Synthetic Generation of RGB-based Tree Instance Segmentation Dataset using 3D Rendered Scenes

Tae Kyung Kim[1], Viduranga[1], Minseok Gang[1], Joonseok Lee[1] *
[1]Seoul National University, Republic of Korea
{bmkim88, vidzrox, msg1907}@snu.ac.kr, joonseok2010@gmail.com

1. Abstract

Despite its significant demand in the industries, tree instance segmentation has not been studied extensively owing to the complexity of tree textures and occlusions one another. In this study, we proposed a synthetic dataset generation framework that creates high-quality segmentation dataset for tree instances. We utilized 3D tree generation softwares and rendered the tree models into fine-resolution images with precise annotation maps. We tested and compared two different approaches: pasting by depth information, and using pre-defined anchor and the other objects' annotation maps. Depth values were not plausible due to the large noises and discontinuity. Instead, Our second approach could produce realistic street scenes with proper augmentation of tree instances. The bottlenecks were mostly originated from unrealistic tree instance images and anchors, and dichotomous occlusion flags ('Front' or 'Behind'). When we tested the instance segmentation accuracy, the model trained with our dataset showed better performance than the plain dataset, and model segmented precise boundaries of tree instances.

2. Introduction

For better understanding of visual scenes, the main interest of computer vision has been progressing from object classification to more sophisticated tasks, including instance localization and segmentation. Of great significance and demand, large scale annotation datasets in various domains have been established and released. Each dataset tackles a variety of classes notwithstanding, they are biased to a narrow range of objects with rigid boundaries (e.g., people, vehicles, traffic signs, etc.), namely 'things' [14]. Conversely, amorphous objects with indistinct nature, such as sky, grass, and trees, are considered as 'stuffs' and segmented as a whole or excluded without proper annotation (Figure 1) [3, 18]).

Among these natural instances, trees show growing demand to monitor, protect, while properly utilizing the resources they provide. In the field of forestry, there has been several attempts to detect and segment the individual tree, using terrestrial or airborne LiDAR [9, 29], RGB-Depth sensors [30], and high resolution airborne RGB imagery [12]. However, these sensors are costly, and lacks in potential to build a large-scale dataset which could be shared and utilized for tackling more generalized tasks.

Another potential bottleneck for establishing tree segmentation dataset is indeterminate boundary of each tree, which is exacerbate by occlusion from other tree instances. Indeed, instance-level manual annotation of trees can be costly, time-consuming, but most importantly, not precise. These limitations might have led previous large-scale datasets containing instance annotation of trees become so rare. Therefore one of the reasonable approach to build such data could be using synthetic data. Synthetic data generation workflows that creates samples in different viewpoints, illuminations, and different backgrounds would be more effective, if done properly.

In this study, we aim to propose an automated framework that could produce large scale tree segmentation data by augmenting pre-existing segmentation datasets of street scenes. More specifically, our framework will include: synthetic generation of tree instances, rendering RGBA and annotation maps, pasting tree images with occlusion, and discrimination between real and synthetic trees in the scenes. In this report, tree instance generation, rendering, augmentation, and preliminary results on pasting methods are included.

We utilize Blender, a 3D modelling software, and its tree generation add-ons to create and render realistic figures of trees. Regarding placement of the rendered trees, we test two distinct approaches: paste-by-depth which considers depth information to simulate occlusions in real-world scenes; and paste-by-anchors that pastes trees into manually drawn anchors in the scene while simulating occlusion using object annotation maps.

Figure 1. **Sample annotation data from COCO (left) and Mapillary Vistas datasets (right).** Unlike other objects, trees are segmented as a whole, not by individual instances. Image adopted and modified from [14, 18].

