J Inf Process Syst, Vol.19, No.5, pp.614~630, October 2023
https://doi.org/10.3745/JIPS.02.0204

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

Automatic Detection of Dead Trees Based on Lightweight YOLOv4 and UAV Imagery

Yuanhang Jin, Maolin Xu*, and Jiayuan Zheng

Abstract
Dead trees significantly impact forest production and the ecological environment and pose constraints to the sustainable development of forests. A lightweight YOLOv4 dead tree detection algorithm based on unmanned aerial vehicle images is proposed to address current limitations in dead tree detection that rely mainly on inefficient, unsafe and easy-to-miss manual inspections. An improved logarithmic transformation method was developed in data pre-processing to display tree features in the shadows. For the model structure, the original CSPDarkNet-53 backbone feature extraction network was replaced by MobileNetV3. Some of the standard convolutional blocks in the original extraction network were replaced by depthwise separable convolution blocks. The new ReLU6 activation function replaced the original LeakyReLU activation function to make the network more robust for low-precision computations. The K-means++ clustering method was also integrated to generate anchor boxes that are more suitable for the dataset. The experimental results show that the improved algorithm achieved an accuracy of 97.33%, higher than other methods. The detection speed of the proposed approach is higher than that of YOLOv4, improving the efficiency and accuracy of the detection process.

Keywords
Dead Tree, Deep Learning, MobileNetV3, Object Detection, YOLOv4

1. Introduction

The tree health assessment in forests has important implications for biodiversity, forest management, and environmental monitoring. Dead trees are a vital indicator of forest biodiversity and ecosystem health, making dead tree detection essential [1]. Dead tree detection is relatively primitive, relying mainly on manual patrols and observations, which are usually labor-intensive and challenging to detect in treacherous terrain. Traditional manual detection methods are inefficient in meeting the increasingly critical needs of dead tree detection.

Numerous studies have used remote sensing images to improve the efficiency of dead tree detection. For example, Otsu et al. [2] utilized point cloud data to detect dead trees, resulting in an overall accuracy of 94.3%. Kaminska et al. [1] employed multispectral images to classify dead trees, leading to an overall accuracy of 95%. However, the hardware cost of LiDAR and multispectral methods is high, and data acquisition can be complex. As a miniature low-altitude remote sensing platform, the unmanned aerial vehicle (UAV) has numerous advantages, such as low operating costs, convenient operations, and free use of time. The small UAV with a visible light camera can easily and quickly monitor the target area on

Manuscript received June 21, 2022; first revision September 27, 2022; second revision October 17, 2022; accepted November 28, 2022.
* **Corresponding Author:** Maolin Xu (xml1964@163.com)
State Key Laboratory of Geo-Information Engineering and School of Civil Engineering, University of Science and Technology Liaoning, Anshan, China (yhjin2@126.com, Zhengjy0221@163.com)

a large scale, which is vital for developing small- and medium-scale remote sensing applications and forestry detection technology.

With its growing applications in target detection using computer vision, machine learning has been utilized for tree detection in remote-sensing images. For example, Malek et al. [3] used scale invariant feature transform (SIFT) to extract a set of key points of palm trees. Then they used an extreme learning machine (ELM) classifier to distinguish palm trees from other vegetation. However, the SIFT algorithm only selects features for a few key points in the sample, which are local and less accurate than the RGB-based global feature method. Regarding deep learning in machine learning, Li et al. [4] applied a deep learning-based convolutional neural network (CNN) approach to detect densely planted Malaysian oil palm trees, resulting in 96% of samples being detected correctly. Culman et al. [5] applied CNN for phoenix tree detection in the Canary Islands and correctly detected 86% of the samples. Guirado et al. [6] proposed a CNN-based shrub detection method using Google Earth images as the data source, achieving better detection results than previous single-tree detection methods. Tao et al. [7] used a CNN to detect dead pine trees photographed using a UAV. Yu et al. [8] used YOLOv4 for the early detection of trees infected with pine nematode disease and achieved 57%–63% accuracy. However, their network structure is relatively simple, with insufficient information extracted and poor capability for detecting complex targets.

Using YOLO as the base model, Junos et al. [9] made improvements (e.g., the addition of the swish activation function and the optimization of the prior bounding box to detect palm fruits) and were able to obtain better detection results. Liu et al. [10] used Fast-RCNN-based algorithms to detect palm trees at three sites in Malaysia and achieved more than 95% accuracy. Yarak et al. [11] combined high-resolution images with the Fast-RCNN [12] phase for automatic detection and health classification of palm trees, which also achieved good detection results. However, the above methods used long training and detection time models, which could not meet real-time demand.

This paper addresses the shortcomings of costly data acquisition, insufficient information extraction, and lengthy training time. To obtain a complete image dataset, we collected tree images using a consumer-grade UAV platform, enhanced them using an improved logarithmic transformation, and used enhancement and image rotation for data augmentation. This paper used the classical YOLOv4 model of the One Stage target detection algorithm as the foundation to reduce network parameters and computations, improve training and detection speed, and ensure detection accuracy.

First, the K-means++ clustering algorithm was used to obtain the prior bounding box instead of K-means in the original YOLOv4 model. The CSPDarkNet-53 backbone network was then changed to a lighter MobileNetV3 network, and depthwise separable convolution was added to the enhanced feature extraction network further to reduce the number of parameters in the model while ensuring the feature extraction capability. Finally, the improved MobileNetV3-YOLOv4 network model was used to train the dataset, and the weight files obtained after training were used to obtain the detection results. The technical flow chart of this paper is shown in Fig. 1.

2. Materials and Study Area

2.1 Data Acquisition

In this paper, dead trees were used as the object of interest. RGB images of dead trees in a scenic area of Southern Liaoning were collected using aerial photography of the forest area with drone-borne cameras. The time slot for image capture was from 3 pm–5 pm. The equipment used in capturing the images was a DJI PHANTOM4 RTK quadcopter, flying at an altitude of 80–120 m. The camera was flown at 90° vertical, with a heading overlap and a side overlap of 80%.

