TREE TRUNK DETECTION USING CONTRAST TEMPLATES

Yan Lu and Christopher Rasmussen

Department of Computer and Information Sciences
University of Delaware, Newark, DE, U.S.A.

ABSTRACT

We propose a simple contrast-based method for tree detection and shape estimation from ground-plane perspective images for purposes of counting, classification, modeling, or robotic obstacle avoidance. Under the assumption that tree trunks are relatively narrow and vertical shapes which strongly differ in appearance from the scene background and have boundaries of opposite contrast, we apply a bank of bar filters parametrized from camera intrinsics combined with trunk location and diameter limits, and integrate results vertically. Non-maximum suppression is applied to candidates in the resulting trunk likelihood image. We present results demonstrating the effectiveness of our tree detection algorithm on a variety of forested images obtained directly from our robot as well as sampled from the web, and quantify performance using ground truth on a subset of those images. We also compare the results of our method with several related published approaches.

***Index Terms*— Feature extraction, image segmentation, robot vision systems**

1. INTRODUCTION

Trees are common objects in outdoor environments, and detecting them robustly in images taken from a ground-level perspective is a challenging problem with several applications. First, it may be desirable to count, classify, or model trees [1, 2] in scenes containing buildings, people, or other types of vegetation, necessitating segmentation of tree-containing regions as an initial step. Also, trees are important in outdoor robot navigation both as obstacles and potential landmarks. In related work [3, 4], we have studied robotic hiking and mountain-bike *trail following* in a variety of terrain types based on color appearance contrast. In forested terrain, trees are of course the most prominent obstacles affecting motion planning, and we would also like to carry out the above classification and modeling tasks autonomously. Therefore, it is both helpful and necessary to explicitly detect trees rather than to simply lump them in with other non-traversable obstacles based on scene structure derived from stereo vision or laser range finders [5, 6, 7, 8, 9] or local appearance contrast [10, 11].

In this paper, we propose a simple yet effective contrast-based approach to detect and parametrize approximately vertical tree trunks in a wide variety of scene types. Fig. 1(a) shows an example scene from our robot dataset. Because it was captured by an omnidirectional camera, the image was first warped to a panoramic projection such that tree trunks become vertical (as seen in cropped format in Fig. 1(b)), but this is not necessary in general. Tree candidates are searched for in the image at discrete trunk diameters and distances (the gray lines in (b)). Rectangular tree trunk regions detected based on the technique described in the next section are shown in Fig. 1(b), and their unwarped locations are shown in the original omnidirectional image in Fig. 1(c).

Author e-mail: yanlu@udel.edu and cer@cis.udel.edu.

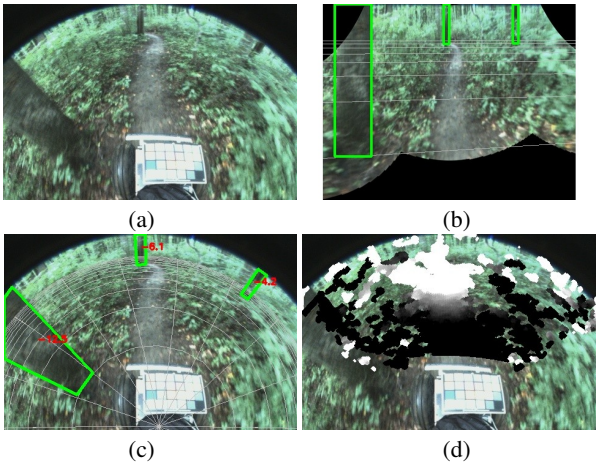


Fig. 1: Sample tree image and detection results. (a) Raw omnidirectional image from robot-collected dataset; (b) Warped panoramic image with depth lines at 1 m intervals and tree detections overlaid; (c) "Unwarped" tree detections; (d) Stereo depth map for same scene (note ragged edges and missing disparity values).

