Tree detection from aerial imagery

Lin Yang[*]
Computer and Information Sciences
University of Alabama at Birmingham
1300 University Blvd.
Birmingham, AL 35294
galabing@cis.uab.edu

Xiaqing Wu, Emil Praun, Xiaoxu Ma
Google, Inc.
1600 Amphitheatre Parkway
Mountain View, CA 94043
{wux,emilp,xiaoxuma}@google.com

ABSTRACT

We propose an automatic approach to tree detection from aerial imagery. First a pixel-level classifier is trained to assign a {tree, non-tree} label to each pixel in an aerial image. The pixel-level classification is then refined by a partitioning algorithm to a clean image segmentation of tree and non-tree regions. Based on the refined segmentation results, we adopt template matching followed by greedy selection to locate individual tree crowns.

As training a pixel-level classifier requires manual generation of ground-truth tree masks, we propose methods for automatic model and training data selection to minimize the manual work and scale the algorithm to the entire globe. We test the algorithm on thousands of production aerial images across different countries. We demonstrate high-quality tree detection results as well as good scalability of the proposed approach.

Categories and Subject Descriptors

I.4 [**Image Processing And Computer Vision**]: Enhancement

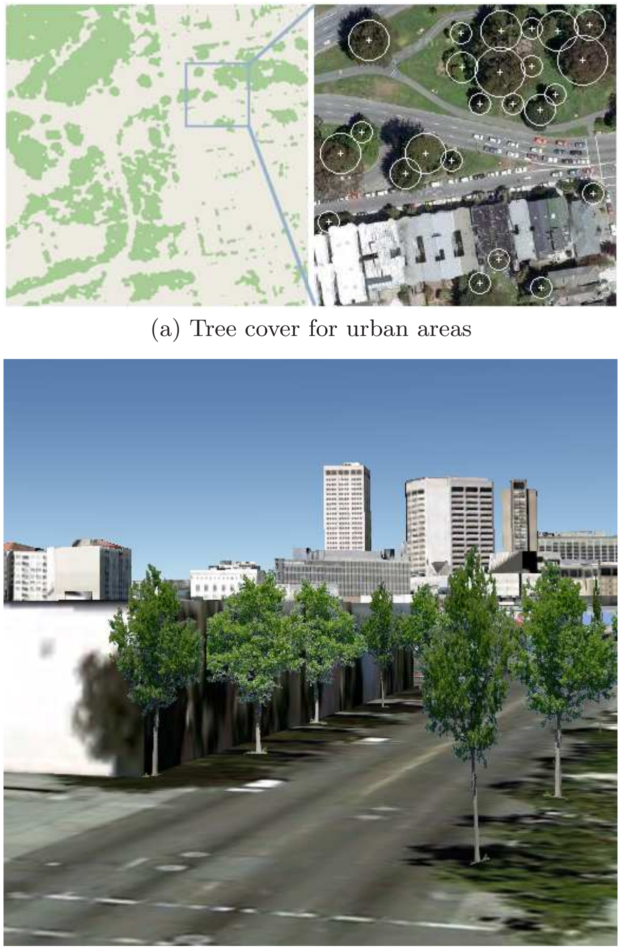
General Terms

Algorithms

1. INTRODUCTION

Aerial imagery captures a wide variety of natural and man-made objects on Earth. Detecting such objects is a classic topic of GIS [15].

We focus on tree detection from aerial imagery. Given an aerial image, the proposed approach can automatically segment the region of trees, locate each individual tree and provide an estimate of its crown size. The detection results can assist in various applications such as urban planning and 3D city modeling (see Figure 1).



(a) Tree cover for urban areas

(b) Augmented 3D city modeling

Figure 1: Two sample applications for tree detection from aerial imagery. (a) (left) shows the tree cover for the north-west region of San Francisco, which is based on the results of tree segmentation discussed in Section 2. (a) (right) shows a zoom-in view of the detected tree crowns. The study of tree cover has a potential impact on urban planning [2]. (b) shows the 3D city modeling for San Francisco. The view is augmented by tree models at detected tree locations from aerial images to enhance the user experience.

[*] This work was done at Google, Inc.

ACM GIS ’09 , November 4-6, 2009. Seattle, WA, USA (c) 2009 ACM
ISBN 978-1-60558-649-6/09/11...$10.00.

1.1 Related work

The current state of the art can be roughly divided into three classes depending on their input data: LiDAR-based [5, 6], NIR-based [10, 17], and image-based [1, 15] methods.

