3D MAP SYSTEM FOR TREE MONITORING IN HONG KONG USING GOOGLE STREET VIEW IMAGERY AND DEEP LEARNING

Michael Li[1], Wei Yao[1*]
[1]Department of Land Surveying and Geo-Informatics, The Hong Kong Polytechnic University, Hung Hom, Hong Kong
michael.jm.li@connect.polyu.hk, wei.hn.yao@polyu.edu.hk

ABSTRACT:

In densely built urban areas such as Hong Kong, the positive effect of urban trees is to help maintain high environmental and social sustainability for the city while unmanaged trees lead to negative effects such as accidents, outbreaks of pests and diseases. The public awareness of urban tree population has been increasing and preserving all the benefits offered by trees, a continuous monitoring concept would be required. In this work, an efficient 3D map system for tree inventory in Hong Kong is presented to the based on automated tree detection from publicly available Google street view (GSV) panorama images. First, Convolutional Neural Networks (CNNs) based object detector and classifier - YOLOv3 with pretrained model is adopted to learn GSV images to detect tree objects. GSV depth image has been utilized to decode depth values of each GSV panorama image and will provide accurate information to calculate the tree geographic position. A "field of view" filter was designed to remove duplicated tree detection within the overlapped areas followed by spatial clustering applied to further increase the tree localization accuracy. The average distance between the detected trees and ground truth data was achieved within 3 meters for selected roads used for the experiment. Second, a 3D Map platform prototype for facilitating the urban tree monitoring and management was developed. Currently, there is no true 3D platform for interpreting the results of tree records in Hong Kong city areas. With the help of webGL technology, contemporary browsers are able to show 3D buildings, terrain and other scene components together with the obtained tree records in an open source 3D GIS platform, the level of visualization is enhanced as all the detected trees are placed on the 3D digital terrain model. Consequently, it is easy for end-users to know the actual position of the trees and their distribution.

KEY WORDS: Convolutional Neural Networks, Tree Management, Google Street View, 3D Map, Urban areas

1. INTRODUCTION

Trees are essential to the future of our city. Trees and woods ensure our urban areas to be environmentally more sustainable. But with uncountable trees in our environment, tree management is a challenging task (Deng et al., 2019). Some old trees and dead trees even will be harmful to our society and would cause fatal accidents. Although Hong Kong has developed (Hong Kong Tree Register. 2016) database which consists of 865 tree records, each record is manually inspected and recorded. While there are 2016,698,523 trees from 554 different species in Hong Kong as recorded in October 2018. It is inefficiency to monitor and develop the tree database manually. Most of the existing work addressed tree detection from LiDAR point clouds (Lahivaara et al., 2014, Wu et al., 2018) or from a combination of LiDAR and aerial images (Polewski et al., 2015, Paris and Bruzzone, 2014, Yang et al., 2009). LiDAR supplies with direct object depth information through 3D point clouds, which can distinguish trees from other objects and the ground. However, the acquisition of LiDAR data is expensive because of booking of dedicated flight arrangement. The dense urban canyons in Hong Kong are also challenging for designing the flight lines to minimize the effect of data shadowing by enhancing the visibility to the ground in-between tall buildings. In this study, a new approach using deep learning is developed to determine trees located at the roadside by using publicly available Google street view (GSV) images only. Comparison with real-life situation will be conducted to evaluate the result by identifying areas for future improvements. This study targets two major objectives:

- The first objective is to adapt a Convolutional Neural Networks (CNNs) based object detector for extracting roadside trees in Hong Kong from GSV images followed by estimation of geographic position of the trees using depth images.

- The second part is to build up a 3D map platform for visualizing the tree distribution. With the technology of webGL, most modern browsers can show 3D models, terrain and scene with the tree records in open source 3D map viewer, the level of visualization is enhanced as all the trees are placed on the 3D terrain.

2. RELATED WORKS

Remote sensing data acquisition by using satellite images, aerial images, hyperspectral data, and point cloud from Light detection and ranging (LiDAR) have been commonly utilized for tree detection, tree health assessment and species recognition, etc. Larsen et al. (2011) have carried out comparison of tree crown detection algorithms. Another comparison by Kaartinen et al., (2012) focused on automatic tree extraction from laser scanner data. LiDAR for remote sensing has been used to assess on dead trees by Yao et al. (2012) for developing a methodology to retrieve individual dead tree in a mixed mountain forest using features that are derived from small-footprint airborne full waveform LIDAR data. Lafarge and Mallet (2012) developed an algorithm which reconstruct simultaneously buildings, trees and topologically complex grounds. The approach is experimentally validated on complex buildings and large urban scenes of millions of points.

Torii et al. (2009) presented a structure-from-motion (SfM) pipeline for visual 3D modelling of a large city area using GSV images only. The technique combines the state-of-the-art techniques for feature detection by relative camera motion estimation. The method produces 3D scene with depth information which would be a supplementary information to calculate the tree position. RegisTree project is a collaboration between ETH Zurich and the California Institute of Technology. It aims to investigate and develop an automated system for the detection and classification of public objects in publicly available GSV imagery. Wegner et al. (2016) developed methods to catalog and list the location of street trees at city scale. Their approach uses deep learning by Faster-RCNN (Ren et al., 2015) to detect the trees in the street-view and aerial images and to classify its

*Corresponding author

species. Tree trunk diameter is approximated by using the training model on tree inventories from Pasadena (LA, California) and yielded a performance of 70.6% average precision. The multi-view detection approach that combines information from multiple sources, such as maps, aerial images, and street-view images, by using a conditional random field (CRF).