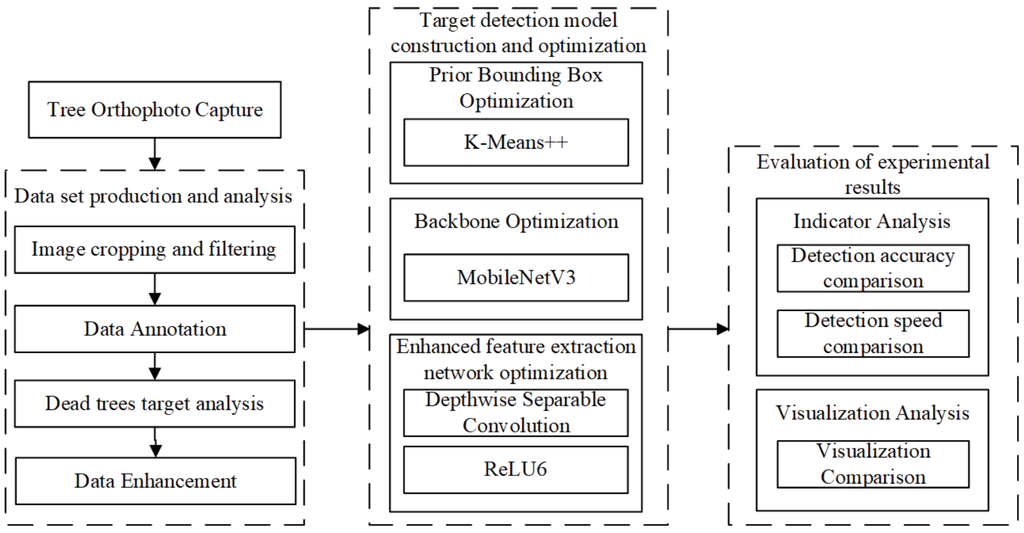

Tree Orthophoto Capture
Data set production and analysis
Image cropping and filtering
Data Annotation
Dead trees target analysis
Data Enhancement
Target detection model construction and optimization
Prior Bounding Box Optimization
K-Means++
Backbone Optimization
MobileNetV3
Enhanced feature extraction network optimization
Depthwise Separable Convolution
ReLU6
Evaluation of experimental results
Indicator Analysis
Detection accuracy comparison
Detection speed comparison
Visualization Analysis
Visualization Comparison

Fig. 1. The flowchart of dead tree detection techniques.

2.2 Image Pre-processing

Because of the large size of the original UAV image and the small amount of helpful information in the original image, using the entire image to train the deep learning network model would be very slow and inefficient. Therefore, in this paper, the original image was cropped to a 912×912 size image with 96 dpi, and the invalid image was eliminated manually.

Finding sufficient data to complete the task in many practical projects would be difficult [13]. Before training, the dataset would need to be augmented with data to provide more images to avoid problems like poor model stability or overfitting. Image flipping and a modified log-transformed image enhancement method were used to process the images of the dead tree training set and add them to the training set.

The light intensity of a scene shot at different angles can interfere with the captured tree image and affect the algorithm's detection. Adjusting the image brightness can reduce the effect of uneven illumination to a certain extent. Logarithmic transformation [14] is a commonly used image enhancement method representing a logarithmic relationship between the output image and its corresponding input image pixel grayscale values. It is used as a component of image processing algorithms to emphasize the lower grayscale portions of an image by partially expanding the grayscale values of the image as well as partially compressing the higher grayscale values. Its conventional Eq. (1) is:

$$f(i,j) = A\,log(|g(i,j)| + 1), \tag{1}$$

where $f(i,j)$ and$g(i,j)$ are the grayscale pixel values of the output image and the corresponding input image, and A is the intensity parameter, which mainly transforms the dynamic range of the shadow feature enhancement to a suitable interval to show more details of the shadow region. When this method was used for image enhancement, the results indicated that the method was too large for image transformation and caused over-brightening or overexposure of the image.

The basic equation for the logarithmic transformation can be changed into different forms depending on the application. To address excessive image transformation when the original logarithmic transformation is applied for image enhancement, an improved transformation was designed for dark area compensation:

$$f(i,j) = A\,lg(V * |g(i,j)| + 1)\ \ (v > 1), \tag{2}$$

where v is the adjustment parameter to make the image transformation more flexible, the value selection for the parameter v has an essential effect on image transformation. Fig. 2 compares the original image with the basic and improved Equations.

After enhancement of the original image, the dead tree targets were annotated using the open-source software LabelImg [15], as shown in Fig. 3. An XML file in PASCAL VOC [16] format was used for annotation.

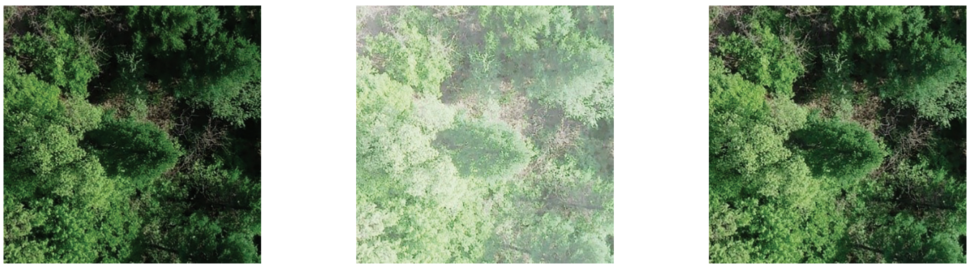

(a) (b) (c)

Fig. 2. Comparison effect before and after the improvement of logarithmic transformation: (a) original image, (b) original log-transformed image, and (c) improved log-transformed image.

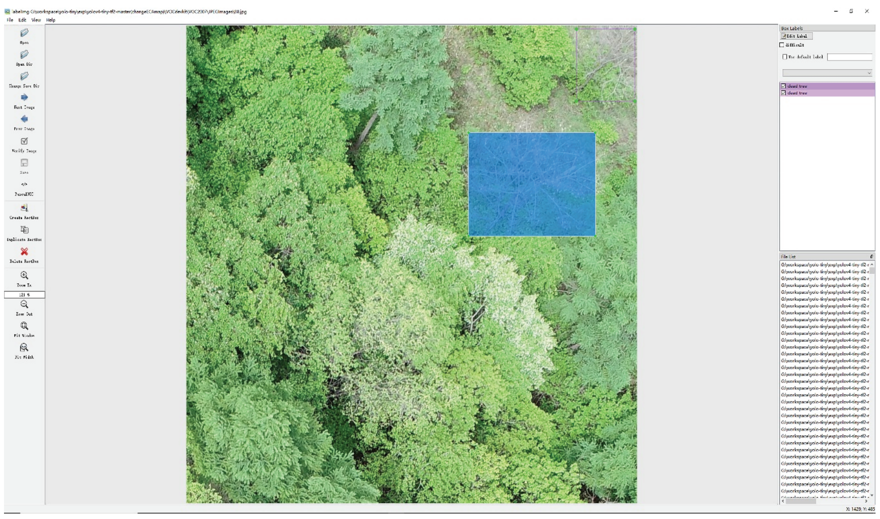


Fig. 3. Data annotation.

