An Experimental Evaluation of Different Features and Nodal Costs for Horizon Line Detection

Touqeer Ahmad[1], George Bebis[1], Emma Regentova[2], Ara Nefian[3], and Terry Fong[3]

[1] Dept. of Computer Science and Engineering, University of Nevada, Reno
sh.touqeerahmad@gmail.com, bebis@cse.unr.edu
[2] Dept. of Electrical and Computer Engineering, University of Nevada, Las Vegas
Emma.Regentova@unlv.edu
[3] NASA Ames Research Center
{ara.nefian,terry.fong}@nasa.gov

Abstract. Horizon line detection is a segmentation problem where a boundary between a sky and non-sky region is searched. Conventionally edge detection is performed as the first step followed by dynamic programming to find the shortest path which conforms to the detected horizon line. Recent work has proposed the use of machine learning to reduce the number of non-horizon edges to accurately detect the horizon line. In this paper, we investigate the suitablity of various local texture features and their combinations to reduce the number of false classifications for a recently proposed horizon detection approach. Specifically, we explore SIFT, LBP, HOG and their combinations SIFT-LBP, SIFT-HOG, LBP-HOG and SIFT-LBP-HOG as features to train the SVM classifier. We further show that using only edge information as the nodal costs is not enough and propose various nodal costs which can result in enhanced accuracy of the detected horizon line as evidenced by the conducted experiments and results. We compare our proposed formulations with an earlier approach relying only on edges and suffers due to faulty assumptions. We report our comparative results for an image set comprising of mountainous images captured during an outdoor robot exploration of Basalt Hills.

1 Introduction

The problem of segmenting an image into sky and non-sky regions is termed as horizon/skyline detection or sky segmentation. Various attempts have been made to horizon line detection in the recent years due to many applications of a detected horizon boundary. Horizon line detection finds its applications in the navigation and obstacle avoidance for aerial vehicles (UAVs, MAVs) [4,5,6,7,8,9], outdoor robot localization [1,10,11,12], visual geo-localization [17,18], ship detection and port security[19,20]. Previous attempts to horizon line detection can be grouped into two major classes; (i) methods modeling sky and non-sky regions using machine learning techniques [4,5,6,7,8,9,22] and (ii) methods relying

G. Bebis et al. (Eds.): ISVC 2014, Part I, LNCS 8887, pp. 193–205, 2014.

on edge detection as an essential preliminary step [1,2,3,21]. It should be mentioned that earlier methods to horizon detection suffer from the assumption that the horizon boundary is linear and hence are limited.

Lie et al. [3] are the first to formulate horizon line detection problem as a multi-stage graph problem. The detected edge map of the query image is converted into a multi-stage graph; each edge pixel becoms a low cost node and each non-edge pixel becoms a high cost node in the corresponding graph. Each column of the image is taken as a stage of the graph and each pixel in the row is treated as a node. Links are established between nodes in one stage(column) to nodes in the next stage(column) with cost associated to the vertical distances of these nodes. Lie et al. [3] fill up the gaps resulted by edge detection by introducing high cost dummy nodes. They introduce source and sink nodes to the left and right most columns of the graph respectively and a shortest path is searched by Dynamic Programming(DP) which conforms to the detected horizon line.

Both Ahmad et al. [1] and Hung et al. [2] have recently proposed an improvement to Lie et al. [3] where they reduce the number of edge pixels significantly by training a classifier. Hung et al. [2] train an SVM classifier using raw pixel intensities as features around the edges and use the trained classifier to reduce the number of edge in the novel test images. Ahmad et al.[1] reduce a significant number of edges by first keeping only those edges which survive a number of edge detection parameter choices [named Maximally Stable Extremal Edges(MSEEs)] and then using a trained classifier to distinguish between horizon and non-horizon edges. They then apply DP on top of these survived positively classified horizon edges (termed as MSEE_{+}). Ahmad et al. [1] trained an SVM classifier and used SIFT descriptors around edge pixels as feature choice unlike pixel intensities as done by [2]. Although their approach works fine; there are misclassifications due to classifier which intoduce gaps (due to false negatives) and increase number of horizon classified edges (due to false positives). Also they reported their preliminary results on a data set comprising only 10 images.

