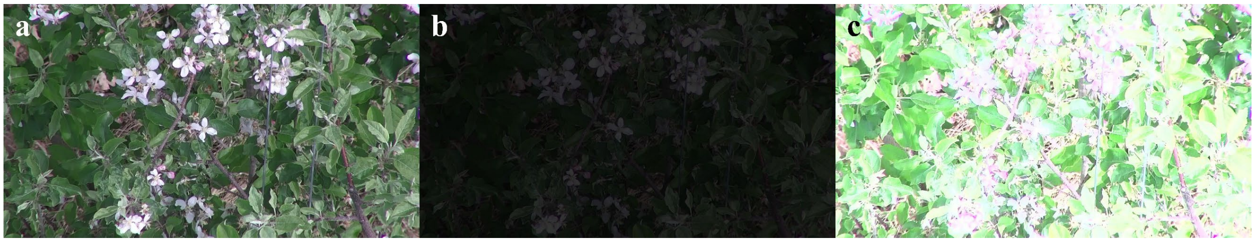

Figure 22. Examples of distorting images with intensity change: (**a**) original image; (**b**) 80% decrease; (**c**) 180% increase.

3.2.3.3. Training Dataset Manipulation

After establishing the performances of the baseline model against different image distortions, the influences of training dataset attributes on YOLOv4 were then explored. The term "attributes" refers to various aspects of a training dataset, such as how many images are in the dataset, whether all instances in the images are properly annotated, whether negative samples are utilized, whether the images are shuffled into a random sequence, and whether noises are present in the images. The importance of a training dataset attribute was evaluated based on the classification accuracy comparison between the baseline model and a newly trained model on the same test dataset.

3.2.3.3.1. Dataset Size

Generally, CNNs trained with larger datasets have better classification accuracies [81]. However, object detectors such as the YOLO series are trained on labeled instance information, which might reduce the significance of training dataset size for model performance since a single image can contain numerous instances. To study how training image quantity would affect YOLOv4 in terms of classifying apple flower buds, seven new models were trained using 90%, 70%, 50%, 20%, 10%, 5%, and 1% of the original training images, which corresponded to 2268, 1764, 1260, 504, 252, 126, and 24 images. In each new training dataset, the number of positive samples for each growth stage were kept equal when possible.

3.2.3.3.2. Image Label Quality

As outlined in the Introduction, labeling plant structures such as apple flower buds can be a challenging task. Inevitably, ground truth quality may vary from image to image. Training datasets with low image annotation qualities were simulated by randomly removing a percentage of annotations from each original training image (Figure 23). Six new models were trained with 90%, 70%, 50%, 20%, 10%, and 5% of the image labels in each original training image. The quantities of labels that should be preserved in each image were calculated by rounding the products of the original image instance number and the percentages to the nearest integers. The removed labels were randomly chosen.

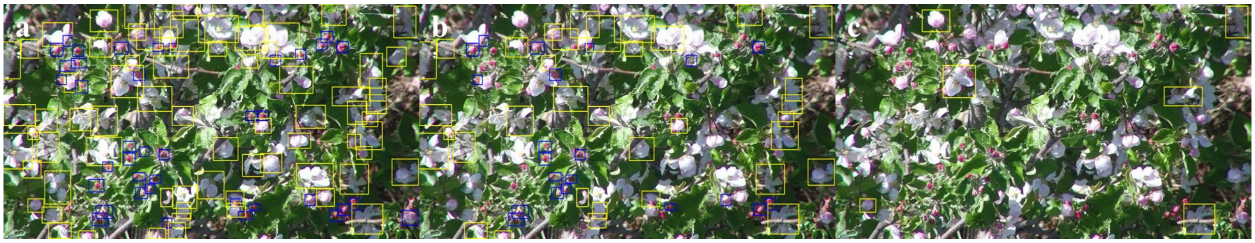

Figure 23. Examples of manipulating training image label quality: (**a**) 100% labels; (**b**) 50% labels; (**c**) 5% labels.

3.2.3.3.3. Negative Sample Presence

Images that contain no target objects, or negative samples are sometimes utilized during training to improve object detector robustness [86]. The value of negative samples was examined for YOLOv4 models trained on large and small datasets, by training two new models with 100% and 1% of only the original positive training samples, respectively.

3.2.3.3.4. Training Image Order

It has been observed in the previous study that training image order might influence the performance of an object detector, meaning when a model is exposed to a dataset that is organized by class during training, its performance possibly will be different from the model that is trained on a dataset with a random image sequence. A new model was trained using the original training dataset whose images were arranged by date, with the purpose of stimulating the model with first tight cluster stage, next pink stage, then bloom stage, and lastly petal fall stage during each training epoch.

3.2.3.3.5. Noise Retraining

From the model training perspective, attempts were made to improve the classification accuracies of YOLOv4 on the distorted test datasets that previously caused the baseline model to fail. With the goal of exposing a model to noise during training, mainly two approaches were adopted to modify the training datasets for the five types of image distortions, namely distorting all original training images with the baseline-model-failing levels of noises, and with evenly distributed levels of noises in between no noise and the baseline-model-failing noise levels. For white noise, two new models were trained and validated with a 40% distorted dataset and a 0 to 40% evenly distorted dataset. For motion blur, two new models were trained and validated with a 25-pixel distorted dataset and a 0 to 25-pixel evenly distorted dataset. For hue, a new model was trained and validated with a 0 to 360° evenly distorted dataset, treating hue as completely a noise source to observe how accurately YOLOv4 would classify apple flower buds without relying on hue. For saturation, three new models were trained and validated with a 90%-decrease distorted dataset, a 900%-increase distorted dataset, and a “90%-decrease to 0 to 900%-increase” evenly distorted dataset. For intensity, three new models were trained and validated with an 80%-decrease distorted dataset, a 180%-increase distorted dataset, and an “80%-decrease to 0 to 180%-increase” evenly distorted dataset. For saturation and intensity, the amounts of images with decrease distortion were equal to the amounts of images with increase distortion in the “decrease to 0 to increase” datasets.

3.2.3.3.6. Ultimate Test

Considering multiple types of noises might occur in an image simultaneously in real life, an “ultimate” dataset having all five types of image distortions in each image was created, by distorting all images from the original training, validation, and test datasets with 0 to 40% evenly distributed white noises, 0 to 25-pixel evenly distributed motion blurs, 0 to 360° evenly distributed hue shifts, 90%-decrease to 0 to 900%-increase evenly distributed saturation changes, and 80%-decrease to 0 to 180%-increase evenly distributed intensity changes. Due to the combination of noises with extreme levels, many images in the dataset had heavily distorted appearances and apple flower buds in these images were no longer recognizable to humans (Figure 24). Aside from building a new model that was directly trained and validated on the ultimate dataset, whether a model trained and validated on datasets where each image only contains one type of noise would be able to recognize an image with multiple types of noises, was also explored. Such a dataset was generated by simply combining the evenly distorted datasets for the five types of noises in the previous section for noise retraining, which contained 12,600 training images, 1,800 validation images, and 900 test images.

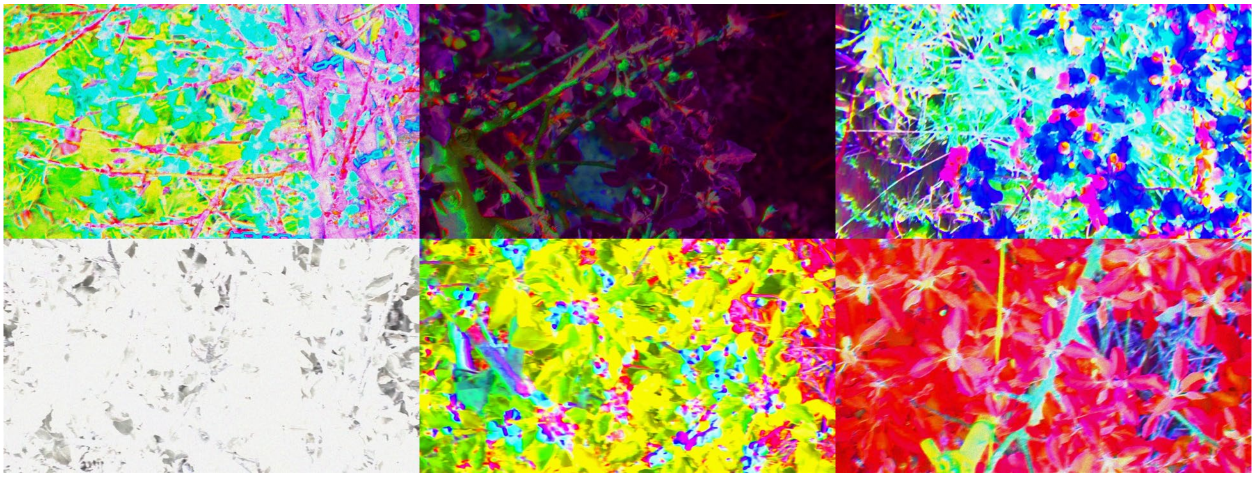

Figure 24. Samples from the ultimate dataset where apple flower buds are no longer recognizable to humans.

3.2.3.4. Dataset Naming Notation

Since a large number of diverse datasets was involved in this study, to minimize confusion, each unique training, validation, or test dataset was named based on how it differed from the original training, validation, or test dataset using unique abbreviations as listed in Table 6, which represented the type of difference between the original dataset and the current dataset. If necessary, the levels of image distortions were specified following the abbreviations.

Table 6. Abbreviations used for naming different training, validation, or test datasets and their meanings.

Abbreviation	Meaning
OD	Original dataset
WN	White noise
MB	Motion blur
HS	Hue shift
SD	Saturation decrease
SI	Saturation increase
ID	Intensity decrease
II	Intensity increase
DS	Training dataset size
LQ	Training image label quality
NA	Negative sample absent
IO	Training images ordered by date
UL	Ultimate dataset where each image was distorted by all five types of noises
DC	All five types of evenly distorted datasets combined

3.3. Results and Discussions

Due to the large size of the statistical results, APs and mAPs of all models on all relevant datasets are presented in Appendix.

3.3.1. Baseline Model

On the original test dataset, the baseline model achieved a mAP of 77.25%. Specifically, the APs for tight cluster, pink, bloom, and petal fall stages were 88.63%, 72.69%, 85.23%, and 62.45%, which were very similar to what was obtained previously.

3.3.1.1. White Noise

The baseline model almost failed at 30% white noise with a 32.91% mAP, and failed at 40% white noise with a 15.17% mAP. Tight clusters and blooms were more sensitive to white noise than pinks and petal falls. At 40% white noise, the APs for tight cluster and bloom were only 4.06% and 8.48%, while for pink and petal fall were above 20%. Interestingly, the images with white noise added, even at the 40% level (Figure 18), did not appear to be difficult for humans to identify apple flower buds. When not trained specifically for white noise, YOLOv4 might rely on local pixel homogeneity for classifying certain instances, such as clustered green pixels for leaves of tight clusters and clustered white pixels for blooms (Figure 25), while humans are able to unconsciously apply low-pass filters on noisy images in their brain before identifying objects. The 40% white noise level represents a very low SNR, whose occurrence in real life is improbable unless camera shutter speed is very high, or the environment is extremely dark and artificial light sources such as flashlights are absent. Based on the results, white noise does not seem to be a concern for most agricultural researchers applying a pretrained object detector to new images.

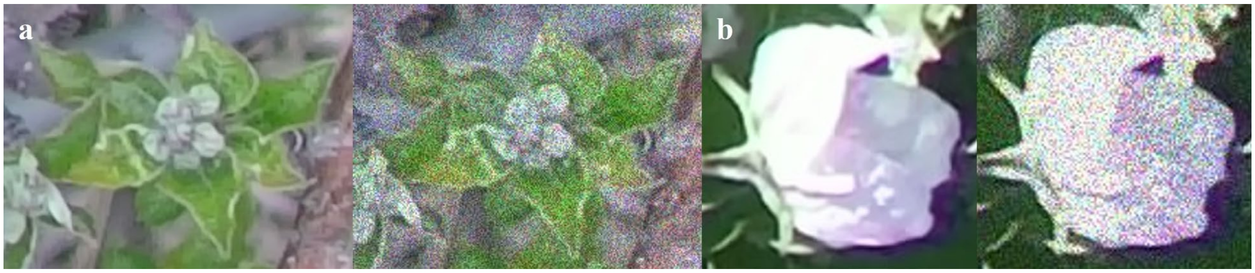

Figure 25. Examples of how the local pixel homogeneity of a flower bud feature changes after a 40% white noise distortion: (**a**) green leaves of a tight cluster; (**b**) white petals of a bloom.

3.3.1.2. Motion Blur

The baseline model failed at 25-pixel motion blur with a 23.28% mAP. Petal fall was the most sensitive stage to motion blur, which had a 4.74% AP, while tight cluster was the least sensitive stage with a 46.15% AP. Instance size or bounding box size might be the most important factor that influenced YOLOv4's detection rates on the blurred apple flower buds. Based on the labeling rules, tight clusters were typically the largest flower buds since each bounding box also included leaves, and petal falls whose bounding boxes mostly contained pistils, stamens, and sepals were the smallest. Comparatively speaking, the same level of blur can cause a larger disturbance to the features of small instances than large instances, since large instances might have large-scale features that are less sensitive to image distortion. For example, a petal fall might completely lose its stage-defining characteristics after a 25-pixel linear motion blur due to its small size relative to the length of motion, while a large tight cluster is very likely to preserve its original overall shape after the same blur (Figure 26).

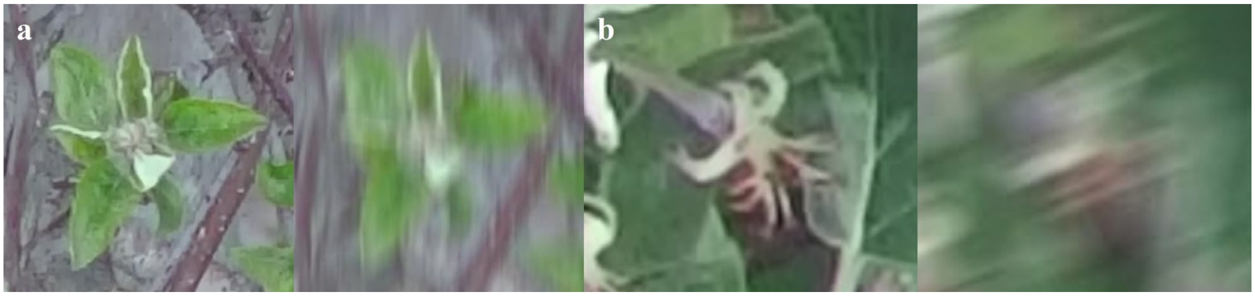

Figure 26. Examples of how large flower buds are more resistant to motion blur than small flower buds: (**a**) a tight cluster that is able to keep its rough shape after a 25-pixel motion blur; (**b**) a petal fall that is entirely unrecognizable after the same blur.

Assuming a shutter speed of 1/60 s and an approximated area of 6.6×10^{-8} m^2 for each image pixel, the 25-pixel motion blur simulated roughly a 0.4 m/s UAV flying speed assuming perfectly stationary apple trees. Ideally, at least in the context of applying the baseline model for apple flower bud classification, new test images should be captured with a UAV speed no more than 0.4 m/s to prevent significant negative impacts from motion blur on model performance based on the assumed camera shutter speed and image spatial resolution. Unfortunately, it can be challenging to manually maintain a UAV speed lower than that, also not all UAV operating software supports such a low flying speed. For example, DJI GS Pro, a native flight controlling app from DJI, only supports speeds above 1 m/s. Considering wind frequently causes substantial tree movements in the field, UAV researchers should be mindful about minimizing motion blur in their image data for optimal object detection accuracies. Increasing camera shutter speed can help mitigate the motion blur issue and potential allow UAVs to fly at higher speeds. However, captured images might also look noisier due to lower camera SNRs.