3. Method

In this experiment, MobileNetV3, a lightweight network, was used to reduce the number of network parameters in backbone feature extraction and tackle the redundant structure of YOLOv4 [17]. Also, to address many convolutional blocks in the enhanced feature extraction network model of YOLOv4, a partial replacement was performed using depthwise separable convolution blocks to reduce the model parameters further while improving the detection speed. The LeakyReLU was replaced with the ReLU6 activation function to improve model robustness. The K-means++ clustering algorithm replaced the K-means algorithm in the YOLOv4 model to obtain a more reasonable and higher accuracy prior bounding box, making the model easier to learn.

3.1 Lightweight Network MobileNetV3

At the heart of lightweight networks is the design of more efficient network computations for CNN, which can reduce the number of network parameters without losing network performance. This paper demonstrates a replaced backbone feature extraction network CSPDarkNet-53 of YOLOv4 with the MobileNetV3 network structure. The lightweight network MobileNetV3 [18] was obtained by improving the lightweight attention model from the depthwise separable convolution of MobileNetV1 [19], the inverted residual structure of the linear bottleneck of MobileNetV2 [20], and the squeeze-and-excitation (SE) [21] structure of MnasNet.

3.2 Depthwise Separable Convolution

Depthwise separable convolution was made up of depthwise and pointwise convolution, as shown in Figs. 4 and 5. Only one convolution kernel convolved each depthwise convolution channel (i.e., one convolution kernel was responsible for only one channel in the feature map, extracting the features inside a single channel). The feature map channel obtained after the channel-by-channel convolution was the same as the original feature map channel; the size of each convolution kernel for pointwise convolution was C×1×1. The pointwise convolution was performed, and the feature information between different channels was fused to obtain a new feature map.

Depthwise separable convolution yields a feature map with the same dimensionality as the standard convolution. However, depthwise separable convolution requires fewer parameters and model computations, significantly improving algorithm efficiency.

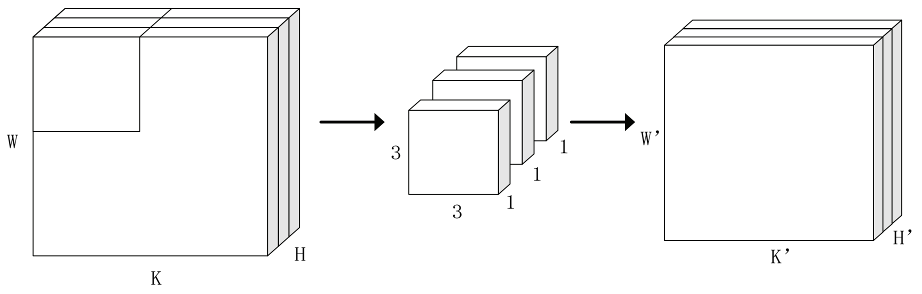


Fig. 4. Depthwise convolution.

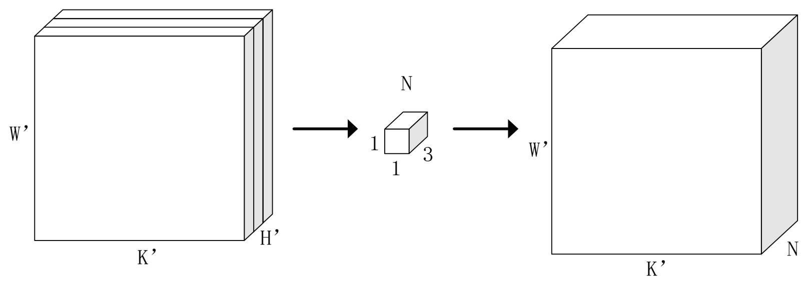


Fig. 5. Pointwise convolution.

3.3 ReLU6

For the model to be made more robust in the low-precision calculations, the original LeakyReLU activation function was replaced by the activation function ReLU6. The two activation functions differ mainly in their treatment when x is greater than 0. The LeakyReLU function does not perform any operation when x>0, while ReLU6 has a function value of 6 when x>6, increasing a boundary. The function equation of ReLU6 is (3):

$$ReLU6 = \min(\max(0, x), 6), \tag{3}$$

3.4 K-Means++

To reduce the dependence of the clustering results on the K-value selection when obtaining the prior bounding box and to make the initial clustering centers as far apart as possible, the original method was replaced by the K-means++ clustering method [22]. The K-means++ algorithm was adopted to optimize the selection of K initial clustering centers to reduce the clustering bias at the initial clustering points effectively. The K-means++ algorithm obtains better-sized prior bounding boxes and matches them to the corresponding feature maps, thus effectively improving the detection accuracy and recall rate.

The K-means++ algorithm randomly selects a sample point as the first clustering center. The shortest distance between each sample and the existing clustering center is calculated, and the sample is classified based on the nearest clustering center. After calculating the probability for each sample, the sample with the highest probability is selected as the next center using Eq. (4):

$$p = \frac{D(x)^2}{\sum_{i=1}^{n} D(x_i)^2}, \tag{4}$$

where $\boldsymbol{D(x)}$ is the shortest distance from each sample point to the current center, the clustering center can recalculate the objects based on the existing clusters, repeating the process until no objects are reassigned to other clusters.

Finally, the K clustering centers are selected. In this study, the anchor box widths and heights obtained using K-means++ clustering are as follows: (30,30), (59,45), (48,62), (79,64), (67,95), (97,87), (107,135), (149,106), and (172,173).

3.5 Improvements on the Enhanced Feature Extraction Network

To further reduce the number of network parameters and optimize the model, some of the general convolutions in the YOLOv4 enhanced feature extraction network were replaced with depthwise separable convolutions. The specific changes introduced in the proposed methodology are as follows:

(1) The activation function in the 3×3 general convolution was changed from LeakyReLU to ReLU6 for more robustness in case of low-precision calculations.

(2) The partial convolution in the three convolution blocks and the five convolution blocks of the PANet network were replaced with depthwise separable convolutions to reduce the number of parameters without affecting feature extraction as much as possible.

(3) The 3×3 general convolution in both down-samplings with a depthwise separable convolution was replaced.

