Vol 1 No 2 (2019) *e*-ISSN: 2682-8383

Journal of Engineering Technology and Applied Physics

Oil Palm Tree Detection from High Resolution Drone Image Using Convolutional Neural Network

Chee Cheong Lee*, See Yee Tan, Tien Sze Lim and Voon Chet Koo
Faculty of Engineering and Technology, Multimedia University, Jalan Ayer Kerah Lama, 75450, Melaka, Malaysia
*1141600033@student.mmu.edu.my
https://doi.org/10.33093/jetap.2019.1.2.2

Abstract – **We propose a method to combine several image processing methods with Convolutional Neural Network (CNN) to perform palm tree detection and counting. This paper focuses on drone imaging, which has a high image resolution and is widely deployed in the plantation industry. Analyzing drone images is challenging due to variable drone flying altitudes, resulting in inconsistent tree sizes in images captured. Counting by template matching or fixed sliding window size method often produces an inaccurate count. Instead, our method employs frequency domain analysis to estimate tree size before CNN. The method is evaluated using two images, ranging from a few thousand trees to a few hundred thousand trees per image. We have summarized the accuracy of the proposed method by comparing the results with manually labelled ground truth.**

Keywords—CNN, fourier analysis, palm tree detection, drone imaging

I. INTRODUCTION

Being one of the largest palm oil producer and exporter [1], palm oil industry plays a very significant role in Malaysia economy. Knowing number and location of trees helps management in several aspects, such as predicting the yield of palm oil, amount of fertilizer required for plantation area. It also helps management to predict the time required to harvest a plantation area, thus, improving the productivity.

In the past, number of trees is counted manually, by sending people to count on site. Some may use satellite image to estimate the number of trees by calculating the density. In past few years, drone imaging becomes popular, due to its capability of producing high resolution image. Computer vision research in this area has produced some semi-automated tree counting and localization solutions. S. Daliman *et al.* [2] used Haar-based rectangular windows to extract features from oil palm tree. The features served as input for Support Vector Machine (SVM) for tree classification and counting. B. Kalantar *et al.* [3] suggested object-based analysis to segment the oil palm tree by its colour, shape, size and pixel. Like [2], these extracted features were classified using SVM. H. M. Rizeei *et al.* [4] combined satellite images and LiDAR point cloud data, grouped image pixels data into non-overlapping homogeneous regions, and performed edge-based segmentation. They also used SVM to perform classification.

Convolution Neural Network (CNN) is being used widely in image processing domain. It achieves very good performance in various applications, such as image classification and object detection. Researchers have been using it for tree counting. N. A. Mubin *et al.* [5] used LeNet for tree detection. They produced another two CNNs based on LeNet to detect young and mature tree separately. Images from from WV-3 dataset is first converted into RGB, then split into sub tiles of 26 × 26 pixels for young tree and 31 × 31 pixels for matured tree to feed into CNN classification. Counting is done by counting the number of "True" outputs from CNN. Cheang *et al.* [6] proposed another CNN method and sliding windows for tree detection. The author used LeNet as classifier and have sliding windows size of 40 × 40 pixels due to each palm tree spans about 40 × 40 pixels in the dataset he used. Besides above methods, LeNet is being used in another methods, suggested by Li *et al.* [1]. From the result, LeNet achieved > 96 % accuracy in tree counting using satellite images.

It is proven CNN is the state-of-art technique in palm tree counting. However, performance of these approaches is bound a limitation: inability to set the tree size prior to processing the image. Also, it requires that trees are very well organized. Often in the actual site, the trees are very crowded, overlapping with each other, and they might have forest nearby.

In this paper, we introduce a CNN-based approach on palm tree detection and counting for high resolution drone images. In this method, we collect many tree samples from drone images. We also include frequently appear non-tree objects around palm oil plantation, such as factory, road, river, and forest. These datasets are split into two classes, "Tree" and

"Non-Tree". We train the CNN using these datasets to get a classifier model. First, we apply Fourier transform and analyses the images, estimate the size of the tree; the size represents sliding window size. The sliding window then slides through the image and perform tree prediction using CNN. Output from sliding window is probability map, and finally tree detection is done by detecting the local maxima of probability map, and tree counting is done by counting the number of local maxima from probability map.

This paper is organized as below: We present the datasets in section II, section III explains the framework, results and discussions of the research are reported in section IV, and finally, we conclude at section V.

II. DATASETS

We deployed DJI Phantom 4 for data collection. The images are preprocessed using Agisoft and exported as Geotiff file. We collected images from three sites. Each of the image has different spatial resolution, as low as 0.0186m/pixel, and with three colour band (RGB). Table I illustrates the location of data source, its spatial resolution, and area of plantation.