Krylov et al. (2018) proposed a special model by using Markov Random Field (MRF) to perform objects triangulation and geotagging of recurring stationary objects from GSV imagery. They developed workflows which combine the use of monocular depth estimation and triangulation to enable automatic mapping for similar objects. The triangulation assists to remove duplicates and reduce the number of false positives simultaneously. They conducted experiments on auto detection and geotagging for traffic lights and telegraph poles and reported high recall rates with accuracy within 2 meters, similar to that obtained from a GPS receiver.

3. METHODOLOGY

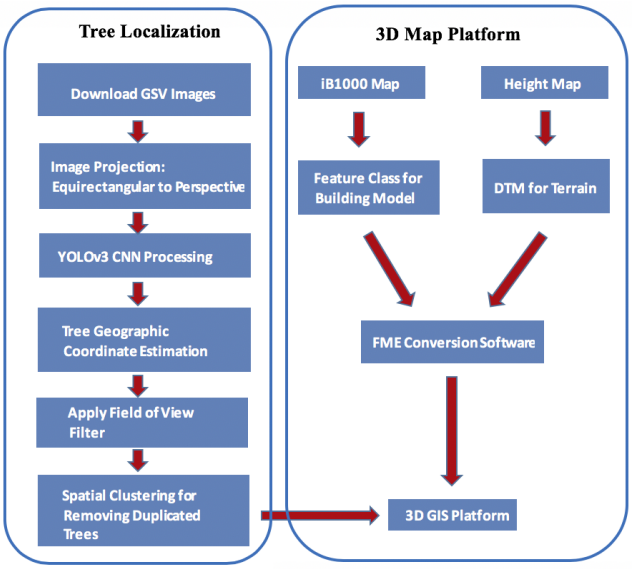


Figure 1. Workflow for automated tree detection and visualization in 3D Map platform

The purpose of this project is to investigate a system and develop a prototype that can automatically estimate the position of individual tree from street-level imagery and 3D map platform for visualization. The overall workflow is stated in Figure 1. The workflow consists of data processing through CNNs and building 3D map platform. The approach is to train deep CNNs model for automatically tree object detection and finally to extract the tree geographic coordinates. First, required street view imagery data is downloaded from Google Map. Second, data will be fed into YOLOv3 network with pre-trained model for transfer learning and processing with respect to the tree detection. Transfer learning is a popular method in computer vision because it allows us to build accurate models in a timesaving way. Instead of starting the training process from scratch, transfer learning starts from patterns that have been learned from a pre-trained model which was trained on a large dataset with many different of classes to solve similar problem. An example of commonly used dataset for pre-trained model on computer vision problems is ImageNet (Russakovsky et al., 2015). Finally, the geographic position of detected trees will be calculated and embedded into 3D Map platform.

3.1 GSV Data collection

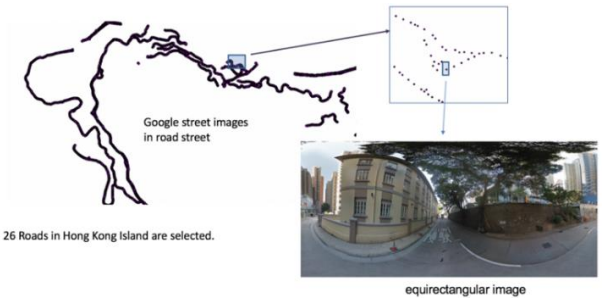


Figure 2. Download google street view images for 26 selected roads. Each dot represents the panorama photo in equirectangular format

According to Google Maps REST API (Google map. 2019), data of panorama photos have been downloaded. We initially use road polygon from open data (Lands Department. 2017) to sample the positions of panorama photos and download the images along each road. In Figure 2, the lines from the left correspond to the roads and each dot represents the position of google street panorama view. Each view provides both the equirectangular image for whole scene in 360° view and the depth map.

3.2 Image Projection

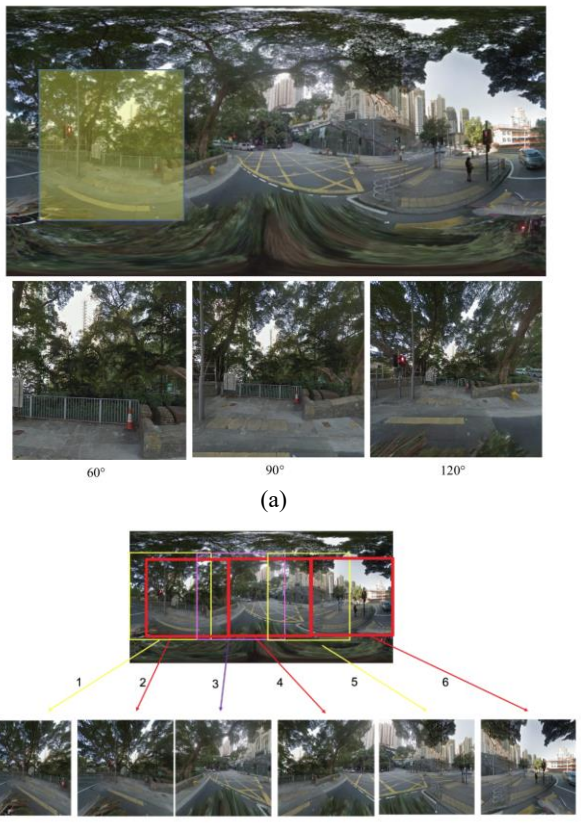


Figure 3. (Top) Equirectangular to perspective projection at different field of view (FOV). Yellow box is an area to carry out the projection. (Bottom) The equirectangular image is exported into 6 perspective images (FOV 114°) for CNNs training