Considering alternative techniques, stereo depth maps such as the one in Fig. 1(d) using OpenCV's semi-global block matching function [12] are problematic due to noise and holes which distort the shape and the boundaries of the trees, especially for cluttered scenes. Pixel or patch/superpixel classification have also been used for appearance-based traversability estimation [10, 11]. Such methods are not directly applicable in our scenario, because our goal is to explicitly estimate the position and the size of trees in the scene, but post-processing of such low-level obstacle classification results is possible. The method of [6] is most similar to ours in that it performs vertical grouping of features, but they detect left and right tree trunk edges separately and then try match them by looking for pairs of edges of opposing contrast. Such a method is prone to errors in the matching stage, and tree trunk segments may not be linked together. In this work, we parametrize tree hypotheses in the ground-plane coordinates, detect both edges of the tree at the same time, and explicitly link detected trunk sections all the way to the ground.

2. TREE DETECTION BASED ON CONTRAST TEMPLATES

Tree trunks that are different in appearance from their background will have boundaries of opposite contrast, and they are nearly vertical in the perspective images with a normally-held camera. Given these two characteristics, we apply *bar filters* as contrast templates to extract vertical features of varying widths in the scene. The tree candidates are searched depth by depth. For each depth, given the range of tree diameters in meters and the camera parameters, we can calculate the range of diameters in pixels, which determines the number of bar filters of different sizes that should be applied at that depth level. In order to facilitate the search procedure, we discretize the depth and the filter sizes. Two levels of 1-D non-maximum suppression are then performed to obtain final detection results.

2.1. Camera Calibration

The omnidirectional camera used to capture the scene is fully calibrated, with the OCamCalib Omnidirectional Camera Calibration Toolbox for Matlab [13] used to obtain intrinsics and extrinsics estimated using ground plane fiducials in the lab. This allows us to warp omnidirectional images to a panoramic view (shown in Fig. 1) such that the edges of tree trunks in the warped image are vertical. We masked out the robot chassis and peripheral pixels which are on the sides of the robot or not imaged.

For web images (described in the Results section) for which the calibration is unavailable, we used reasonable default parameters for horizontal and vertical fields of view and manually indicated the horizon line to estimate camera tilt.

2.2. Depth and Filter Size Discretization

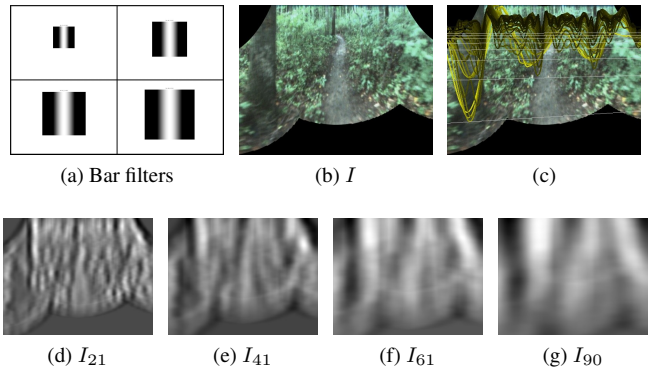
For navigation and classification purposes, we are most interested in tree trunks that are within a certain distance of the camera, so we only search for tree trunks whose bottom positions are within a meters in depth. To facilitate the search procedure, we discretize the depths. Fig. 1(b) shows the images overlaid with depth lines up to 8 m in front of the robot with a depth step of 1 m. Due to a slight rotation offset in the roll angle introduced at the stage of camera mounting, these depth lines are not perfectly horizontal in the robot panoramic images.

We define the range of tree diameters in meters to be $r = [d_1, d_2]$ and assume that trees' bases are on the ground plane. Then, the range of tree diameters in pixels at a particular depth a_n is $R_{a_n} = [D_{1_{a_n}}, D_{2_{a_n}}]$, where $D_{1_{a_n}} = d_1 * a_{n_p}/a_n$, $D_{2_{a_n}} = d_2 * a_{n_p}/a_n$, and a_{n_p} is the depth of a_n in pixels which can be calculated from camera parameters. The range R_{a_n} determines the sizes of bar filters to be applied at depth a_n. It is time-consuming and also not necessary to apply bar filters associated with every value in R_{a_n}. Similar to how we discretize the depth, we select from R_{a_n} a few values for the sizes of bar filters at depth a_n such that these values spread out among R_{a_n}.