3. Related Work

3D modelling based synthetic datasets. The labourous and costly nature of manual annotation workflows is widely known, and recently there have been rapidly increasing trend to utilize synthetic data within the computer vision community [23]. Typical approaches to the synthetic data generation problem could be listed as two distinct methods: using 3-dimensional objects and rendered scenes, and 2-dimensional cut-and-paste approach. Method based on 3D modelling were mainly utilized in several object detection studies that tackle variety of tasks in [21, 27, 16]. Regarding segmentation problems, several studies have also taken advantage of synthetic approaches, as virtual indoor scenes [10, 20], and outdoor scenes [23].

Cut and Paste. As mention above, 3D rendering is widely used to generate large synthetic datasets[17, 24, 26]. Even though this method is capable of generating large datasets, creating 3D models is not a simple task. Rendering a 3D model from a real-world object requires skillful 3D model designer, and also is time-consuming [38]. Another drawback of 3D rendering methods is that it suffers from the domain gap problem. Several approaches were proposed to overcome the domain gap issues [31, 22], but the most cost-effective manner is using the 2D cut-and-paste technique to generate synthetic datasets. This approach has been applied in indoor and outdoor synthetic data generation scenarios in numerous studies [5, 7, 38, 40, 33].

Tree segmentation with image sensors. Over the years, the use of aerial image data to detect, classify and separate individual trees has increased, with the use of Unmanned Aerial Vehicles in forestry applications[28, 8]. Segmentation algorithms using three-dimensional (3D) information provided by light detection and ranging (LiDAR) technology have been developed [32, 34, 35]. Research with high-resolution RGB images and RGB-D images using stereo cameras have also been proposed[39, 6, 15, 19, 25]. Some researchers have applied terrestrial image sensors to tree segmentation [37, 13]. Methods using these imagery are becoming cheaper and their application range is growing in the forestry field. It still takes a lot of temporal and human resources to build large scale data sets.

4. Proposed Method

Synthetic Tree Generation. To obtain sufficient tree instances with large variation yet realistic figures, tree generation modules in 3D modelling software were utilized. For the base modelling software to generate, modify, and render tree instances, Blender 2.82 was adopted. Then, off-the-shelf tree generation softwares and Blender add-ons were utilized. We have listed plausible options for our task in Table 1. As described in the table, tree generation tools can be typified into two categories: parameterized generation, and library based generation. Among the variety of tree generation tools, 'the Grove 10' is a Blender-based parameterized tree generation add-on, and provides powerful two-staged growth simulation: branch simulation, and twigs allocation. Branch simulation supports the growth procedure with various parameters including the chance of branch split, number of nodes, lengths, and angles of each branch. It was followed by assigning twigs and leaf textures to the branches with pre-defined chances and sizes. Despite the flexibility and robustness of producible trees, the add-on required purchase of additional twigs and texture models. Thus, in this study, library-based generation add-ons were mainly utilized.

Botaniq and Real Trees are tree model libraries, each containing more than 80 and 20 tree species, respectively. They include various types of trees including coniferous and deciduous species, along with seasonal variations that exhibit leaf coloring and leaf fall. As the street scenes in the target dataset mostly contained deciduous trees with large crowns, we sampled the most suitable tree instances among the tree libraries, resulting in 36 unique tree instances. The process of spawning and rendering each tree model was automated by a python script applied to the Blender engine. Rendering was followed by a manual cropping process that removed any redundant spaces around the trees.

Pre-rendering Augmentation and Rendering. Generated tree instances were augmented before rendering, using the following methods: illumination adjustment and camera rotation. Unlike conventional image augmentation approaches, illumination adjustment directly alters the position, color, and magnitudes of light sources and thus produces totally different image samples from the original one.

Name	Platform	Classification	Free
Botaniq	Blender	Library	N
Cinema 4d	Windows	Procedural	N
Modular Tree	Blender	Procedural	Y
PlantFactory	Windows	Procedural	N
Real Trees	Blender	Library	N
Sapling Tree Gen	Blender	Procedural	Y
SpeedTree	Windows	Procedural	N
The Grove 10	Blender	Procedural	N

Table 1. **List of 3D tree generation softwares.** Among the variety of generation softwares, Botaniq, Real Trees, and The Grove 10 were considered as the most suitable option for our task. Due to the lack of resources, we only tested library based add-ons in this project.