Light Detection and Ranging (LiDAR) and Near Infrared (NIR) are remote sensing technologies which provide geometric and radiometric measures on the Earth surface. Given that most trees fall into a small range on both measures, LiDAR and NIR data provide a strong heuristic on tree detection. However, the availability of remote sensing imagery is very limited compared to aerial imagery. On the other hand, significant progress has been made on image-based object recognition in the computer vision community. It is necessary to explore tree detection methods that operate on pure images, which is the focus of our work.

There is a wide literature on object detection from aerial imagery. [18, 7, 16, 1] propose image-based features for extracting roads, intersections, buildings and compound objects such as harbors from aerial imagery. However, such man-made objects have a strong prior in shapes and recurrent patterns, so the features cannot be directly applied to tree detection.

Porway *et al.* [15] propose a method for parsing aerial imagery into an object hierarchy. An aerial image is learned at both scene and object levels with color histogram and bag of SIFT features [12]. Contextual constraints are then applied to resolve the ambiguities of learned results (*e.g.*, cars on top of trees). However, since the object inference is learned in the context of multiple objects, the discriminating power is lowered. Their final results contain many false positives (about 20% for trees), even after contextual constraints are applied.

1.2 Overview

Our approach to tree detection proceeds in two stages. During the first stage, a pixel-level classifier is trained to assign a {tree, non-tree} label to each pixel in the aerial image based on a set of visual features. The pixel-level classification is then refined by considering the local smoothness of pixel labeling to generate a clean segmentation of tree and non-tree regions. During the second stage, a set of tree templates are used to correlate with the classification results and locate candidate tree crowns. The tree crowns are then selected in a greedy manner to maximize correlation scores while minimizing the overlap.

Compared to previous work, our method only requires the RGB channels of aerial imagery for tree detection, and achieves $> 90\%$ accuracy in pixel-level tree classification. On the other hand, the training procedure for the pixel-level classifier is open for any pixel-level features (such as LiDAR and NIR data). Therefore our method can be conveniently incorporated into existing methods to boost the performance for tree detection.

We also address the scalability of the proposed approach. Since training a pixel-level classifier requires manual creation of ground-truth tree masks, we introduce methods for automatic model and training data selection, so that the amount of training data can be minimized. We demonstrate the methods on a large urban area using only 1% of the aerial images as training data. By applying model and training data selection, we achieve $> 90\%$ accuracy in pixel-level tree classification, which advances the baseline training method by 5%.

2. TREE/NON-TREE SEGMENTATION

In this section, we discuss the first stage of our approach: segmenting the aerial image to tree and non-tree regions. Our method is inspired by the computer vision literature. Tree and non-tree regions are segmented by applying a pixel level classification based on a set of visual features followed by a partitioning algorithm for refinement. While many previous works exist on object-oriented image segmentation, few are dedicated to large-scale GIS data. We empirically tune the selection of visual features and classifiers for an optimal balance of speed and accuracy.

2.1 Feature selection

Since we apply pixel-level classification, all the visual features are selected at the pixel level. Higher order geometric features (edges, shapes, *etc.*) or bag of words models (quantized SIFT, textons, *etc.*) are not considered in our current work. However, we shall show that the pixel-level features can already capture the most distinctive properties of trees, and the adoption of pixel-level features allows convenient incorporation of LiDAR and NIR data into our method without changing the pipeline.

Color. Color is undoubtedly the most revealing feature for trees. We convert the RGB channels of each aerial image to CIE L*a*b* color space for a better perceptual uniformity, and to the illumination-invariant color space [8] for some robustness against the change of lighting condition. We concatenate the two color representations to form a 6 dimensional feature vector at each pixel.