Table I. Datasets source.

	Location	*Coordinate*	*Spatial Resolution*	*Area (Hectare)*
1	Kluang	1.9172573, 103.481824	0.031m/pixel	26.081
2	Palong Timur	2.7960827, 102.715298	0.0186m/pixel	31.3000
3	Ulu Belitong	1.9353581, 103.477460	0.0475m/pixel	265.350

All sites are located in Johor, which is the largest palm oil plantation area in West Malaysia. Locations are illustrated in Fig. 1.

Fig. 1. Site locations mark as 1, 2, 3 accordingly.

Images from site 3 are used to generate palm tree datasets for CNN training. We evaluate on site 1 and site 2 images.

III. FRAMEWORK

A. Overview

Given an image, the target is to detect and get the number of palm tree. In contrast with other CNN approaches, we perform image preprocessing before CNN. We begin by obtaining sliding window size, followed by CNN, and finally image smoothing to filter noisy peaks, and find local maxima. Block diagram of proposed method is shown as Fig. 2.

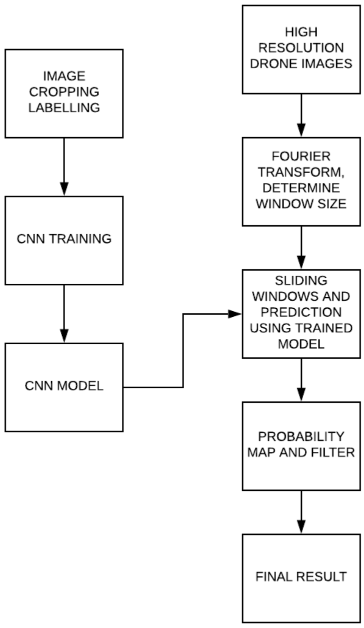


Fig. 2. Framework of proposed method.

B. Datasets Preparation and Training Process

Geotiff image is produced using hundreds to thousands of snapshots from drone and stitch together. Snapshots collected from site 3 are used to prepare datasets. We randomly select 274 snapshots to prepare datasets. It is done by manually cropping from snapshots and label tree and non-tree objects from the images. Example datasets are shown in Fig. 3.

Fig. 3. Example datasets. Top row are tree objects. Bottom are non-tree object.

There are in total 5093 images, 4942 are positive image, and 151 are negative images. All datasets are resized to 299 × 299 pixels to fit the input size of CNN model. We use InceptionV2 [7] as our CNN model. We applied image augmentation to boost the performance of the network. Among them are, image rotation, width shift, height shift, image shear, image zoom, horizontal flip and vertical flip. The datasets are split into two groups, 70 % for training, 30 % for evaluation. We have less negative image because we train the network on top of pre-trained model, where pre-trained model has been trained for general object recognition and perform quite well on non-tree objects.

C. Sliding Window Technique

Tree detection is done by sliding window technique. It is important to determine the right sliding window size. Large size may consist several trees in a window, expected result from CNN will be the window has high probability being a tree, but not how many are them. This will lead to inaccurate count. In the other hand, smaller sliding windows size may not fit a complete tree in a window, hence cause detection with low probability through the whole sliding window process. Sliding window size can be set by determine tree size in an image. An ideal windows size is the size of the one tree; however, trees has minor variation in size. As such, we take an average tree size as sliding windows size.

D. Fourier Analysis

Image contains many trees with consistent in size due to there is standard plantation pattern to optimize land used. We can safely assume the tree density is uniform across the image. These tree usually occupy hundreds of pixel and they are planted side by side, hence, they form repetitive pattern in the image. These uniform repetitive features can be captured by Fourier Analysis $f(\xi)$ [8]. Periodical occurrence will be shown as peak in frequency domain. We convert image to gray scale, and apply Gaussian filter to remove high frequency component (tree leaves) to avoid it appears as false peak in frequency domain.

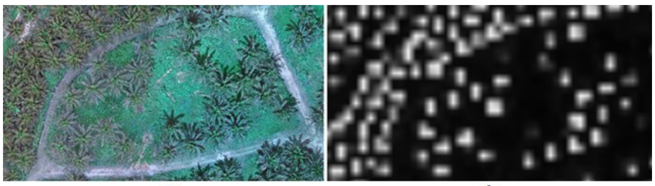
The peak has a coordinate *(X,Y)* representing the contribution of the repetitive pattern along horizontal and vertical axis with image resolution (W_I, H_I). The sliding windows size (W_S, H_S) can be calculated using Eq. (1).

$$W_S = W_I/X$$
$$H_S = H_I/Y \qquad (1)$$

E. Tree Detection

From (W_S, H_S) obtain from Fourier Analysis, we use sliding window technique, to perform tree detection. The sliding window size is dynamic according to the tree size. Thus, sliding window image should be resized to 299 × 299 pixels to match with CNN input. The output from sliding window is a probability map.