2.3. Bar Filters

Considering that tree trunks are strong vertical features of certain widths in the image, we apply bar filters of different sizes on the image to extract the features. The kernel function of the bar filter is

$$\begin{aligned} x' &= y\sin(\theta), \qquad y' = y\cos(\theta), \\ B(x,y) &= \exp(-0.5 * (\frac{x'^2}{s_x} + \frac{y'^2}{s_y}))\cos(2\pi f x'), \end{aligned} \quad (1)$$



(a) Bar filters (b) I (c) (d) I_{21} (e) I_{41} (f) I_{61} (g) I_{90}

Fig. 2: (a) Subset of bar filters with $s_x \in \{21, 41, 61, 90\}$ ($s_y = s_x$, $f = 1/(2 * s_x)$, and $\theta = 90^o$); (b) Input panoramic image I; (c) Values of $S_{bar}(j, a_n)$ across every column visualized by overlaid curves when a_n ranges from 1 m to 8 m and s_x ranges from 11 to 107. The value of s_x is indicated by the intensity of the curve. Larger s_x is colored in higher intensity. (d)-(g) Results after convolving the bar filters with the green channel of I.

where s_x and s_y are half width and height of the kernel, $x \in [-s_x, s_x]$, $y \in [-s_y, s_y]$. A bar filter of size s_x is applied to find trees of width s_x in the image. In the spatial domain, the bar filter is a Gaussian kernel function modulated by a 1-D sinusoidal wave in the horizontal direction. We adapt our bar filter from Gabor filter, and the difference is that, for Gabor filter, the Gaussian kernel function is modulated by a 2-D sinusoidal plane wave. We make such a change because we do not want the value varies in the y direction for the bar filter. The frequency and orientation of the bar filter are set to be $f = 1/(2 * s_x)$ and $\theta = 90^o$. Fig. 2a shows a subset of bar filters we apply for our experiment. The results, I_{s_x}, after convolving these bar filters with the green channel of Fig. 2b are shown in Figures 2d to 2g, and the results are normalized to 0-255 for display purpose.

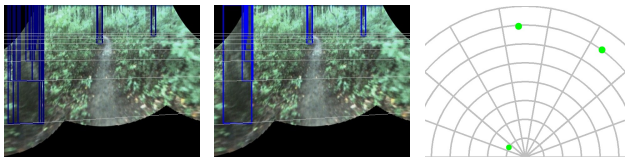
From the results, an s_x-sized bar filter correctly reacts to vertical features of width s_x. For dark trees with bright background as the case shown in Fig. 2b, their convolved values with bar filters are negative, and they appear dark in the resulting images. The results are more smooth when s_x is larger due to the effect of Gaussian smoothing incorporated in the bar kernel.

2.4. Tree Candidate Selection

We look for tree candidates depth by depth. For each discretized depth a_n, we apply bar filters whose s_x are chosen from R_{a_n}. Once the bar filtered result I_{s_x} is obtained, a normalized summed-up value $S_{bar}(j, a_n)$ is calculated for every column j of I_{s_x} such that

$$S_{bar}(j, a_n) = \sum_{i=0}^{i=r_{n_j}} I_{s_x}(i, j)/(s_x * s_x * n * a_n). \quad (2)$$

In the above equation, $I_{s_x}(i, j)$ is the pixel value of I_{s_x} at row i and column j, and r_{n_j} is the row number of the depth line a_n overlaid on the image at column j. We add values of $I_{s_x}(i, j)$ starting from row r_{n_j} all the way up to the top of the image along column j for $S_{bar}(j, a_n)$. Then, $S_{bar}(j, a_n)$ is normalized with respect to filter size s_x, number of pixels added n, and the depth a_n, so that $S_{bar}(j, a_n)$ is comparable across different values of s_x, a_n, and n.



(a) (b) (c)

Fig. 3: (a) Tree candidates extracted across all depths and values of s_x for the image from Fig. 2(b); (b) Candidates after non-maximum suppression along the horizontal direction; (c) Bottom positions and sizes of detected trees on polar grid with circles at 1 m intervals.