Considering that the figures of tree vary significantly depending on the angle of view, camera rotation was applied to produce image samples that cover 360° degrees. 3 different light source conditions and 6 different rotation angles, with 60° degrees interval, were applied. Then, final rendering was performed by adopting and modifying python modules developed by [36], with the following settings: RGB image, and two annotation rendering of instance and depth.

Post-rendering Augmentation. Using the rendered images of single tree instance, data augmentation of rendered output was performed. First, object size resizing was performed. The background-subtracted object size was calculated by using the contour information of the image. OpenCV was utilized to find the contour of binary image of each tree instance and resize each instance. Resizing was followed by object merge which generates occluded trees from the tree instances. We could merge several objects to create more complex synthetic images (Figure 2). These techniques have been used in [5, 7] to generate synthetic training datasets. We used another OpenCV program to combine two or more instances in a background-subtracted image. CityScape test dataset were utilized as background images and were randomly pasted object-merged images as the final stage.

Cityscapes Dataset. After numerous trial-and-errors, our finalized approach to create synthetic dataset was to apply augmentation of tree instances to the pre-existing street scene datasets with instance annotations. Among the variety of datasets, Cityscapes were adopted [4]. The dataset contained precise annotation maps of a sufficient number of classes (30 classes; including people, vehicles, construction objects, and also vegetation but annotated as a whole). However, it was challenging to cope with the scenes that already contained a large proportion of vegetation (Figure 3), and required much labor to delineate and annotate each

Figure 2. **Resized and merged objects.** Each tree instance was resized using contour informations and merged to reproduce occlusion effect.

tree instance. In this project, we filtered out scenes that contained only a small proportion of vegetation and augmented the scenes with our rendered tree instances. Only images that contained vegetation less than 2% were filtered, which gave 394 images in total.

Figure 3. **Sample scenes that contained large vegetation.** Tree instances were highly occluded one another, resulting in complex textures that were challenging to delineate.

Paste by Depth. In order to synthesize the occluded tree image using the cut and paste method, the depth value for each object in the background images was required. Some public datasets such as CityScape [4] and UASOL [1] provide both depth and RGB data. We separated the image by removing pixels with a threshold value in the depth map. The previously created tree object images were pasted to the separated image, and the removed part was recombined to generate occlusion at a specific depth level as shown in the Figures 4 and 9. Dense depth maps without missing values are required for the image separation process. The dense maps could be approximated using annotated objects [11]. The depth level and position of the tree object was determined randomly, and the tree was resized again using the background object according to the depth. The synthesized result will be labeled whether it is natural or not, and learned through a CNN model in our future works.

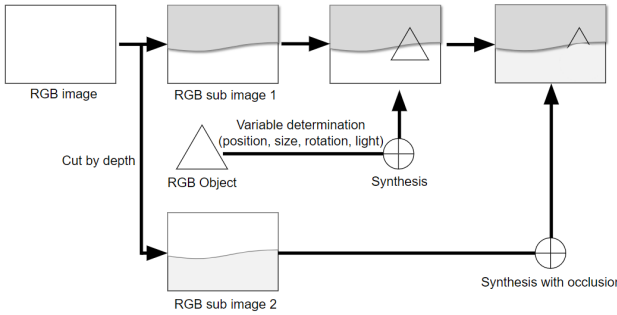

Figure 4. **Flowchart of paste by depth approach.** Using the base RGB and depth image, we first thresholded the images by depth and separated into different *layers*. The occlusion was simulated by overlaying the layer with smaller depth onto the synthetic tree instance.