The number of model parameters of the replaced MobileNetV3-YOLOv4 network was reduced from 64,363,101 to 11,692,029, equivalent to 1/5 of the model parameters of the original network, significantly affecting network optimization. The improved MobileNetV3-YOLOv4 network structure is shown in Fig. 6, with the orange part of the figure highlighting the improvements.

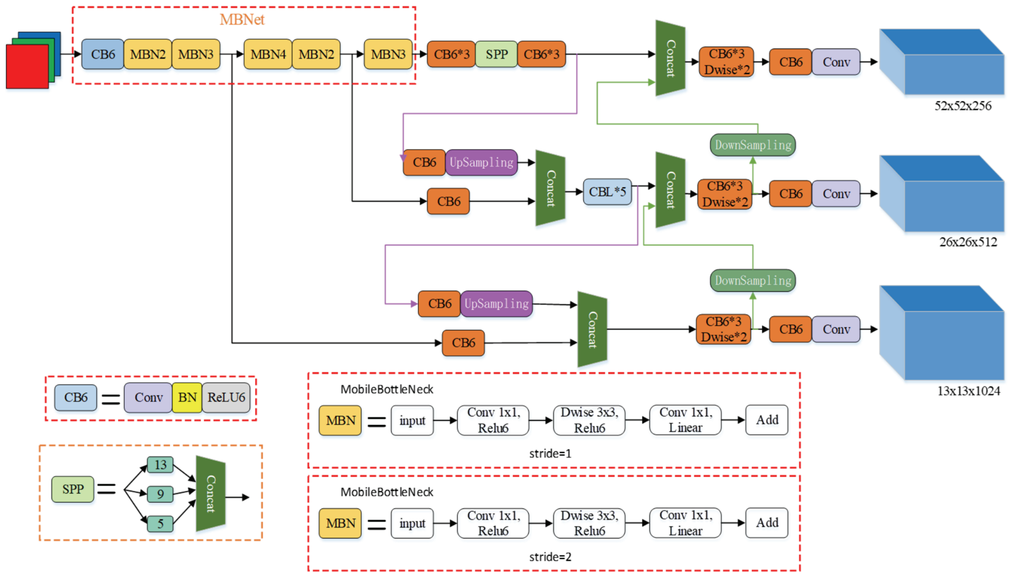


Fig. 6. The improved MobileNetV3-YOLOv4 network model structure.

4. Results

The dead tree image dataset used in the experiments contained 10,000 images, 80% of which were used for training and validation and the remaining 20% for the test set. Fig. 7 shows some of the image data.

4.1 Experimental Environment & Evaluation Indicators

The experiments used the open-source TensorFlow framework to implement the network model in Python, with an NVIDIA RTX 3080 (10 GB) graphics card and 32 G of RAM. The initial learning rate of the training model was set to 0.0001, and the cosine annealing decay strategy was simulated to adjust the learning rate of the network with epoch set to 600.

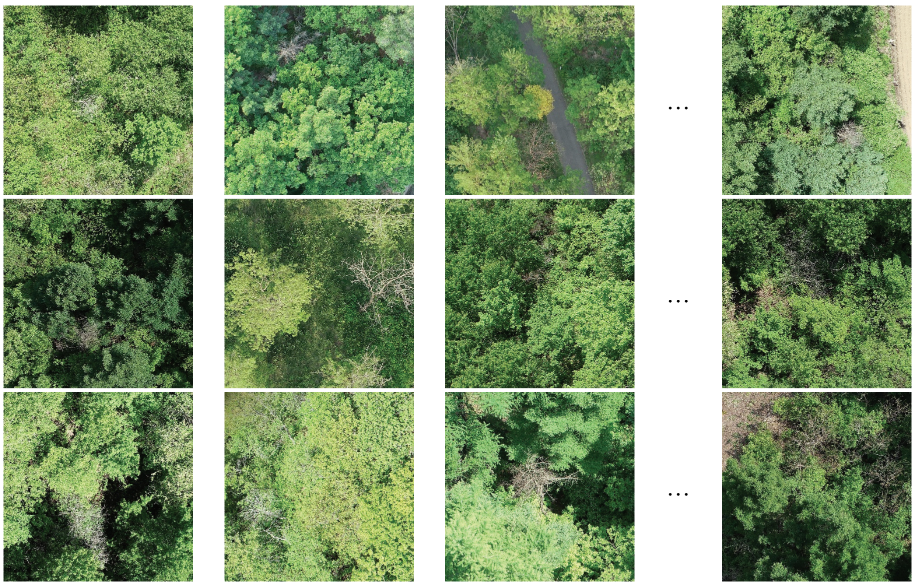


Fig. 7. Partial dataset images.

Subjective and objective evaluations were used, with the subjective evaluation being mainly human eye interpretation and the objective indicators being frames per second (FPS) and average precision (AP). FPS is the frame rate per second of detected images. Precision-Recall (P-R) is used to analyze the accuracy of the network predictions, where each metric is calculated as follows:

$$\text{Precision} = \frac{TP}{TP+FP}, \tag{5}$$

$$\text{Recall} = \frac{TP}{TP+FN}, \tag{6}$$

$$\text{AP} = \int_0^1 P(r)dr, \tag{7}$$

where true positive (TP) is the number of positive class samples that have been correctly detected and judged to be positive; false positive (FP) is the number of negative samples that have been incorrectly detected and judged to be positive; false negative (FN) is the number of positive class samples that have not detected and judged to be negative; $\boldsymbol{r}$ is the recall value, and $\boldsymbol{P(r)}$ is the precision value corresponding to the r-value [23].

The area of the P-R curve is then AP. In target detection tasks, the AP metric is used to measure the accuracy of a particular class of target detection for the good or bad performance of single target detection algorithms. Since only dead trees were the subject of target detection, the AP metric was used.

4.2 Analysis of Experimental Results

4.2.1 Impact of different lightweight backbone networks on model performance

Three lightweight networks (i.e., MobileNetV1, MobileNetV2, and MobileNetV3) were used to replace the backbone network in the original YOLOv4 model to show the superiority of replacing the backbone network; the same dataset was trained using the three networks. The weights obtained after training were also used to predict and compare the test set. The experimental results are shown in Table 1.

