Semantic Segmentation of Aerial Images Using Fusion of Color and Texture Features

Mahdie Rezaeian[1], Rasoul Amirfattahi[1], Saeid Sadri[1]

[1]*Department of Electrical and Computer Engineering, Isfahan University of Technology, Isfahan, 84156-83111, Iran*

(Fax: +98-311-3912451)

Emails: m.rezaeian@ec.iut.ac.ir, {fattahi, sadri}@cc.iut.ac.ir

Abstract

This paper presents a semantic method for aerial image segmentation. Multi-class aerial images are often featured with large intra-class variations and inter-class similarities. Furthermore, shadows, reflections and changes in viewpoints, high and varying altitude and variability of natural scene pose serious problems for simultaneous segmentation. The main purpose of segmentation of aerial images is to make subsequent recognition phase straightforward. The present algorithm combines two challenging tasks of segmentation and classification in a manner that no extra recognition phase is needed. This algorithm is supposed to be part of a system which will be developed to automatically locate the appropriate site for *UAV* landing. With this perspective, we focused on segregating natural and man-made areas in aerial images. We compared different classifiers and explored the best set of features for this task. In addition, a certainty based method has been used for integrating color and texture descriptors in a more efficient way. The experimental results show the overall segmentation accuracy rate of 91.34%.

Keywords *aerial images, semantic segmentation, classification, local binary patterns, feature fusion, Artificial Neural Network, Support Vector Machine, Random Forest*

1 Introduction

Segmentation is the process of partitioning an image into non-overlapping meaningful regions such that each region is uniform based on a specific homogeneity measure and no union of any adjacent segments fulfills the homogeneity criterion (Blaschke, 2010). Most of the tasks based on processing of aerial images demand a full description of the scene. The main purpose of segmentation of aerial images is to make subsequent recognition phase straightforward. High and varying altitude, effects of shadow and reflection, varying natural scene, and changes in viewpoint pose a great challenge to the area of aerial image segmentation. Furthermore, most recognition methods benefit from structure information. However, even in high resolution aerial images, there is no enough detail and structure information. Therefore, common recognition methods are not directly applicable to aerial images. These considerations add to the complexity of the segmentation problem which is already yet to be fully resolved in the computer vision and image processing community. As a result, any attempt that can reduce the online computation demand while making the system robust to the above challenges is of significant importance. For this purpose, a semantic segmentation approach is proposed that incorporates invariant and robust descriptors to encode distinctive image information.

Semantic or class specific segmentation is the task of labeling image pixels (or areas) to a set of semantic classes. It combines two challenging tasks of segmentation and classification in a manner that no extra recognition phase is needed. By this method, not only different areas of the test image are segmented but also similar segments take the same label. Classification can be performed either unsupervised or supervised. In unsupervised or clustering techniques, samples in the feature space are not labeled. To distinguish different classes, samples are divided based on some similarity/ dissimilarity measure. Due to the variable number of classes and combination variety in different aerial images, unsupervised techniques may result in over- or under-segmentation. On the other hand, supervised approaches need a training phase prior to segmentation. During this phase, a set of input features is computed for each sample image of favorite classes. These features form a database of predefined classes. A label is assigned to each sample in the database which shows the class that the sample belongs to.

Finally, representative features of an unknown sample are compared with a database of a certain number of classes and the most likely label is computed.

Ojala and Pietikäinen (1999) used pixel-classification as the final step of their segmentation algorithm to improve the localization of the boundaries. Hu et al. (2005) applied the proposed method by Ojala, and Pietikäinen on seven aerial images and reported overall misclassification rate of 15.7%. They used an adaptive weighted combination of texture, intensity, and color features. Each image was segmented into four classes including water, residential area, wood and crop.

Permuter et al. (2006) segmented natural and man-made areas in aerial images on the basis of Gaussian mixture models (*GMM*s). They tested the performance of their method over a database including seven gray-scale aerial images with average classification accuracy of 85.2%.

Yang and Newsam (2008) evaluated Gabor texture features and Scale-Invariant Feature Transform (*SIFT*) descriptors for extracting 11 land cover classes. They applied these spatial features to maximum a posteriori (*MAP*) and support vector machine (*SVM*) classifiers. Their results are shown in table 1.

Xu et al. (2010) presented a Bag-of-Visual Words (*BOV*) representation for a four-class land-use segmentation problem. By means of a combination of spectral and texture features with *SVM* classifier they achieved the overall classification accuracy of 93.12% over 882 "single-class" images.