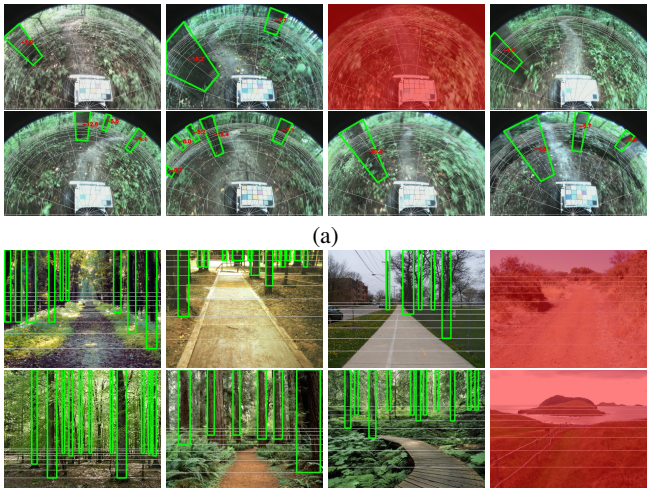
Masked pixels are excluded for this calculation. In Fig. 2c, the values of $S_{bar}(j, a_n)$ are visualized by curves overlaid on the images when $a_n \in [1, 8]$ m and $s_x \in [11, 107]$.

From Fig. 2c, trees align well with the positions where local minima of $S_{bar}(j, a_n)$ occur. Therefore, we first extract tree candidates by searching depth by depth across all sizes of bar filters for local minima whose values of $S_{bar}(j, a_n)$ also satisfy $S_{bar}(j, a_n) < \alpha$, where α is the threshold that triggers a detection. The shape of a tree candidate is parameterized by a rectangle with its bottom aligned with r_{n_j}. Its width is the value of s_x, and its center is $(r_{n_j}/2, j)$. Fig. 3a shows all tree candidates extracted from the local minima shown in Fig. 2c with $\alpha = -4$. Then, we apply two levels of non-maximum suppression, first in the horizontal direction (Fig. 3b) and then in the vertical direction on the value of $S_{bar}(j, a_n)$ that is associated with each candidate, to obtain the final detections already shown in Fig. 1(b). The corresponding locations in the omnidirectional image are shown in Fig. 1(c) with the value of $S_{bar}(j, a_n)$ for each detection indicated. Fig. 3c shows the bottom positions and the sizes of these detected trees in the obstacle map, given camera parameters.

3. RESULTS

Here we show results of testing our tree detection algorithm on three sources of images: (1) an omnidirectional image dataset collected from our robot along a hiking trail in a forested area of the mid-Atlantic U.S.; (2) a set of 38 tree and non-tree images collected from the web, and (3) selected images from other papers [6, 11] that propose methods to solve similar problems. We have also tried the popular machine-learning-based HOG object detection algorithm [14] on our dataset.

There are about $5,200$ total image frames captured at 10 Hz in the omnidirectional image robot dataset. Of these, we generated ground truth by manually labeling trees in a subset of 118 images. Our algorithm searched for trees over a range of diameters $r = [0.2, 0.8]$ m and a depth range of $[1, 8]$ m with a depth step of 1 m. Five bar filters were applied at each depth with sizes picked to evenly cover the tree diameter range in pixels at that depth level. The images in Fig. 4(a) show the results of our tree detector on some images from this dataset when the trigger threshold $\alpha = -4$. Generally, our method does find tree up to 8 m away with a good estimate of their positions and sizes despite some significant variation in appearance. For example, the appearance of the detected nearby trees is quite different from those detected far away due to different levels of details of tree bark textures. Moreover, our detector is robust to partial occlusion by under-story foliage that blurs the tree and background boundaries. We also show some failure cases in the last column of Fig. 4(a). In the first row the size of the tree is not detected correctly due to its having a very dark region whose contrast with the



(a)

(b)

Fig. 4: Sample results of our tree detector on (a) Images from robot-collected omnidirectional image dataset; (b) Tree/non-tree images collected from the web. Images with no tree detections are tinted red.

rest of the tree is higher than the contrast between the entire tree and the background. In the second row a tree is erroneously detected in a tree-free area because of its clear contrast with the bright trail region, indicating a need for further texture analysis of detection regions.