Table 1. Prediction results of different backbone networks

Model	Number of parameters	AP (%)	FPS
MobileNetV1-YOLOv4	12,692,029	84.70	90.52
MobileNetV2-YOLOv4	10,801,149	80.10	60.34
MobileNetV3-YOLOv4	11,729,069	88.49	58.43

Table 1 shows that the YOLOv4 model using MobileNetV3 as the backbone network achieves an 88.49%, which outperforms the other two models by 3.79% and 8.39%. Table 1 also shows the number of parameters and FPS of the three models; the number of MobileNetV3-YOLOv4 parameters is in between the three models. Due to the more straightforward structure of the MobileNetV1 network, it has a more significant FPS and faster detection speed, while the difference in detection speed between MobileNetV3-YOLOv4 and MobileNetV2-YOLOv4 is more negligible. Thus, in a comprehensive view, MobileNetV3-YOLOv4 provides better dead tree detection than the three models.

4.2.2 Effect of different enhanced feature extraction networks on model performance

The training results from the original and improved extraction networks were analyzed and compared using the same dataset to see if the addition of ReLU6 and depthwise separable convolution can optimize the model. Table 2 summarizes the results of the two models.

Table 2. Comparison of different enhanced feature extraction networks

Model	AP (%)	FPS
Original enhanced feature extraction network	88.49	58.43
Improved enhanced feature extraction network	89.77	69.15

Table 2 shows that the improved extraction network increased the model’s detection and detection speed, with an AP value of 89.77%, improving the original network by 1.28%. This increment in AP value means that the improved network can retain features better and reduce feature loss while enhancing feature learning. The proposed algorithm can retain image information to a greater extent and is more suitable for detecting small targets that usually have sparse and difficult-to-learn features. After replacing the general convolution with a lighter depthwise separable convolution, the FPS was improved by 10.72, considerably improving the model’s detection speed.

4.2.3 Impact of clustering algorithms on model performance

This paper analyzed and compared clustering using the K-means and K-means++ methods in network training to evaluate the prior bounding box obtained by K-means++. The clustering methods' prior bounding boxes were then used for training and prediction on the same test set, with the results shown in Table 3.

The AP for clustering using K-means++ was 97.33%, equivalent to 7.56% higher than the YOLOv4 using K-means. The results show that K-means++ can achieve optimized clustering centers with aspect ratios that better match the characteristics of the dead tree dataset. Therefore, using the K-means++ algorithm for training and testing makes it easier to fit the prior bounding box to the target, reducing model training difficulties, enhancing localization, and improving the algorithm's detection accuracy.

Table 3. Comparison of different clustering algorithms

Clustering method	AP (%)	FPS
K-means	89.77	69.15
K-means++	97.33	68.68

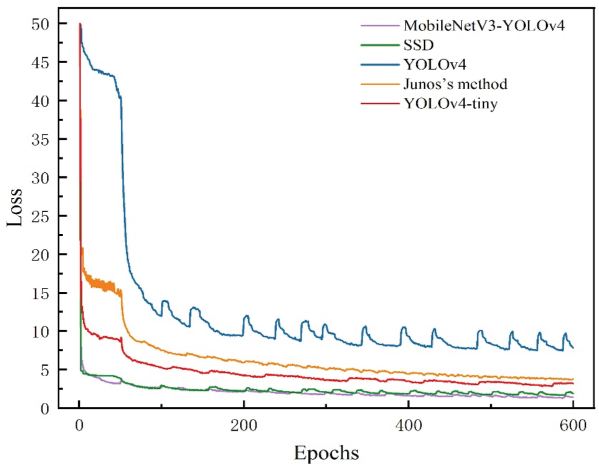


Fig. 8. The changing trend of loss in model training.

4.2.4 Comparison experiments of different detection models

To further evaluate model reliability, the same experimental data were used on four other models (i.e., YOLOv4, YOLOv4-tiny, SSD [24], Junos' method [9]) and the proposed MobileNetV3-YOLOv4 model. Fig. 8 presents the trends for the five training models. As shown in Fig. 8, YOLOv4, YOLOv4-tiny, and Junos' methods had higher losses than the other two. The SSD algorithm iterates to a plateau at epoch 200 but is slightly higher than the proposed algorithm. The proposed algorithm has the fastest decreasing loss between epochs 0–200, and the loss curve flattens out at epoch 400.

To verify the effectiveness of the improved algorithm in detecting dead trees and visually comparing the models, each of the five models was used to detect the test set images. Fig. 9 presents some of the results. In Fig. 9(a), the SSD model misidentified similarly colored roads as dead trees, while the YOLOv4, YOLOv4-tiny, and Junos models could not detect them. In Fig. 9(b), the other four models missed detection when the background was complex, while the proposed algorithm accurately detected dead trees in the images. In Fig. 9(c), the proposed algorithm performed target detection well, while the other four models missed detection.

The five algorithms previously mentioned were selected and tested on a test set to verify the performance of the improved MobileNetV3-YOLOv4 algorithm, as shown in Table 4.

As shown in Table 4, the MobileNetV3-YOLOv4 detection model has a higher AP value than the YOLOv4 model by 14.62%, higher than the YOLOv4-tiny model by 13.66%, higher than the SSD model by 16.65%, and higher than the Junos' method by 8.74%. The MobileNetV3-YOLOv4 model has a higher FPS than YOLOv4 and is lower than the SSD, YOLOv4-tiny, and Junos' method.