In this paper we investigate various other texture descriptors (SIFT[14], LBP[13], HOG[15]) and their combinations with each other (SIFT-LBP, SIFT-HOG, LBP-HOG, SIFT-LBP-HOG) to further reduce the number of false classifications and enhance the accuracy of shortest path found by DP. In addition to these textural features we investigate the use of gradient magnituded, difference of gradient magnitude, normalized classification scores and combined gradient and classifier scores as the node cost choices for edge pixels or horizon classified edge pixels. We report our analysis on an extended data set comprising of 45 images with considerable viewpoint and scene changes, hand picked from Basalt Hills data set acquired by an out door robot's field navigation[23]. The rest of the paper is organized as follows: Section 2 details about Ahmad et al. [1] and Lie et al.[3], section 3 and 4 describe the details about various texture features and formalism about the various nodal costs that we consider. We conclude the paper in section 6 after listing and detailing the experiments and results in section 5.

2 Background

2.1 Lie et al. [3]

Lie et al. [3] formulated horizon detection problem as a multi-stage graph problem where edge pixels are used as graph nodes and links are established between these edge nodes belonging to adjacent stages. They dealt with edge discontinuties by introducing high cost dummy nodes based on a neighborhood search strategy. They assumed horizon line exists in the upper half of the image and hence a bias towards solutions lying in the upper half of the image is induced by initializing the nodes in the first and last stage of the graph according to the vertical positions of nodes. There are three major steps involved in Lie et al. approach i.e. edge detection, graph formulation and initialization and dynamic programming. First edge detection is applied on a given image to get a binary map $I(x, y)$ where each edge pixel represented by 1 and non-edge pixel by 0. This edge map is used to initialize the nodal costs of an M× N multistage graph $G(V, E, \Psi, \Phi)$ where $\Psi(i, j)$ is the cost associated with node i in stage j.

$$\Psi(i,j) = \begin{cases} l, & \text{if } I(x,y) = 1. \\ \infty, & \text{if } I(x,y) = 0. \end{cases} \tag{1}$$

Next, the links are established for each node i belonging to stage j with nodes in $j + 1$ within a vertical neighborhood defined by a user specified parameter δ. These link costs are associated with the absolute vertical disances between the nodes belonging to stages j and $j + 1$.

$$\Phi(i,k,j) = \begin{cases} |i-k|, & \text{if } I(i,j) = I(k,j+1) = 1 \\ & \text{and } |i-k| \leq \delta \\ \infty, & \text{otherwise.} \end{cases} \tag{2}$$

To deal with edge gaps; Lie et al. use a parameter tolerance-of-gap(tog) to specify a fanout search window for node i in stage j where an edge node is searched in case no edge node is found in stage $j + 1$ within the δ vertical neighborhood. Once an edge node k is discovered in $tog + \delta$ search window; the two edge nodes are connected by introducing high cost dummy nodes in between. Next, the bias towards horizon being present in upper half of image is encoded by forcing the nodes in the stages 1 and N to be initialized according to their vertical positions.

$$\Psi(i,j) = \begin{cases} (i+1)^2, & \text{if } j = 1 \text{ or } j = N \\ \Psi(i,j), & \text{otherwise.} \end{cases} \tag{3}$$

Finally, zero cost source and destination nodes s and t are introduced to the left of stage 1 and right of stage N. Zero cost links are established from source node to all the nodes in stage 1 and from all the nodes in stage N to node t. Dynamic programming is then applied on this formulated multi-stage graph to

find the shortest path extending from node s to t. The shortest path thus found confroms to the detected horizon line.