The format for GSV image is in equirectangular projection which is unwrapped from spherical format. Equirectangular image is a common format used by panorama camera vendors

as its rows and columns are scaled in equal distance. To take the advantages of a pre-trained model based on large dataset of perspective images, it requires to project the image from equirectangular to multiple perspective projection. Taking Figure 3 as an example, it shows the multiple perspective images generated from the whole equirectangular image at different FOV (60°, 90° and 120°). The higher the angle of FOV, the wider the view for same size of perspective image. Wikipedia (2019) states that human' eye normally cover 114° (horizontally) of FOV. We used this value of FOV for the parameter of equirectangular to perspective projection in this study. The equirectangular image is exported into 6 overlapped perspective images. Finally, the perspective images will be passed to YOLOv3 network for tree object annotation, training and detection.

3.3 CNNs Processing

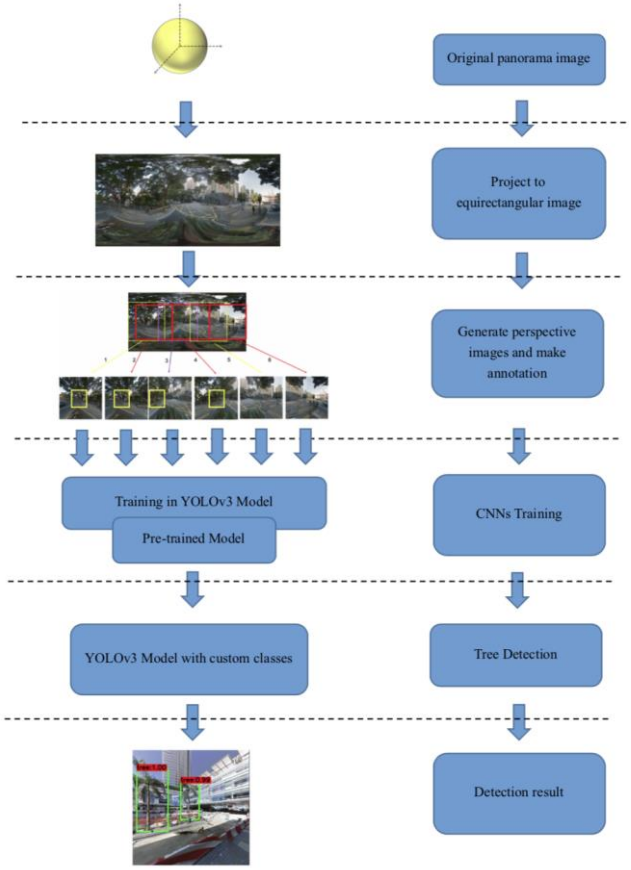


Figure 4. CNNs Processing chain

YOLOv3 Network YOLO stands for "You Only Look Once" which is an extremely fast real time multi-object detection algorithm. It is developed by Redmon et al. (2018). YOLO applies a single neural network to the full image and divides the image into regions to predict the bounding boxes and probabilities for each region. These bounding boxes are weighted by the predicted probabilities. The model has several advantages over classifier-based systems. It looks at the whole image at test time so its predictions are informed by global context in the image. It also makes predictions with a single network evaluation unlike systems such as R-CNN (Girshick et al., 2015) which require thousands for a single image. Redmon et al. (2018) stated that this model is more than 1000x faster than R-CNN and 100x faster than Fast R-CNN.

We refer to the approach recommended by Yang et al. (2018) and Strategy II is selected for the purpose of training and detection. The workflow is illustrated in Figure 4. First, the original downloaded panorama image is projected to equirectangular image. Second, 6 perspective images (Figure 3) which have 114° FOV having size of 416x416px are generated from the equirectangular image. Selection of 416x416px fits to the optimal size for the YOLOv3 model. Third, as this is a machine learning task with supervised learning, we have to enrich the training data with example desired trees planted in local environment. Annotation with tree bounding box on the training data has to be completed before carrying out the training process. Fourth, the annotated images are passed into YOLOv3 model for training. The pre-trained model from ImageNet (Russakovsky et al., 2015) is used for training our own tree object detector instead of learning from scratch. After the training, a newly enriched YOLOv3 model with object classes used only for tree detection task is obtained. Finally, input images will be fed to the new YOLOv3 model for tree detection task.