3.3.1.3. Hue

The baseline model started to fail when hue shift passed 60 to 75°. Clockwise 60° hue shift had a 43.85% mAP, and counterclockwise 60° hue shift had a 33.30% mAP. Bloom seemed to be the least sensitive stage to hue shift, having substantially higher APs than the other three stages at most shift levels. Blooms primarily had only white petals within their bounding boxes, while tight clusters had green leaves and unopened grayish sepals, pinks had unopened pink petals and grayish sepals, and petal falls had green sepals and yellowish pistils and stamens. It might be presumed that the more colors the identification of an object requires, the more reliant YOLOv4 would be on the object's hue information. Nevertheless, large hue distortions such as the 180° shift still can lead to near-zero APs of the baseline model for all growth stages. In real life, significant hue changes such as the ones simulated in this study would never occur under a natural setting, hence hue will not likely be a factor negatively affecting the performance of an object detector. However, when a scene is illuminated by artificial light sources, such as colored LED lights commonly applied in greenhouses [87], a pretrained detector might not work well for identifying objects with unnatural hues.

3.3.1.4. Saturation

The baseline model barely failed at 90% saturation decrease with a 29.31% mAP, and 900% saturation increase with a 29.74% mAP. Expectedly, bloom was the most robust stage against saturation distortion with over 50% APs, since saturation is irrelevant to grayscale colors such as white. An image with a 90% saturation decrease is very close to a pure grayscale image, implying YOLOv4 was not heavily dependent on pixel saturation information. Increasing image saturation did not seem to negatively affect the baseline model much either until the distortion reached extreme levels. Considering the unlikelihood of capturing images lacking saturation completely or being oversaturated in real life, saturation does not seem to be an image parameter that can influence the performance of an object detector for agricultural applications substantially.

3.3.1.5. Intensity

The baseline model failed at 80% intensity decrease with a 25.69% mAP, and 180% intensity increase with a 27.43% mAP. Bloom stage again was the most robust stage, having 48.26% and 60.01% APs. Tight cluster was the most sensitive stage, with only 1.08% and 3.42% APs. From the perspective of human recognition, after the 180% intensity increase distortion, the center flower structures of tight clusters were nearly impossible to recognize (Figure 27a), while many pinks could be distinguished from the background by its pink color (Figure 27b), most blooms looked similar to their undistorted appearances (Figure 27c), and some petal falls were identifiable by their unique shapes (Figure 27d). Note the distorted images with increased intensities were not entirely equivalent to overexposed images taken in real life, which have not only high intensities but also low saturations. The 80% decrease distortion simulated images captured under low light conditions, which generally did not seem to be as challenging for humans to identify apple flower buds as the 180% increase distortion, potentially because human brains have already been sufficiently “trained” to recognize objects under poor illuminance at night. Since white has the highest intensity value of any other color, it was not surprising to see bloom was the least influenced stage by decreased intensity distortion (Figure 22). One interesting observation was that the AP of tight cluster dropped drastically from 47.14% at 70% intensity decrease to 1.08% at 80% intensity decrease. Upon further inspection, rather than falsely classifying tight clusters as flower buds in other growth stages, the baseline model simply failed to detect most tight clusters at 80% intensity decrease (Figure 27e). The true reason behind this phenomenon is still unclear, since the images with the 70% and 80% intensity decrease distortions were relatively similar to human eyes. However, it is suspected that the baseline model might have learned an “intensity threshold” during training, such that once the overall intensity of a tight cluster was below this threshold, the model excluded the possibility of the tight cluster being a positive instance or simply returned an extremely low confidence score. The auto mode of modern RGB cameras, such as automatically adjusting shutter speed, aperture, ISO and flash settings, generally secures the intensity level of an image to be in the optimal range. For that reason, intensity is unlikely a variable which a pretrained object detector would have difficulty with in agricultural research.

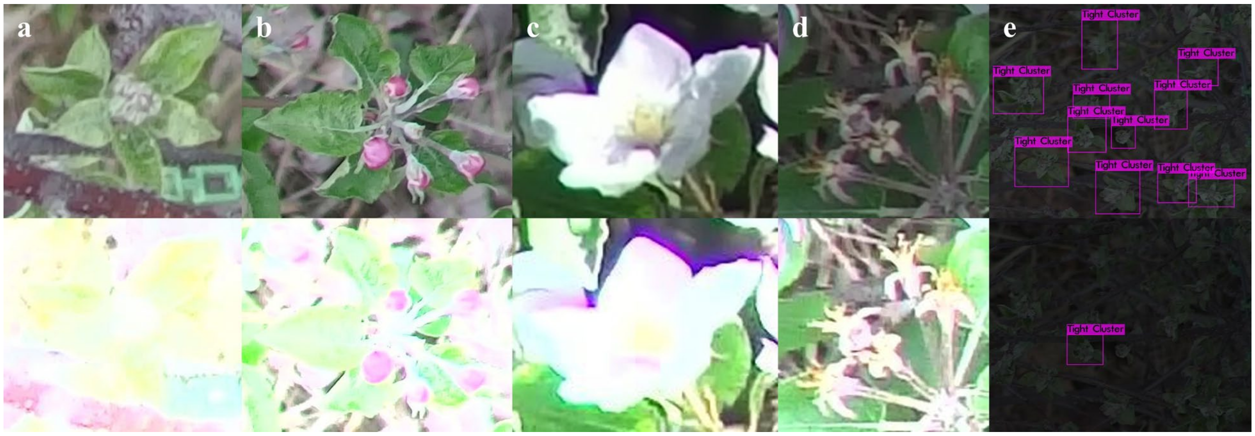

Figure 27. Phenomena observed during the intensity distortion test: at 180% intensity increase, tight clusters (**a**) lost most of their stage-defining characteristics, while pinks (**b**), blooms (**c**), and petal falls (**d**) were able to perverse some of their unique color or shape traits; (**e**) an example of

how the baseline model was able to detect many tight clusters at the 70% intensity decrease but not at the 80% decrease, which appeared to be relatively similar to human eyes.

3.3.1.6. Critical Levels and Failing Levels

Figure 28 shows how the performance of the baseline model in terms of mAP changed with various image distortion levels from no noise to the model-failing noise levels at 30% mAP. Based on the spline interpolations, Table 7 presents the critical level and the exact baseline-model-failing level of each noise obtained. As mentioned previously, the critical points were determined based on normalized mAP and noise level scales.

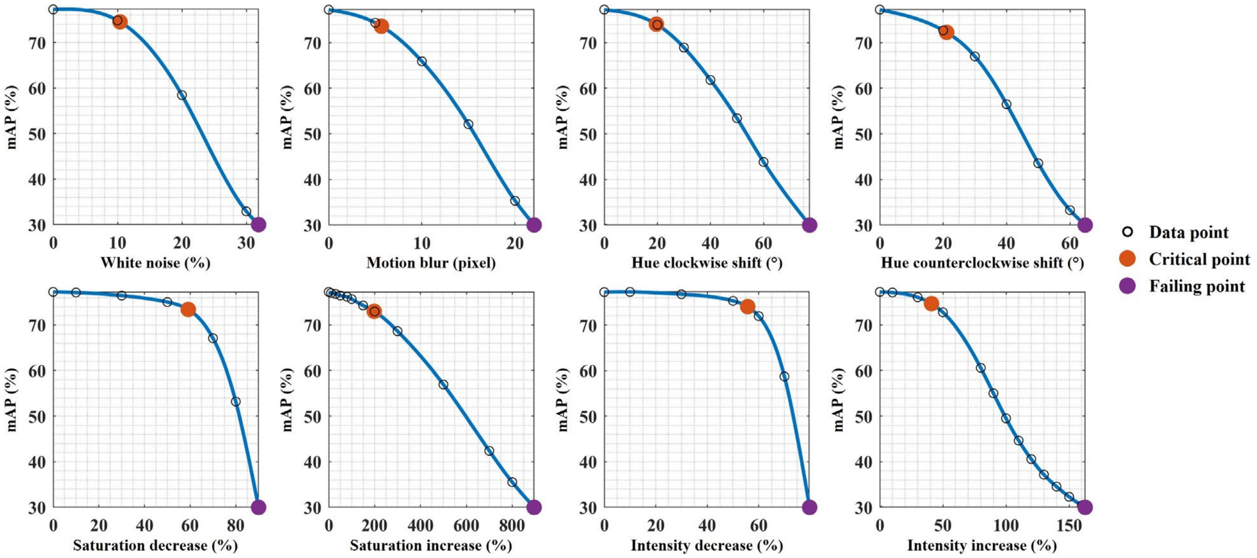

Figure 28. Relationships between the classification accuracies of the baseline model and levels of different types of image distortions.

Table 7. Critical levels and failing levels of the baseline model regarding different types of image distortions.

Distortion	Critical level	Failing level
White noise	10.39%	31.91%
Motion blur	5.66-pixel	22.05-pixel
Hue clockwise shift	19.65°	77.38°
Hue counterclockwise shift	21.11°	64.81°
Saturation decrease	59.17%	89.98%
Saturation increase	198.38%	895.35%
Intensity decrease	55.73%	79.80%
Intensity increase	40.96%	162.71%

For the best model performance in practice, before applying a pretrained object detector to new images, it is recommended that researchers either collect high-quality data in the first place, or remove images with high levels of noises from their existing datasets. The critical and failing distortion levels in this study serve as a reference for future studies to roughly gauge the classification accuracies of a pretrained model that needs to be utilized in complicated agricultural scenes involving many unexpected noise sources.

3.3.2. Retraining

In this section, the performance comparison between the baseline model and the models retrained from scratch with the manipulated training and validation datasets is presented.

3.3.2.1. Training Dataset Size

As shown in Appendix, YOLOv4 demonstrated its exceptionally strong robustness against low training image number. The baseline model was trained on 2520 images and achieved a 77.25% mAP. When trained with only 5% of the original training images, that is 126 images, YOLOv4 was able to attain a 55.49% mAP on the original test dataset. Trained with only 1% of the original training images, that is three positive samples of each growth stage and 12 negative samples, in total 24 images, YOLOv4 was still capable of accomplishing a 39.63% mAP. It is worth noting that generally the mAPs of the size-modified training datasets were negatively correlated with the corresponding training image numbers, which increased from 87.66% to

99.79% as the training image numbers decreased from 2520 to 24, implying it is more likely for YOLOv4 to maximize its learnable "knowledge" from smaller training datasets.

Conventional wisdom suggests building a robust CNN model requires large training datasets. It is not rare for researchers to train models on tens of thousands of images [66]. The counterintuitive results of achieving decent object detector classification accuracies with extremely small training datasets obtained in this study, can be explained from the perspective of instance number rather than image number. One of the unique aspects of the datasets was that each image often contained many apple flower buds. For example, during bloom stage, an image could easily include more than 100 blooms. Unlike pure CNN models that classify whole images, object detectors classify image regions of interests. Theoretically, a single image can contain an infinite number of unique regions. In the current study's case, 12 positive samples might already allow YOLOv4 to observe and learn from hundreds of different instances, which is confirmed in the subsequent analysis presented in the following sections. Based on the results, training instance number seems to be a more critical factor than training image number for object detectors.

3.3.2.2. Image Label Quality

Unlike in the previous section where training image number was modified, the training datasets for studying the influence of image label quality all had the same number of images, 2520. However, different percentages of image labels in the original training images were randomly removed depending on the dataset, simulating poor image label qualities by human annotators. Based on the results, labeling every single instance in an image does not seem to be essential for building an object detector, at least in the context of classifying apple flower buds, although higher training image label qualities always led to higher model performances. Trained with 50% of the original image labels, implying the dataset would require only half of the original training image annotation workload, YOLOv4 achieved a 71.16% mAP, in contrast to the baseline model performance of a 77.25% mAP. Trained with only 5% the original image labels, having the majority of flower buds in the training images unlabeled, YOLOv4 still was able to accomplish a 40.73% mAP. Overall, the performance of YOLOv4 with regard to training image label quality roughly resembled its performance regarding training dataset size. For example, the model trained on 100% original images with a 70% label quality had similar classification accuracies as the model trained on 70% original images with a 100% label quality.

An interesting but abnormal observation was that when training image label quality dropped below 70%, YOLOv4 returned lower APs and mAPs for training datasets than validation and test datasets, while for all other models training datasets always had higher APs and mAPs than validation and test datasets. This phenomenon implies that YOLOv4 has a certain degree of "integrity" against imperfect ground truth, meaning even when YOLOv4 is trained by a mix of correct and incorrect ground truth, such that some instances in training images have labels and some do not, YOLOv4 might still be able to identify an instance that do not have a label, going against the incorrect ground truth.

3.3.2.3. Image Quantity vs. Label Quality

As mentioned in introduction, the need for substantially sized training datasets as well as the difficulty of labeling agricultural images pose a dilemma to researchers who would like to develop robust object detectors: given limited time and human annotators, whether to collect a large amount of images and label all of them with suboptimal quality, such as missing some instances in images, or collect a small dataset but concentrate on annotating each image as perfectly as possible. Figure 29a and 29b show the line plots of how the YOLOv4 performance changed with training image number and training image label quality respectively, as discussed in the previous two sections. When training dataset size decreased but image label quality stayed the same, YOLOv4 performance decreased (Figure 29a). When training image label quality decreased but dataset size stayed the same, YOLOv4 performance again decreased (Figure 29b). The common variable that changed in both scenarios was training instance number, suggesting it is likely a critical factor for objector detectors. The performances of the YOLOv4 models trained on the datasets with compromised image quantities or label qualities were then compared in terms of instance quantity, by calculating the total labeled instance numbers in the training datasets. Figure 29c shows how the two types of YOLOv4 models from Figure 29a and 29b compared to each other when their training instance numbers were equal. Although no substantial difference existed between the two types of models when their training instance numbers were larger than 30,000, high-label-quality-low-image-quantity models always outperformed high-image-quantity-low-label-quality models at the same instance number, especially when their training instance numbers were lower than 23,000. This result potentially has two implications. First, regardless of training instance number, training image label quality matters more than training image quantity. Second, regardless of training image quantity and label quality, higher training instance numbers can lead to better model performances. The second implication was further confirmed by Figure 29d, which shows the relationship between APs and instance numbers of the models from Figure 29a and 29b at each growth stage. Overall, APs and training instance numbers showed a strong positive correlation. APs of all growth stages drastically increased with instance numbers smaller than approximately 3,000. The rate of increase of AP started to decline when instance number exceeded approximately 4,000, indicating an increasing ratio of "work to reward", or image annotation workload to model performance improvement.

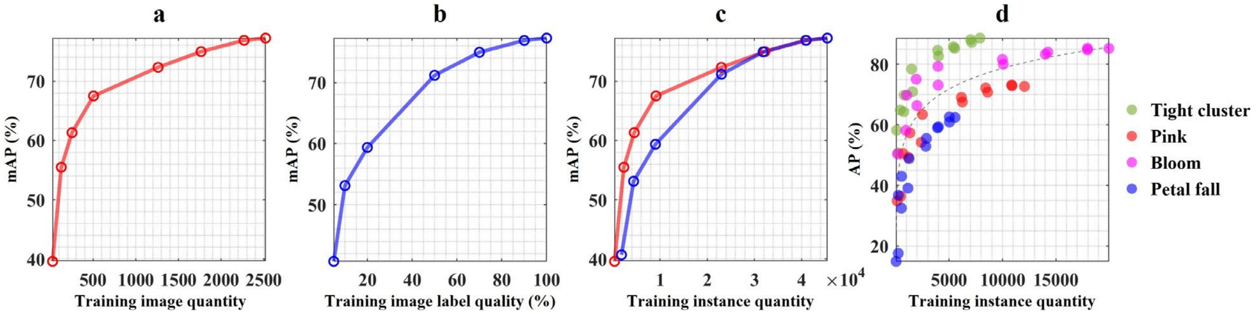

Figure 29. Relationships between training dataset size, training image label quality, training instance quantity and mAP and AP of YOLOv4 models on the original test dataset: (**a**) training image quantity vs. mAP; (**b**) training image label quality vs. mAP; (**c**) total training instance number vs. mAP where red points represent models trained with compromised dataset sizes (a) and blue points represent models trained with compromised image label qualities (b); (**d**) training instance number of each growth stage vs. AP with a fitted two-term power series trend line for the models in (a) and (b).