Paste by Anchors. We proposed another pasting method that we called 'Paste by Anchors', a three-staged image synthesis procedure that includes (1) manual annotation of anchors, (2) placement and resizing tree instances to match each corresponding anchor, and (3) cut-out regions that are overlapped with pre-existing objects in the scenes. An openCV-based annotation tool was implemented that can easily draw the anchors in the scene and save them as json objects (Figure 5. As shown in the Figure 5, anchors for each tree were manually drawn, and marked as 'Front' or 'Behind' which tells whether the tree should be masked with the occluding objects or not. Then, tree instances were randomly sampled from the 36 unique tree images (*Proposed Method, Synthetic Tree Generation*), and were placed and resized to match the location and the size of each corresponding anchor. It was followed by cut-out process, which masked out the regions in trees that occlude with the other objects (e.g., people, vehicles, poles, traffic signs). This cut-out process was performed only when the anchor was marked as 'Behind'.

Figure 5. **Demonstration of anchor annotation process.** Each anchor was defined as 5 points: two points that consists vertical range of tree (blue), another two points of horizontal range (red), and the last point that indicates the lower end of the crown.

Instance Segmentation Model. To generate the tree segmentation model, we used a state-of-the-art real-time instance object segmentation model called YOLACT(You Only Look at Coefficients) [2]. YOLACT breaks instance segmentation into two parallel tasks, (1) generating a set of prototype masks and (2) predicting pre-instance mask coefficients. And then produce full-image instance masks by linearly combining the prototypes with the corresponding mask coefficients. They claim that the YOLOCT can reach 30fps even when using the ResNet-101 backbone and generate high-quality masks compared to the FCIS and Mask R-CNN approaches. We trained all models with the batch size of 32 on a single RTX 3090 GPU using ResNet-50 as backbone and ImageNet pre-trained weights. The model consists of the original CityScape dataset class list, which contains 35 categories. In the CityScape dataset, trees are labeled as vegetation class. We trained with SGD for 8,000 iterations at a learning rate of 103 and a momentum of 0.9.

5. Results and Discussion

Synthetic Tree Generation. Tree instances were generated with large variety of species, branch and crown shapes, and illuminations (Figure 6). Despite the parameterized tree generation module, The Grove 10, could produce new tree instances consistently, the available types of twigs and leaves were limited to a small number, resulting in tree library modules providing the better output. After rendering with built-in render engines in Blender (e.g., Cycles, and Eevee), highly realistic tree instances with corresponding annotation map could be produced (Figure 7). As shown in the figure, high-resolution RGB and depth images were generated which could be pasted directly into any backgrounds.

Figure 6. **Tree instances generated from the add-ons.** Tree instances with variety of species, shapes, and color were generated.

Figure 7. **Render output with different settings.** Images on the left, middle, and right represents the rendered results of instance annotation, RGB, and depth map, respectively. Instance annotation map was rendered as a very dark images as the pixel values were assigned as one.

Paste by Depth. We tested our method of pasting rendered tree objects into existing datasets, especially with depth information. As shown in Figure 9, pasting by depth generated occluded tree instances with the objects in the scenes. We also considered and calibrated the size of tree instance image using the depth information. Here, we found several limitations in our current approach. First, discontinuous quality of occlusion was shown owing to the sparse nature of depth map. As the source depth map was not continuous (Figure 9), the cut-out regions were also not even, resulting in large noises in the occlusion. Hence, the other variables for pasting the objects should be considered, including position, color space, rotation, and light condition. As shown in the figure, the color and brightness of pasted tree instance did not matched with other instances in the same scene.

Paste by Anchors. As mentioned in previous section, *Proposed Methods, Paste to Anchors*, we created our tree segmentation dataset by pasting tree instances to Cityscapes dataset and generated fine-resolution annotation maps including trees. Figure 8 shows the output scenes using our synthetic generation framework. As shown in the figure, our proposed method successfully created realistic scenes with ready-to-use annotation labels with precise boundary. Hence, utilizing the annotation polygons of pre-existing objects (e.g., vehicles, people, traffic signs) gave more reasonable occlusion effects to the tree instances. However, as shown in Figure 10 that shows some unrealistic output results, it was mostly originated from (1) unrealistic tree instance images (color, shadows, and textures), (2) faulty anchors with excessively small sizes, (3) dichotomous occlusion effects (could not placed in the middle of two objects). These limitations can be improved by utilizing procedural tree generation softwares, assigning more realistic anchors, and adding some occlusion rules that reflects more realistic relationships between the occluding objects.