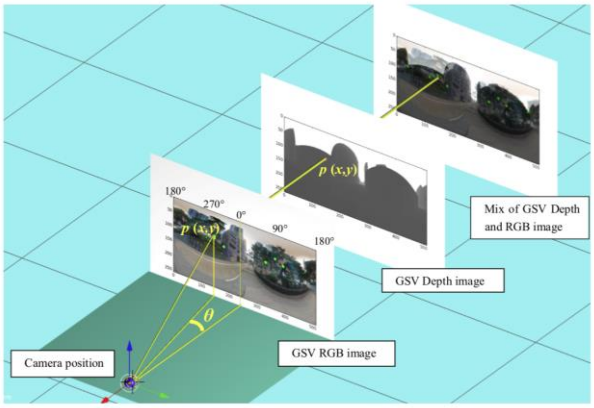
3.4. Estimation of Tree Geographic Coordinate

Referring to Figure 5, each pixel from the GSV Depth image provides accurate distance information from center of camera taking the photo. Fusion of GSV Depth image and RGB color image enable to provide the accurate geographic coordinates for the detected trees. GSV RGB and Depth images are resized to same 512x256px, which are aligned in the same position and orientation framework. Once CNNs detect a tree object, the detected pixel p (x,y) is calculated in GSV RGB image. The aligned GSV Depth image will have same p (x,y) and provide the distance D between the center of camera and the detected tree. As we know the camera geographic coordinate P (lat, lon), the geographic coordinate of the detected tree can be estimated by remapping the detected point p (x,y) into depth image space. The zero yaw angle θ is set to zero at the center of depth image. The yaw angle increases from right to left until going through original point from 0° to 360°. The calculation of geographic coordinate P'(lat, lon) of the detected tree is explained in Eq. (1).

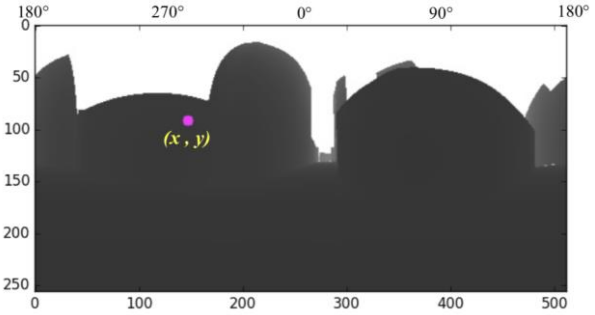
$$\begin{aligned} &\text{If } x <= W/2,\ x' = W/2 - x \\ &\text{If } x > W/2,\ x' = W - (x - W/2) \\ &\theta = 360/W * x' \\ &\theta' = \theta 0 + \theta \end{aligned} \quad (1)$$

P' (lat, lon) = GetDistance(P (lat, lon), D, θ'), where

H is the height of the depth image (256px)
W is the width of the depth image (512px)
x' is the remapped x coordinate in depth map, the center is defined as zero
p (x,y) is the coordinate of the detected point in depth map image space
P (lat, lon) is geographic coordinate of the camera center
P' (lat, lon) is geographic coordinate of the detected tree
D is the distance from the camera center to the detected point
θ is the yaw angle in depth image space
θ^0 is the default GSV tile yaw angle
θ' is the final yaw angle
GetDistance is the function for calculating geographic coordinate *P' (lat, lon)* of the detected tree from camera center point *P (lat0, lon0)* having yaw angle *θ'* and distance *D*



(a)



(b)

Figure 5. (a) Fusion of GSV Depth and RGB color images. The detected beam passes through a same pixel for GSV RGB and Depth images. (b) Depth image

3.5 Field of View Filter

When Google car drives along a road, it takes shoots of spherical panorama imagery continuously. Each panorama imagery 360° field of view around the camera position. Multiple GSV panorama images along a road in same direction could record same trees in their snapshots. The scenario is demonstrated in Figure 6. Panorama images in P1 and P2 are in same direction. Trees (1, 3) are detected from P1 while trees (2, 4) are detected in P2. Under current workflow, the tree geographic position is calculated. There are 4 trees in the map but actually there exist 2 trees only.

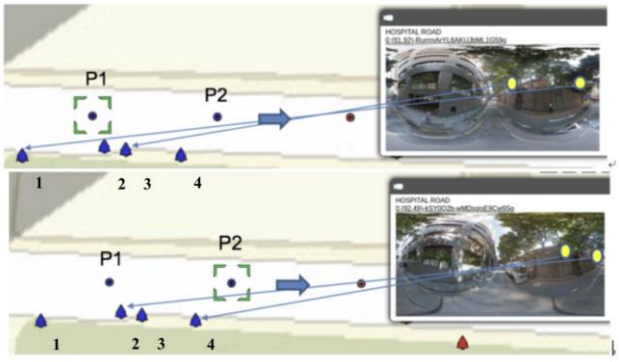
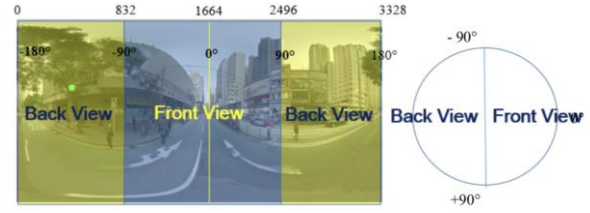
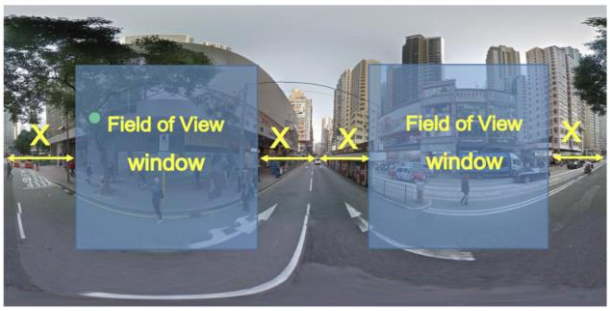