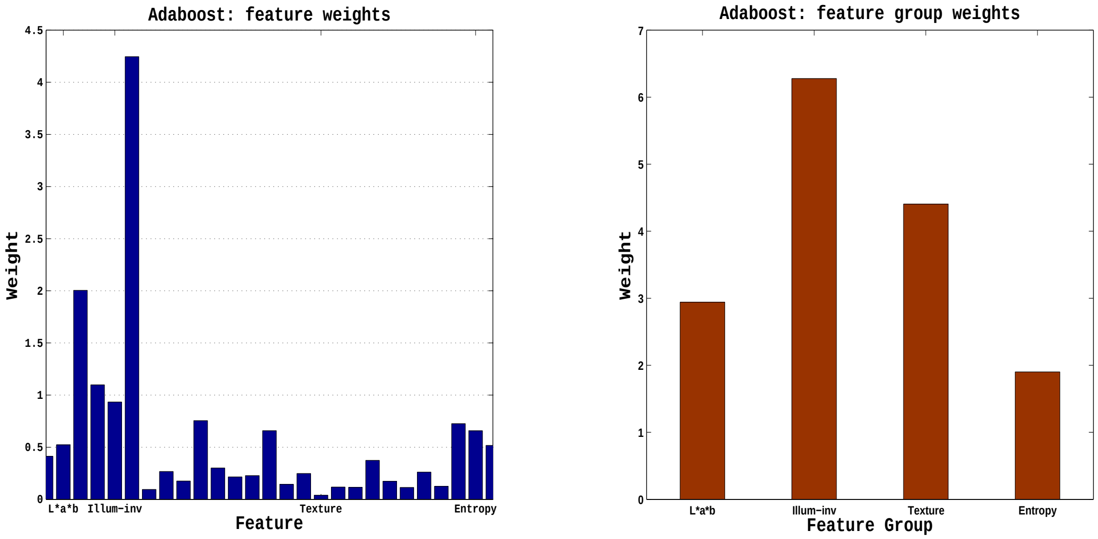
Texture. The texture pattern formed by tree leaves often distinguishes trees from similarly colored objects such as grass and tennis courts. We convolve the L channel of each aerial image with a filter-bank to generate a set of filter responses at each pixel as its texture feature. We empirically choose the Gaussian derivative filters to form the filter-bank [13], although other kernels such as Gabor filter achieve a similar performance. Each Gaussian derivative filter is a Gaussian in the X direction and a second derivative of Gaussian in the Y direction. The filter-bank consists of filters on 3 scales (with $\sigma = 1, \sqrt{2}, 2$) and 6 orientations uniformly sampled in $[0, \pi)$, which generates an 18 dimensional feature vector at each pixel.

Entropy. Entropy measures the uncertainty of a random variable (in our case, the L channel of aerial images). Because of shadows, tree leaves tend to exhibit a substantially higher entropy compared to man-made objects such as roads and buildings. We compute the entropy of each pixel within 5×5, 9×9, and 17×17 search windows on the L channel of the image. The later two can be efficiently approximated by iteratively downsampling the image and keeping the search window at 5×5. We concatenate the entropy values to form a 3 dimensional feature vector at each pixel.

Finally, by concatenating color, texture and entropy features, a 27 dimensional feature vector is formed at each pixel. If other informative features are available (*e.g.*, LiDAR and NIR), they can be conveniently concatenated here, without any changes on the rest of the pipeline.

2.2 Pixel-level classification

Each of the 27 visual features alone can be used to construct a weak classifier for tree detection. In order to form a discriminative combination, we adopt Adaboost [9] to train a strong classifier based on these weak classifiers:



(a) Individual feature weights

(b) Grouped feature weights

Figure 2: The weights of (a) individual features and (b) grouped features learned by Adaboost.

$$H(x) = \sum_{t=1}^{T} \alpha_t h_t(x), \tag{1}$$

where x is the feature vector at a pixel to be predicted, and $H(x)$ is the output of the strong classifier. $sign(H(x)) \in \{+, -\}$ gives a prediction of {tree, non-tree} at the pixel, and $|H(x)|$ gives a confidence score of the prediction. In the sequel, the confidence score is normalized to tree probability in $[0, 1]$ by the sigmoid function:

$$P_{tree}(x) = \frac{1}{1 + e^{-H(x)}}. \tag{2}$$

We adopt the basic decision stump on a single feature as the weak classifier $h_t(x)$:

$$h_t(x) = \begin{cases} 1 & s x_{i(t)} > s\theta_t, \\ 0 & otherwise. \end{cases} \tag{3}$$

In all our experiments, we use $T = 200$ weak classifiers. The parameterization for each weak classifier $s \in \{+, -\}$, $i(t) \in \{1, ..., 27\}$ and θ_t, along with their weights α_t, are automatically learned by the Adaboost procedure.

Once a strong classifier is trained, predicting each pixel only takes a linear combination of stump decisions. In our initial study, a 512 × 512 aerial image takes 15 seconds to be classified by Adaboost, as opposed to 15 minutes by non-linear SVM [4], while both achieving comparable accuracies.