3.3.2.4. Negative Sample Presence

The model trained with 100% of the original positive samples or 1260 images had a 77.10% mAP, which was almost identical to the baseline model's 77.25% mAP. The model trained with only 1% of the original positive samples or 12 images had a 36.64% mAP, which was very close to the mAP of the model trained with 1% of both the original positive and negative samples, 39.63%. Although negative samples seemed to marginally improve model mAP, certain growth stages actually had lower APs when negative samples were utilized during training. Overall, the results did not suggest any substantial positive influences of negative samples on model performance, and negative samples do not appear to be a necessary component for the training of object detectors, at least in the context of apple flower bud classification using YOLOv4.

3.3.2.5. Training Image Order

No substantial performance difference between the model trained on date-ordered dataset and the baseline model was observed, whose mAPs were 77.51% and 77.25% respectively. The results refuted the speculation regarding the potential influence of training image sequence on object detector classification accuracies.

3.3.2.6. Noise Retraining

Generally two strategies were employed to improve YOLOv4's robustness against artificial image distortions: training with datasets having all images distorted to the baseline-model-failing noise levels, and training with datasets having image distorted with equally distributed noise levels from no noise to the baseline-model-failing noise levels. The results indicated that incorporating distorted images into model training was able to improve all models' classification accuracies against image noises by 13 to 390% when compared to the baseline model (Table 8). However, YOLOv4 generally performed the best on a test dataset when it was trained on a similar training dataset in terms of noise type and level. As a few examples, the baseline model had the best mAP on the original test dataset than any other model since it was trained on the original training dataset, the model trained on the training dataset with 25-pixel motion blur had the best mAP on the test dataset with 25-pixel motion blur compared to other models, the model trained on the training dataset with 90%-decrease to 0 to 900%-increase evenly distributed saturation distortion had the best mAP on the test dataset with 90%-decrease to 0 to 900%-increase evenly distributed saturation distortion compared to other models, etc.

Table 8. The mAPs of retrained YOLOv4 models on test datasets with different image distortion types and levels. “Original” refers to the original undistorted dataset, “Extreme” refers to the baseline-model-failing noise levels, and “Even” refers to evenly distributed noise levels from no noise to the baseline-model-failing noise levels.

Distortion	**Training and validation datasets**	**Test Dataset mAP (%)**			
		Original	Extreme		Even
			Decrease	Increase	
White noise	Original (OD)	77.25	15.17		54.71
	Extreme (WN40%)	71.79	74.40		73.29
	Even (WN0-40%)	77.03	73.84		76.15
Motion blur	Original (OD)	77.25	23.28		59.53
	Extreme (MB25Pixel)	54.27	72.08		67.85
	Even (MB0-25Pixel)	75.82	70.45		74.90
Hue shift	Original (OD)	77.25	NA		25.76
	Even (HS0-360°)	74.85			74.86
Saturation change	Original (OD)	77.25	29.31	29.74	61.89
	Extreme decrease (SD90%)	42.41	75.44	0.04	26.33
	Extreme increase (SI900%)	35.65	12.32	75.23	48.89
	Even (SD90%-0-SI900%)	77.23	68.84	75.36	76.23
Intensity change	Original (OD)	77.25	25.69	27.43	65.50
	Extreme decrease (ID80%)	48.13	76.42	0.97	34.49
	Extreme increase (II180%)	31.91	0	69.59	36.16
	Even (ID80%-0-II180%)	76.34	73.64	64.97	73.89

Training with datasets having evenly distributed noise levels seemed to be an effective strategy to build object detectors that are good at recognizing both normal and distorted images, while training with datasets having all images distorted to the extreme levels only led to models that were good at detecting heavily distorted images. As shown in Table 8, for all five types of noises, the models trained with evenly distorted datasets have much better overall performance on the three types of test datasets than the baseline model as well as the models trained with the baseline-model-failing levels of noises. Interestingly, hue did not seem to be an important information source for YOLOv4 to identify apple flower buds. The model trained on the random-hue dataset had a 74.85% mAP on the original test dataset, which was only 3% lower than the baseline model’s 77.25% mAP.

The results suggested that in a practical sense, training an object detector “specifically” for its applications seems to be the tactic that researchers should follow. For example, if an object detector will only be applied to high quality image data without noise, there is no need to train the model for noise. When an object detector is expected to be utilized on identifying noisy images such as blurred pictures taken by a UAV, it is important to expose the expected noise types with expected noise levels to the model during training. As demonstrated in this study, artificially distorting training images can be a way of improving model robustness against image noise. From the perspective of data collection, researchers should be mindful about ground truth annotation difficulty when they plan to directly collect noisy training images. Unlike white noise and intensity change that are naturally straightforward for humans to discern, certain noises such as blurriness can easily distort objects in images to the unrecognizable extent for humans. Therefore, when possible, capturing high quality images, labeling instances in the images, and artificially distorting the training and validation images might be a more efficient protocol of improving the performance of object detectors against image noise.

3.3.2.7. Ultimate Test

Due to the extreme levels of distortions, the baseline model was only able to have a 4.56% mAP on the ultimate test dataset, while the model trained on the ultimate dataset achieved a 54.81% mAP. The model that was trained on the combination of five types of evenly distorted datasets had a 35.24% mAP, which was substantially higher than the mAP of the baseline model but still considerably lower than the mAP of the other model. This result again confirmed that researchers should train their object detectors specifically for applications. If a model will be applied on datasets having multiple types of distortions in each image, training dataset should also have the same types and levels of distortions in each image.

3.4. Conclusions

The following statements were made in the context of classifying apple flower bud growth stages using YOLOv4. The baseline model that was trained on a normal dataset demonstrated sufficient robustness against image distortions from white noise, hue shift, saturation change, and intensity change. Practically, motion blur seemed to be a concerning image noise source, especially when UAVs are involved in data collection. Negative samples and training image sequence were proven to have no considerable influence on model classification accuracies. Annotating small datasets with high ground truth quality showed its superiority over annotating large datasets with low ground truth quality. Nonetheless, training instance quantity is a critical factor that determines the performance of an object detector, regardless of training dataset size and training image label quality. Annotating at least 3,000 to 4,000 instances of each class for complex agricultural scenes is recommended to achieve optimal object detection results. Before training a model, considering model applications first can help design appropriate training datasets, which should resemble test datasets as much as possible in terms of image distortion types and levels on both individual image level and whole dataset level. Incorporating artificially distorted images during training might be a viable and efficient strategy to improve model robustness against image noise.

CHAPTER 4
ORCHARD NAVIGATION MAP SYNTHESIS

4.1. Introduction

Many advantages of current off-the-shelf UAVs, such as low-cost, compact size, robust hardware, and user-friendly software, ensure flight and data collection flexibility and promote the growing adoption of UAVs in agriculture. Aside from the various types of imagers such as RGB cameras [88], thermal cameras [89], multispectral cameras [90], and hyperspectral cameras [91], LiDAR scanners emerge as another instrument that are being often installed on a UAV platform. Because of the hardware advancement over the years, LiDARs have evolved from capturing only a single return per laser pulse to recording the entire returned waveform [92], and their sizes and costs have also decreased significantly, allowing advanced processing methods with increased pulse detection reliability, accuracy, and resolution as well as easier integration with UAVs and wider sensor adoption and application.

Since a LiDAR measures distances relative to itself, when it is in motion, such as being mounted on a flying UAV, additional sensor inputs or point cloud registration techniques are required to align raw LiDAR data. As the most common method, the combination of GNSS and inertial measurement unit (IMU) provides location and orientation information that allows point clouds to be registered to a geographic coordinate system. GNSSs are able to locate receiver positions continuously worldwide utilizing radio signals from at least four satellites, while IMUs measure attitude angles via built-in accelerometers, gyroscopes, and sometimes magnetometers. As an alternative method, point cloud registration algorithms have the potential to align LiDAR data even under GNSS-denied conditions [93], and some of them when implemented in frameworks that are able to construct and update geometrically consistent maps in an unknown environment with real-time speed are often categorized as simultaneous localization and mapping (SLAM) algorithms. SLAM is a fundamental research area in robotics as it is promising in solving challenges related to autonomous navigation of artificial intelligence robots. Aside from LiDARs, many other exteroceptive sensors such as sonars, monocular cameras, stereo cameras, and RGB-depth (RGB-D) cameras and proprioceptive sensors such as odometers and compasses can also be utilized within SLAM methods to map environments and ascertain robot poses [94].

The application of UAV-LiDAR systems in plant-related studies started to boom in the past two to three years, since remote sensing-based methods have the natural advantages of scanning large areas such as agricultural or forested fields with low time and labor costs. Many agricultural studies have experimented with directly estimating or indirectly predicting plant parameters with UAV-LiDAR systems while using manual measurements as ground truths to assist timely crop management. Liu et al. [95] obtained relative errors of 3.86% to 12.73% for cotton height using the average height of the highest five points in a regional point cloud to represent crop height. Zhou et al. [96] achieved a coefficient of determination (R^2) of 0.96 and a root mean square error (RMSE) of 0.13 m for the height of maize plants with different degrees of lodging through canopy height models (CHMs). Hu et al. [97] studied maize self-recovery ability after lodging and attained R^2s of 0.95 and 0.98 and RMSEs of 0.06 m and 0.13 m for plant heights at tasseling and filling stages. Lei et al. [98] estimated leaf area index (LAI) of maize at upper, middle, and lower layers and obtained normalized RMSEs (NRMSEs) of 10.8%, 12.4%, 42.8% respectively using voxel-based point cloud processing. Luo et al. [99] used two methods for crop height estimation, namely CHM and optimal point density, which resulted in R^2s of 0.65 and 0.83 and RMSEs of 0.24 m and 0.16 m for maize, and R^2s of 0.4 and 0.48 and RMSEs of 0.09 m and 0.08 m for soybean. Phan and Takahashi [100] calculated rice height by taking the difference between 1st and 95th point cloud percentile and found an R^2 of 0.85 and an RMSE of 0.07 m. Xu et al. [101] explored six modeling techniques to predict sugarcane above ground fresh weight, and their best results were an R^2 of 0.97 and an RMSE of 1.33 kg/m^2. Sofonia et al. [102] exported sugarcane heights from the point cloud processing software 3DReshaper (TECHNODIGIT, France) and got an R^2 of 0.91 for max block height and an R^2 of 0.93 for mean block height. Later they predicted sugarcane biomass with an R^2 of 0.52 using LiDAR-derived predictors [103]. Masjedi et al. [104] reported their sorghum biomass predictions multiple times. Using recurrent neural network they obtained NRMSEs of 7.5% to 15%; using support vector regression (SVR) and multi-layer perception while combining LiDAR and hyperspectral data, they stated R^2s of 0.07 to 0.48 and 0.19 to 0.55 [105]; using partial least squares regression (PLSR), SVR and random forests (RF) and having RGB, LiDAR, and hyperspectral features as inputs, they achieved R^2s of 0.64 to 0.89 [106]. Maimaitijiang et al. [107] estimated sorghum height by taking the 1st and 99th plot-level point cloud percentile difference and attained an R^2 of 0.98 and an RMSE of 0.05 m. They also predicted sorghum LAI with different canopy density through RF and obtained R^2s of

0.42 to 0.95 and RMSEs of 0.27 to 0.32. Dhami et al. [108] extracted wheat plot heights using three methods including voxel-filtered max height, non-voxel-filtered max height, and 99th percentile height, and reported error ranges of -0.5 to 0.1 m, -0.05 to 1.05 m, and -0.15 to 0.2 m respectively. Ten Harkel et al. [109] averaged top 10, 90, and 30 points of potato, sugar beet, and winter wheat point clouds and achieved R^2s of 0.5, 0.7, and 0.78 and RMSEs of 0.12 m, 0.07 m, and 0.03 m for plant height. Their biomass estimations for the three crops through a 3-dimensional profile index model had R^2s of 0.24, 0.68, and 0.82 and NRMSEs of 22.09%, 17.47%, and 13.94%. Hadas et al. [110] were able to detect more than 99% of apple trees in an orchard and maintained a crown area RMSE of 1 m^2 and a tree top location RMSE of 0.38 m. Their RMSEs for tree height and crown base height were both 0.09 m while the R^2s were 0.92 and 0.49. Aside from agriculture, forestry is another important plant-related field where UAV-LiDAR systems are commonly employed. Replacing traditional laborious manual measurements, tree parameters now can be extracted or predicted through aerial point cloud-derived metrics with ease, and some of the relevant works can be found in [111–128].

Given UAV-LiDAR systems' promising capability of assessing vegetation structural attributes as demonstrated in the reviewed literature, aerial LiDAR data of apple orchard were collected and utilized to extract field locations that were safe for UGV traveling, for the purpose of potentially guiding the navigation of autonomous UGV-based heating for frost protection in the future. The objectives of the study included: (1) developing a low-cost UAV-LiDAR system integrating a single-board computer and UAV's built-in GNSS units and IMUs; (2) generating binary orchard obstacle and open space maps from georeferenced orchard point clouds; (3) simulating orchard navigation maps that can be deployed to guide UGVs in terms of both path planning and heat application.

4.2. UAV-LiDAR System Development

4.2.1. Hardware

The UAV platform chosen was DJI Matrice 600 Pro. Its 6 kg payload capacity ensured that the UAV could sufficiently complete flights under 15 min even with a LiDAR and an external power source. The UAV was equipped with DJI's A3 Pro flight control system, which featured three sets of global positioning system (GPS) Compass Pro modules and IMUs and allowed the UAV to hover with ±1.5 m horizontal and ±0.5 m vertical accuracies. The A3 Pro was also compatible with DJI Onboard SDK (OSDK) which enabled users to retrieve onboard sensor readings such as GNSS and IMU data and control the UAV with customized programs in an onboard computer. The LiDAR scanner employed on the UAV was Puck LITE (Velodyne Lidar, USA), formerly known as VLP-16. It recorded distance and reflectivity information through 16 lasers with a 903 nm wavelength, and had a vertical field of view (FOV) of 30° with a 2° resolution and customizable horizontal FOV range and resolution. Depending on the return mode, the LiDAR could take up to 578,688 laser measurements per second with a ±3 cm range accuracy. A Raspberry Pi 4 Model B (Raspberry Pi Foundation, UK) was selected as the low-cost onboard computer that communicates with the UAV and the LiDAR. The computer had a 1.5 GHz quad-core processor and 8 GB of RAM. A 128 GB micro-SD card was used for operating system loading and data storage. A 21,000 mAh power bank BP101 (BEATIT, China) was used to power the LiDAR and the computer simultaneously, which provided both 12 V and 5 V DC outputs. A summary of hardware key specifications can be found in Table 9.

Table 9. Key specifications of the UAV-LiDAR system components.

Hardware	Model	Key Specifications
UAV	DJI Matrice 600 Pro	6 kg payload, 16 to 32 min hovering time, ±0.5 m vertical and ±1.5 m horizontal hovering accuracy
LiDAR	Velodyne Puck LITE	16 lasers with a 903 nm wavelength, 30°×360° FOV, 2° vertical resolution, 0.1° to 0.4° horizontal resolution, 100 m measurement range, ±3 cm range accuracy, max 578,688 measurements per second
Onboard computer	Raspberry Pi 4 Model B	1.5 GHz quad-core processor, 8 GB SDRAM, 128 GB storage, gigabit Ethernet, 2.4/5.0 GHz Wi-Fi
Battery	BEATIT BP101	21,000 mAh, 12 V 10 A DC output, 5/9/12 V USB output

The installation of all hardware components to the UAV was achieved by a piece of steel sheet that could be attached to and detached from the UAV bottom via mounting brackets (Figure 30a). Aside from the LiDAR which was mounted on a 3D-printed plate so that its lasers pointed towards ground during data collection, the rest of the components were directly fixed onto the sheet metal. Since the A3 Pro could only utilize signals from one GNSS unit at a time and the active GNSS unit could change for different flights, in order to keep the lever arm offsets [129] between the LiDAR and the active GNSS unit constant, two sets of GNSS units were disconnected from the UAV (Figure 30b).