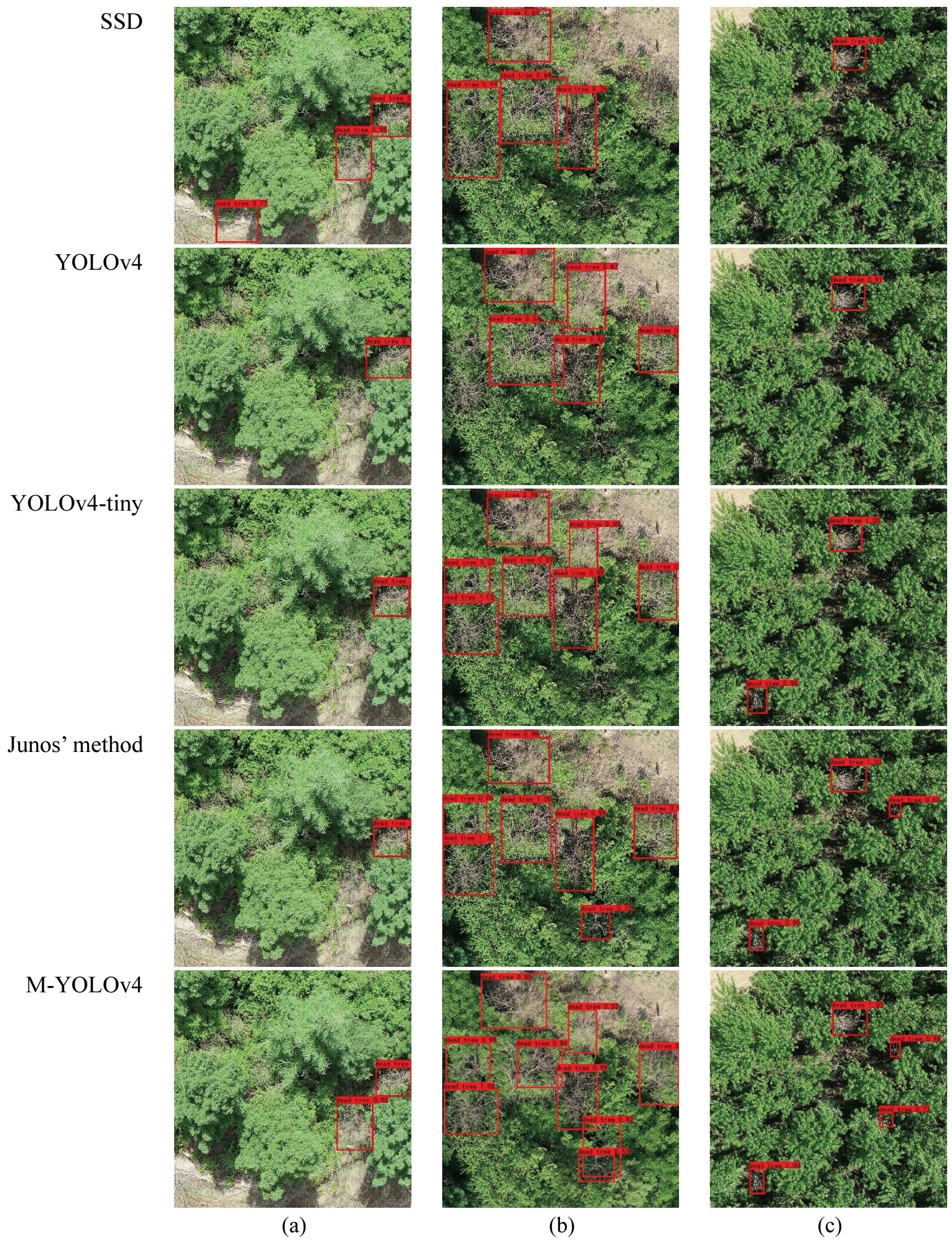


Fig. 9. Image test results: (a) comparison of confusing target detection, (b) comparison of complex background detection, and (c) comparison of small target detection.

Table 4. AP value of four models for test set detection

Model	AP (%)	FPS
YOLOv4	82.71	47.45
YOLOv4-tiny	83.67	196.22
SSD	80.68	110.37
Junos' method	88.59	179.54
MobileNetV3-YOLOv4	97.33	68.68

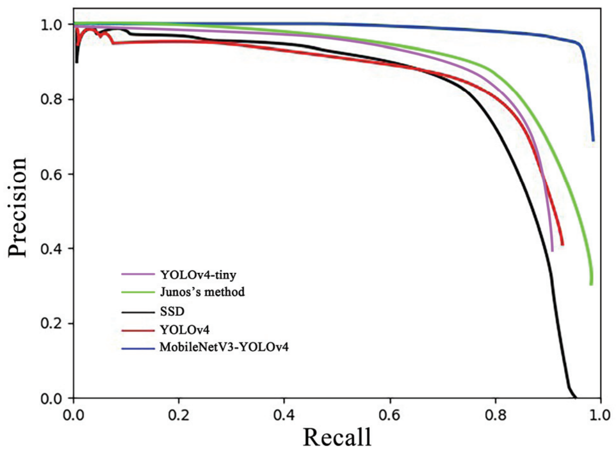
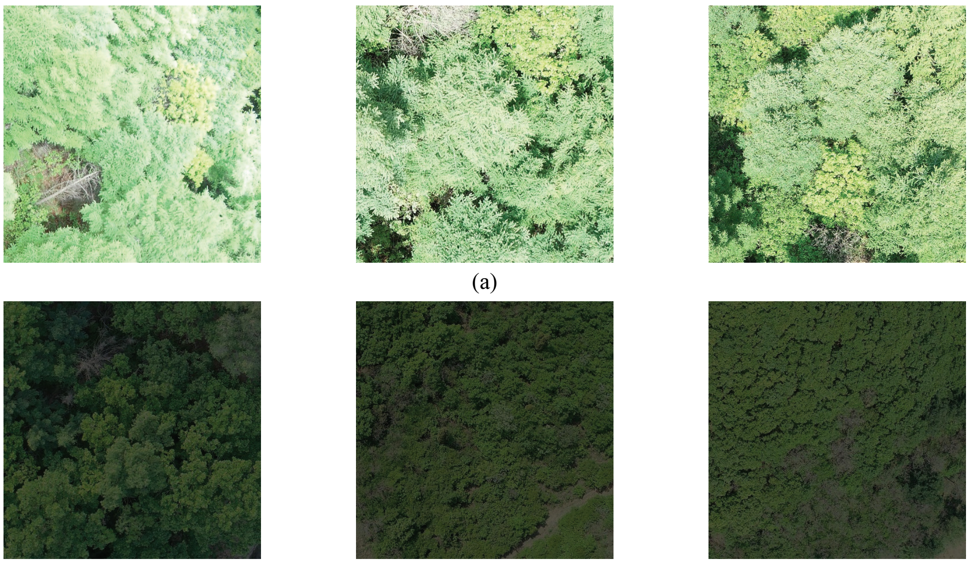


Fig. 10. P-R curve.

The AP values were recorded during the training process. Fig. 10 shows the P-R curves for YOLOv4, YOLOv4-tiny, SSD, Junos' method, and MobileNetV3-YOLOv4 models for dead tree detection. The more extensive the coverage of the P-R curve on the coordinate system, the higher the detection accuracy and the better the model effect. As presented in Table 4, the AP values for YOLOv4, YOLOv4-tiny, and SSD were around 80%, indicating that the P-R curves for the three models are similar. The curve for MobileNetV3-YOLOv4 covers almost the entire coordinate system and also lies above the other curves, which suggests that the MobileNetV3-YOLOv4 model outperforms the other four models.