The results can be quantified in several ways. From the perspective of classification, we can calculate the true positive rate and false positive rate at the pixel level of the detected tree trunk regions and generate an ROC curve as shown in Fig. 5 by varying the trigger threshold α. Another way of quantifying results at the pixel level is to calculate an overlap score between the detected tree regions and the ground truth. We use the formula suggested by [15]: $Overlap(\mathcal{R}_1, \mathcal{R}_2) = A(\mathcal{R}_1 \cap \mathcal{R}_2)^2/(A(\mathcal{R}_1)A(\mathcal{R}_2))$. The median $Overlap$ is 0.58 when $alpha = -4$ for the robot-collected dataset. In addition to pixel level quantification, we can measure the absolute errors in meters in the bottom center position $|\Delta_p|$ and trunk diameter $|\Delta_w|$ for true positives. The median $|\Delta_p|$ is 0.28 m and the median $|\Delta_w|$ is 0.1 m when $\alpha = -4$.

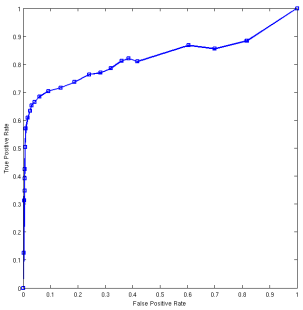


Fig. 5: ROC curve of detector results from robot dataset

To further validate the generality of our method, we tested it on a set of 38 tree and non-tree images culled from Flickr and Google and cropped and scaled to 320×240 as necessary. Some of the results are shown in Fig. 4(b), and they are fairly good qualitatively except for some misalignment of tree bottoms due our discretization of depth for efficiency.

We compared the results from our tree detector with other avail-

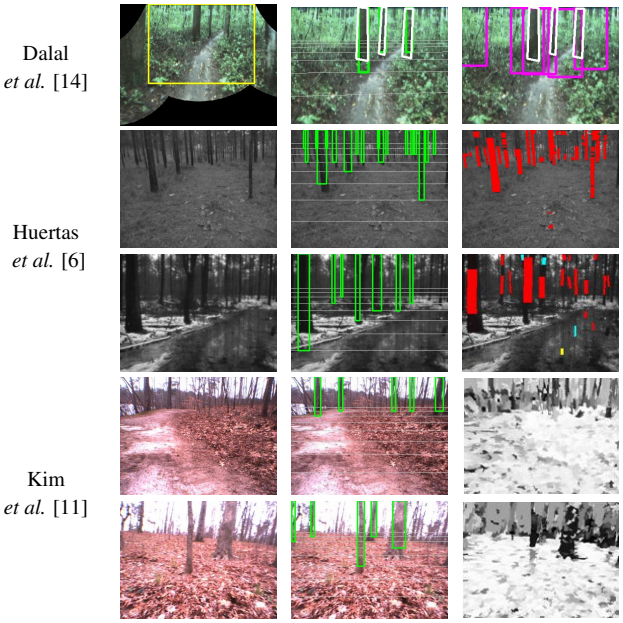


Fig. 6: Comparison with other appearance-based algorithms. (a) Original images; (b) Detection results with our method; (c) Results from alternative method. For the first row, in columns (b) and (c) the white polygons are ground truth.

able methods that have been proposed to solve similar detection problems, with some sample results shown in Fig. 6. The first column shows the input image, the second our method's detections in green, and the third the result of the alternative published method. The first comparison method we tried was the histogram of oriented gradients (HOG) detector with SVM [14], a baseline algorithm for person detection which learns the target object appearance from a large set of positive and negative examples. The HOG method was applied to look for tree trunks as objects by training with default parameters using scaled ground truth tree regions as positive examples and randomly-picked background patches as negative examples. In order to avoid masked-out regions, we cropped the panoramic image to the yellow rectangle shown in the first image of the first row of Fig. 6. The median $Overlap$ for HOG was a poor 0.10, while the median $Overlap$ was 0.60 on this cropped image set for our method.

We also tested our method on images from [6] and [11]. As with the web images, we manually indicated the horizon line in each such image in order to estimate the camera's tilt angle. From the results in Fig. 6, our method is able to find most of the trees in the images except those far away and tiny ones that are beyond our detection scope. By comparison, the advantage of our detector is that we can localize a tree in the images as a connected component, and this is especially helpful for robot navigation. Note how for [6] several of the tree trunk detections are in disconnected pieces. For [11] darker patches indicate higher obstacle likelihoods, but without explicit grouping there is no recognition of trees *per se*.