Instance Segmentation Performance. Due to the limitation of time, we could not test and compare different models. We trained our dataset with 8,000 iteration and compare results with original cityscape dataset vegetation class mean average precision(mAP). Our dataset got 39.74% and 38.06% mAP for mask and bounding box respectively. Hence the cityscape dataset only achieved 20.81% and 27.23% mAP for mask and bounding box respectively. Inference results of the YOLACT model is shown in Figure 11. In the figure, the model could delineate boundaries of each tree with high precision, including the trees that were not synthesized but already existed in the original scene. However, it seemed to show more errors especially when delineating the occluded trees, and the vegetation in the far background (which is actually forests or mountains rather than 'trees'). Hence, there are also possibility that the model might have overfit to the exact shape of 36 tree model images. In future works, we will create the dataset with having much variety of tree instances, and test the robustness in real-world conditions.

6. Conclusion

In this study, we have generated 3D tree models and rendered them into fine-resolution images with precise annotation maps. Regarding the placement of the trees, depth

Figure 8. **Synthecized Cityscapes scenes.** Original Cityscapes scenes that contained vegetation less than 2% were augmented using the rendered tree instances. Each tree instance were resized to match the size of pre-defined anchors (*Proposed Method, Paste by Anchors*), which was followed by brightness adjustment and blurring with bilateral filter. Each six images on the top, bottom left, and bottom right represents the synthesized scenes, original scenes, and generated annotation maps, respectively.

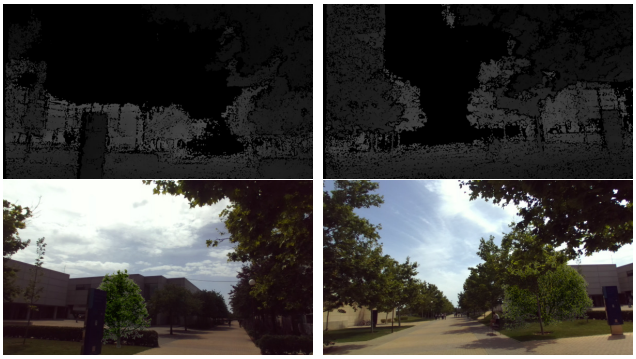

Figure 9. **Sample outputs produced by paste-by-depth.** Images above represent the depth map of each scenes, and images below show the occluded output generated by paste-by-depth method. Sparsely recorded data of source depth maps resulted in noisy occlusion in the final output. High-resolution figure is hosted on figshare, doi.org/10.6084/m9.figshare.14562684.v1

Figure 10. **Samples of unrealistically synthesized scenes.** It was mainly due to mismatching textures and colors, small anchors, and unrealistic occlusion owing to the dichotomous occlusion flags.