2.3 Classification refinement

One disadvantage of pixel-level classification is that each pixel is predicted independently. The consistency among adjacent pixels is not considered. Therefore the results are noisy (*e.g.*, shadowed tree leaves are labeled as non-tree; sparse points on a grass field are labeled as tree; *etc.*). In this section, we refine the pixel-level classification results to a clean segmentation of tree and non-tree regions by optimizing the consistency between all pairs of neighboring pixels.

The refinement is formalized as an energy minimization problem and solved by the graph cuts algorithm presented in [3]. The energy function is defined as follows:

$$E = \sum_{p \in \mathcal{P}} D(p, L(p)) + \sum_{p,q \in \mathcal{N}} V(L(p), L(q)), \tag{4}$$

where $\mathcal{P}$ is the set of all pixels, $L(p) \in$ {tree, non-tree} gives the label of pixel p, and $\mathcal{N}$ is the set of all neighboring pairs of pixels (pixels are 8-connected). The data term D measures the agreement between the assigned label and the tree probability given by Adaboost, and the smoothness term V measures the local smoothness of labeling:

$$D(p, L(p)) = \begin{cases} log(1 - P_{tree}(p)) & L(p) = tree, \\ log(P_{tree}(p)) & otherwise, \end{cases} \tag{5}$$

$$V(L(p), L(q)) = \begin{cases} 0 & L(p) = L(q), \\ \beta & otherwise, \end{cases} \tag{6}$$

where β is empirically set to 1 in all our experiments.

The bottom row of Figure 3 shows the clean segmentation after graph cuts optimization. Small holes in tree crowns are filled with tree labels. More importantly, the false tree labels on the soccer field and the river are corrected, which is fundamental for the subsequent tree localization.

Notice that, while simpler techniques for image denoising exist such as morphological close and open operations (a dilation followed by erosion), they are incapable of taking into account the labeling probabilities. For example, a dilation followed by erosion on the classification results shown in Figure 3 is likely to generate small tree regions on the soccer field and the river because of the locally dense tree labels, even though the probabilities supporting those labels are very low.