Sliding step is also an important factor for the accuracy and processing time. By default, it is set to 1, which is smallest, and has highest accuracy. Probability map generated using this method will have image size of $(W_I\text{-}W_S\text{-}1, H_I\text{-}H_S\text{-}1)$, and total number of label prediction is performed $(W_I\text{-}W_S\text{-}1)*(H_I\text{-}H_S\text{-}1)$, times. To reduce processing time, we can set sliding window to larger value, but not too large to make it miss the tree. From experimental result, optimized sliding step is $(W_S/5, H_S/5)$. Figure 4 shows extracted RGB image and its probability map.



a b

Fig. 4. (a) Input image. (b) Probability map.

F. Tree Counting

After tree prediction process, it generates a probability map. A region with high probability indicates there is a tree. Apply local max filter to probability map, we can obtain the tree count. Nevertheless, sliding window may slide through the same tree many times and generate many high probability pixels, especially when we set sliding step to 1. It will reflect as false peaks in probability map, and result in local max filter return many counts for one tree. Our goal is to count one peak per tree. To solve this problem, we apply a gaussian filter before local max filter. The tree size is known and distance between trees are uniform, we can set the filter kernel size (W_K, H_K) to one-fifth of sliding window size to filter noise peaks, while keeping the tree probability pattern. Figure 5 shows noise peaks filtering. To ensure the filter kernel size is odd number, we can always add 1 if kernel size if not odd number as shown in Eq. (2):

$$W_K = \begin{cases} W_s/5, & W_s/5 = Odd \\ W_s/5 + 1, & W_s/5 = Even \end{cases}$$

$$H_K = \begin{cases} H_s/5, & H_s/5 = Odd \\ H_s/5 + 1, & H_s/5 = Even \end{cases} \qquad (2)$$

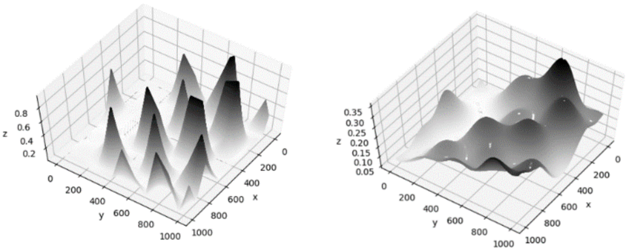


Fig. 5. Before and after filtering probability map.

IV. RESULT AND DISCUSSION

To evaluate the performance of the proposed method, we used confusion matrix to record True Positive, True Negative, False Positive, and False Negative. Evaluation metrics: Precision, Recall and F1 as describe in Eqs. (4), (5) and (6) will be calculated from confusion matrix.

$$Precision = \frac{Correctly\ Detected\ Trees}{Correctly\ Detected\ Trees + False\ Detected\ Trees} \qquad (4)$$

$$Recall = \frac{Correctly\ Detected\ Trees}{Total\ Number\ of\ Trees\ in\ Ground\ Truth} \qquad (5)$$

$$F1 = 2 * \frac{Precision * Recall}{Precision + Recall} \qquad (6)$$

Table II. Tree counting result.

Evaluation Matrix	***Site 1***	***Site 2***
Actual Count	2830	2331
True Positive	2781	2269
False Negative	49	62
False Positive	30	6
Precision(%)	98.93	99.74
Recall(%)	98.27	99.38
F1 Score(%)	98.6	98.54

Table II illustrates the accuracy of proposed method. F1 score for site 1 and site 2 are 98.6 % and 98.54 %. The result shows the effectiveness of the method in detection and counting.

From the result, proposed method performs well in most of the area in image. There is trend indicates false positive is lesser than false negative. It represents the proposed methods perform very well in detecting non-tree object, and it more likely to miss detect a tree rather than detect a random object as tree. Figure 6 shows the output image, which RGB image overlay with probability map and tree counting label.