We show several step of Lie et al. for a synthetic image in figure 1 and to also highlight the problem with their underlying faulty assumption of horizon belonging to upper half of image. Figure 1-(a) shows the output of edge detection; black and white reflects a pixel being an edge pixel or not. Figure 1-(b) highlights a search window $tog+\delta$ for $i = 5$ node in $j = 5$ stage when $tog = 4$ and $\delta = 1$ are used as parameter choices. Figures 1-(c) and 1-(d) show two edge nodes being diecovered within the fanout search window $tog + \delta$ and then being connected to the node 5 in stage 5 by introduction of high cost dummy nodes. The bias towards solutions in the upper half is then induced according to equation 3 as shown by increasing intensity of stages 1 and N in figure 1-(e). For graph shown in figure 1-(e) there exist two paths of equal cost ignoring stages 1 and N. Howere due to the bias induced towards horizon being in upper half of image would reduce the cost of the upper path which would be found by the DP. It is probable that the upper right edge segment in figure 1-(a) was due to a clound and not part of horizon but ended up being detected as horizon part due to the bias while the edge segment in the lower right belongs to true horizon but was missed.

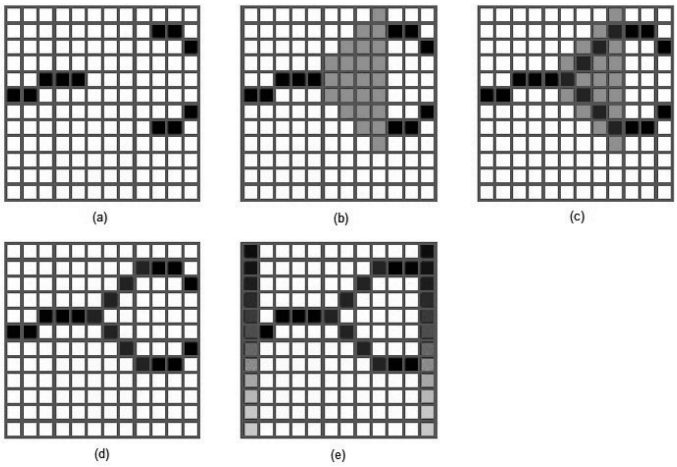

Fig. 1. Steps of horizon line detection approach by Lie et al.

2.2 Ahmad et al. [1]

The use of a trained classifier in this scnario is two fold. Not only it helps to reduce the number of non-horizon edges considerably but also reduces the chances of non-horizon edges being detected as part of the horizon line. Ahmad et al. [1] proposed a machine learning based horizon detection method where they

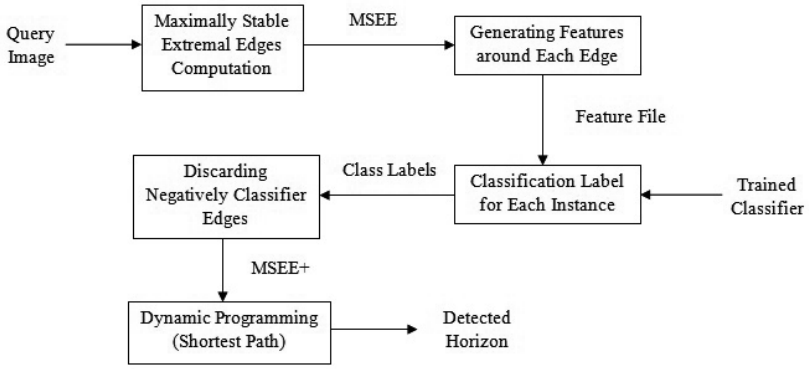

Fig. 2. Steps of horizon line detection approach by Ahmad et al.