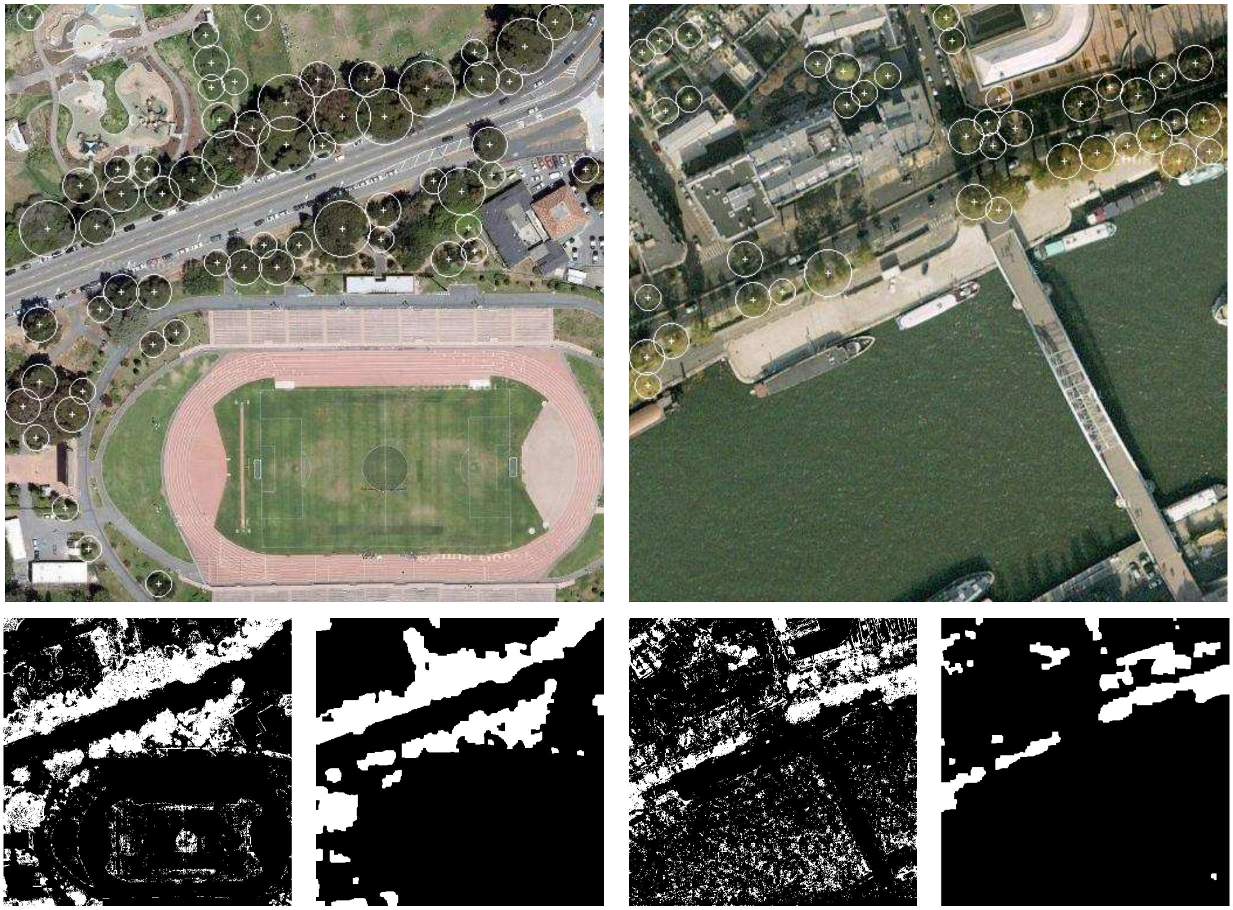


Figure 3: The top row shows the tree detection results for two sample aerial images, where the intermediate results for classficiation (left) and classification refinement (right) are shown below each image.

3. TREE LOCALIZATION

Based on the refined tree segmentation, we discuss in this section the second stage of our approach: locating each individual tree and providing an estimate of its crown size. For this task, we turn to the old technique of template matching, which has been adopted in a previous work on NIR-based tree detection [10].

Our template matching is applied to 4 channels: the R, G, B channels of the original aerial image, and the channel P of tree probabilities given by Adaboost classification. Values of the R, G, B, P channels are scaled to $[0, 1]$.

During the training stage, the R, G, B channels of a tree template is formed by averaging multiple ground-truth tree crowns with the same radius. The P channel of a tree template is simply a binary disk (with 1 inside and 0 outside) with the same radius as the template. Multiple templates with various radiuses can be formed depending on the geographic region. In our experiment, we form 4 tree templates for each data set with radiuses ranging from 2m to 8m.

Based on the refined tree segmentation given by graph cuts, template matching is constrained within the tree regions at discrete locations (sampled at half-radius step). At each location, the tree templates are correlated with the aerial image and the normalized correlation score is stored along with the template itself to a candidate list. The template matching strategy does not handle trees crossing image boundaries. In practice, the boundary effect can be reduced by stitching adjacent images together.

The candidate list contains a number of overlapping template matches covering the tree regions of the aerial image. In order to maximize the likelihood of all template matches while minimizing their overlaps, the final set of template matches are selected in a greedy manner. First the candidate list is thresholded by a minimum correlation score of θ_{corr}. Template matches with scores lower than θ_{corr} are removed. The rest of the candidate list is sorted in descending order of correlation score. Then in each round, the top template match of the list is selected into the final set, and the rest of the template matches in the list are removed if their overlap with the selected one exceeds certain threshold $\theta_{overlap}$. The process iterates until all the template matches in the candidate list are either selected or removed.

We empirically set $\theta_{corr} = 0.25$ and use a simple metric to define the overlap between two templates:

$$Overlap(T_i, T_j) = \frac{(R_i + R_j - distance(C_i, C_j)}{min(R_i, R_j)}, \qquad (7)$$

where R_i and C_i are the radius and center of template match T_i, and the threshold $\theta_{overlap}$ is set to 0.25 (the outer quarter of a crown can overlap with other crowns).

4. EXPERIMENTATION

We collected aerial images for San Francisco (about 1500 512×512 pixel tiles), Paris (about 4000 tiles), and New York (about 4500 tiles) for experimentation. The resolution for each aerial image is between 0.5m and 1m per pixel. 1% of the tiles from each data set are manually masked for training the pixel-level classifier (see Section 5 for a discussion on model and training data selection) and collecting tree templates. The tree detection procedure is applied to the rest of the tiles. An end-to-end processing for each tile takes less than 30 seconds in average. An illustration for an end-to-end processing is shown in Figure 3.