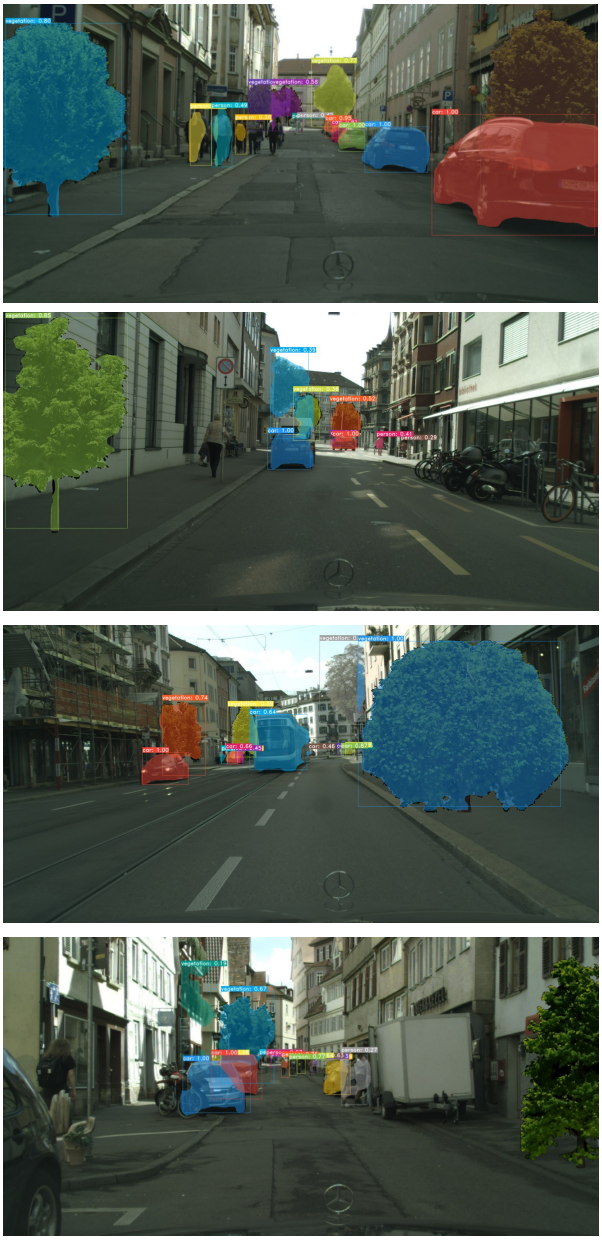

Figure 11. **Final inference results.** YOLACT was utilized, and was trained with 8,000 iterations using our dataset.

values were not suitable for our task owing to the excessive noises and discontinuity. Instead, we proposed a three-staged approach, which uses pre-defined anchor location to paste and scale the tree, and then cuts out the occluding regions with pre-existing objects in the scene. Our proposed methods could produce realistic street scenes with proper augmentation of tree instances (Figure. 8). However, the current approach had apparent limitations as follows: limited to the scenes with less vegetation, disparity with the real trees in terms of color, texture (shadow, simplified crown

shapes), and complicated occlusion with different level of orders (Figure 10). The segmentation performance was improved using our dataset, and the trained model seemed to delineate exact boundaries of tree instances.

References

[1] Zuria Bauer, Francisco Gomez-Donoso, Edmanuel Cruz, Sergio Orts-Escolano, and Miguel Cazorla. Uasol, a large-scale high-resolution outdoor stereo dataset. *Scientific data*, 6(1):1–14, 2019. 4

[2] Daniel Bolya, Chong Zhou, Fanyi Xiao, and Yong Jae Lee. YOLACT: real-time instance segmentation. *CoRR*, abs/1904.02689, 2019. 4

[3] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Cocostuff: Thing and stuff classes in context. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1209–1218, 2018. 1

[4] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016. 3, 4

[5] Debidatta Dwibedi, Ishan Misra, and Martial Hebert. Cut, paste and learn: Surprisingly easy synthesis for instance detection. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 2, 3

[6] Matheus Pinheiro Ferreira, Danilo Roberti Alves de Almeida, Daniel de Almeida Papa, Juliano Baldez Silva Minervino, Hudson Franklin Pessoa Veras, Arthur Formighieri, Caio Alexandre Nascimento Santos, Marcio Aurélio Dantas Ferreira, Evandro Orfanó Figueiredo, and Evandro José Linhares Ferreira. Individual tree detection and species classification of amazonian palms using uav images and deep learning. *Forest Ecology and Management*, 475:118397, 2020. 2