train an SVM classifier based on SIFT features to reduce the number of edges found by an edge detector. Edge detection is the preliminary step to the horizon detection approach proposed by Lie et al. However unlike Lie et al. [3]; they first reduce the number of edges considerably and then formulate the reduced edge map as a multi-stage graph. Figure 2 describes various steps involved in Ahmad et al. approach. To train the classifier the ground truth horizons for the training images are marked manually. Then the positive key points are chosen uniformly from ground truth horizon locations from all training images. To generate the negative key points; they compute the Maximallsy Stable Extremal Edges Images (MSEE) by applying Canny edge detector with fixed σ and varied higher and lower thresholds. For a number of threshold choices; different binary edge maps are gnerated based on which MSEE image comprising of pixels which survived k different thresholds is computed. Mathematically,

$$E(x,y) = \begin{cases} 1, & \text{if } \sum_{i=1}^{N} I(x,y)_i > k. \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

where, I_i through I_N are the binary edge images generated for N different parameter choices, $I(x,y)_j$ is a pixel at x,y location in edge map j and E is the resultant Maximally Stabe Extremal Edges Image corresponding to the train image. In their results Ahmad et al. [1] have shown that MSEE reduces the number of edges considerably without harming the horizon edges. We also verify this for our extended data set in the experimental section.

Once the MSEE is computed; the negative key points are chosen randomly from the non-horizon edge locations from MSEE image.Ahmad et al. [1] then generate the SIFT features around these positive and negative key points and train an SVM classifier. This trained classifier is used for classifying the edges of a query image. On the testing side, MSEE image $E(x,y)$ is gnerated for a given query image according to equation 4. The edges in the MSEE image are

then classified as horizon or non-horizon edges by the trained classifier. The resuntant edge image comprising of only horizon classified edges is named E_+. If the classifier is assumed to be a binary function assigning 1/0 labels to the input edge pixel then mathematically E_+ can be written as,

$$E_+(x,y) = \begin{cases} 1, & \text{if } E(x,y) = 1 \,\&\, \text{Classifier}[E(x,y)] = 1. \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

Instead of using the output of edge detector; Ahmad et al. use this horizon classified edge MSEE image E_+ to formulate the multi-stage graph $G(V, E, \Psi, \Phi)$. So, the equations 2 and 3 are modified accordingly.

$$\Psi(i,j) = \begin{cases} l, & \text{if } E_+(x,y) = 1. \\ \infty, & \text{if } E_+(x,y) = 0. \end{cases} \tag{6}$$

$$\Phi(i,k,j) = \begin{cases} |i-k|, & \text{if } E_+(i,j) = E_+(k,j+1) = 1 \\ & \text{and } |i-k| \leq \delta \\ \infty, & \text{otherwise.} \end{cases} \tag{7}$$

Since, the number of candidate horizon edges are reduced considerably due to the use of MSEE and the trained classifier; Ahmad et al. do not enforce the bias towards horizon solutions in the upper half so equivalent to equation 4 is skipped in their formulation. However, any kind of gaps are filled following the conventional method. Next two nodes i.e. a source and a sink are added, essential links between them and nodes in stages 1 and N are established with zero cost and DP is used to find the horizon line.

3 Exploring Texture Features and Their Combinations

We have investigated three texture descriptors as feature choices to train the SVM classifier namely Scale Invariant Feature Transform(SIFT)[14], Local Binary Patterns(LBP)[13] and Histogram of Oriented Gradients(HOG)[15]. We also explore their combinations with each other and then all of them combined as feature choices for our classifier. We use the implementation of these descriptor available from vlfeat [16] which provides 128, 58 and 31 dimensional vectors for SIFT, LBP and HOG respectively. We investigate individual descriptors as well as their combinations as the feature choices to train the SVM classifier. The combinations are formed by mere concatenation of the descriptor vectors for each training and testing instance. The feature sizes for each combination : SIFT-LBP, SIFT-HOG, LBP-HOG and SIFT-LBP-HOG are 186, 159, 89 and 217 respectively. We have found SIFT-HOG combination as the best choice when compared with individual descriptors and other combinations as described in the results section.