Figure 6. Duplicated tree detected for two street view images P1 and P2 are the panorama points. The arrow shows the direction of driving

Current processing procedures by CNNs cannot filter out the duplicated trees in P1 and P2. Manual filtering is required to apply to the panorama images by finding the best solution to remove the duplicated records. Each equirectangular image represents the panorama view from Google car's camera shooting. The image size is 1664px in height and 3328px in width. The view is 360° to capture a scene. Referring to Figure 7, the front view window is sized as 832px to 2496px in image space which reflects the 180° front view angle from -90° to +90°. A filter (Figure 7) is applied to the equirectangular image by adding an offset X to narrow the field of view window for tree object detection. The offset is defined in pixel dimension. The smaller the rectangle window, the narrow the field of view. Duplicated tree records will be filter out.



(a)



(b)

Figure 7.(a) Front view and back view in equirectangular image. (b) window filter with offset setting (X) is applied to reduce duplicated tree detection.

3.6. Spatial Clustering for Removing Duplicated Trees

Spatial clustering is applied to further reduce the duplicated trees with a defined distance called epsilon in which DBSCAN (Wikipedia. 2019) algorithm would filter spatial data set based on two parameters: a physical distance from each point, and a minimum cluster size. It is assumed that the trees are planted in interval of at least 4 meters so that epsilon is set to 4.

4. EXPERIMENT AND RESULTS

We studied the performance of the state-of-the-art detector YOLOv3 (Redmon et al., 2018) with multiple perspective images generated from GSV equirectangular images. The result will be compared to ground truth data from onsite survey to determine the accuracy of the detected tree both in detection rate and position.

4.1 Data Set

26 Road sections in Hong Kong Island are selected (Gloucester Road, Hennessy Road, Man Cheung Street, Lockhart Road, Johnston Road, Lung Wo Road, Upper Albert Road, Lower Albert Road, Garden Road, Kennedy Road, Macdonnell Road, Hornsey Road, Robinson Road, Conduit Road, Lyttelton Road, Lee Nam Road, Shek Pai Wan Road, Cyberport Road, Victoria Road, Pok Fu Lam Road, Shing Sai Road, Bonham Road, Hospital Road, Po Shan Road, Stubbs Road, Mount Butler Drive). Totally, there are 6618 GSV

equirectangular images downloaded along the roads for the experiment.

4.2 Results

For CNNs training process, it involves splitting the data into two sets randomly.

Training Dataset: Depending on the amount of data, 1500 and 2500 images are randomly selected for two sets of training. We have to input examples of desired trees which only planted in local environment. Annotation with tree bounding box on training data has to be completed before. We used an opensource labelling tool (*https://github.com/tzutalin/labelImg*) to complete the task.

Test Dataset: This is the data set on which the trained model is tested. No image should be part of the both the training and the test set. 200 and 300 images are selected for two sets of testing.

```
1          2          3          4          5          6
1: 8790.451172, 8790.451172 avg, 0.000000 rate, 6.481886 seconds, 64 images
2: 8790.721680, 8790.478516 avg, 0.000000 rate, 6.413727 seconds, 128 images
3: 8809.923828, 8792.422852 avg, 0.000000 rate, 6.710170 seconds, 192 images
4: 8788.920898, 8792.072266 avg, 0.000000 rate, 6.519844 seconds, 256 images
5: 8820.444336, 8794.909180 avg, 0.000000 rate, 6.427377 seconds, 320 images
```

Figure 8. Training log output from YOLOv3 network with pretrained model

During the training process as shown in Fig.8, it returns the values to reflect the status and result of each training step. The values comprise batch number (1), loss in the current batch (2), average loss till the current batch (3), current learning rate (4), time taken for the batch (5) and images used till current batch (6) from YOLOv3 network training. Updating the weights of the CNNs is processing iteratively which is based on how many mistakes it is making on the training dataset.

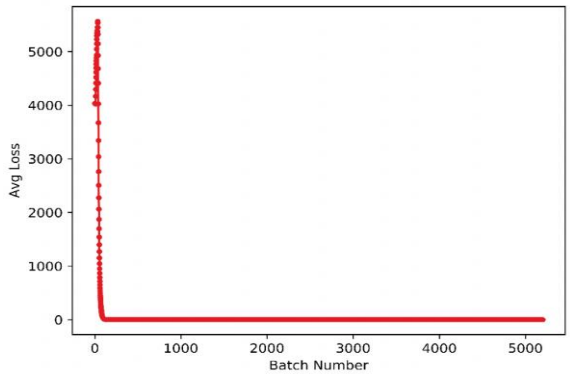


Figure 9. The training plots. Average loss vs batch number

During the training process, the log file is generated which contains the average loss in each batch. When it is reached a certain threshold value (as small as possible), the training processing can be stop for checking the accuracy. Figure 9 shows the loss plotted against the batch number for the tree detector. After 12 hours of training, the average loss reaches the threshold value that is almost reaching zero.