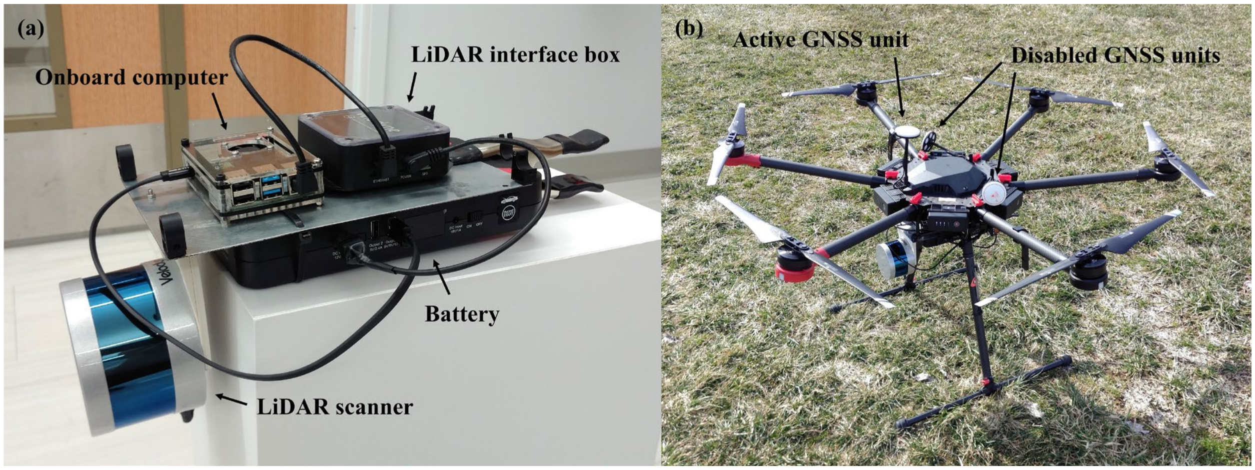

Figure 30. The UAV-LiDAR system: (**a**) the LiDAR payload; (**b**) the UAV platform with the LiDAR installed.

4.2.2. LiDAR Configuration

The LiDAR was able to provide a 360° horizontal view utilizing an internal motor. Here LiDAR data belonging to the same motor rotation from 0° to 360° is defined as one LiDAR "frame". Since it is impossible for the LiDAR to remain perfectly stationary in the air during data collection, theoretically speaking unique GNSS and IMU measurements are needed for every single LiDAR point's georeferencing. Considering the low UAV flying speed of regular flight missions and UAV hardware limitations such as GNSS update rate, in-frame LiDAR motion distortion was ignored. However, to mitigate the distortion, the LiDAR motor was configured with the fastest rotational speed, which was 1200 RPM and corresponded to a 0.4° horizontal resolution. As the aerial application implied that the LiDAR would always be above target plants, the horizontal FOV of the LiDAR was configured as 135° to 225°, ignoring background data generated from objects above the UAV (e.g. insects, birds). The LiDAR was also capable of capturing more than one return from each laser firing. The LiDAR was configured with Dual Return mode, which recorded the strongest and last returns.

4.2.3. Hardware Communication

The LiDAR and the onboard computer were connected via a Cat6 Ethernet cable, and the data transfer between them was completed through User Datagram Protocol (UDP). For each flight, collected LiDAR data were saved in a unique pcap file. The A3 Pro was connected to the onboard computer via a USB to TTL serial cable. After setting up the software environment in the onboard computer for OSDK, a customized C++ program was used to access A3 Pro telemetry data, which were saved in a txt file at 50 Hz during data collection. Descriptions of the telemetry parameters are listed in Table 10.

Table 10. UAV telemetry parameters recorded during data collection [131].

Parameter	Unit	Accuracy	Sensor Source	Note
Latitude	rad	±1.5 m	GNSS, IMU	GNSS and IMU fused position
Longitude	rad			
Altitude	m	±0.5 m	GNSS, IMU, barometer	GNSS, IMU and barometer fused height from sea level using the International Civil Aviation Organization (ICAO) model
Quaternion w	N/A	Pitch and roll ±0.5°, yaw ±1.5°	IMU	Ground frame to UAV frame rotation
Quaternion x				
Quaternion y				
Quaternion z				

The onboard computer had an Ubuntu MATE 20.04.1 operating system, which was responsible for controlling data collection through customized Bash programs as well as storing collected data. The synchronization between LiDAR frames and UAV telemetry readings was achieved utilizing the computer's internal Unix time. During flights, the onboard computer was remotely controlled by a Windows 10 laptop hosting an Ubuntu 18.04.3 LTS virtual machine to start and end data collection. In the field the two computers were connected via Wi-Fi and communicated with each other based on Secure Shell Protocol (SSH). Figure 31 summarizes the connection and communication between all hardware components.

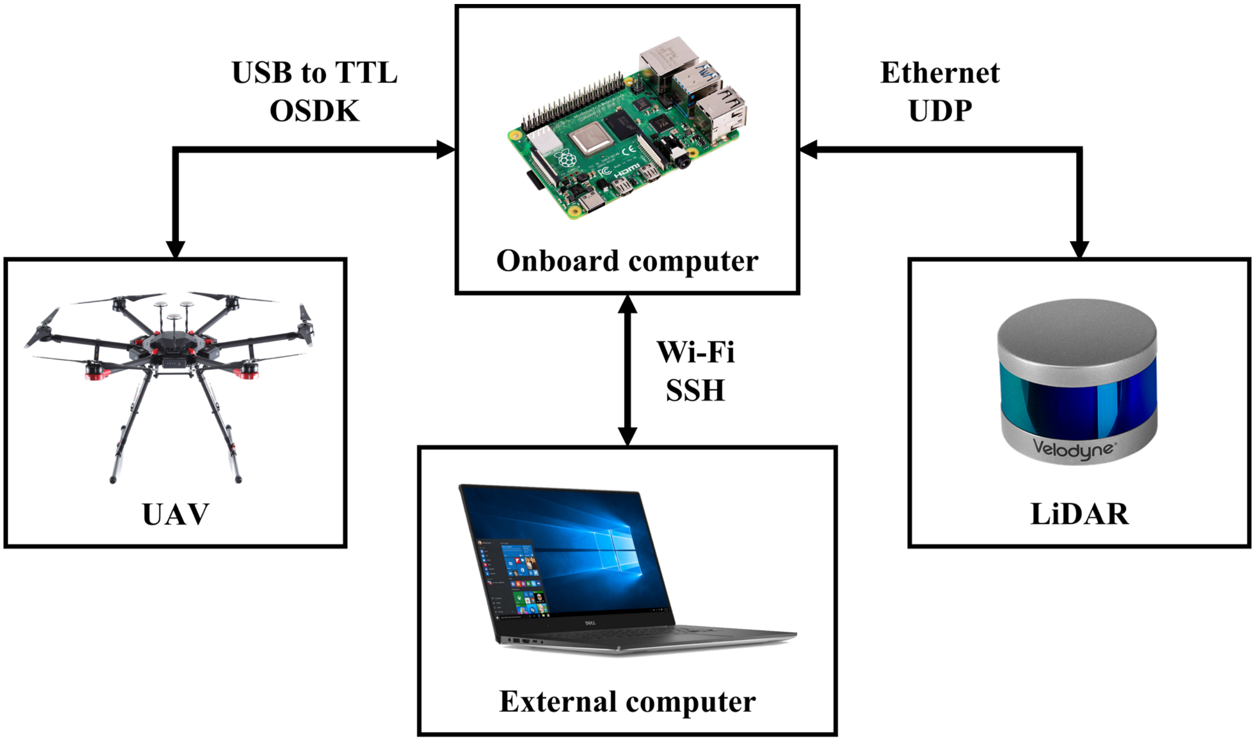

Figure 31. Connections and communications between hardware components of the UAV-LiDAR system.

4.3. Methods

4.3.1. LiDAR Data Collection

LiDAR point clouds of the same apple orchard where the previous studies were conducted, were collected throughout the growing season of 2021. To minimize redundancy, single-pass flight missions following the tree row directions were conducted at 20 m AGL (Figure 32), since the LiDAR could cover the four rows of trees simultaneously during a flight. The UAV-LiDAR system flew at a 1 m/s speed and completed a typical flight mission within a minute. Approximately 500 LiDAR frames were collected in each flight. The LiDAR dataset used for demonstrating obstacle maps, open space maps, and navigation maps was collected on March 21, 2021.

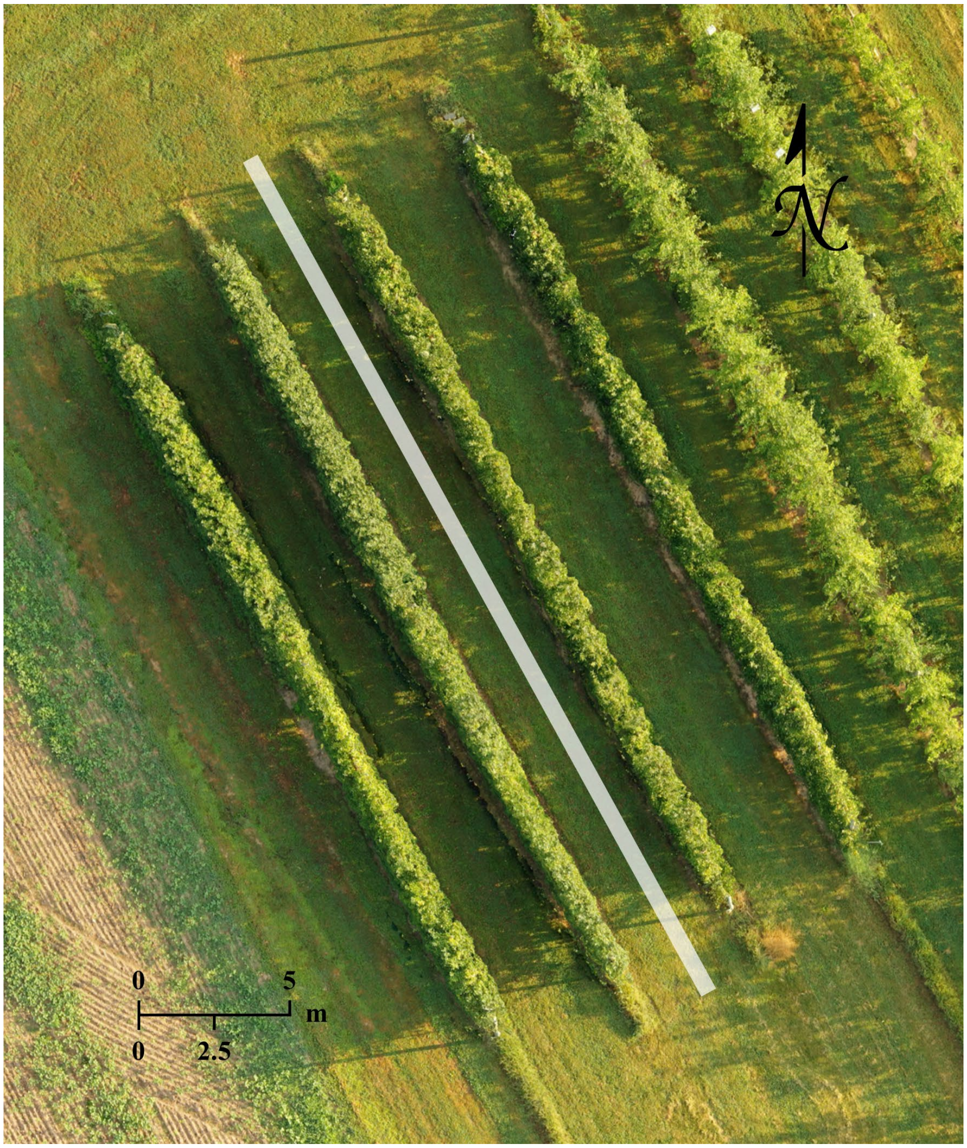

Figure 32. Flight path of the LiDAR flight missions conducted over the apple orchard.

4.3.2. LiDAR Data Georeferencing

By default, the coordinate systems of geography, the LiDAR and IMU of the UAV-LiDAR system, and typical point cloud viewing software do not match with each other (Figure 33), and axis redefining is constantly needed for data processing and point cloud viewing. Before georeferencing, the coordinate systems of UAV telemetry and LiDAR data were redefined to match with the software coordinate system, or geographic Cartesian coordinate system with flipped Easting axis.

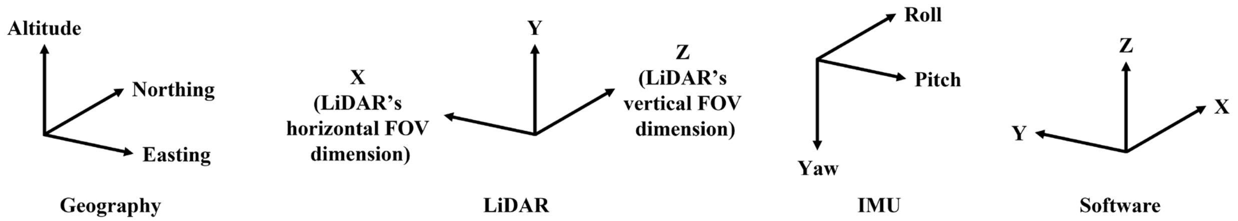

Figure 33. Illustration of coordinate system mismatches between geography, LiDAR, IMU and point cloud viewing software assuming the UAV-LiDAR system is level, its heading points true north, and no boresight errors exist between LiDAR and IMU.

Given the axis-adjusted LiDAR frames and their corresponding UAV telemetry readings, LiDAR data georeferencing was completed using the following equation:

$$\begin{bmatrix} N_G \\ E_G \\ A_G \end{bmatrix} = R_A R_B \begin{bmatrix} X_R \\ Y_R \\ Z_R \end{bmatrix} + R_A \begin{bmatrix} L_X \\ L_Y \\ L_Z \end{bmatrix} + \begin{bmatrix} N_R \\ E_R \\ A_R \end{bmatrix}, \tag{3}$$

where (X_R, Y_R, Z_R) are the raw coordinates of a point in an axis-adjusted LiDAR frame, (N_G, E_G, A_G) are the georeferenced coordinates of the point, (N_R, E_R, A_R) are the Universal Transverse Mercator (UTM) coordinates converted from a set of raw GNSS latitude, longitude, and altitude readings, R_A is a rotation matrix derived from IMU attitude angles (A_R, A_P, A_Y) in Euler angle format converted from quaternions [132], R_B is a rotation matrix derived from the boresight angles (B_R, B_P, B_Y) in Euler angle format [132], (L_X, L_Y, L_Z) are the lever arm offsets between LiDAR scanner center and GNSS receiver. The boresight errors between the IMU and the LiDAR of the UAV-LiDAR system were estimated utilizing orchard ground as a calibration field [133], while the lever arm offsets between the GNSS receiver and the LiDAR were manually measured [129]. Final orchard point clouds were generated using MATLAB R2021a by combining georeferenced LiDAR frame data from the same data collection dates into the same coordinate system (Figure 34).

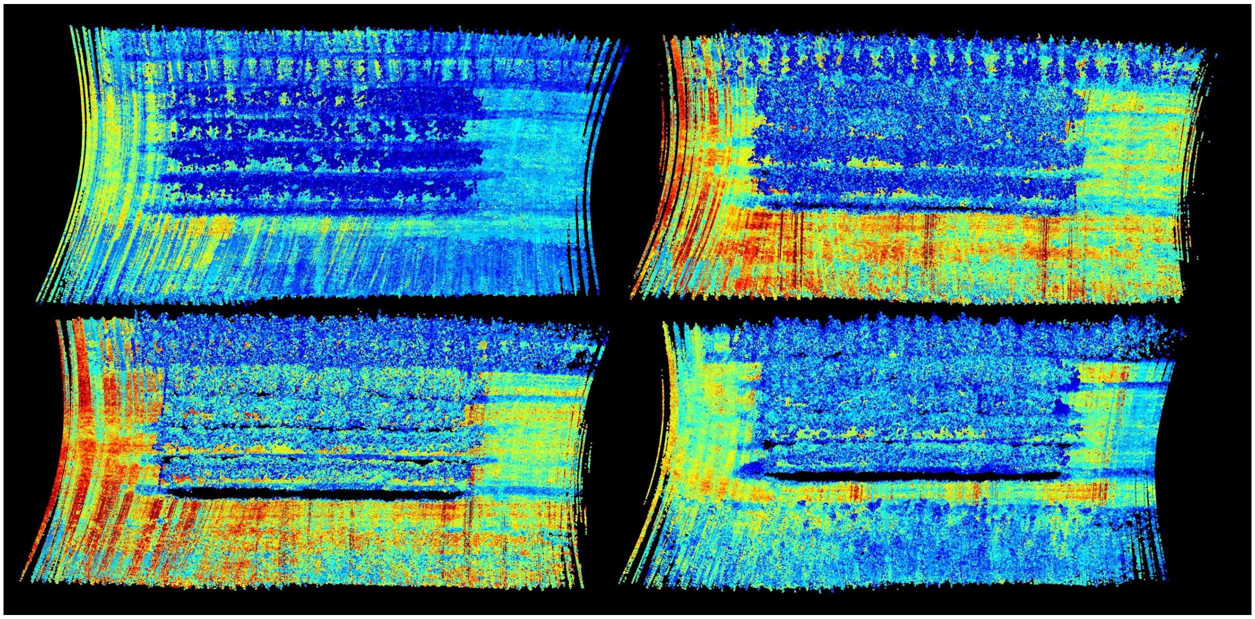

Figure 34. Examples of georeferenced orchards point clouds collected in the growing season of 2021. Point color represents near-infrared reflectivity.