4 Proposed Nodal Costs

In [3] Lie et al. proposed the use of edges where they use binary costs to encode the information in the multi-stage graph about a node being an edge pixel or not and then initialize the dummy nodes to high costs. Although Ahmad et al.[1] and Hung et al.[2] reduce the number of horizon candidate edges considerably by the use of MSEE and trained classifiers; they still use the same nodal costs as proposed by Lie et al. We show in our experiments that using only the information about pixels being edge or non-edge is not enough to initialize the nodal costs as it is possible for DP to choose falsely classified horizon edges as part of the solution. We propose the use of various nodal costs that provide further evidence about the positively classified edges for being true horizon edges.

4.1 Gradient Magnitude and Difference of Gradient Magnitude

We use the information due to gradient magnitudes and difference of gradient magnitudes to initialize the node costs of our multi-stage graph. Unlike Lie et al. and others who formulate the multi-stage graph based only on edges we propose here to make a dense multi-stage graph where each pixel would be a node of the graph and would be conneted to it neighbors in the next stage. However, the nodes are initialized according to the gradient information. As gradient magnitudes are used as an intermediate part of edge detection the DP should find a solution where the sum of the gradient magnitude is maximized but for continuity we should also enforce that the difference between the gradient magnitudes of two adjacent neighbors should be minimized. We should also note that gradient based approach does not involve any kind of training and is presented here to establish that using just the constant low and high costs for edge and non-edge pixels (horizon classified edges in case of Ahmad et al. and Hung et al.) are not enough to find the accurate horizons.

In gradient based approach; given a query image $Q(x, y)$ the gradient magnitude for each pixel of the image is computed. Mathematically,

$$\nabla(x, y) = \Gamma[Q(x, y)] \tag{8}$$

Where, Γ is the function which takes a gray scale image as an input and returns the corresponding gradient magnitude image ∇. Next, the difference of the gradient magnitude image is computed. Since, a node i in stage j can be connected to as many nodes as defined by the δ neighborhood; one should generate as many gradient difference images where the difference should be taken with the node in next stage to which the current node is being connected. The equation below shows the making of difference of gradient mask for connections at the same level.

$$d\nabla(i, j) = |\nabla(i, j) - \nabla(i, j + 1)| \tag{9}$$

Since, we want to maximize the gradient magnitude while minimizing the difference of gradient magnitude we normalize the magnitude and difference

images between 0 and 1. The nodal costs $\Psi(i,j)$ of the graph are then set as a weighted combination of these two images depending upon to which node in the next stage the current node is being connected; identified by the sub-script k in the equation below.

$$C_1(i,j) = \Psi(i,j) = w * d\nabla_k(i,j) + (1-w) * (1 - \nabla(i,j)) \tag{10}$$

where w is the weight assigned to the difference of magnitude and the gradien magnitude image; we use a value of 0.5 hence equally weighting both. Next, the links costs may be initialized the way in equation 7; however for our experiments we consider all the link weights to be equal and set them to zero since we are considering a small neighborhood i.e. $\delta = 1$.

4.2 SIFT+HOG Classified Edges

The 2nd formulation that we investigate is fairly similar to Ahmad et al. the only difference is that our classifier is trained based on SIFT-HOG as we would show in experiments section the SIFT-HOG choice outperforms all other feature choices. So eventually equation 5 through 7 are used to initialize the graph and set the node/link costs.

4.3 SIFT+HOG Classifier Scores

As described earlier using a fixed low cost for edge pixels provides only partial information and no confidence at all to compare two positively classified nodes where one might be misclassified. To enfoce this knowledge in our dynamic programming formulation we propose a two fold use of the classifier; first to distinguish between horizon and non-horizon edges as realised by equation 5 and second to provide a confidence about and edge pixel of horizon-ness. We normalize the raw scores provided by the classifier between 0 and 1 without using any thresholding. The node costs are then initialized by the actual classification scores instead of initializing all positively classified edges to fixed low cost. The equation 6 would be altered to reflect this information where Ω is realized as a classifier which returns a value between [0–1]. Since, we want to find a shortest path through DP we assume that the values have been reversed so a smaller value reflects an edge pixel is more probable to be a horizon pixel.