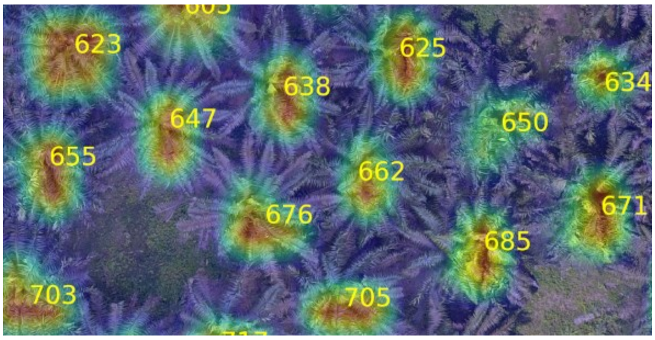


Fig. 6. Tree counting result. Image overlay with probability map.

It can be observed the tree shape is heavily distorted at the edge of image, as shown in Fig. 7. It causes CNN to be unable to recognize the image successfully and contribute most of false negative in the result. This is due to there is no distorted image sample in the training sample. It is our intention not to include distorted images as part of training datasets. The features of distorted images are very random, and it will affect CNN performance by including such images under palm tree class. Nonetheless, it is unavoidable to have such tree in drone image, because image stitching algorithm nature.

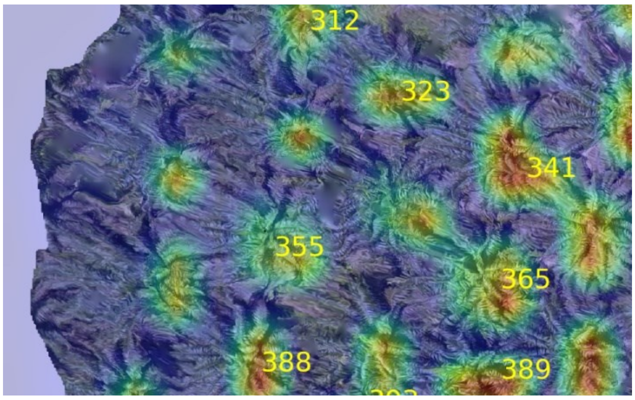


Fig. 7. Heavily distorted image at the edge of image result in false negative.

Comparing to other CNN methods by [1, 5, 6], the advantage of proposed method is to determine windows size dynamically. It can analyze drone image with different flying altitude. Considering some plantation areas have matured and young tree, and dynamic window size will only fit either matured or young tree. It will not perform well in this plantation area. Method [5] propose manually tune network and window size to detect matured and young tree separately, thus the method can be applied in this plantation area.

V. CONCLUSION

In this paper, we implemented and evaluated the performance of InceptionV2 with sliding window technique for palm tree detection. The experiment is done with drone image, illustrating convincing result in tree detection, which having F1 Score 98.6 % and 98.54 % respectively. The methods can also dynamically decide sliding window size and sliding step. In other methods by [1, 5, 6], CNN is also used for tree classification, but it requires window size to be set manually. In our coming work, we will focus to improve the processing time of the algorithm. Sliding window perform well, but it is computationally intensive technique and slow. Furthermore, we will continue to explore other method to detect distorted tree at the edge of image.

ACKNOWLEDGEMENT

The author would like to acknowledge support from iRadar Sdn Bhd in providing datasets as well as high performance computing for this research.

REFERENCES

[1] W. Li, "Deep Learning Based Oil Palm Tree Detection and Counting for High-Resolution Remote Sensing Images," *Remote Sens,* vol. 9(1), 22, 2017.

[2] S. Daliman, "Development of Young Oil Palm Tree Recognition Using Haarbased Rectangular Windows" *IGRSM,* vol. 37, 012041, 2018.

[3] B. Kalantar, "Smart Counting – Oil Palm Tree Inventory with UAV," *Coordinates*, vol. XIII, no. 05, pp. 17–22, 2017.

[4] H. M. Rizeei, "Oil Palm Counting and Age Estimation from WorldView-3 Imagery and LiDAR Data Using an Integrated OBIA Height Model and Regression Analysis," *Journal of Sensors,* vol. 2018, 2536327, 2018.

[5] N. Mubin, "Young and Mature Oil Palm Tree Detection and Counting using Convolutional Neural Network Deep Learning Method," *International Journal of Remote Sensing,* vol. 40, no. 19, pp. 7500 – 7515, 2019.

[6] E. K. Cheang, "Using Convolutional Neural Networks to Count Palm Trees in Satellite Image," *Unpublished*, 2017.

[7] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," *International Conference on Machine Learning,* PMLR vol. 37, pp. 448-456, 2015.

[8] H. Idrees, "Multi-source Multi-scale Counting in Extremely Dense Crowd Images," *IEEE Conference on Computer Vision and Pattern Recognition,* 10.1109, 2013.