Kluckner et al. (2010) proposed a novel feature representation based on covariance matrices and Sigma Points. For accurate semantic classification into five classes, they applied multiple appearance cues including color, edge responses, and height information to multi-class random forest (*RF*) classifier. The experimental results of their method are given in table 1.

Nitze et al. (2012) compared the performance of machine learning techniques including Artificial Neural Network (ANN), Support Vector Machine (SVM) and Random Forest (RF) in the task of agricultural crop classification in remote sensing images. The best result they achieved was 88.1% overall accuracy for SVM. Their results are given in table 1.

Table 1 provides a brief literature review on class based segmentation of high resolution aerial images.

Table 1- brief literature review on class based segmentation of aerial images.

Author	Year	Method	Overall classification Accuracy Rate
Hu et al.	2005	*LBP*/C + saturation/hue distribution + grey-level histograms	84.29%
Permuter et al.	2006	energies of wavelet subbands and DCT regions + mean and covariance of the RGB and LAB values	85.2%
Yang et al.	2008	SIFT & Gabor + MAP & SVM as classifier	<table><tr><td></td><td>SIFT</td><td>Gabor</td></tr><tr><td>MAP</td><td>84.5%</td><td>73.9%</td></tr><tr><td>SVM</td><td>76.2%</td><td>89.8%</td></tr></table>
Xu et al.	2010	the mean and standard deviations of three spectral bands (i.e., RGB) + 48 texture features computed from 12 GLCMs + SVM	93.12%
Kluckner et al.	2010	covariance matrices and Sigma Points + randomized forest	<table><tr><td>building</td><td>water</td><td>grass</td><td>tree</td><td>street</td></tr><tr><td>92.7%</td><td>85.8%</td><td>94.4%</td><td>92.6%</td><td>95.3%</td></tr></table>
Nitze et al.	2012	five different vegetation indices including ground cover, NDVI, MTVI2, NDVIRE, MTCI	<table><tr><td>ANN</td><td>SVM</td><td>RF</td></tr><tr><td>87.1%</td><td>88.1%</td><td>87.4%</td></tr></table>

This paper provides: 1) semantic segmentation of aerial images, 2) a comparison between different learning algorithms and evaluation of texture features for semantic segmentation, and 3) the fusion of color and texture features for better segmentation.

The rest of the paper is organized as follows. In section 2 the proposed segmentation method is described. Different classifiers using color histogram are investigated on multi-class aerial images in section 3. The texture descriptors tested in this work are explained in section 4. Section 5 shows experimental results of the proposed segmentation algorithm using color and texture fusion on aerial images and final conclusion is presented in section 6.

2 Proposed method

Before detailing our approach, it is necessary to point out that most of previous works in aerial image segmentation area, such as those mentioned in the introduction, have used high quality and high resolution satellite images. Here, to investigate the performance of the proposed method, we assumed that proper images are available and applied our proposed method to color images obtained from Google Earth. However, in real application a preprocessing and image

restoration or enhancement step may be necessary. Fig 1 shows the block diagram of the proposed segmentation algorithm.

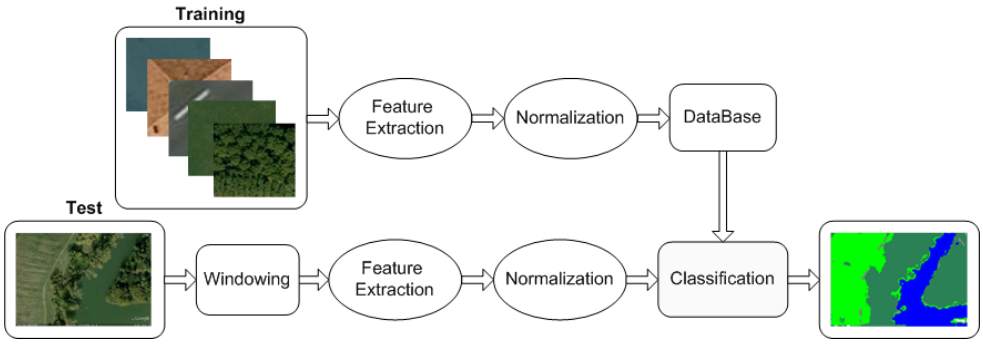

Fig 1- Proposed image segmentation scheme.

Different steps of training and testing the proposed algorithm are described in the following sections.