$$C_2(i,j) = \Psi(i,j) = \begin{cases} \Omega(E_+(x,y)), & \text{if } E_+(x,y) = 1. \\ \infty, & \text{if } E_+(x,y) = 0. \end{cases} \tag{11}$$

4.4 SIFT+HOG Classifier Scores+ Gradient Information

In this formulation we combine the classifier information with the gradient information. By fusing equations 10 and 11 we get a new initialization for the nodal costs.

$$\Psi(i,j) = \begin{cases} w_2 * C_2(x,y) + (1-w_2) * C_1(x,y), & \text{if } E_+(x,y) = 1. \\ \infty, & \text{if } E_+(x,y) = 0. \end{cases} \tag{12}$$

where, w_2 is a scalar and we use 0.5 value to weight both the scores equally.

5 Experiments and Results

5.1 Ground Truth and Importance of MSEE

Since, we want to compare the detected horizon lines by various approaches; we have manually created the ground truth for our data set which is comprised of 45 images hand picked from Basalt Hills data set. The images chosen have considerable viewpoint and scene changes and we make sure a full visible horizon exists in these images. Edge detection is used as the preliminary step for the manual labeling. The edges which belong to horizon line visually are kept for further processing. Whereas pixels are linked manually if there exist any gaps due to edge detection and become the part of ground truth horizon.

Following Ahmad et al. as a first step to our proposed feature analysis we compute the Maximally Stable Extremal Edge (MSEE) Images for all the images in our data set. We compare the number of edges survived after MSEE compared with the number of edges found by Canny edge detector of Matlab and see on an average a 66.37% reduction in the number of candidate horizon edges which are further reduced by the classifier.

5.2 Reduction in Horizon Candidate Edges due to Features and Classifier

We investigate various texture desciptors and their combinations as the feature choices to train our SVM classifier. We perform a 5-fold validation where for each fold the data set is divided into non-overlapping train (9 images) and test sets (36 images). Since, we have the ground truth horizons at our disposal we know which egdes belong to true horizon and which do not. Table 1 shows the percentage false positive and false negative errors averaged over the five folds of training and the respective standar deviations. Figure 3 shows a graphical view of the same information. Since, false negative error is of more importance we choose the classifier based on SIFT-HOG combination for further evaluation.

5.3 Best Nodal Cost

We compare our proposed formulations (section 4) againts the state of the art horizon detection method based on edges and DP i.e. Lie et al. [3]. To compare the detected horizon lines found by each method with the ground truth horizons, we compute an average pixelwise absolute error. Since, in all of our formulations we do

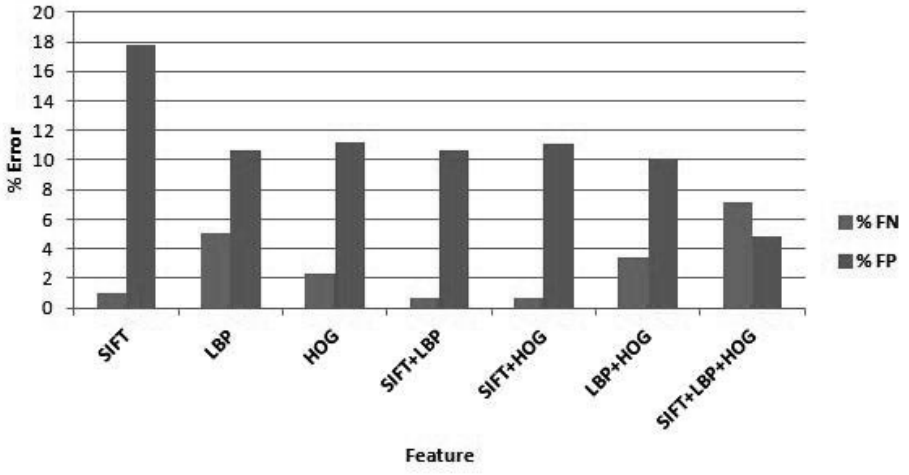

Fig. 3. Mean of % False Positive and False Negative Errors when various Features and their combinations used for training the SVM Classifier