[7] Georgios Georgakis, Arsalan Mousavian, Alexander C. Berg, and Jana Kosecka. Synthesizing training data for object detection in indoor scenes. *CoRR*, abs/1702.07836, 2017. 2, 3

[8] Tristan RH Goodbody, Nicholas C Coops, Peter L Marshall, Piotr Tompalski, and Patrick Crawford. Unmanned aerial systems for precision forest inventory purposes: A review and case study. *The Forestry Chronicle*, 93(1):71–81, 2017. 2

[9] Hamid Hamraz, Marco A. Contreras, and Jun Zhang. A robust approach for tree segmentation in deciduous forests using small-footprint airborne lidar data. *International Journal of Applied Earth Observation and Geoinformation*, 52:532–541, 2016. 1

[10] Ankur Handa, Viorica Patraucean, Vijay Badrinarayanan, Simon Stent, and Roberto Cipolla. Synthcam3d: Semantic understanding with synthetic indoor scenes. *arXiv preprint arXiv:1505.00171*, 2015. 2

[11] Maximilian Jaritz, Raoul De Charette, Emilie Wirbel, Xavier Perrotton, and Fawzi Nashashibi. Sparse and dense data with cnns: Depth completion and semantic segmentation. In *2018 International Conference on 3D Vision (3DV)*, pages 52–60. IEEE, 2018. 4

[12] Teja Kattenborn, Jana Eichel, and Fabian Ewald Fassnacht. Convolutional neural networks enable efficient, accurate and fine-grained segmentation of plant species and communities from high-resolution uav imagery. *Scientific reports*, 9(1):1–9, 2019. 1

[13] Qiujie Li, Pengcheng Yuan, Xu Liu, and Hongping Zhou. Street tree segmentation from mobile laser scanning data. *International Journal of Remote Sensing*, 41(18):7145–7162, 2020. 2

[14] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 1, 2

[15] Daliana Lobo Torres, Raul Queiroz Feitosa, Patrick Nigri Happ, Laura Elena Cué La Rosa, José Marcato Junior, José Martins, Patrik Olã Bressan, Wesley Nunes Gonçalves, and Veraldo Liesenberg. Applying fully convolutional architectures for semantic segmentation of a single tree species in urban environment on high resolution uav optical imagery. *Sensors*, 20(2):563, 2020. 2

[16] Javier Marin, David Vázquez, David Gerónimo, and Antonio M López. Learning appearance in virtual scenarios for pedestrian detection. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 137–144. IEEE, 2010. 2

[17] Yair Movshovitz-Attias, Takeo Kanade, and Yaser Sheikh. How useful is photo-realistic rendering for visual learning? In Gang Hua and Hervé Jégou, editors, *Computer Vision – ECCV 2016 Workshops*, pages 202–217, Cham, 2016. Springer International Publishing. 2

[18] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulò, and Peter Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *International Conference on Computer Vision (ICCV)*, 2017. 1, 2

[19] Masanori Onishi and Takeshi Ise. Explainable identification and mapping of trees using uav rgb image and deep learning. *Scientific reports*, 11(1):1–15, 2021. 2

[20] Jeremie Papon and Markus Schoeler. Semantic pose using deep networks trained on synthetic rgb-d. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 774–782, 2015. 2

[21] Xingchao Peng, Baochen Sun, Karim Ali, and Kate Saenko. Learning deep object detectors from 3d models. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1278–1286, 2015. 2

[22] Aayush Prakash, Shaad Boochoon, Mark Brophy, David Acuna, Eric Cameracci, Gavriel State, Omer Shapira, and Stan Birchfield. Structured domain randomization: Bridging the reality gap by context-aware synthetic data. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 7249–7255, 2019. 2

[23] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M Lopez. The synthia dataset: A large

collection of synthetic images for semantic segmentation of urban scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3234–3243, 2016. 2

[24] Artem Rozantsev, Vincent Lepetit, and Pascal Fua. On rendering synthetic images for training an object detector. *Computer Vision and Image Understanding*, 137:24–37, 2015. 2