4.2.5 Testing model image adaptation experiments

We screened 600 images of dead trees in bright and dark conditions through manual processing to obtain the effect of the model on detection accuracy, as shown in Fig. 11.



(b)

Fig. 11. Different state partial images (a) in a bright state and (b) in a dark state.

Then, by comparing and verifying the adaptability of the models in detecting images, the YOLOv4, YOLOv4-tiny, SSD, and proposed algorithms were used on various images at different brightness. The experimental results are shown in Table 5.

The algorithm had an accuracy of 97.85% for bright images, which is 12.35%, 14.3%, and 11.29% higher than YOLOv4, YOLOv4-tiny, and SSD, respectively. In images with darker settings, the algorithm achieved good detection results; the SSD algorithm also achieved accurate results, although slightly lower than the proposed algorithm by 2.24%. The YOLOv4 and YOLOv4-tiny algorithms were less effective in detecting images in the dark, achieving an accuracy of 88.80% and 86.23%, lower than the proposed algorithm by 9.35% and 11.92%. The results suggest that the proposed approach is better at image detection of dead trees for different environmental conditions.

Table 5. Comparison of four models for different image detection results (unit: %)

State	YOLOV4	SSD	YOLOv4-tiny	MobileNetV3-YOLOv4
Brightness	85.50	86.56	83.55	97.85
Darkness	88.80	95.91	86.23	98.15

4.2.6 Impact of image enhancement on model accuracy

A total of 6,000 images (original and processed) were used to investigate the effects of image enhancement on model detection accuracy; the experimental results are shown in Table 6.

Table 6. Image enhancement contrast experiment

Type	AP (%)	FPS
Original images	93.27	63.66
Enhanced images	95.87	65.21

The dead tree detection accuracy after training with image enhancement was 95.87%, an improvement of 2.60% compared to the original image. The results show that the image enhancement process allows the model to detect more feature points and learn dead tree features more accurately, improving the model's detection capability.

4.2.7 Effects of different parameters on the accuracy of the model

(1) Effects of different batch sizes on model accuracy

Different batch sizes (i.e., 4, 8, and 16) were used in the experiments to evaluate the influence of batch sizes on model accuracy. The experimental results are shown in Table 7.

From Table 7, the highest AP value was 97.33% for batch size 8, which is 0.69% and 7.03% higher compared to batch sizes 4 and 16. It is possible that when the batch size is set too small, the model slowly converges; when set too high, the memory becomes insufficient, causing the model's generalization ability to weaken. Therefore, the detection effect is better in batch size 8.

Table 7. Comparison of different batch sizes

Batch size	AP (%)	FPS
4	96.64	63.68
8	97.33	68.68
16	90.30	64.85

(2) Effect of the number of different datasets on the accuracy of the model

Eight datasets analyzed the effect of different numbers of datasets on model accuracy with varying image counts (i.e., 1,000, 2,000, 3,000, 4,000, 5,000, 6,000, 7,000, and 8,000) were used in the experiments. The experimental results are shown in Table 8.

As the number of datasets increased, the AP value for model detection also increased. The model detection accuracy improved by 8.5% and 8.92% when the images were increased from 1,000 to 2,000 and from 3,000 to 4,000, respectively. The model achieved better detection accuracy when the number of datasets reached 6,000, after which the accuracy improved by approximately 2% for every additional 1,000 datasets. The highest detection accuracy was 97.33% when images reached 8,000.

Table 8. Comparison of different datasets

Quantity	AP (%)	FPS
1,000	63.07	64.69
2,000	71.57	63.62
3,000	76.24	61.93
4,000	85.16	63.22
5,000	89.85	64.36
6,000	93.27	63.66
7,000	95.92	64.61
8,000	97.33	68.68

5. Discussion and Conclusion

This paper applied deep learning and drones for dead tree detection, developing an innovative approach that could significantly improve efficiency, reduce costs, and decrease terrain-related dangers. This proposed methodology is based on the MobileNetV3-YOLOv4 network of UAV imagery, using consumer-grade UAVs in acquiring tree imagery. An improved logarithmic transformation method is introduced in data pre-processing to address the overbrightening or overexposure of images after image enhancement. Adjusting the parameter V in the improved equation increases the flexibility of image adjustment and enhances the target features. Images processed using the proposed enhancement method were fed into the model for training, and the trained model improved the accuracy of dead tree detection. In terms of detection, the lightweight model MobileNetV3-YOLOv4 is proposed to reduce the number of model parameters and improve detection accuracy for small targets while ensuring extraction performance. The K-means++ meth-od was also introduced to generate a prior bounding box closer to the ground truth box, significantly improving the model detection performance. The detection accuracy rate reached 97.33%, and the proposed approach to detect dead trees confirmed the proposed method's reliability. By significantly improving the efficiency of dead tree detection, the proposed image detection approach has novelty and application values.

This paper contains some shortcomings to be improved in subsequent studies. For example, there were still cases of false detection in the case of complex tree images, and the accuracy would have to be further improved. In addition, the image data used in this paper were taken by a consumer-grade UAV and hence were relatively homogeneous. In future research, more complex and diverse image data can be collected and produced from different devices. At the same time, the model can be further improved to enhance its generalizability and verify the method's feasibility to maximize the algorithm's value.

Acknowledgement

This research was supported by the Fund project of the Provincial Education Department (No. LJKMZ20220638) and the Open Fund Project of the Marine Information Technology Innovation Center of the Ministry of Natural Resources.

References

[1] A. Kaminska, M. Lisiewicz, K. Sterenczak, B. Kraszewski, and R. Sadkowski, "Species-related single dead tree detection using multi-temporal ALS data and CIR imagery," *Remote Sensing of Environment*, vol. 219, pp. 31-43, 2018. https://doi.org/10.1016/j.rse.2018.10.005

[2] K. Otsu, M. Pla, A. Duane, A. Cardil, and L. Brotons, "Estimating the threshold of detection on tree crown defoliation using vegetation indices from UAS multispectral imagery," *Drones*, vol. 3, no. 4, article no. 80, 2019. https://doi.org/10.3390/drones3040080

[3] S. Malek, Y. Bazi, N. Alajlan, H. AlHichri, and F. Melgani, "Efficient framework for palm tree detection in UAV images," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 7, no. 12, pp. 4692-4703, 2014. https://doi.org/10.1109/JSTARS.2014.2331425

[4] W. Li, H. Fu, L. Yu, and A. Cracknell, "Deep learning based oil palm tree detection and counting for high-resolution remote sensing images," *Remote Sensing*, vol. 9, no. 1, article no. 22, 2016. https://doi.org/10.3390/rs9010022