not allow nodes to be connected within the same stage, there exists a one-to-one mapping between the pixels of the detected(d) horizon and the ground(g) truth horizon. The absolute average error for a detected horizon can be written as,

$$S = \frac{1}{N} \sum_{j=1}^{N} |P_{d(j)} - P_{g(j)}| \tag{13}$$

where $P_{d(j)}$ and $P_{g(j)}$ are the positions (rows) of the detected and true horizon pixels in column j and N is the number of columns in the test image. For each of our formulation and Lie at al. we computer the average and standard deviation over all images in the data set listed in the table 2. Clearly, SIFT+HOG

Table 1. Mean and standar deviations for % False Positive and False Negative Errors due to Various Features and their Combinations

Feature	%FN		%FP	
	Mean	Std. Dev.	Mean	Std. Dev.
SIFT	1.0224	0.7890	17.8090	5.6089
LBP	5.0332	8.3747	10.6366	8.0770
HOG	2.3285	2.3498	11.2331	5.6616
SIFT+LBP	0.6915	1.1827	10.6065	5.8949
SIFT+HOG	0.6624	0.7436	11.0801	5.1797
LBP+HOG	3.3647	3.6415	10.0737	6.394
SIFT+LBP+HOG	7.1887	8.1177	4.8302	4.0438

Scores outperforms all others strengthing our understanding that using only edge information is not enough for the nodal costs.

Table 2. Average absolute errors between the detected and ground truth horizons for various choices of Nodal Costs

Nodal Costs	Absolute Error	
	Mean	Std. Dev.
Lie et al. (Edges)	5.5548	9.4599
Gradient Info.	3.9908	6.3530
SIFT+HOG Edges	0.5783	1.0227
SIFT+HOG Scores	0.4124	0.8120
SIFT+HOG Scores + Gradient Info.	0.4358	0.8124

Figure 4 shows few sample test images from our data set with detected horizon lines overlaid (in red) when SIFT-HOG classifier score is used as nodal cost.

Fig. 4. Detected Horizon Lines using SIFT-HOG Classifier Score as Nodal Costs; highlighted in red

6 Conclusion

We have investigated various texture features and nodal costs for a recently proposed edge and DP based horizon line detection approach. We have established by experimental evidence that using the binary scores in DP formulation is not enough to find the true horizon line. Using the proposed nodal costs we were able to show that even gradient information contributes more that mere edge detection. Recently proposed approaches use the trained classifier to reduce the number of candidate horizon edges whereas we extend the use of classifier not only to classify the edges but to use the normalized classification scores as the nodal costs.

The current analysis is based on a relatively small data set comprising of 45 images only; which though have significant scene and viewpoint changes but come from the same geographical sight and are taken under same weather conditions. In future, we hope to extend and verify this analysis for a more challenging data set preferably mountain images with visible horizon collected from a search engine.

Acknowledgement. This material is based upon work supported by NASA EPSCoR under Cooperative Agreement No. NNX10AR89A.