Since we adopt a collection of visual features to capture the properties of trees, one interesting question we consider is which of the visual features are more discriminative for the task of tree detection. Given that Adaboost associates a weight to each weak classifier, and each weak classifier is a decision stump based on a single visual feature, we aggregate the weights of all the weak classifiers into the bins for their associated feature to approximate the total weight for each visual feature. In Figure 2, we show the weights for individual features as well as grouped features (L, a, b grouped together as L*a*b*, *etc.*). Since color takes a dominant amount of weight, we keep L*a*b* and illumination-invariant color features as separate groups. It is shown that color features are the most discriminative feature for trees, and the illumination-invariant representation is more powerful than CIE L*a*b*. Texture features on a single scale and orientation is weak (which is reasonable because texture of tree leaves has random orientations), but combined together still have a significant impact on the final classification. Entropy as a group feature is the least discriminative among all, but the entropies of each of the individual scales have comparable weights to the individual L, a, b color features.

In large-scale tree detection, false detections are inevitable. In our experiments, typical failure cases are caused by locally indistinguishable objects such as grass, river, *etc.* or by an unrepresentative training set (see Section 5 for details). Figure 4 shows a few examples. Notice that if LiDAR or NIR data is available, the pixel-level classifier should be able to eliminate most false detections on the river or man-made objects (Figure 4 (b) and (c)). The incorporation of LiDAR and NIR data is left for future work.

5. TREE DETECTION FOR THE GLOBE

One common bottleneck for GIS applications is the gigantic amount of data that needs to be processed. The bottleneck is especially significant if supervised training is involved. In our approach, the training of a pixel-level classifier requires manual creation of ground-truth tree masks. If it takes a large percentage of the data to train a discriminative classifier, the manual work required by processing the entire globe will be prohibitively expensive.

In this section, we address the scalability of pixel-level classification and propose methods for model and training data selection in order to minimize the percentage of data used for training while achieving a resonable accuracy.

One principle for machine learning algorithms is that a representative set of positive and negative samples must be seen in the training data in order to train a robust classifier. Take tree classification for example, if a geographic region contains a lake, but none of the lake views appears in the training data, then the classifier is likely to perform poorly on the lake because it looks more similar to positive samples (trees) than negative ones (ground, buildings, *etc.*). On the other hand, if we enlarge the geographic region to a larger land coverage, the diversity of geographic features (*e.g.*, ground, grass, trees, buildings, lakes, *etc.*) will grow at a lower rate, because the total number of geographic features in the region is very limited compared to the area. Based on this observation, we propose to cluster large-scale aerial imagery with similar geographic features, and apply model and training data selection from the clustering results.

Since the pixel-level classifier is trained on visual features, aerial images are also clustered into visually coherent clusters. We adopt the color-augmented Gist descriptor to encode the geographic features for aerial images. The Gist descriptor [14] computes a low dimensional representation for the scene structure of an image, and has been shown to convey higher level semantics of images in the computer vision community [11]. We form the Gist descriptor by convolving an aerial image with Gabor filters on 3 scales with 8, 8, and 4 orientations respectively, and aggregating the filter responses into 4×4 spatial bins, which gives a 320 dimensional descriptor for each aerial image. We also compute an $8 \times 8 \times 8$ joint color histogram in the L*a*b* space to form a 512 dimensional color descriptor. The Gist and color descriptors are both normalized to unit L-1 norm and concatenated together to form the descriptor for an aerial image. After the descriptors are computed for all aerial images, Principal Component Analysis (PCA) is applied to reduce the dimensionality of descriptors to 12 while preserving about 95% of the total variance of the original descriptors.

We experimented with three training methods on the New York data set with about 4500 tiles. All three methods are constrained to use 1% of the tiles (45 tiles) as training data.

- Baseline: 45 training tiles are uniformly sampled in the geographic region.
- Cluster-I: k-means are applied to the image descriptors to divide the data set into 45 clusters, and the training tiles are selected as cluster centroids.
- Cluster-II: Two-level clustering is applied to the image descriptors. The first level k-means divide the data set into 4 clusters. Within each cluster, the method of Cluster-I is used to select 1% as training tiles. The training tiles of each cluster are used to train a **separate** classifier dedicated to the tiles in that cluster.