2.1 Training

In this paper, the problem is restricted to detecting 5 classes include "*tree*", "*grass*" and "*water body*" as natural and "*building*" and "*road*" as man-made sites. To train the classes, homogenous rectangular areas of different sizes on images of dataset were selected. Then each area is labeled to one of the predefined classes. We provided 160 images of each class and therefore in total we had 800 training images. After choosing training images for each class, a set of features is computed for them. To make comparison of different samples possible, it may be necessary to compute a suitable transformation like normalization for the feature vectors.

2.2 Test

In order to extract features from the test image, a simple strategy is to do measurements in a window. This operation has two stages:

- Feature extraction: in this stage the window is placed and moved over the image pixels and a set of descriptors is calculated for each window.
- Classification: the calculated descriptors of the window are compared with those in the dataset and the most likely label is assigned to the window.

The window size depends on image resolution and fineness of its texture. In general, window size should be large enough to reflect the local characteristics of the test image in feature vectors. On the other hand, large window causes increase in computations and loss of image details and boundary localization accuracy. For

more accurate segmentation, sweeping windows have overlap. Adjacent pixels in natural images generally have similar characteristics and belong to the same class. So, larger overlapping means more redundant computation. Besides, partly cover between sweeping windows will results in a more uniform and accurate segmentation. Briefly, windows overlap specifies computation time and classification accuracy. We heuristically subdivide the input image to overlapping windows of size 45×45 with sweeping step of 10 pixels. This means areas of 10×10 are assumed to have same label.

3 Comparison between different classifiers

There are different types of machine learning algorithms applicable in segmentation task. After introducing a brief description of some prominent cases of these, the segmentation results using these algorithms have been provided for comparison.

3.1 k-Nearest Neighbor (k-NN)

k-Nearest neighbor (*k-NN*) works based on the intuitive principle that in a feature space, the samples with similar characteristics generally exist close together. Because of possible outliers, a judgment solely based on the nearest neighbor may cause error. For achieving robustness against outliers, *k-NN* classifier finds the *k* closest feature samples in the training set and returns the most frequently class label within the *k*-subset (Nixon and Aguado, 2008). The two parameters of a *k-NN* classifier are *k* and a distance metric. *k* is set to 10 with respect to the number of the trained samples per class. The choice of the distance metric varies according to the features. The formulas for computation of the distance metrics used in this work are given in table 2. The parameter S is the length of the two vectors which their distance is computed.

Table 2- Computation formulas for some typical distance metrics

Distance metric	formula	
Minkowsi (L_k-norm) (Nixon and Aguado, 2008)	$D_k(A,B) = \left(\sum_{i=1}^{S} \left\lvert A_i - B_i \right\rvert^k\right)^{1/k}$	(1)
L_1-norm (Manhattan)	$D_1(A,B) = \sum_{i=1}^{S} \lvert A_i - B_i \rvert$	(2)

L_2-norm (Euclidean)	$D_2(A,B)=\sqrt{\sum_{i=1}^{S}(A_i-B_i)^2}$	(3)
Non-Intersection (Cha, 2001)	$D(A,B)=\sum_{i=1}^{S}A_i-\sum_{i=1}^{S}\min_i(A_i,B_i)$	(4)
Chi-square	$D(A,B)=\sum_{i=1}^{S}\frac{(A_i-B_i)^2}{A_i+B_i}$	(5)

3.2 Artificial Neural Network (ANN)

Neural networks consist of a number of simple processors or perceptrons (Ripley, 2008). An *ANN* is typically characterized by its architecture and its learning process. Here, we used a feed forward Multi-Layer-Perceptrons (MLP) consisting of one input layer, one hidden layer and one output layer. The number of hidden layers depending on the complexity of the problem and input features is selected via trial and error. The number of neurons comprising the input layer is equal to the sum of the lengths of the input features. According to our experience it is better to choose the number of hidden neurons to be the square root of the input neurons. The number of output neurons depends on the number of classes. For binary classification, a single output neuron is sufficient.