4.3 Tree Detection and Localization

Tree detection is started by the process to feed the perspective images to YOLOv3 model. Once a tree is detected, it will be indicated by a bounding box and also return the accuracy of detection. We only select the detected tree which has accuracy higher than 0.9. For the selected 26 roads, it consists of 6618 equirectangular GSV images captured totally. Each equirectangular image is splitted into 6 perspective images. Totally 39708 images are fed to the YOLOv3 network for tree detection. When the CNNs detects a tree in an image. It will provide a bounding box and shows the accuracy rate. Referring to Figure 10, the bounding box cannot represent tree position. To make it feasible to calculate the tree position, we assume the bottom middle of the bounding box as the position of the tree in which the tree base is regarded as lowest position of the tree stem.

The position of the detected trees in perspective image space is projected back to the equirectangular image space. The final image is illustrated in Figure 11. As the centre coordinates for each equirectangular image is known, geographic coordinate of detected trees can be calculated through the GSV depth map.

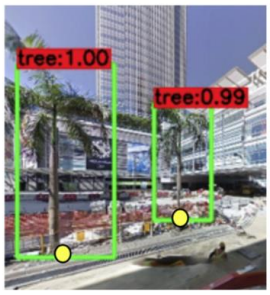


Figure 10. Yellow dots represent the position of the detected trees in perspective image

Figure 11. Green dot represents position of the tree in equirectangular image

4.4 Accuracy

4.4.1 Accuracy in Object Detection

Figure 12 shows the mistaken cases from CNNs model in which it detected tree-like objects but not really existent. Common mistake is found for lamp posts, water diversion channel and building gap. To minimize this kind of mistake (False detection), careful annotation for training object selection should be considered.

Figure 12. False tree detection

Another type of mistakes is misdetection. Figure 13 shows that the CNNs model cannot distinguish trees with great variety of appearances. To reduce this kind of mistake, increasing the number training data set with more variety of tree appearances followed by submitting to CNNs model for re-training would

improve the accuracy. Table 1 list the summary of the results. For the selected 26 roads, they consist of 6618 equirectangular images. Each equirectangular image is splitted into 6 perspective images. Totally 39708 images are fed to the YOLOv3 network for tree detection. Two set of training and testing images are created for the processing. For smaller training set (a), there are 10076 correctly detected trees with accuracy higher than 0.9. But cases of wrong detections (1035) and fail detection (1801) occurs. Overall, the detection rate is 0.78. For larger training set (b), the overall detection rate is 0.81.

Figure 13. Miss-detection highlighted in yellow circle

	(a) Train image: 1500 Test image: 200	(b) Train images: 2500 Test images: 300
Number of roads	26	26
Number of 360 GSV images	6618	6618
Number of correct detections	10076	11026
Number of wrong detections	1035	1005
Number of fail detection	1801	1505
Accuracy rate	0.78	0.81

Table 1. Summary of YOLOv3 network training and detection with different training set

4.4.2 Accuracy in Tree Geographic Position

To compare the difference in tree's geographic positions between estimation and real scene, two roads have been selected for onsite checking to determine the ground truth data: they are Hospital Road and Johnston Road featuring following properties.

1. Hospital Road: this road is extended along the hilly and mountainous terrain with steep slopes.
2. Johnston Road: this road is extended over relatively flat terrain.

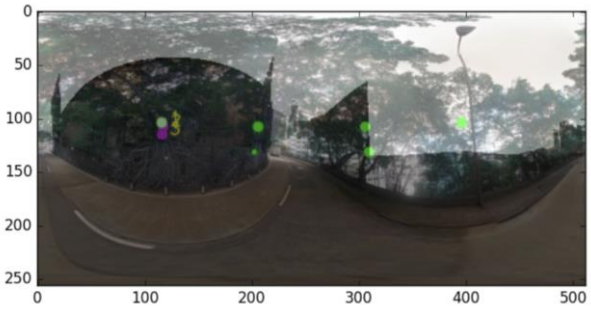


Figure 14. Result for a processed equirectangular image with detected points. This image is generated by fusing RGB with Depth images. The detected points (green and purple dots) are marked inside the image.

Figure 14 shows the result of a processed panorama image. The detected points (green and purple dots) are the detected tree position. There are multiple detected points for same tree position. After applying spatial clustering, the multiple tree points will be grouped into a single object. Figure 15 shows how to calculate the mean planimetric deviation. It is simply to calculate all the distance between detected trees and ground truth data, finally divided by total number of detected trees. The smaller the value, the optimal value for selection.

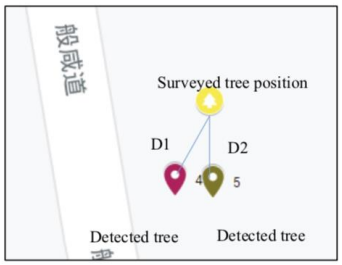


Figure 15. Calculation of mean planimetric deviation w.r.t. ground truth data. Detected trees labeled 4 and 5 belong to same surveyed tree

	Hospital Road		Johnston Road	
Number of trees recorded in onsite survey (Ground Truth)	28		11	
Average distance (m) of clustered trees w.r.t. ground truth data with offset X (field of view)				
	Difference	Average distance (m)	Difference	Average distance (m)
X = 100px	4	2.1	5	2.9
X = 200px	2	2.1	6	2.9
X = 300px	1	2.2	2	2.8
X = 400px	-1	2.4	2	2.9
X = 500px	-9	2.2	1	2.5
X = 600px	-11	2.3	-2	2.6

Table 1. Summary of results for number of clustered trees in different offset value X for the "field of view" filter.