4.3.3. Orchard Height Maps

To find out the orchard areas where the heights of obstacles such as vegetations and artificial structures are sufficiently low for safe UGV traveling, a threshold-based height differencing approach was utilized to synthesize orchard obstacle and open space maps from the georeferenced orchard point clouds. An area was assumed to be obstacle-free if the height difference between the highest point and the lowest point within the area was below a specified threshold.

Given an orchard point cloud, an empty 2D map was first initialized as the template for storing point cloud information. In the map, each grid or pixel represented a unit area whose size was customizable and chosen to be 0.01 m^2. The center of each row of pixels represented a constant northing, and the center of each column of pixels represented a constant easting. The map size depended on the easting and northing ranges of the point cloud. Two height maps were generated by filling relevant information of the point cloud into the empty map template: a maximum height map where the intensity of each pixel represented the highest height within each pixel (Figure 35a), and a minimum height map where the intensity of each pixel represented the lowest height within each pixel (Figure 35b). A height difference map was generated by subtracting the minimum height map from the maximum height map (Figure 35c). The open space map, or the inverse of an obstacle map of the orchard was synthesized by a two-step procedure: first identifying obstacle pixels with large within-pixel height differences by thresholding the height difference map with a specified value, which was selected as 1 m for demonstration, and then using an image morphological closing operation to close the pixels and form complete obstacle areas that UGVs should avoid during navigation. An optional dilation operation was applied to the initial obstacle map to expand the overall obstacle regions, which ensured UGVs would be far from any object that might potentially influence its safe traveling. The final open space map was simply the inverse the dilated obstacle map.

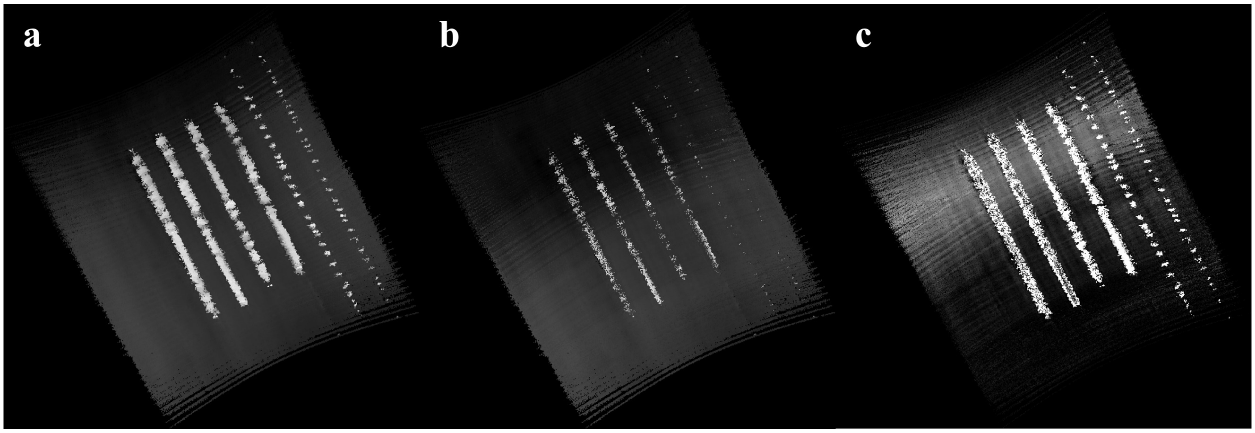

Figure 35. Examples of orchard height maps derived from the same orchard point cloud: (**a**) maximum height map, (**b**) minimum height map, and (**c**) height difference map.

4.3.4. Navigation Map

The open space map itself is sufficient for preventing UGVs from crashing into objects during traveling. However, it does not provide a destination for UGVs in terms of where to apply heat treatments and how long to stay at a region while applying heat. Therefore, a valid orchard navigation map needs to contain both the open space information and the heating requirement information. A simulated navigation map was generated by fusing obstacle and open space maps with a simulated heating requirement map (Figure 16). As the easting and northing coordinates of each pixel in both the obstacle map and heating requirement map were known, the correspondence between the heating requirement pixels and the obstacle pixels were established based on the association between the UTM coordinates of the two maps. Since the heating requirement map had a much higher spatial resolution (3.23×10^{-4} m^2/pixel) than the obstacle map (0.01 m^2/pixel), the heating requirement pixels that belonged to the same obstacle pixel were first identified by a searching algorithm based on whether their pixel UTM coordinates were located within the obstacle pixel area, and their values were then averaged and used in the navigation map. All map processing and synthesis were performed in MATLAB R2021a.

4.4. Results and Discussions

4.4.1. Obstacle and Open Space Maps

Using the collected LiDAR frame data and UAV telemetry data, a georeferenced orchard point cloud that contained roughly 1.7 million points could be generated in 40 s (Figure 34). After obtaining the height difference map, which could be created in seconds, the obstacle and open space maps of the orchard could be synthesized quickly as shown in Figure 36.

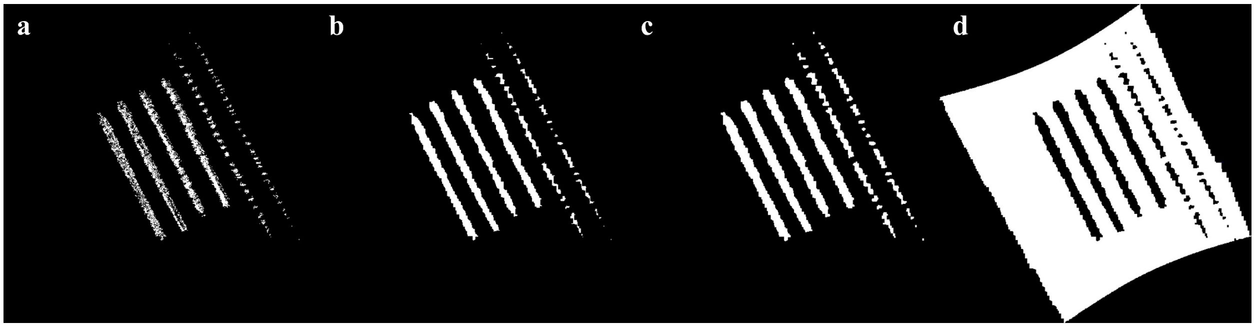

Figure 36. Examples of orchard obstacle and open space maps: (**a**) initial obstacle map with disconnected pixels; (**b**) closed obstacle map with complete obstacle regions; (**c**) dilated obstacle map with expanded obstacle regions; (**d**) open space map.

As indicated in Table 10, the navigation units of the UAV do not output highly accurate GNSS data. One way to improve orchard point cloud georeferencing accuracy is to correct UAV GNSS signals through techniques such as real time kinematic (RTK) and post processed kinematic (PPK) with the help of ground-based reference stations. However, by dilating the closed obstacle map to the degree where the obstacle regions in the map include any true obstacle location even with the maximum possible location errors reported by the UAV's GNSS units (e.g. ±1.5 m), UGV navigation can be safely conducted according to the open space map. Similar to flower bud growth stage maps, it is ideal to have orchard obstacle and open space maps prepared before a frost event. However, considering the slow growth of tree structures, obstacle and open space maps might only need to be updated weekly or even monthly.

4.4.2. Navigation Map

Figure 37a shows the result of fusing the obstacle map and the simulated heating requirement map based on their corresponding UTM coordinates. Figure 37b shows an updated navigation map for viewing convenience where blue pixels represent orchard open space, black pixels represent obstacles or unexplored areas, and yellow-red pixels represent tree regions with specified heating requirements.

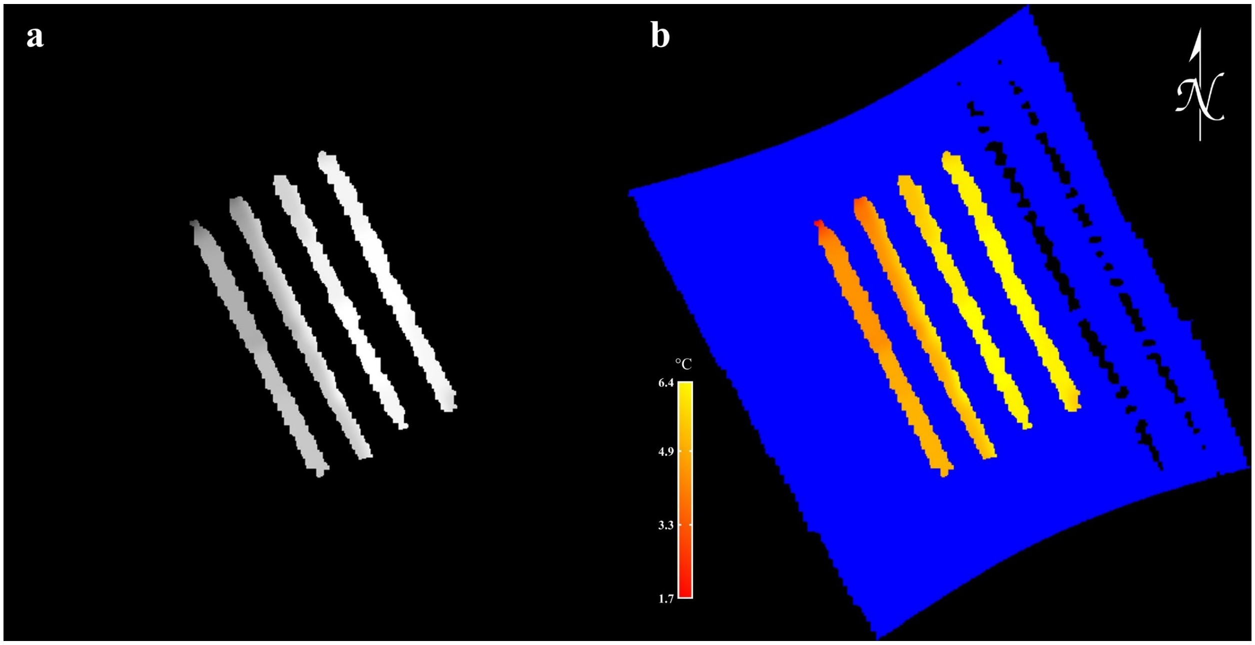

Figure 37. Simulated orchard navigation map examples: (**a**) fusion result of the dilated obstacle map and the heating requirement map; (**b**) final navigation map containing obstacle, open space, and heating requirement information of the orchard.

Depending on the resolution of the heating requirement map and obstacle map, generation time of the final navigation map may vary. Using the developed searching algorithm, the fusion between the original heating requirement map and obstacle map costed about five minutes. For the purpose of determining heat application requirements in orchards, even if the heating requirement map is downscaled significantly such that it has the same or lower spatial resolution than the obstacle map, it would still be sufficient for guiding UGVs in applying heat treatments to trees. Consequently, the generation time of the navigation map could be dramatically reduced, allowing the possibility of real-time orchard navigation map adjustment as a UGV continuously applies heat to the orchard. For example, when the heating requirement map was resized to the same spatial resolution as the open space map, the searching algorithm only needed 0.38 s to generate the navigation map. Alternative approaches could also be utilized to fuse heating requirement maps and obstacle maps, such as resizing and cropping the map with higher resolution to match the map with lower resolution, and masking the heating requirement map with the obstacle map after establishing their pixel correspondence based on their UTM coordinates.

4.5. Conclusions

UAV-LiDAR systems were shown to be a promising tool for fast and convenient plant structure estimation, which can be developed at a much lower cost than purchasing current off-the-shelf aerial LiDAR payloads. Point spatial information of a point cloud alone was sufficient for identifying and mapping obstacle regions of agricultural environments. Utilizing georeferenced orchard point clouds and heating requirement maps, orchard navigation maps that provide orchard open space and tree heating requirement information can be generated in a timely fashion, which relieves UGVs from the need of having dedicated sensors and sophisticated algorithms for object avoidance during navigation.

CHAPTER 5
CONCLUSION AND FUTURE WORK

The research work presented in this dissertation pioneered the concept of “precise frost management”, which has never been discussed in existing literature. During a natural disaster such as frost when management actions need to be quickly carried out to prevent crop damage, a CPS that can spontaneously respond to critical environmental conditions demonstrates much potential as a future research direction. Combining the power of thermal imagery, RGB imagery, LiDAR point clouds, and data processing algorithm such as feature-based image registration algorithm, CNN-based object detector, and GNSS and IMU-based georeferencing, UAV was proven to be a valuable sensor-carrying platform for efficient plant status monitoring and assessment, as well as a promising component of a responsive frost managing CPS for orchards.

The original contributions of this dissertation to research community mainly comprised three aspects. First, aside from plant temperature, the consideration of plant growth stage and critical temperature variation helped with minimizing unnecessary heat treatments during frost events. The development and evaluation of the apple flower bud classifier demonstrated the feasibility of replacing humans with machines for plant growth monitoring. Second, the utilization of UAV-borne thermal cameras and the developed thermal image stitching algorithm allowed quick and detailed assessment and visualization of orchard temperature variation. Lastly, the integrated mapping framework harnessed multi-sensor information and generated georeferenced maps that could be easily interpreted by ground heating vehicles for both path planning and heat treatment application, achieving convenient and efficient orchard sensing and potential managing to minimize frost damage.

Although this dissertation has laid an important foundation of a fully-integrated, autonomous CPS for precise apple orchard frost protection, much work is still needed in follow-up studies to realize the system. With the georeferenced orchard navigation maps generated by the multi-dimensional mapping framework, the development of a UGV-based heating system, the integration and communication between UAV and UGV to form a complete CPS, and the optimization of heating logistics during frost events to minimize low temperature plant damage are the three major areas that require extensive research and engineering. The UGV-based heating system, which should consist of multiple ground heating vehicles, needs to have the capability of navigating precisely and promptly to designated orchard locations based on GNSS coordinates, as well as adjusting heater powers and angles depending on orchard temperature deficits and weather conditions. The communication between UAV and UGV, which will likely require a computer serving as a mediator to receive and process data from UAV and send commands to UGV, should happen in real time with a robust wireless connection. Lastly, the heating logistics that determine the sequence of locations where UGVs need to apply heat treatments, should not only potentially take orchard temperature deficit, topography, weather, and tree structure into consideration, but also be able to decide when to turn on or off, or up or down the heaters to achieve maximum fuel utilization efficiency.

APPENDIX

FULL AP AND MAP RESULTS OF ALL YOLOV4 MODELS

Table A1. The baseline model on original datasets.

Dataset		OD	OD	OD
		Training	Validation	Test
AP	Tight cluster	93.9	88.54	88.63
	Pink	85.5	75.32	72.69
	Bloom	92.28	85.4	85.23
	Petal fall	78.95	62.55	62.45
mAP		87.66	77.96	77.25

Table A2. The baseline model on white noise added datasets.

White noise		WN10%	WN20%	WN30%	WN40%
Dataset		Test	Test	Test	Test
AP	Tight cluster	85.27	52.79	16.42	4.06
	Pink	71.51	61.73	45.03	25.95
	Bloom	82.51	64.73	28.32	8.48
	Petal fall	60.09	54.51	41.89	22.19
mAP		74.84	58.44	32.91	15.17

Table A3. The baseline model on motion blurred datasets.