[5] M. Culman, S. Delalieux, and K. Van Tricht, "Individual palm tree detection using deep learning on RGB imagery to support tree inventory," *Remote Sensing*, vol. 12, no. 21, article no. 3476, 2020. https://doi.org/10.3390/rs12213476

[6] E. Guirado, S. Tabik, D. Alcaraz-Segura, J. Cabello, and F. Herrera, "Deep-learning versus OBIA for scattered shrub detection with Google earth imagery: Ziziphus Lotus as case study," *Remote Sensing*, vol. 9, no. 12, article no. 1220, 2017. https://doi.org/10.3390/rs9121220

[7] H. Tao, C. Li, D. Zhao, S. Deng, H. Hu, X. Xu, and W. Jing, "Deep learning-based dead pine tree detection from unmanned aerial vehicle images," *International Journal of Remote Sensing*, vol. 41, no. 21, pp. 8238-8255, 2020. https://doi.org/10.1080/01431161.2020.1766145

[8] R. Yu, Y. Luo, Q. Zhou, X. Zhang, D. Wu, and L. Ren, "Early detection of pine wilt disease using deep learning algorithms and UAV-based multispectral imagery," *Forest Ecology and Management*, vol. 497, article no. 119493, 2021. https://doi.org/10.1016/j.foreco.2021.119493

[9] M. H. Junos, A. S. Mohd Khairuddin, S. Thannirmalai, and M. Dahari, "Automatic detection of oil palm fruits from UAV images using an improved YOLO model," *The Visual Computer*, vol. 38, pp. 2341-2355, 2022. https://doi.org/10.1007/s00371-021-02116-3

[10] X. Liu, K. H. Ghazali, F. Han, and I. I. Mohamed, "Automatic detection of oil palm tree from UAV images based on the deep learning method," *Applied Artificial Intelligence*, vol. 35, no. 1, pp. 13-24, 2021. https://doi.org/10.1080/08839514.2020.1831226

[11] K. Yarak, A. Witayangkurn, K. Kritiyutanont, C. Arunplod, and R. Shibasaki, "Oil palm tree detection and health classification on high-resolution imagery using deep learning," *Agriculture*, vol. 11, no. 2, article no. 183, 2021. https://doi.org/10.3390/agriculture11020183

[12] R. Girshick, “Fast R-CNN,” in *Proceedings of the IEEE International Conference on Computer Vision*, Santiago, Chile, 2015, pp. 1440-1448. https://doi.org/10.1109/ICCV.2015.169

[13] A. Buslaev, V. I. Iglovikov, E. Khvedchenya, A. Parinov, M. Druzhinin, and A. A. Kalinin, “Albumentations: fast and flexible image augmentations,” *Information*, vol. 11, no. 2, article no. 125, 2020. https://doi.org/10.3390/info11020125

[14] X. Wang, Z. Jia, J. Yang, and N. Kasabov, “Change detection in SAR images based on the logarithmic transformation and total variation denoising method,” *Remote Sensing Letters*, vol. 8, no. 3, pp. 214-223, 2017. https://doi.org/10.1080/2150704X.2016.1258125

[15] M. P. Mathew and T. Y. Mahesh, “Leaf-based disease detection in bell pepper plant using YOLO v5,” *Signal, Image and Video Processing*, vol. 16, no. 841-847, 2022. https://doi.org/10.1007/s11760-021-02024-y

[16] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, “The pascal visual object classes (VOC) challenge,” *International Journal of Computer Vision*, vol. 88, pp. 303-338, 2010. https://doi.org/10.1007/s11263-009-0275-4

[17] A. Bochkovskiy, C. Y. Wang, and H. Y. M. Liao, “YOLOv4: optimal speed and accuracy of object detection,” 2020 [Online]. Available: https://arxiv.org/abs/2004.10934.

[18] A. Howard, M. Sandler, G. Chu, L. C. Chen, B. Chen, M. Tan, et al., “Searching for MobileNetv3,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, Seoul, South Korea, 2019, pp. 1314-1324. https://doi.org/10.1109/ICCV.2019.00140

[19] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, “MobileNets: efficient convolutional neural networks for mobile vision applications,” 2017 [Online]. Available: https://arxiv.org/abs/1704.04861.

[20] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L. C. Chen, “MobileNetv2: inverted residuals and linear bottlenecks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, 2018, pp. 4510-4520. https://doi.org/10.1109/CVPR.2018.00474

[21] J. Hu, L. Shen, and G. Sun, “Squeeze-and-excitation networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Salt Lake City, UT, 2018, pp. 7132-7141. https://doi.org/10.1109/CVPR.2018.00745

[22] D. Arthur and S. Vassilvitskii, “K-means++ the advantages of careful seeding,” in *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms*, New Orleans, LA, 2007, pp. 1027-1035.

[23] C. Donmez, O. Villi, S. Berberoglu, and A. Cilek, “Computer vision-based citrus tree detection in a cultivated environment using UAV imagery,” *Computers and Electronics in Agriculture*, vol. 187, article no. 106273, 2021. https://doi.org/10.1016/j.compag.2021.106273

[24] [24] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C. Y. Fu, and A. C. Berg, “SSD: single shot multibox detector,” in *Computer Vision–ECCV 2016*. Cham, Switzerland: Springer, 2016, pp. 21-37. https://doi.org/10.1007/978-3-319-46448-0_2

Yuanhang Jin **https://orcid.org/0000-0003-1482-3027**

He is currently pursuing an M.E. degree at the School of Civil Engineering, University of Science and Technology Liaoning, Anshan, China. In 2019, he received a B.S. degree from the school of civil engineering, Liaoning University of Science and Technology, Anshan, China. His research interests include remote sensing, image detection and deep learning.

Maolin Xu **https://orcid.org/0000-0002-6986-1189**

He is currently a professor at the School of Civil Engineering, University of Science and Technology Liaoning, Anshan, China. His research interest includes precise deformation monitoring technology, artificial intelligence and remote sensing image detection.

Jiayuan Zheng **https://orcid.org/0000-0003-1944-1167**

She is currently pursuing an M.E. degree at the School of Civil Engineering, University of Science and Technology Liaoning, Anshan, China. In 2019, she received a B.S. degree from the School of Civil Engineering, Liaoning University of Science and Technology, Anshan, China. Her research interests include remote sensing, geographical information science and deep learning.