Table 2 lists the result for different offset value X (referring to Figure 7) for the "field of view" filter. "Difference" is estimated tree numbers minus ground truth. The less the different value, the more the accuracy in estimated tree number. Average distance is the mean planimetric deviation w.r.t. ground truth data. The ground truth data for the surveyed tree position is recorded through mobile phone GPS receiver. The less the value in average distance, the more the accuracy in estimated tree geographic position. The average distance is within 3 meters slightly larger than the value obtained by GPS receiver with accuracy of up to 2 meters (Krylov et al., 2018).

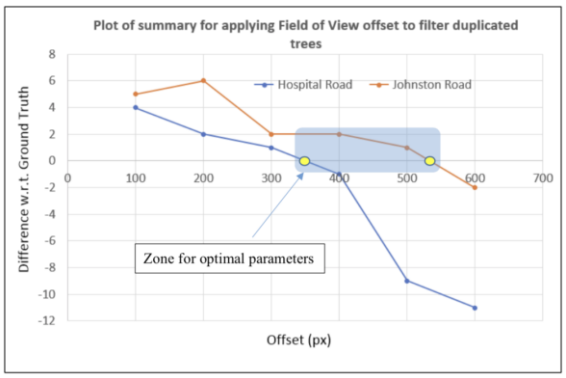


Figure 16. Plot of summary for Table 2.

Figure 16 plots the results for Table 2. It aims to find out the optimal offset value X. The yellow circle, which intersects with x and y axis, reflects the offset value when estimated tree number and ground truth are same and produce zero "difference". It is found that there is no universal offset value for the 2 selected roads. For Hospital road, the best offset value is 350px while 530px for Johnston Road. The average best offset value is 440px.

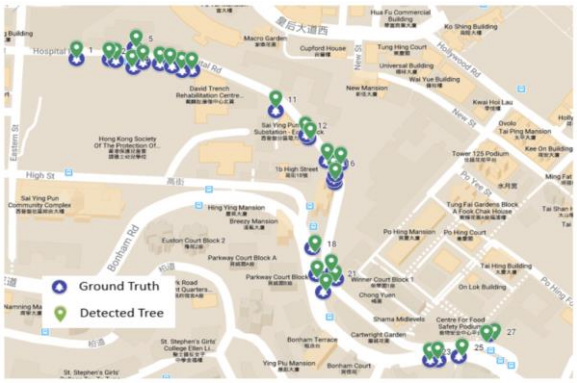


Figure 17. Result for Hospital Road in map view.

Figure 18. Result for Johnston Road in map view.

Figure 17 and Figure 18 display the result in map for Hospital Road and Johnston Road respectively. The offset value for field of view is set to 440px, epsilon for spatial clustering is 4 meters and mean planimetric deviation is within 3 meters.

4.7 3D Map Visualization

Figure 19. The distribution of detected trees along the Hospital road in 3D Map viewer

Fig.19 demonstrates the estimated tree distribution in the 3D Map prototype which is based on an opensource Cesium library (Cesiumjs.org. 2018). In the prototype, it is possible to visualize trees with surrounding facilities such as roads, buildings, facilities, premises and infrastructures, etc. The rendered 3D environment provides enhanced Web based 3D Map system for tree management.

5. LIMITATION

The main challenge in this work remains to increase the accuracy for estimation of geographic position of detected trees. Although "field of view" filter and spatial clustering were applied to remove duplicated trees inside Google Street-view panorama images, the effectiveness of the filter depends on the distribution of trees. We selected 2 roads (Hospital Road and Johnston Road) for comparison with ground truth data. The average offset value is 440px. As the scenario for tree distribution for each road is different, variation will still occur with respect to ground truth. If there are more roads selected with ground truth data, the average offset value should more reflect the optimal value. Spatial clustering algorithm DBSCAN is applied to reduce the duplicated trees within 4 meters. The result is further evaluated by defining a mean distance which is a statistical value for the separation between the model-estimated value with its on-site measurement. Currently, we did not make formal comparison with the results with other research and compare the results with other kinds of spatial clustering algorithm such as K-means and agglomerative hierarchical clustering. It is worth it for further study to provide results with better justification. Moreover, this study is only focussed on tree detection along roadside, which does not provide information for tree species. Tree species detection involves large amount image data for tree species planted along roadside but currently it seems that such data does not exist in Hong Kong.