Motion blur		MB5Pixel	MB10Pixel	MB15Pixel	MB20Pixel	MB25Pixel
Dataset		Test	Test	Test	Test	Test
AP	Tight cluster	87.01	80.72	69.24	57.49	46.15
	Pink	68.71	60.31	47.61	33.66	23.56
	Bloom	82.15	71.7	55.62	33.99	18.65
	Petal fall	59.64	50.96	35.95	16.16	4.74
mAP		74.37	65.92	52.1	35.32	23.28

Table A4. The baseline model on hue shifted datasets.

Hue shift	Counterclockwise	HS20°	HS30°	HS40°	HS50°	HS60°	HS120°	HS180°
	Clockwise	HS340°	HS330°	HS320°	HS310°	HS300°	HS240°	HS180°
Dataset		Test	Test	Test	Test	Test	Test	Test
AP	Tight cluster	84.69	78.84	65.9	45.94	31.7	7.55	0.57
	Pink	67.35	58.55	40.5	24.08	13.59	1.48	0.18
	Bloom	81.63	77.78	73.07	67.15	60.23	7.92	3.83
	Petal fall	57.12	52.81	46.62	37.12	27.67	5.32	0
mAP		72.7	67	56.52	43.57	33.3	5.57	1.14

Hue shift	Counterclockwise	HS240°	HS300°	HS310°	HS320°	HS330°	HS340°
	Clockwise	HS120°	HS60°	HS50°	HS40°	HS30°	HS20°
Dataset		Test	Test	Test	Test	Test	Test
AP	Tight cluster	0.72	38.56	50.39	62.07	74.22	83.43
	Pink	0.45	28.85	42.96	55.09	64.25	69.91
	Bloom	7.88	65.6	72.27	77.46	80.64	83.27
	Petal fall	0.53	42.38	48.14	52.68	56.66	59.09
mAP		2.39	43.85	53.44	61.82	68.94	73.93

Table A5. The baseline model on saturation decreased datasets.

Saturation decrease		SD10%	SD30%	SD50%	SD70%	SD80%	SD90%
Dataset		Test	Test	Test	Test	Test	Test
AP	Tight cluster	88.18	88.21	86.02	71.59	44.73	12.59
	Pink	72.69	71.99	71.05	63.6	50.14	21.56
	Bloom	85.01	84.55	84.03	81.06	75.05	61.37
	Petal fall	62.61	60.98	58.92	52.05	42.83	21.72
mAP		77.12	76.43	75.01	67.08	53.19	29.31

Table A6. The baseline model on saturation increased datasets.

Saturation increase		SI10%	SI30%	SI50%	SI80%	SI100%	SI150%	SI200%	SI300%	SI500%	SI700%	SI800%	SI900%
Dataset		Test	Test	Test	Test	Test	Test	Test	Test	Test	Test	Test	Test
AP	Tight cluster	88.39	88.21	87.71	87.89	87.61	86.85	86.23	83.51	70.4	48.26	35.78	24.37
	Pink	73	72.65	72.04	71.47	70.56	69.35	67.69	64.43	52.38	39.7	35.01	31.36
	Bloom	84.69	84.78	84.45	84.55	84.12	82.81	81.88	79.19	70.82	60.42	55.66	51.35
	Petal fall	61.92	61.58	61.58	60.77	60.39	58.03	55.94	47.38	34.11	21.19	15.67	11.89
mAP		77	76.8	76.44	76.17	75.67	74.26	72.93	68.63	56.93	42.39	35.53	29.74

Table A7. The baseline model on intensity decreased datasets.

Intensity decrease		ID10%	ID30%	ID50%	ID60%	ID70%	ID80%
Dataset		Test	Test	Test	Test	Test	Test
AP	Tight cluster	88.33	87.82	85.08	78.58	47.14	1.08
	Pink	72.87	72.7	71.57	69.83	59.91	27.03
	Bloom	85.53	85.47	84.22	82.36	76.23	48.26
	Petal fall	62.42	61.12	60.41	57.09	51.65	26.38
mAP		77.29	76.78	75.32	71.96	58.73	25.69

Table A8. The baseline model on intensity increased datasets.

Intensity increase		II10%	II30%	II50%	II80%	II90%	II100%	II110%
Dataset		Test	Test	Test	Test	Test	Test	Test
AP	Tight cluster	88.45	87.31	84.07	60.22	47.5	36.04	25.58
	Pink	73.12	72.2	68.43	56.2	51.73	47.3	43.84
	Bloom	84.86	83.64	81.34	77.69	75.82	73.99	72.9
	Petal fall	62.07	61.08	57.21	48.18	44.99	40.72	36.22
mAP		77.12	76.06	72.76	60.57	55.01	49.51	44.64

Intensity increase		II120%	II130%	II140%	II150%	II160%	II180%
Dataset		Test	Test	Test	Test	Test	Test
AP	Tight cluster	18.33	12.85	9.11	7.02	5.4	3.42
	Pink	41.08	38.6	36.55	34.31	32.75	30.62
	Bloom	71.24	69.37	67.66	66.12	64.28	60.01
	Petal fall	31.61	27.9	24.94	21.81	19.31	15.66
mAP		40.57	37.18	34.56	32.31	30.44	27.43

Table A9. The training-dataset-size-reduced models on original datasets.

Training dataset size		DS90%	OD	OD	DS70%	OD	OD	DS50%	OD	OD
Dataset		Training	Validation	Test	Training	Validation	Test	Training	Validation	Test
AP	Tight cluster	94.06	88.17	88.12	94.46	86.51	85.77	96.26	85.06	84.6
	Pink	85.36	74.95	73.15	86.39	73.28	70.77	90.37	70.48	67.55
	Bloom	92.06	84.81	85.27	92.65	83.63	83.98	95.06	81.44	81.6
	Petal fall	79.2	63.12	60.92	80.88	59.35	59.36	87.17	55.86	55.52
mAP		87.67	77.76	76.87	88.6	75.69	74.97	92.22	73.21	72.32

Training dataset size		DS20%	OD	OD	DS10%	OD	OD	DS5%	OD	OD
Dataset		Training	Validation	Test	Training	Validation	Test	Training	Validation	Test
AP	Tight cluster	93.66	80.31	78.44	96.12	70.23	69.81	98.76	68.44	64.89
	Pink	85.35	64.78	63.43	91.97	60.31	57.36	98.18	50.92	50.52
	Bloom	92.94	78.44	79.29	95.84	73.74	75.07	99.54	69.64	69.8
	Petal fall	79.16	50.91	48.86	87.42	45.09	43.02	97.6	32.02	36.78
mAP		87.78	68.61	67.5	92.84	62.34	61.32	98.52	55.25	55.49

Training dataset size		DS1%	OD	OD
Dataset		Training	Validation	Test
AP	Tight cluster	99.17	55.82	58.23
	Pink	100	34.04	34.92
	Bloom	100	50.68	50.42
	Petal fall	100	10.42	14.95
mAP		99.79	37.74	39.63

Table A10. The training-label-quality-reduced models on original datasets.

Training Image label quality		LQ90%	OD	OD	LQ70%	OD	OD
Dataset		Training	Validation	Test	Training	Validation	Test
AP	Tight cluster	86.6	87.53	87.2	69.24	84.95	85.23
	Pink	77.64	74.79	72.93	62.09	72.61	72.17
	Bloom	84.38	84.6	84.73	67.07	83.06	83.3
	Petal fall	71.27	62.16	62.61	56.5	59.38	59
mAP		79.97	77.27	76.87	63.73	75	74.93

Training Image label quality		LQ50%	OD	OD	LQ20%	OD	OD
Dataset		Training	Validation	Test	Training	Validation	Test
AP	Tight cluster	52.47	83.12	82.69	16.61	72.72	70.95
	Pink	45.51	69.5	69.07	12.84	55.35	54.24
	Bloom	50.05	79.68	79.99	16.1	73.42	73.13
	Petal fall	42.15	54.74	52.91	8.5	36.21	39.1
mAP		47.55	71.76	71.16	13.51	59.42	59.36

Training Image label quality		LQ10%	OD	OD	LQ5%	OD	OD
Dataset		Training	Validation	Test	Training	Validation	Test
AP	Tight cluster	8.5	61.1	64.31	4.3	49.71	50.73
	Pink	6.71	50.02	49.3	2.96	37.42	36.43
	Bloom	7.94	66.94	66.37	4.05	58.63	58.17
	Petal fall	3.72	29.1	32.5	1.65	15.82	17.6
mAP		6.72	51.79	53.12	3.24	40.39	40.73

Table A11. The negative-sample-missing models on original datasets.

Negative sample absent		NA	OD	OD	DS1%NA	OD	OD
Dataset		Training	Validation	Test	Training	Validation	Test
AP	Tight cluster	95.83	88.64	87.72	98.53	41.41	40.74
	Pink	89.21	74.72	73.8	100	38.17	37.1
	Bloom	94.5	84.34	84.33	100	52.25	50.32
	Petal fall	86.3	62.59	62.56	100	13.92	18.41
mAP		91.46	77.57	77.1	99.63	36.44	36.64

Table A12. The training-image-ordered-by-date model on original datasets.

Training image ordered by date		IO	OD	OD
Dataset		Training	Validation	Test
AP	Tight cluster	93.13	88.54	87.77
	Pink	84.67	75.15	74.27
	Bloom	91.6	85.22	86.11
	Petal fall	77.92	63.1	61.9
mAP		86.83	78	77.51

Table A13. The extreme-white-noise-trained model on relevant datasets.

White noise		WN40%	WN40%	WN40%	OD	WN0-40%
Dataset		Training	Validation	Test	Test	Test
AP	Tight cluster	90.32	85.37	85.22	78.88	82.99
	Pink	81.56	72.79	71.54	68.28	69.81
	Bloom	89.13	82.2	83.15	83.71	83.37
	Petal fall	72.56	58.01	57.69	56.31	57
mAP		83.39	74.59	74.4	71.79	73.29

Table A14. The even-white-noise-trained model on relevant datasets.

White noise		WN0-40%	WN0-40%	WN0-40%	OD	WN40%
Dataset		Training	Validation	Test	Test	Test
AP	Tight cluster	92.02	87.02	86.31	86.91	85.29
	Pink	82.71	74.05	72.6	74.28	69.51
	Bloom	90.55	84.33	83.74	85.03	82.52
	Petal fall	75.26	59.72	61.94	61.89	58.06
mAP		85.13	76.28	76.15	77.03	73.84

Table A15. The extreme-motion-blur-trained model on relevant datasets.

Motion blur		MB25Pixel	MB25Pixel	MB25Pixel	OD	MB0-25Pixel
Dataset		Training	Validation	Test	Test	Test
AP	Tight cluster	91.44	84.71	84.96	61.5	79.19
	Pink	81.41	70.73	68.88	58.88	69.25
	Bloom	88	79.64	79.98	62.95	74.88
	Petal fall	71.82	52.19	54.5	33.76	48.08
mAP		83.17	71.82	72.08	54.27	67.85

Table A16. The even-motion-blur-trained model on relevant datasets.

Motion blur		MB0-25Pixel	MB0-25Pixel	MB0-25Pixel	OD	MB25Pixel
Dataset		Training	Validation	Test	Test	Test
AP	Tight cluster	92.24	87.23	86.21	87.78	82.89
	Pink	82.33	73.26	71.88	71.84	67.4
	Bloom	89.32	82.85	83.23	84.66	78.29
	Petal fall	72.09	56.42	58.31	59.01	53.22
mAP		83.99	74.94	74.9	75.82	70.45

Table A17. The even-hue-shift-trained model on relevant datasets.

Hue shift		HS0-360°	HS0-360°	HS0-360°	OD
Dataset		Training	Validation	Test	Test
AP	Tight cluster	91.63	87.29	85.31	86.15
	Pink	80.29	72.68	71.32	71.5
	Bloom	89.09	83.52	83.69	83.55
	Petal fall	70.96	59.4	59.14	58.23
mAP		82.99	75.72	74.86	74.85

Table A18. The extreme-saturation-decrease-trained model on relevant datasets.

Saturation change		SD90%	SD90%	SD90%	OD	SI900%	SD90%-0-SI900%
Dataset		Training	Validation	Test	Test	Test	Test
AP	Tight cluster	91.43	86.27	86.59	47.9	0.01	32.6
	Pink	81.13	72.07	71.15	29.15	0.05	23.14
	Bloom	89.95	84.26	85.04	68.8	0.1	32.57
	Petal fall	71.66	59.46	58.99	23.78	0	17.03
mAP		83.55	75.52	75.44	42.41	0.04	26.33

Table A19. The extreme-saturation-increase-trained model on relevant datasets.

Saturation change		SI900%	SI900%	SI900%	OD	SD90%	SD90%-0-SI900%
Dataset		Training	Validation	Test	Test	Test	Test
AP	Tight cluster	91.03	88.04	86.84	16.67	1.63	44.78
	Pink	79.43	72.77	71.12	36.78	2.91	43.78
	Bloom	89.23	83.7	84.18	64.5	40.29	71
	Petal fall	70.37	59.92	58.79	24.65	4.45	36.02
mAP		82.51	76.11	75.23	35.65	12.32	48.89

Table A20. The even-saturation-change-trained model on relevant datasets.

Saturation change		SD90%-0-SI900%	SD90%-0-SI900%	SD90%-0-SI900%	OD	SD90%	SI900%
Dataset		Training	Validation	Test	Test	Test	Test
AP	Tight cluster	92.2	87.81	87.92	88.78	78.26	87.43
	Pink	82.33	73.48	72.06	73.08	63.28	70.38
	Bloom	90.69	84.32	84.29	84.91	80.5	83.29
	Petal fall	74.32	62.56	60.66	62.13	53.31	60.32
mAP		84.89	77.04	76.23	77.23	68.84	75.36

Table A21. The extreme-intensity-decrease-trained model on relevant datasets.

Intensity change		ID80%	ID80%	ID80%	OD	II180%	ID80%-0-II180%
Dataset		Training	Validation	Test	Test	Test	Test
AP	Tight cluster	92.07	86.96	87.38	30.5	0	30.86
	Pink	83.03	73.53	72.04	47.57	1.87	34.99
	Bloom	91.05	83.94	85	75.47	1.99	45.3
	Petal fall	75.39	61.36	61.25	38.98	0	26.8
mAP		85.39	76.45	76.42	48.13	0.97	34.49

Table A22. The extreme-intensity-increase-trained model on relevant datasets.

Intensity change		II180%	II180%	II180%	OD	ID80%	ID80%-0-II180%
Dataset		Training	Validation	Test	Test	Test	Test
AP	Tight cluster	88.91	83.28	83.1	0.96	0.01	17.37
	Pink	76.86	67.37	66.22	38.75	0	45.74
	Bloom	86.7	80.1	80.17	49.09	0	52.52
	Petal fall	62.92	48.59	48.86	38.85	0	29.02
mAP		78.85	69.83	69.59	31.91	0	36.16

Table A23. The even-intensity-change-trained model on relevant datasets.

Intensity change		ID80%-0-II180%	ID80%-0-II180%	ID80%-0-II180%	OD	ID80%	II180%
Dataset		Training	Validation	Test	Test	Test	Test
AP	Tight cluster	89.58	84.49	86.29	87.81	83.35	75.3
	Pink	80.54	73.46	69	71.9	69.8	61.33
	Bloom	89.32	82.9	82.53	84.87	83.94	79.32
	Petal fall	70.65	58.5	57.74	60.79	57.47	43.92
mAP		82.52	74.84	73.89	76.34	73.64	64.97

Table A24. The ultimate-dataset-trained model on relevant datasets.

Combined noises		UL	UL	UL	OD	DC
Dataset		Training	Validation	Test	Test	Test
AP	Tight cluster	76.9	73.63	65.49	81.18	80.19
	Pink	59.96	50.48	50.56	63.83	62.17
	Bloom	73.95	66.2	66.35	78.16	76.56
	Petal fall	46.32	33.1	36.83	50.48	48.69
mAP		64.28	55.85	54.81	68.42	66.9

Table A25. The combined-dataset-trained model on relevant datasets.