In order to compare the performance of the three methods, we randomly sampled 100 tiles from the data set (none appeared in any of the three training sets) and manually created the ground-truth masks for validation. The confusion matrices and accuracies for all three methods are shown in Table 1. It is shown that Cluster-II exceeds Cluster-I by 2% and Baseline by 5% in classification accuracy. For 4500 512×512 tiles of aerial images, a 5% improvement in accuracy means that about 60 million more pixels are correctly labeled. More importantly, the improvement mainly comes from reducing false positives (shown in the second row of confusion matrices), to which most applications are more sensitive (*e.g.*, placing a false tree on the water is much worse than missing a tree in 3D city modeling).

Table 1: confusion matrices and accuracies for the three training methods on the New York data set.

		Baseline		Cluster-I		Cluster-II	
		Tree	NonT.	Tree	NonT.	Tree	NonT.
Ground-Truth Labels	Tree	5.6%	0.5%	5.4%	0.7%	5.1%	0.9%
	NonT.	12.0%	81.0%	10.3%	83.7%	8.4%	86.6%
Accuracy		86.6%		89.1%		91.7%	

6. CONCLUSIONS

In this paper we propose an automatic approach to tree detection from aerial imagery. Given an aerial image, we combine pixel-level classification followed by global optimization to generate an image segmentation of tree and non-tree regions. Based on the image segmentation, we adopt template matching to locate each individual tree crown and provide an estimate of its crown size.

The proposed approach operates on pure images only, and is efficient for tree detection in large volumes. The pipeline for pixel-level tree classification is also open for convenient incorporation of traditional LiDAR and NIR data to further boost the accuracy.

To address the scalability of the proposed approach, we exploit the limited geographic features on Earth and introduce methods for model and training data selection based on two-level clustering. On large data sets, we are able to train the classifier on only 1% of the data while achieving $> 90\%$ accuracy for pixel-level tree classification.

Our current focus is on the general topic of tree detection. One interesting direction for future research would be distinguishing trees of different species. Our adoption of template matching can implicitly infer certain parameters on the tree crowns. However, substantially more information is still required for robust tree species recognition.

7. ACKNOWLEDGMENTS

The authors would like to thank Paul Strauss for rendering the tree models in Google Earth.

8. REFERENCES

[1] S. Bhagavathy and B. Manjunath. Modeling and detection of geospatial objects using texture motifs. *Geoscience and Remote Sensing, IEEE Transactions on*, 44(12):3706–3715, Dec. 2006.

[2] S. Boriah, V. Kumar, M. Steinbach, C. Potter, and S. Klooster. Land cover change detection: a case study. In *KDD '08: Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 857–865, New York, NY, USA, 2008. ACM.

[3] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:359–374, 2001.

[4] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[5] A. Charaniya, R. Manduchi, and S. Lodha. Supervised parametric classification of aerial lidar data. In *Computer Vision and Pattern Recognition Workshop, 2004. CVPRW '04. Conference on*, pages 30–30, June 2004.

[6] G. Chen and A. Zakhor. 2d tree detection in large urban landscapes using aerial lidar data. In *ICIP 2008.*, Feb. 2009.

[7] Y.-Y. Chiang and C. A. Knoblock. Automatic extraction of road intersection position, connectivity, and orientations from raster maps. In *GIS '08: Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems*, pages 1–10. ACM, 2008.

[8] H. Y. Chong, S. J. Gortler, and T. Zickler. A perception-based color space for illumination-invariant image processing. *ACM Trans. Graph.*, 27(3):1–7, 2008.

[9] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting, 1995.

[10] J. A. Greenberg, S. Z. Dobrowski, and S. L. Ustin. Shadow allometry: Estimating tree structural parameters using hyperspatial image analysis. *Remote Sensing of Environment*, 97(1):15 – 25, 2005.

[11] J. Hays and A. A. Efros. Scene completion using millions of photographs. *ACM Trans. Graph.*, 26(3):4, 2007.

[12] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60:91–110, 2004.

[13] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. 8th Int'l Conf. Computer Vision*, volume 2, pages 416–423, July 2001.