6. CONCLUSIONS

360° Google street-view imagery is free and can be open accessed through Internet. By applying deep learning with CNNs using YOLOv3 network model an efficient method for automatic tree detection and accurate geographic localization is proposed. This work has performed the experiment by processing the 26 roads in Hong Kong Island to validate the approach it achieved mean distance of 3 meters with respect to ground truth data for 2 selected roads. Once the complete dataset for GSV images is downloaded, the pipeline is easy to be applicable to whole Hong Kong territory. With the development of the 3D GIS porotype system embedding automatic tree detection function based on deep learning, it would complement the Hong Kong tree register database by reducing the traditional work of manual inspection. Once the tree database is completed, on top of the inventory of the detected trees, it will be extended to tree species classification and health monitoring by further developing and applying advanced machine learning approaches.

ACKNOWLEDGEMENTS

The work described in this paper was partially supported by a grant from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. PolyU 25211819)

REFERENCES

Cesiumjs.org. (2018). CesiumJS - Geospatial 3D Mapping and Virtual Globe Platform. Retrieved from https://cesiumjs.org/. Accessed on 16 May 2019

Deng, L., Lin, Y., Yan, L., Tesfamichael, S., Billen, R., Yao, Y., Yao, W., Chen, X., Fang, X., Wang, C., and Jing, X. (2019). Urban plant phenology monitoring: Expanding the functions of widespread surveillance cameras to nature rhythm understanding. Remote Sensing Applications: Society and Environment, 15, 100232.

Google map. (2019). Google map street view for developer. Retrieved from https://developers.google.com/maps/documentation/javascript/streetview. Accessed on 17 June 2019

Hong Kong Tree Register. (2016). Retrieved from https://www.greening.gov.hk/treeregister/map/treeIndex.aspx. Accessed on 14 June 2019.

Kaartinen, H., Hyypp'a, J., Yu, X., Vastaranta, M., Hyypp'a, H., Kukko, A., Holopainen, M., Heipke, C., Hirschmugl, M., Morsdorf, F., et al. (2012). An international comparison of individual tree detection and extraction using airborne laser scanning. Remote Sensing 4(4), 950-974.

Krylov, V. A., Kenny, E., & Dahyot, R. (2018). Automatic discovery and geotagging of objects from street view imagery. Remote Sensing, 10(5), 661.

Lahivaara, T., Seppanen, A., Kaipio, J. P., Vauhkonen, J., Korhonen, L., Tokola, T., and Maltamo, M. (2014). Bayesian approach to tree detection based on airborne laser scanning data. IEEE Transactions on Geoscience and Remote Sensing, 52(5), 2690-2699.

Lafarge, F. and Mallet, C. (2012). Creating large-scale city models from 3d-point clouds: a robust approach with hybrid representation. International Journal of Computer Vision, 99(1), 69-85.

Lands Department. (2017). iB1000 Digital Topographic Map. Retrieved from http://www.hkmapmeta.gov.hk/mcs/home/web/data/lands/iB1000.faq.html. Accessed on 21 May 2019

Larsen, M., Eriksson, M., Descombes, X., Perrin, G., Brandtberg, T., and Gougeon, F. A. (2011). Comparison of six individual tree crown detection algorithms evaluated under varying forest conditions. International Journal of Remote Sensing, 32(20), 5827-5852.

Paris, C. and Bruzzone, L. (2014). A three-dimensional model-based approach to the estimation of the tree top height by fusing low-density LiDAR data and very high resolution optical images. IEEE Transactions on Geoscience and Remote Sensing, 53(1), 467-480.

Polewski, P., Yao, W., Heurich, M., Krzystek, P., and Stilla, U. (2016). Combining active and semisupervised learning of remote sensing data within a Renyi entropy regularization framework. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 9(7), 2910-2922.

Redmon, J. and Farhadi, A. (2018). "YOLOv3: An Incremental Improvement" in CVPR.

Ren, S., He, K., Girshick, R., and Sun, J. (2015). Faster r-cnn: Towards real-time object detection with region proposal networks. In Advances in neural information processing systems, pages 91–99.

Torii, A., Havlena, M. and Pajdla, T. (2009). From Google Street View to 3D city models. In 2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops (pp. 2188-2195). IEEE.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C., and Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV).

Wikipedia (2019). Field of view. Retrieved from https://en.wikipedia.org/wiki/Field_of_view. Accessed on 17 June 2019

Wegner, J. D., Branson, S., Hall, D., Schindler, K. and Perona, P. (2016). Cataloging public objects using aerial and street-level images-urban trees. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 6014-6023.

Wikipedia. (2019). Density-based spatial clustering of applications with noise (DBSCAN). Retrieved from https://en.wikipedia.org/wiki/DBSCAN. Accessed on 14 November 2019.

Wu, J., Yao, W., and Polewski, P. (2018). Mapping individual tree species and vitality along urban road corridors with LiDAR and imaging sensors: Point density versus view perspective. Remote Sensing, 10(9), 1403.

Yang, L., Wu, X., Praun, E., and Ma, X. (2009). Tree detection from aerial imagery. In ACM GIS'09.

Yao, W., Krzystek, P., and Heurich, M. (2012). Identifying standing dead trees in forest areas based on 3d single tree detection from full waveform lidar data. ISPRS Annals of the Protogrammetry, Remote Sensing and Spatial Information Sciences 1, 7.

Yang, W., Qian, Y. L., Cricri, F., Fan, L. X. and Kamarainen, J. K. (2018). Object Detection in Equirectangular Panorama. In 2018 24th International Conference on Pattern Recognition (ICPR) (pp. 2190-2195). IEEE.