The goal of the training procedure is to find a set of weights that allow the network to perform correctly on the training examples. In the *MLP* network used in our study, the relationship between the input neurons (i_m) and the output neuron (o) is determined by:

$$o = f[\sum_n w_n g(\sum_m w_{nm} i_m + \theta_{in}) + \theta_{hid}] \quad (6)$$

The activation function for both the hidden and output layers is sigmoid function: $f(x)=g(x)=1/1+e^{-x}$. The activation function defines the output of neurons in terms of their weighted inputs. In equation (6) w_n is the weight from the nth hidden neuron to the output neuron, w_{nm} is the weight from mth input neuron to the nth hidden neuron, θ_{in} and θ_{hid} are the input and hidden biases, respectively. Network weights are iteratively adjusted and computed based on back-propagation (*BP*) with momentum terms, as follows:

$$\Delta w_{jk}(t+1) = \alpha \delta_k Z_j + \mu \Delta w_{jk}(t) \quad (7)$$

Where $Z_j = g(\theta_j + \sum_k w_{jk} x_k)$, x_k is the activity level of the kth neuron in the previous layer and w_{jk} is the weight of the connection between the jth and kth neurons. δ_k is the error between the desired and actual *ANN* output value. α is the learning rate, μ is the momentum and t is the number of iterations. The momentum term determines the effect of past weight changes on the current weight update. *BP* is a gradient descent method and can get stuck in local minima. The momentum term prevents the network from trapping into local minima and speeds up the convergence of the network. In this study, the learning rate and momentum were optimized through trial and error and the weights were randomly initialized within [-1,1].

3.3 Support Vector Machine (SVM)

In this method, data is transformed to a P-dimensional vector using a kernel function so that it becomes separable using a P-1dimensional hyper plane. There are many possible hyper planes but, a rational choice is to choose the hyper plane creating maximum margin between the separating hyper plane and the samples on either side of it (Steinwart and Christmann, 2008). *SVM* algorithm originally solves binary classification problems. In this work, generalization to multi-class classification is accomplished by training multiple one-against-the-others classifiers. The kernel function is *homogeneous polynomial* of degree 3 and the soft margin constant which determines the upper bound on the Lagrange multipliers is 1000. Interested readers are referred to (Taylor and Cristianini, 2000) for more details on *SVM*.

3.4 Random Forest (RF)

Random forest is a combination of decision trees such that each tree is formed based on a set of random decision functions selected independently. In the training phase, the distribution of different classes for each tree is computed, then in the test phase the class that is the mode of the class's output by individual trees is selected for assigning label to a new sample (Breiman, 2001). Each node in a tree (except leaf nodes) divides the data into two disjoint subsets. The decision function used in this study is a simple comparison with a normally distributed

random threshold which is generated between the minimum and maximum values of the trained features.

In the training phase, each random split function is scored using the Shannon entropy (Moosmann et al., 2008). Shannon entropy quantifies the homogeneity of the labeled samples in the child nodes:

$$H = -\sum_{c} \frac{n_c}{N} \log \frac{n_c}{N} \tag{8}$$

Where N is the number of the samples passing through the current node and n_c is the number of samples among the N inputs belonging to class c. For perfectly homogeneous data containing only a single class, the entropy is 0. Therefore, efficient decision rules can be selected based on the condition of minimum entropy. In our tests, the number of the tress is 100.

3.5 Comparison of Classifiers

In this section, the four abovementioned classifiers are evaluated for semantic segmentation using the histogram of RGB components as the color descriptor. The feature vector is constructed by concatenating three histograms of three color channels in RGB space. Each histogram has 32 bins and the color descriptor is a vector of size 3×32=96 in length. As mentioned before, the parameters of each classifier are tuned to achieve the best result. The results are reported from average over 5 runs (except for k-NN). Parameters were kept fixed for all the experiments.

Table 3 shows the confusion matrix of the pixel-wise semantic labeling. The diagonal elements of this matrix show the correct per-class semantic segmentation rates. Other elements show the "inter- class" misclassification rate. In addition to the confusion matrix, the quantitative segmentation results of binary segmentation are given in fig 2. In the binary case, the segmentation is considered as the problem of segregating man-made and natural areas. In fig 2 error type I is the percentage of image pixels belonging to building or road classes and labeled as *natural* and error type II pertains to those pixels of the three *natural* classes labeled as *man-made*.

The segmentation errors are calculated over 25 aerial images. These values are calculated based on the comparison between manually generated ground-truth and the computerized results.