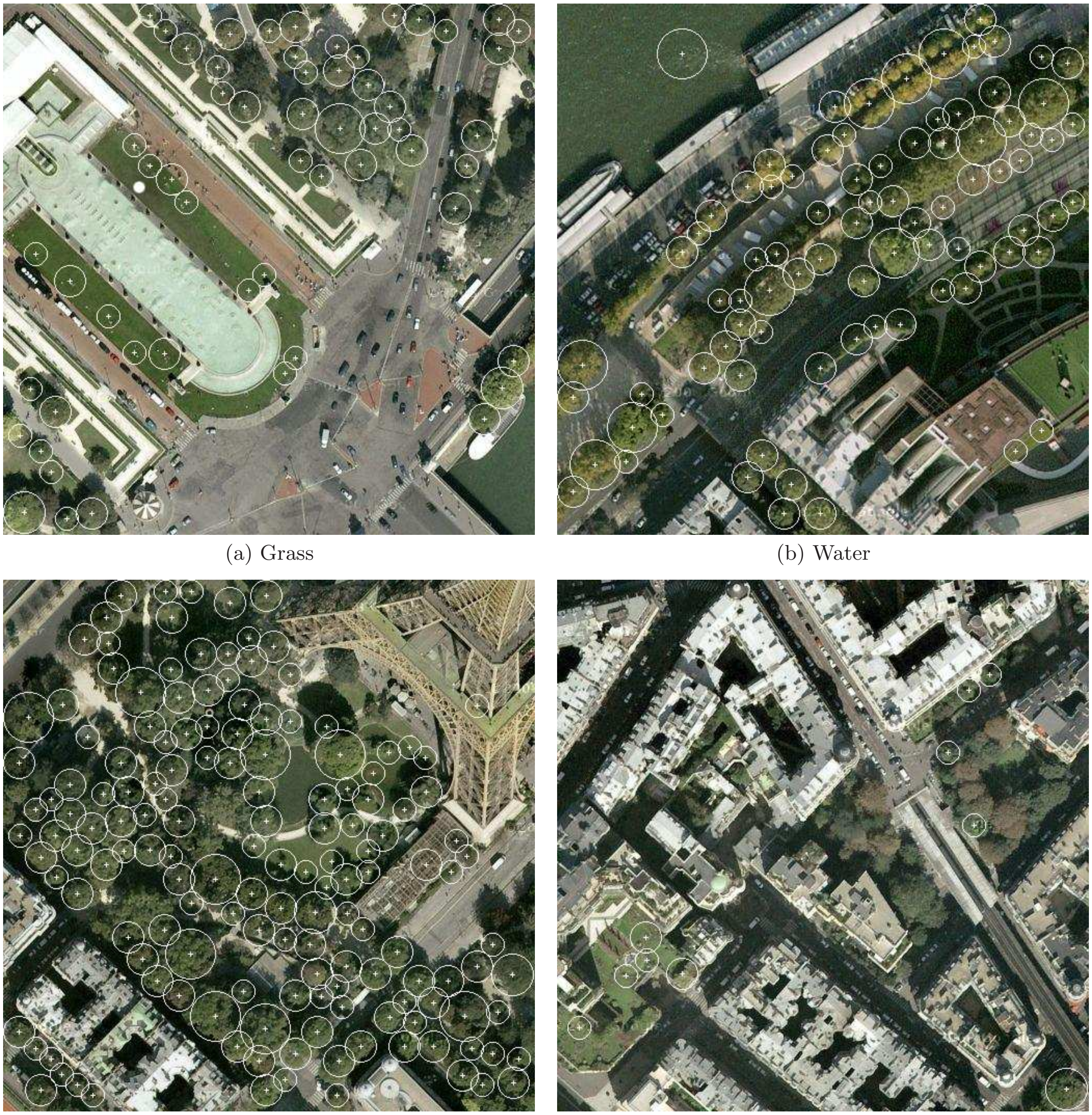
[14] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42:145–175, 2001.

[15] J. Porway, K. Wang, B. Yao, and S. C. Zhu. A hierarchical and contextual model for aerial image understanding. In *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, pages 1–8, June 2008.

[16] B. Sirmacek and C. Unsalan. Urban-area and building detection using sift keypoints and graph theory. *Geoscience and Remote Sensing, IEEE Transactions on*, 47(4):1156–1167, April 2009.

[17] B.-M. Wolf and C. Heipke. Automatic extraction and delineation of single trees from remote sensing data. *Mach. Vision Appl.*, 18(5):317–330, 2007.

[18] X. Wu, R. Carceroni, H. Fang, S. Zelinka, and A. Kirmse. Automatic alignment of large-scale aerial rasters to road-maps. In *GIS '07: Proceedings of the 15th annual ACM international symposium on Advances in geographic information systems*, pages 1–8. ACM, 2007.



(a) Grass

(b) Water

(c) Objects with similar color and texture to trees (tower)

(d) Un-trained tree types

Figure 4: Typical failure cases.