4. CONCLUSION

This paper has presented an effective contrast-based algorithm for tree detection in images for counting, classification, modeling, and robot navigation. Our experimental results show good performance using several metrics and relative to existing work in the field. Because our detector finds trees based on a flexible measure of contrast between them and the background it does not require an *a priori* appearance model and thus works well in variable lighting conditions. Moreover, the detector is relatively robust to partial occlusions and distractions because the detection decision is made based on a cumulative score along the vertical extent of each tree hypothesis for both left and right tree boundaries simultaneously.

We are exploring several avenues for improvement. First, due to the discretization of search depths the tree bottom positions are not exact. Smaller depth step sizes can be used to further reduce the errors at the cost of time, which can be mitigated by some optimizations neglected here. Additional region-based analysis can be performed at the end of the pipeline to help exclude false positives similar to the one in Fig. 4(h). Along these lines, we are experimenting with color and texture classification methods [16]. Finally, though our stereo depth maps are not currently accurate enough for direct tree detection (e.g., by depth contrast), we can use them as an additional check for our detections in the manner of [6].

5. REFERENCES

[1] Z. Huang, C. Zheng, J. Du, and Y. Wan, "Bark classification based on textural features using artificial neural networks," in *Advances in Neural Networks - ISNN 2006*, Lecture Notes in Computer Science, pp. 355–360. Springer, 2006.

[2] L. Lopez, Y. Ding, and J. Yu, "Modeling complex unfoliaged trees from a sparse set of images," *Pacific Graphics*, vol. 29, no. 7, pp. 192–204, 2010.

[3] C. Rasmussen, Y. Lu, and M. Kocamaz, "Appearance contrast for fast, robust trail-following," in *Proc. Int. Conf. Intelligent Robots and Systems*, 2009.

[4] C. Rasmussen, Y. Lu, and M. Kocamaz, "Trail following with omnidirectional vision," in *Proc. Int. Conf. Intelligent Robots and Systems*, 2010.

[5] L. H. Matthies, "Stereo vision for planetary rovers: stochastic modeling to near real-time implementation," *Int. J. Computer Vision*, vol. 8, pp. 71–91, 1992.

[6] A. Huertas, L. Matthies, and A. Rankin, "Stereo-based tree traversability analysis for autonomous off-road navigation," in *IEEE Workshop on Application of Computer Vision (WACV/MOTION)*, 2005.

[7] J. F. Lalonde, N. Vandapel, D. Huber, and M. Hebert, "Natural terrain classification using three-dimensional ladar data for ground robot mobility," *J. Field Robotics*, vol. 23, no. 10, pp. 839–861, 2006.

[8] K. Konolige, M. Agrawal, M. R. Blas, R. C. Bolles, B. Gerkey, J. Sol, and A. Sundaresan, "Mapping navigation and learning for off-road traversal," *J. Field Robotics*, vol. 26, no. 1, pp. 88–113, 2009.

[9] M. McDaniel, T. Nishihata, C. Brooks, and K. Iagnemma, "Ground plane identification using lidar in forested environments," in *Proc. IEEE Int. Conf. Robotics and Automation*, 2010.

[10] I. Ulrich and I. Nourbakhsh, "Appearance-based obstacle detection with monocular color vision," in *Proceedings of the National Conference on Artificial Intelligence*, 2000.

[11] D. Kim, S. Oh, and J. Rehg, "Traversability classification for ugv navigation: A comparison of patch and superpixel representations," in *Proc. Int. Conf. Intelligent Robots and Systems*, 2007.

[12] H. Hirschmuller, "Stereo processing by semi-global matching and mutual information," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 25, no. 2, pp. 328–341, 2008.

[13] D. Scaramuzza, *Omnidirectional Vision: from Calibration to Robot Motion Estimation*, Ph.D. thesis, ETH Zurich, Switzerland, 2008.

[14] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 2005, pp. 886–893.

[15] S. Sclaroff and L. Liu, "Deformable shape detection and description via model-based region grouping," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 23, no. 5, 2001.

[16] M. Varma and A. Zisserman, "A statistical approach to texture classification from single images," *Int. J. Computer Vision*, vol. 26, 2005.