[25] Anastasiia Safonova, Emilio Guirado, Yuriy Maglinets, Domingo Alcaraz-Segura, and Siham Tabik. Olive tree biovolume from uav multi-resolution image segmentation with mask r-cnn. *Sensors*, 21(5):1617, 2021. 2

[26] Hao Su, Charles R. Qi, Yangyan Li, and Leonidas J. Guibas. Render for cnn: Viewpoint estimation in images using cnns trained with rendered 3d model views. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015. 2

[27] Baochen Sun and Kate Saenko. From virtual to reality: Fast adaptation of virtual object detectors to real domains. In *BMVC*, volume 1, page 3, 2014. 2

[28] Lina Tang and Guofan Shao. Drone remote sensing for forestry research and practices. *Journal of Forestry Research*, 26(4):791–797, 2015. 2

[29] Shengli Tao, Fangfang Wu, Qinghua Guo, Yongcai Wang, Wenkai Li, Baolin Xue, Xueyang Hu, Peng Li, Di Tian, Chao Li, Hui Yao, Yumei Li, Guangcai Xu, and Jingyun Fang. Segmenting tree crowns from terrestrial and mobile lidar data by exploring ecological theories. *ISPRS Journal of Photogrammetry and Remote Sensing*, 110:66–76, 2015. 1

[30] S. Tejaswi Digumarti, L. M. Schmid, G. M. Rizzi, J. Nieto, R. Siegwart, P. Beardsley, and C. Cadena. An approach for semantic segmentation of tree-like vegetation. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 1801–1807, 2019. 1

[31] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 23–30, 2017. 2

[32] Luke Wallace, Arko Lucieer, and Christopher S Watson. Evaluating tree detection and segmentation routines on very high resolution uav lidar data. *IEEE Transactions on Geoscience and Remote Sensing*, 52(12):7619–7628, 2014. 2

[33] Hao Wang, Qilong Wang, Hongzhi Zhang, Jian Yang, and Wangmeng Zuo. Constrained online cut-paste for object detection. *IEEE Transactions on Circuits and Systems for Video Technology*, pages 1–1, 2020. 2

[34] Wanqian Yan, Haiyan Guan, Lin Cao, Yongtao Yu, Sha Gao, and JianYong Lu. An automated hierarchical approach for three-dimensional segmentation of single trees using uav lidar data. *Remote Sensing*, 10(12):1999, 2018. 2

[35] Wanqian Yan, Haiyan Guan, Lin Cao, Yongtao Yu, Cheng Li, and JianYong Lu. A self-adaptive mean shift tree-segmentation method using uav lidar data. *Remote Sensing*, 12(3):515. 2

[36] Lei Yang. bpycv: computer vision and deep learning utils for blender. https://github.com/DIYer22/bpycv, 2020. 3

[37] Ting Yun, Kang Jiang, Hu Hou, Feng An, Bangqian Chen, Anna Jiang, Weizheng Li, and Lianfeng Xue. Rubber tree crown segmentation and property retrieval using ground-based mobile lidar after natural disturbances. *Remote Sensing*, 11(8):903, 2019. 2

[38] Woo-Han Yun, Taewoo Kim, Jaeyeon Lee, Jaehong Kim, and Junmo Kim. Cut-and-paste dataset generation for balancing domain gaps in object instance detection. *IEEE Access*, 9:14319–14329, 2021. 2

[39] Dafeng Zhang, Jianli Liu, Wenjian Ni, Guoqing Sun, Zhiyu Zhang, Qinhuo Liu, and Qiang Wang. Estimation of forest leaf area index using height and canopy cover information extracted from unmanned aerial vehicle stereo imagery. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(2):471–481, 2019. 2

[40] Wenwei Zhang, Zhe Wang, and Chen Change Loy. Multi-modality cut and paste for 3d object detection. *CoRR*, abs/2012.12741, 2020. 2