Combined noises		DC	DC	DC	OD	UL
Dataset		Training	Validation	Test	Test	Test
AP	Tight cluster	90.85	85.95	85.41	87.9	39.84
	Pink	81.09	71.25	69.45	73.28	32.21
	Bloom	89.12	82.47	82.77	85.86	46.1
	Petal fall	72.22	58.11	57.64	61.52	22.79
mAP		83.32	74.44	73.82	77.14	35.24

References

1. Moeletsi, M.E.; Tongwane, M.I. Spatiotemporal variation of frost within growing periods. *Adv. Meteorol.* **2017**, *2017*, 1–11.
2. Papagiannaki, K.; Lagouvardos, K.; Kotroni, V.; Papagiannakis, G. Agricultural losses related to frost events: use of the 850 hPa level temperature as an explanatory variable of the damage cost. *Nat. Hazards Earth Syst. Sci.* **2014**, *14*, 2375–2386.
3. Snyder, R.L.; de Melo-Abreu, J.P. *Frost protection: fundamentals, practice, and economics*; 2005; Vol. 1; ISBN 9251053286.
4. Yue, Y.; Zhou, Y.; Wang, J.; Ye, X. Assessing wheat frost risk with the support of GIS: an approach coupling a growing season meteorological index and a hybrid fuzzy neural network model. *Sustainability* **2016**, *8*, 1–21.
5. Teitel, M.; Peiper, U.M.; Zvieli, Y. Shading screens for frost protection. *Agric. For. Meteorol.* **1996**, *81*, 273–286.
6. Eccel, E.; Rea, R.; Caffarra, A.; Crisci, A. Risk of spring frost to apple production under future climate scenarios: the role of phenological acclimation. *Int. J. Biometeorol.* **2009**, *53*, 273–286.
7. Ribeiro, A.C.; de Melo-Abreu, J.P.; Snyder, R.L. Apple orchard frost protection with wind machine operation. *Agric. For. Meteorol.* **2006**, *141*, 71–81.
8. Ballard, J.K.; Proebsting, E.L. Frost and frost control in Washington orchards. *Washingt. State Univ. Ext. Bull. 634* 1972.
9. Yin, N.; Liu, R.; Zeng, B.; Liu, N. A review: UAV-based Remote Sensing. *IOP Conf. Ser. Mater. Sci. Eng.* **2019**, *490*.
10. Tsouros, D.C.; Triantafyllou, A.; Bibi, S.; Sarigannidis, P.G. Data acquisition and analysis methods in UAV- based applications for precision agriculture. In Proceedings of the 2019 15th International Conference on Distributed Computing in Sensor Systems (DCOSS); 2019; pp. 377–384.
11. Hassler, S.C.; Baysal-gurel, F. Unmanned Aircraft System (UAS) Technology and Applications in Agriculture. *Agronomy* **2019**, *9*, 618.
12. Li, L.; Zhang, Q.; Huang, D. A review of imaging techniques for plant phenotyping. *Sensors* **2014**, *14*, 20078–20111.

13. Ezenne, G.I.; Jupp, L.; Mantel, S.K.; Tanner, J.L. Current and potential capabilities of UAS for crop water productivity in precision agriculture. *Agric. Water Manag.* **2019**, *218*, 158–164.

14. Lin, Y. LiDAR: An important tool for next-generation phenotyping technology of high potential for plant phenomics? *Comput. Electron. Agric.* **2015**, *119*, 61–73.

15. An, W.; Wu, D.; Ci, S.; Luo, H.; Adamchuk, V.; Xu, Z. Agriculture Cyber-Physical Systems. In *Cyber-Physical Systems: Foundations, Principles and Applications*; 2017; pp. 399–417 ISBN 9780128038741.

16. Islam, N.; Rashid, M.M.; Pasandideh, F.; Ray, B.; Moore, S.; Kadel, R. A review of applications and communication technologies for internet of things (Iot) and unmanned aerial vehicle (uav) based sustainable smart farming. *Sustainability* **2021**, *13*, 1–20.

17. Xie, C.; Yang, C. A review on plant high-throughput phenotyping traits using UAV-based sensors. *Comput. Electron. Agric.* **2020**, *178*, 105731.

18. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. *Commun. ACM* **2017**, *60*, 84–90.

19. Fukushima, K. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biol. Cybern.* **1980**, *36*, 193–202.

20. Kattenborn, T.; Leitloff, J.; Schiefer, F.; Hinz, S. Review on Convolutional Neural Networks (CNN) in vegetation remote sensing. *ISPRS J. Photogramm. Remote Sens.* **2021**, *173*, 24–49.

21. Wang, Y.; Li, Y.; Song, Y.; Rong, X. The influence of the activation function in a convolution neural network model of facial expression recognition. *Appl. Sci.* **2020**, *10*.

22. Dhillon, A.; Verma, G.K. Convolutional neural network: a review of models, methodologies and applications to object detection. *Prog. Artif. Intell.* **2020**, *9*, 85–112.

23. Zhang, Z. Derivation of Backpropagation in Convolutional Neural Network (CNN) 2016, 1–7.

24. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition; 2014; pp. 580–587.

25. Girshick, R. Fast R-CNN. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV); 2015; pp. 1440–1448.

26. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 1137–1149.
27. He, K.; Gkioxari, G.; Dollár, P.; Girshick, R. Mask R-CNN. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV); 2017; pp. 2980–2988.
28. Pang, J.; Chen, K.; Shi, J.; Feng, H.; Ouyang, W.; Lin, D. Libra R-CNN: towards balanced learning for object detection. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2019; pp. 821–830.
29. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: unified, real-time object detection. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); IEEE, 2016; pp. 779–788.
30. Redmon, J.; Farhadi, A. YOLO9000: better, faster, stronger. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2017; pp. 6517–6525.
31. Redmon, J.; Farhadi, A. YOLOv3: an incremental improvement. *arXiv:1804.02767* **2018**.
32. Bochkovskiy, A.; Wang, C.-Y.; Liao, H.-Y.M. YOLOv4: optimal speed and accuracy of object detection. *arXiv:2004.10934* **2020**.
33. Wang, C.-Y.; Bochkovskiy, A.; Liao, H.M. Scaled-YOLOv4: Scaling Cross Stage Partial Network. *arXiv:2011.08036* **2020**.
34. Lu, X.; Li, Q.; Li, B.; Yan, J. MimicDet: Bridging the Gap Between One-Stage and Two-Stage Object Detection. *arXiv:2009.11528* **2020**.
35. Wang, C.-Y.; Liao, H.-Y.M.; Yeh, I.-H.; Wu, Y.-H.; Chen, P.-Y.; Hsieh, J.-W. CSPNet: A New Backbone that can Enhance Learning Capability of CNN. *arXiv:1911.11929* **2019**.
36. Misra, D. Mish: a self regularized non-monotonic neural activation function. *arXiv:1908.08681* **2019**.
37. He, K.; Zhang, X.; Ren, S.; Sun, J. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *37*, 1904–1916.
38. Liu, S.; Qi, L.; Qin, H.; Shi, J.; Jia, J. Path aggregation network for instance segmentation. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2018; pp. 8759–8768.

39. Xu, B.; Wang, N.; Chen, T.; Li, M. Empirical Evaluation of Rectified Activations in Convolution Network. *arXiv:1505.00853* **2015**.

40. Lin, T.-Y.; Maire, M.; Belongie, S.; Bourdev, L.; Girshick, R.; Hays, J.; Perona, P.; Ramanan, D.; Zitnick, C.L.; Dollár, P. Microsoft COCO: common objects in context. In Proceedings of the 13th European Conference on Computer Vision (ECCV); 2014; pp. 740–755.

41. Gai, R.; Chen, N.; Yuan, H. A detection algorithm for cherry fruits based on the improved YOLO-v4 model. *Neural Comput. Appl.* **2021**, *0123456789*.

42. Parico, A.I.B.; Ahamed, T. Real time pear fruit detection and counting using yolov4 models and deep sort. *Sensors* **2021**, *21*, 1–32.

43. Yijing, W.; Yi, Y.; Xue-Fen, W.; Jian, C.; Xinyun, L. Fig fruit recognition method based on YOLO v4 deep learning. In Proceedings of the 2021 18th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON); IEEE, 2021; pp. 303–306.

44. Jiang, M.; Song, L.; Wang, Y.; Li, Z.; Song, H. Fusion of the YOLOv4 network model and visual attention mechanism to detect low-quality young apples in a complex environment. *Precis. Agric.* **2021**.

45. Chen, W.; Lu, S.; Liu, B.; Li, G.; Qian, T. Detecting citrus in orchard environment by using improved Yolov4. *Sci. Program.* **2020**, *2020*, 8859237.

46. Wu, D.; Lv, S.; Jiang, M.; Song, H. Using channel pruning-based YOLO v4 deep learning algorithm for the real-time and accurate detection of apple flowers in natural environments. *Comput. Electron. Agric.* **2020**, *178*, 105742.

47. Lyu, S.; Zhao, Y.; Li, R.; Chen, Q.; Li, Z. The accurate recognition system of citrus flowers using YOLOv4-Tiny lightweight neural network and FPGA embedded platform. In Proceedings of the International Conference on Mechanical Engineering, Measurement Control, and Instrumentation (MEMCI 2021); 2021; p. 119302E.

48. Zhou, B.; Song, Z.; Wang, Y.; Hu, F. Flower Gender Recognition Based on YOLO V4. In *3D Imaging Technologies—Multidimensional Signal Processing and Deep Learning*; 2021; pp. 43–49 ISBN 9789811631795.

49. Yang, B.; Gao, Z.; Gao, Y.; Zhu, Y. Rapid detection and counting of wheat ears in the field using yolov4 with attention module. *Agronomy* **2021**, *11*, 1–17.

50. Richey, B.; Shirvaikar, M. Deep learning based real-time detection of northern corn leaf blight crop disease using YoloV4. In Proceedings of the Conference on Real-Time Image Processing and Deep Learning; 2021; p. 1173606.

51. Song, C.C.; Wang, C.L.; Yang, Y.F. Automatic Detection and Image Recognition of Precision Agriculture for Citrus Diseases. In Proceedings of the 2020 IEEE Eurasia Conference on IOT, Communication and Engineering (ECICE); 2020; pp. 187–190.

52. Hu, D.; Ma, C.; Tian, Z.; Shen, G.; Li, L. Rice Weed detection method on YOLOv4 convolutional neural network. In Proceedings of the 2021 International Conference on Artificial Intelligence, Big Data and Algorithms (CAIBDA); IEEE, 2021; pp. 41–45.

53. Ying, B.; Xu, Y.; Zhang, S.; Shi, Y.; Liu, L. Weed detection in images of carrot fields based on improved YOLO v4. *Trait. du Signal* **2021**, *38*, 341–348.

54. Wang, D.; Wang, Y.; Li, M.; Yang, X.; Wu, J.; Li, W. Using an Improved YOLOv4 Deep Learning Network for Accurate Detection of Whitefly and Thrips on Sticky Trap Images. *Trans. ASABE* **2021**, *64*, 919–927.

55. Bai, Z.; Tang, Z.; Diao, L.; Lu, S.; Guo, X.; Zhou, H.; Liu, C.; Li, L. Video target detection of East Asian migratory locust based on the MOG2-YOLOv4 network. *Int. J. Trop. Insect Sci.* **2021**.

56. Chen, J.W.; Lin, W.J.; Cheng, H.J.; Hung, C.L.; Lin, C.Y.; Chen, S.P. A smartphone-based application for scale pest detection using multiple-object detection methods. *Electronics* **2021**, *10*, 1–14.

57. Li, S.; Kang, X.; Feng, Y.; Liu, G. Detection method for individual pig based on improved YOLOv4 Convolutional Neural Network. In Proceedings of the DSIT 2021: 2021 4th International Conference on Data Science and Information Technology; 2021; pp. 231–235.

58. Wu, F.; Zhao, H.; Wang, M. Nighttime cattle detection based on YOLOv4. In Proceedings of the 12th International Conference on Graphics and Image Processing (ICGIP 2020); 2021; p. 1172008.

59. Prasetyo, E.; Suciati, N.; Fatichah, C. Yolov4-Tiny and Spatial Pyramid Pooling for Detecting Head and Tail of Fish. In Proceedings of the 2021 International Conference on Artificial Intelligence and Computer Science Technology (ICAICST); IEEE, 2021; pp. 157–161.

60. Hu, X.; Liu, Y.; Zhao, Z.; Liu, J.; Yang, X. Real-time detection of uneaten feed pellets in underwater images for aquaculture using an improved YOLO-V4 network. *Comput. Electron. Agric.* **2021**, *185*, 106135.

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

62. Geirhos, R.; Rubisch, P.; Michaelis, C.; Bethge, M.; Wichmann, F.; Brendel, W. ImageNet-Trained CNNs Are Biased Towards Texture. In Proceedings of the 7th International Conference on Learning Representations; 2019; pp. 1–20.

63. Alcorn, M.A.; Li, Q.; Gong, Z.; Wang, C.; Mai, L.; Ku, W.S.; Nguyen, A. Strike (with) a pose: Neural networks are easily fooled by strange poses of familiar objects. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2019; pp. 4845–4854.

64. Barbu, A.; Mayo, D.; Alverio, J.; Luo, W.; Wang, C.; Gutfreund, D.; Tenenbaum, J.; Katz, B. ObjectNet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 1–11.

65. Dodge, S.; Karam, L. Understanding how image quality affects deep neural networks. *2016 8th Int. Conf. Qual. Multimed. Exp. QoMEX 2016* **2016**.

66. Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. ImageNet Large Scale Visual Recognition Challenge. *Int. J. Comput. Vis.* **2015**, *115*, 211–252.

67. Ghosh, S.; Shet, R.; Amon, P.; Hutter, A.; Kaup, A. Robustness of Deep Convolutional Neural Networks for Image Degradations. In Proceedings of the ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings; 2018; pp. 2916–2920.

68. Geirhos, R.; Schütt, H.H.; Medina Temme, C.R.; Bethge, M.; Rauber, J.; Wichmann, F.A. Generalisation in humans and deep neural networks. In Proceedings of the 32nd Conference on Neural Information Processing Systems (NeurIPS 2018); 2018; pp. 1–28.

69. Dodge, S.; Karam, L. A study and comparison of human and deep learning recognition performance under visual distortions. *2017 26th Int. Conf. Comput. Commun. Networks, ICCCN 2017* **2017**.

70. Dodge, S.; Karam, L. Quality Resilient Deep Neural Networks. *arXiv:1703.08119* **2017**.

71. Dodge, S.F.; Karam, L.J. Quality Robust Mixtures of Deep Neural Networks. *IEEE Trans. Image Process.* **2018**, *27*, 5553–5562.

72. Zhou, Y.; Song, S.; Cheung, N.M. On classification of distorted images with deep convolutional neural networks. *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.* **2017**, 1213–1217.

73. Lecun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-Based Learning Applied to Document Recognition. *Proc. IEEE* **1998**, *86*, 2278–2324.

74. Krizhevsky, A. Learning Multiple Layers of Features from Tiny Images 2009.

75. Vasiljevic, I.; Chakrabarti, A.; Shakhnarovich, G. Examining the Impact of Blur on Recognition by Convolutional Networks. *arXiv:1611.05760* **2016**.

76. Tadros, T.; Cullen, N.C.; Greene, M.R.; Cooper, E.A. Assessing neural network scene classification from degraded images. *ACM Trans. Appl. Percept.* **2019**, *16*.

77. Zhou, B.; Lapedriza, A.; Xiao, J.; Torralba, A.; Oliva, A. Learning deep features for scene recognition using places database. *Adv. Neural Inf. Process. Syst.* **2014**, *1*, 487–495.

78. Cho, J.; Lee, K.; Shin, E.; Choy, G.; Do, S. How much data is needed to train a medical image deep learning system to achieve necessary high accuracy? *arXiv:1511.06348* **2016**.

79. Dutta, S.; Gros, E. Evaluation of the impact of deep learning architectural components selection and dataset size on a medical imaging task. In Proceedings of the Medical Imaging 2018: Imaging Informatics for Healthcare, Research, and Applications; 2018; p. 1057911.

80. Schouten, J.P.E.; Matek, C.; Jacobs, L.F.P.; Buck, M.C.; Bošnački, D.; Marr, C. Tens of images can suffice to train neural networks for malignant leukocyte detection. *Sci. Rep.* **2021**, *11*, 1–8.

81. Luo, C.; Li, X.; Wang, L.; He, J.; Li, D.; Zhou, J. How Does the Data set Affect CNN-based Image Classification Performance? *2018 5th Int. Conf. Syst. Informatics, ICSAI 2018* **2019**, 361–366.