References

1. Ahmad, T., Bebis, G., Regentova, E., Nefian, A.: A Machine Learning Approach to Horizon Line Detection using Local Features. In: Proceedings of 9th International Symposium on Visual Computing, ISVC (2013)
2. Hung, Y.-L., Su, C.-W., Chang, Y.-H., Chang, J.-C., Tyan, H.-R.: Skyline Localization for Mountain Images. In: Proceedings of International Conference on Multimedia and Expo, ICME (2013)
3. Lie, W.-N., Lin, T.C.-I., Lin, T.-C., Hung, K.-S.: A robust dynamic programming algorithm to extract skyline in images for navigation. Pattern Recognition Letters 26, 221–230 (2005)
4. Nasim, S., Boroujeni, S., Etemad, A., Whitehead, A.: Robust Horizon Detection Using Segmentation for UAV Applications. In: Proceedings of IEEE 2012 Ninth Conference on Computer and Robot Vision (2012)
5. de Croon, G.C.H.E., Remes, B.D.W., De Wagter, C., Ruijsink, R.: Sky Segmentation Approach to Obstacle Avoidance. In: IEEE Aerospace Conference (2011)
6. Ettinger, S.M., Nechyba, M.C., Ifju, P.G., Waszak, M.: Vision-Guided Flight Stability and Control for Micro Air Vehicles. In: Proceedings of International Conference on Intelligent Robots and Systems (IEEE/RSJ) (2002)
7. McGee, T.G., Sengupta, R., Hedrick, K.: Obstacle Detection for Small Autonomous Aircraft Using Sky Segmentation. In: Proceedings of International Conference on Robotics and Automation, ICRA (2005)
8. Thurrowgood, S., Soccol, D., Moore, R.J.D., Bland, D., Srinivasan, M.V.: A Vision Based System for Altitude Estimation of UAVs. In: Proceedings of International Conference on Intelligent Robots and Systems, IEEE/RSJ (2009)
9. Todorovic, S., Nechyba, M.C., Ifju, P.G.: Sky/Ground Modeling for Autonomous MAV Flight. In: Proceedings of International Conference on Robotics and Automation, ICRA (2003)
10. Gupta, V., Brennan, S.: Terrain Based Vehicle Orientation Estimation Combining Vision and Inertial Measurements. Journal of Field Robotics 25(3), 181–202 (2008)
11. Ho, N., Chakravarty, P.: Localization on Freeways using the Horizon Line Signature. In: Proceedings of International Conference on Robotics and Automation, ICRA (2014)
12. Dumble, S.J., Gibbens, P.W.: Efficient Terrain-Aided Visual Horizon Based Attitude Estimation and Localization. Journal of Intelligent and Robotic Systems (2014)
13. Ojala, T., Pietikainen, M., Maenpaa, T.: Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) 24(7), 971–987 (2002)
14. Lowe, D.G.: Distinctive Image Features from Scale-Invariant Keypoints. International Journal of Computer Vision (IJCV) 68(2), 91–110 (2004)
15. Dalal, N., Triggs, B.: Histograms of Oriented Gradients for Human Detection. In: Proceedings of Computer Vision and Pattern Recognition(CVPR) (2005)
16. `http://www.vlfeat.org/index.html`
17. Baatz, G., Saurer, O., Köser, K., Pollefeys, M.: Large Scale Visual Geo-Localization of Images in Mountainous Terrain. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part II. LNCS, vol. 7573, pp. 517–530. Springer, Heidelberg (2012)
18. Liu, W., Su, C.: Automatic Peak Recognition for Mountain Images. In: Advanced Technologies, Embedded and Multimedia for Human-centric Computing (2014)

19. Fefilatyev, S., Smarodzinava, V., Hall, L.O., Goldgof, D.B.: Horizon Detection Using Machine Learning Techniques. In: ICMLA, pp. 17–21 (2006)
20. Gershikov, E., Libe, T., Kosolapov, S.: Horizon Line Detection in Marine Images: Which Method to Choose? International Journal on Advances in Intelligent Systems 6(1-2), 79–88 (2013)
21. Kim, B.-J., Shin, J.-J., Nam, H.-J., Kim, J.-S.: Skyline Extraction using a Multistage Edge Filtering. World Academy of Science, Engineering and Technology 55 (2011)
22. Yazdanpanah, A.P., Regentova, E.E., Mandava, A.K., Ahmad, T., Bebis, G.: Sky segmentation by fusing clustering with neural networks. In: Bebis, G., et al. (eds.) ISVC 2013, Part II. LNCS, vol. 8034, pp. 663–672. Springer, Heidelberg (2013)
23. Nefian, A.V., Bouyssounouse, X., Edwards, L., Kim, T., Hand, E., Rhizor, J., Deans, M., Bebis, G., Fong, T.: Planetary Rover Localization within Orbital Maps. In: Proceedings of International Conference on Image Processing, ICIP (2014)