82. Sun, M.-J.; Edgar, M.P.; Phillips, D.B.; Gibson, G.M.; Padgett, M.J. Improving the signal-to-noise ratio of single-pixel imaging using digital microscanning. *Opt. Express* **2016**, *24*, 10476.

83. Finlayson, G.; Schaefer, G. Hue that is invariant to brightness and gamma. In Proceedings of the Proceedings of the British Machine Vision Conference; 2001; pp. 32.1-32.10.

84. Welch, E.; Moorhead, R.; Owens, J.K. Image processing using the HSI color space. In Proceedings of the IEEE Proceedings of the SOUTHEASTCON '91; 1991; pp. 722–725.

85. Schiller, F.; Valsecchi, M.; Gegenfurtner, K.R. An evaluation of different measures of color saturation. *Vision Res.* **2018**, *151*, 117–134.

86. Zhao, J.; Li, C.; Xu, Z.; Jiao, L.; Zhao, Z.; Wang, Z. Detection of passenger flow on and off buses based on video images and YOLO algorithm. *Multimed. Tools Appl.* **2021**.

87. Choi, H.G.; Moon, B.Y.; Kang, N.J. Effects of LED light on the production of strawberry during cultivation in a plastic greenhouse and in a growth chamber. *Sci. Hortic. (Amsterdam).* **2015**, *189*, 22–31.

88. Matese, A.; Di Gennaro, S.F. Practical applications of a multisensor UAV platform based on multispectral, thermal and RGB high resolution images in precision viticulture. *Agriculture* **2018**, *8*, 1–13.

89. Messina, G.; Modica, G. Applications of UAV thermal imagery in precision agriculture: State of the art and future research outlook. *Remote Sens.* **2020**, *12*, 1–26.

90. Candiago, S.; Remondino, F.; De Giglio, M.; Dubbini, M.; Gattelli, M. Evaluating multispectral images and vegetation indices for precision farming applications from UAV images. *Remote Sens.* **2015**, *7*, 4026–4047.

91. Adão, T.; Hruška, J.; Pádua, L.; Bessa, J.; Peres, E.; Morais, R.; Sousa, J.J. Hyperspectral imaging: A review on UAV-based sensors, data processing and applications for agriculture and forestry. *Remote Sens.* **2017**, *9*, 1–30.

92. Mallet, C.; Bretar, F. Full-waveform topographic lidar: State-of-the-art. *ISPRS J. Photogramm. Remote Sens.* **2009**, *64*, 1–16.

93. Tang, J.; Chen, Y.; Niu, X.; Wang, L.; Chen, L.; Liu, J.; Shi, C.; Hyyppä, J. LiDAR scan matching aided inertial navigation system in GNSS-denied environments. *Sensors* **2015**, *15*, 16710–16728.

94. Taheri, H.; Xia, Z.C. SLAM; definition and evolution. *Eng. Appl. Artif. Intell.* **2021**, *97*, 104032.

95. Liu, K.; Dong, X.; Qiu, B. Analysis of cotton height spatial variability based on UAV-LiDAR. *Int. J. Precis. Agric. Aviat.* **2018**, *3*, 72–76.

96. Zhou, L.; Gu, X.; Cheng, S.; Yang, G.; Shu, M.; Sun, Q. Analysis of plant height changes of lodged maize using UAV-LiDAR data. *Agriculture* **2020**, *10*.

97. Hu, X.; Sun, L.; Gu, X.; Sun, Q.; Wei, Z.; Pan, Y.; Chen, L. Assessing the self-recovery ability of maize after lodging using UAV-LiDAR data. *Remote Sens.* **2021**, *13*, 1–22.

98. Lei, L.; Qiu, C.; Li, Z.; Han, D.; Han, L.; Zhu, Y.; Wu, J.; Xu, B.; Feng, H.; Yang, H.; et al. Effect of leaf occlusion on leaf area index inversion of maize using UAV-LiDAR data. *Remote Sens.* **2019**, *11*, 1–15.

99. Luo, S.; Liu, W.; Zhang, Y.; Wang, C.; Xi, X.; Nie, S.; Ma, D.; Lin, Y.; Zhou, G. Maize and soybean heights estimation from unmanned aerial vehicle (UAV) LiDAR data. *Comput. Electron. Agric.* **2021**, *182*, 106005.

100. Phan, A.T.T.; Takahashi, K. Estimation of rice plant height from a low-cost uav-based lidar point clouds. *Int. J. Geoinformatics* **2021**, *17*, 89–98.

101. Xu, J.X.; Ma, J.; Tang, Y.N.; Wu, W.X.; Shao, J.H.; Wu, W. Ben; Wei, S.Y.; Liu, Y.F.; Wang, Y.C.; Guo, H.Q. Estimation of sugarcane yield using a machine learning approach based on uav-lidar data. *Remote Sens.* **2020**, *12*, 1–13.

102. Sofonia, J.; Shendryk, Y.; Phinn, S.; Roelfsema, C.; Kendoul, F.; Skocaj, D. Monitoring sugarcane growth response to varying nitrogen application rates: A comparison of UAV SLAM LiDAR and photogrammetry. *Int. J. Appl. Earth Obs. Geoinf.* **2019**, *82*, 101878.

103. Shendryk, Y.; Sofonia, J.; Garrard, R.; Rist, Y.; Skocaj, D.; Thorburn, P. Fine-scale prediction of biomass and leaf nitrogen content in sugarcane using UAV LiDAR and multispectral imaging. *Int. J. Appl. Earth Obs. Geoinf.* **2020**, *92*, 102177.

104. Masjedi, A.; Crawford, M.M. Prediction of sorghum biomass using time series UAV-based hyperspectral and LiDAR data. In Proceedings of the IGARSS 2020 - 2020 IEEE International Geoscience and Remote Sensing Symposium; 2020; pp. 3912–3915.

105. Masjedi, A.; Zhao, J.; Thompson, A.M.; Yang, K.W.; Flatt, J.E.; Crawford, M.M.; Ebert, D.S.; Tuinstra, M.R.; Hammer, G.; Chapman, S. Sorghum biomass prediction using uav-based remote sensing data and crop model simulation. In Proceedings of the International Geoscience and Remote Sensing Symposium (IGARSS); IEEE, 2018; pp. 7719–7722.

106. Masjedi, A.; Crawford, M.M.; Carpenter, N.R.; Tuinstra, M.R. Multi-temporal predictive modelling of sorghum biomass using UAV-based hyperspectral and LiDAR data. *Remote Sens.* **2020**, *12*, 1–35.

107. Maimaitijiang, M.; Sagan, V.; Erkbol, H.; Adrian, J.; Newcomb, M.; Lebauer, D.; Pauli, D.; Shakoor, N.; Mockler, T.C. UAV-based sorghum growth monitoring: A comparative

analysis of LiDAR and photogrammetry. *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci.* **2020**, *V-3–2020*, 489–496.

108. Dhami, H.; Yu, K.; Xu, T.; Zhu, Q.; Dhakal, K.; Friel, J.; Li, S.; Tokekar, P. Crop height and plot estimation for phenotyping from unmanned aerial vehicles using 3D LiDAR. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS); 2021; pp. 2643–2649.

109. ten Harkel, J.; Bartholomeus, H.; Kooistra, L. Biomass and crop height estimation of different crops using UAV-based Lidar. *Remote Sens.* **2020**, *12*, 1–18.

110. Hadas, E.; Jozkow, G.; Walicka, A.; Borkowski, A. Apple orchard inventory with a LiDAR equipped unmanned aerial system. *Int. J. Appl. Earth Obs. Geoinf.* **2019**, *82*, 101911.

111. Neuville, R.; Bates, J.S.; Jonard, F. Estimating forest structure from UAV-mounted LiDAR point cloud using machine learning. *Remote Sens.* **2021**, *13*, 1–19.

112. Parmehr, E.G.; Amati, M. Individual tree canopy parameters estimation using UAV-based photogrammetric and LiDAR point clouds in an urban park. *Remote Sens.* **2021**, *13*, 1–17.

113. Peng, X.; Li, X.; Wang, C.; Zhu, J.; Liang, L.; Fu, H.; Du, Y.; Yang, Z.; Xie, Q. SPICE-based SAR tomography over forest areas using a small number of P-band airborne F-SAR images characterized by non-uniformly distributed baselines. *Remote Sens.* **2019**, *11*, 1–21.

114. Picos, J.; Bastos, G.; Míguez, D.; Alonso, L.; Armesto, J. Individual tree detection in a eucalyptus plantation using unmanned aerial vehicle (UAV)-LiDAR. *Remote Sens.* **2020**, *12*, 1–17.

115. Prata, G.A.; Broadbent, E.N.; De Almeida, D.R.A.; Peter, J.S.; Drake, J.; Medley, P.; Corte, A.P.D.; Vogel, J.; Sharma, A.; Silva, C.A.; et al. Single-pass UAV-borne GatorEye LiDAR sampling as a rapid assessment method for surveying forest structure. *Remote Sens.* **2020**, *12*, 1–17.

116. Sankey, T.; Donager, J.; McVay, J.; Sankey, J.B. UAV lidar and hyperspectral fusion for forest monitoring in the southwestern USA. *Remote Sens. Environ.* **2017**, *195*, 30–43.

117. Wang, D.; Wan, B.; Qiu, P.; Zuo, Z.; Wang, R.; Wu, X. Mapping height and aboveground biomass of mangrove forests on Hainan Island using UAV-LiDAR sampling. *Remote Sens.* **2019**, *11*, 1–25.

118. Liu, K.; Shen, X.; Cao, L.; Wang, G.; Cao, F. Estimating forest structural attributes using UAV-LiDAR data in Ginkgo plantations. *ISPRS J. Photogramm. Remote Sens.* **2018**, *146*, 465–482.

119. Hu, T.; Sun, X.; Su, Y.; Guan, H.; Sun, Q.; Kelly, M.; Guo, Q. Development and performance evaluation of a very low-cost UAV-Lidar system for forestry applications. *Remote Sens.* **2020**, *13*, 1–21.

120. Guo, Q.; Su, Y.; Hu, T.; Zhao, X.; Wu, F.; Li, Y.; Liu, J.; Chen, L.; Xu, G.; Lin, G.; et al. An integrated UAV-borne lidar system for 3D habitat mapping in three forest ecosystems across China. *Int. J. Remote Sens.* **2017**, *38*, 2954–2972.

121. D'Oliveira, M.V.N.; Broadbent, E.N.; Oliveira, L.C.; Almeida, D.R.A.; Papa, D.A.; Ferreira, M.E.; Zambrano, A.M.A.; Silva, C.A.; Avino, F.S.; Prata, G.A.; et al. Aboveground biomass estimation in Amazonian tropical forests: A comparison of aircraft- and gatoreye UAV-borne LIDAR data in the Chico mendes extractive reserve in Acre, Brazil. *Remote Sens.* **2020**, *12*, 1–19.

122. Cunha Neto, E.M. da; Rex, F.E.; Veras, H.F.P.; Moura, M.M.; Sanquetta, C.R.; Käfer, P.S.; Sanquetta, M.N.I.; Zambrano, A.M.A.; Broadbent, E.N.; Corte, A.P.D. Using high-density UAV-Lidar for deriving tree height of Araucaria Angustifolia in an Urban Atlantic Rain Forest. *Urban For. Urban Green.* **2021**, *63*, 127197.

123. Corte, A.P.D.; Souza, D.V.; Rex, F.E.; Sanquetta, C.R.; Mohan, M.; Silva, C.A.; Zambrano, A.M.A.; Prata, G.; Alves de Almeida, D.R.; Trautenmüller, J.W.; et al. Forest inventory with high-density UAV-Lidar: Machine learning approaches for predicting individual tree attributes. *Comput. Electron. Agric.* **2020**, *179*, 105815.

124. Corte, A.P.D.; Rex, F.E.; de Almeida, D.R.A.; Sanquetta, C.R.; Silva, C.A.; Moura, M.M.; Wilkinson, B.; Zambrano, A.M.A.; da Cunha Neto, E.M.; Veras, H.F.P.; et al. Measuring individual tree diameter and height using gatoreye high-density UAV-lidar in an integrated crop-livestock-forest system. *Remote Sens.* **2020**, *12*, 1–15.

125. Chen, X.; Jiang, K.; Zhu, Y.; Wang, X.; Yun, T. Individual tree crown segmentation directly from uav-borne lidar data using the pointnet of deep learning. *Forests* **2021**, *12*, 1–22.

126. Cao, L.; Liu, K.; Shen, X.; Wu, X.; Liu, H. Estimation of forest structural parameters using UAV-LiDAR data and a process-based model in ginkgo planted forests. *IEEE J. Sel.*

Top. Appl. Earth Obs. Remote Sens. **2019**, *12*, 4175–4190.

127. Cao, L.; Liu, H.; Fu, X.; Zhang, Z.; Shen, X.; Ruan, H. Comparison of UAV LiDAR and digital aerial photogrammetry point clouds for estimating forest structural attributes in subtropical planted forests. *Forests* **2019**, *10*, 1–26.
128. Brede, B.; Calders, K.; Lau, A.; Raumonen, P.; Bartholomeus, H.M.; Herold, M.; Kooistra, L. Non-destructive tree volume estimation through quantitative structure modelling: Comparing UAV laser scanning with terrestrial LIDAR. *Remote Sens. Environ.* **2019**, *233*, 111355.
129. Gautam, D.; Lucieer, A.; Watson, C.; McCoull, C. Lever-arm and boresight correction, and field of view determination of a spectroradiometer mounted on an unmanned aircraft system. *ISPRS J. Photogramm. Remote Sens.* **2019**, *155*, 25–36.
130. Aalerud, A.; Dybedal, J.; Subedi, D. Reshaping field of view and resolution with segmented reflectors: Bridging the gap between rotating and solid-state lidars. *Sensors* **2020**, *20*, 1–26.
131. DJI onboard SDK: Telemetry topics Available online: https://developer.dji.com/onboard-api-reference/group__telem.html (accessed on Jul 7, 2021).
132. Diebel, J. Representing attitude: Euler angles, unit quaternions, and rotation vectors. *Matrix* **2006**, *58*, 1–35.
133. Skaloud, J.; Lichti, D. Rigorous approach to bore-sight self-calibration in airborne laser scanning. *ISPRS J. Photogramm. Remote Sens.* **2006**, *61*, 47–59.

VITA

WENAN YUAN

Education

- The Pennsylvania State University (State College, PA, USA) Aug. 2019 – May 2022
    - Ph.D. in Agricultural and Biological Engineering
- University of Nebraska–Lincoln (Lincoln, NE, USA) Aug. 2016 – Aug. 2019
    - M.S. in Agricultural and Biological Systems Engineering
- Colorado State University (Fort Collins, CO, USA) Aug. 2014 – May 2016
    - B.S. in Soil and Crop Sciences
- China Agricultural University (Beijing, China) Sep. 2012 – Jun. 2014
    - B.S. in Agronomy

Grant

- Yuan, W.; Choi, D. GNE21-274: Apple Blossom Density Estimation and Mapping through Unmanned Aerial Vehicle-Based Photogrammetry. *Northeast SARE*. $14,997. August 1, 2021 - July 31, 2022. Graduate Investigator.

Publication

- Yuan, W.; Choi, D. (2021). UAV-Based Heating Requirement Determination for Frost Management in Apple Orchard. *Remote Sensing*, 13(2):273.
- Yuan, W.; Wijewardane, N.K.; Jenkins, S.; Bai, G.; Ge, Y.; Graef, G.L. (2019). Early Prediction of Soybean Traits through Color and Texture Features of Canopy RGB Imagery. *Scientific Reports*, 9:14089.
- Yuan, W.; Li, J.; Bhatta, M.; Shi, Y.; Baenziger, P.S.; Ge, Y. (2018). Wheat Height Estimation Using LiDAR in Comparison to Ultrasonic Sensor and UAS. *Sensors*, 18(11):3731.
- Bai, G.; Jenkins, S.; Yuan, W.; Graef, G. L.; Ge, Y. (2018). Field-Based Scoring of Soybean Iron Deficiency Chlorosis Using RGB Imaging and Statistical Learning. *Frontiers in Plant Science*, 9:1002.