Table 3- the comparison of segmentation error between four classifiers using RGB histograms.

		building	grass	road	tree	water
K-NN	building	76.80%	5.90%	6.90%	9.20%	1.20%
	grass	1.20%	81.30%	4.90%	12.60%	0
	road	2.70%	2.80%	81.20%	12.40%	0.90%
	Tree	3.10%	4.30%	3.30%	87.90%	1.40%
	water	0.60%	9.90%	1.30%	4.60%	83.60%
ANN	building	78.20%	0.90%	7.50%	13.40%	0.00%
	grass	1.30%	74.20%	7.10%	17.40%	0
	road	2.00%	2.00%	82.00%	12.90%	1.10%
	Tree	3.40%	5.00%	3.90%	86.20%	1.50%
	water	0.00%	13.10%	1.80%	5.10%	80.00%
SVM	building	77.90%	0.00%	3.30%	18.80%	0.00%
	grass	2.00%	63.70%	4.40%	29.90%	0
	road	4.70%	1.20%	76.30%	16.90%	0.90%
	tree	3.40%	3.20%	2.50%	89.70%	1.20%
	water	1.40%	13.10%	0.70%	5.90%	78.90%
RF	building	46.90%	0.20%	10.90%	42.00%	0.00%
	grass	0.90%	64.20%	8.90%	26.00%	0
	road	1.30%	0.30%	55.70%	31.70%	11.00%
	tree	0.60%	3.70%	2.90%	92.60%	0.20%
	water	0.50%	7.20%	31.30%	8.80%	52.20%

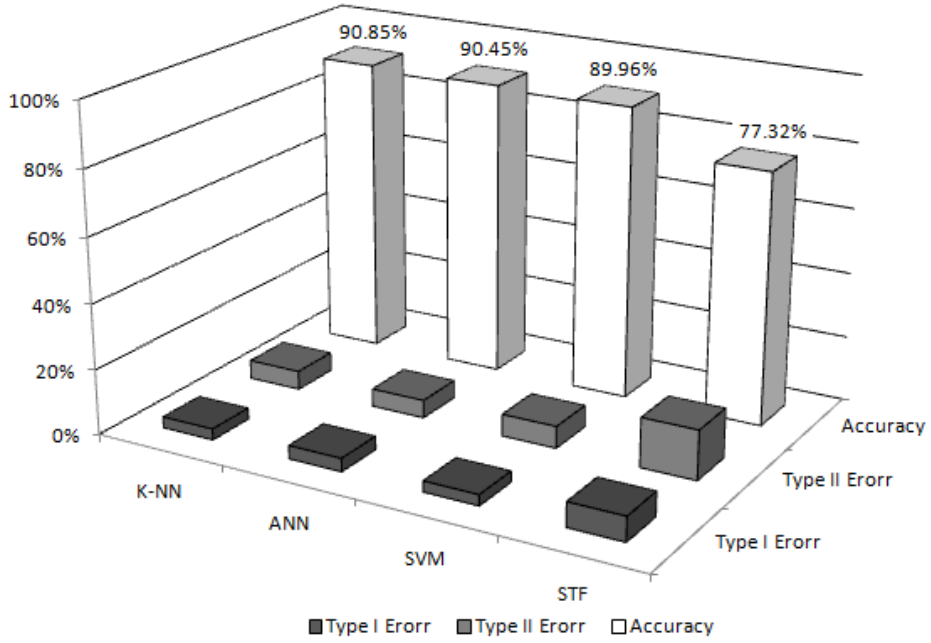

Fig 2. Comparing classifiers using RGB histograms

As shown in table 3 and fig 2, among the four classifiers, *k-NN* classifier yielded the best result. *ANN* and *SVM* exhibited nearly the same mean overall accuracies as *k-NN* and *RF* produced the worst results.

In the next step, to improve the segmentation accuracy we took advantage of texture information.

4 Texture description

Texture descriptors are values and arrays that encode important distinctive information about a texture area. Texture descriptors can be applied to a rectangular or alternatively freeform image area. In an ideal case, they should be compact and invariant to changes in viewpoint, rotation angle, illumination and scale.

Among different available texture descriptors, some variations of Local Binary Patterns (*LBP*) have the advantage of being invariant to rotation, small scale changes and monotonic changes in intensity. *LBP* can be very compact, easily computed and compared against. All these characteristics make *LBP* a perfect choice for our application.

Since the segmentation algorithm presented in this work relies on *LBP* to describe texture, a brief description of *LBP* and its variations is presented in the following subsections.

4.1 Local Binary Pattern

To explain *LBP* descriptors we first describe *LBP* codes. *LBP* codes are defined for each pixel in the area which is to be described. The *LBP* code for each pixel is defined over a circular symmetric neighborhood about the pixel for which the *LBP* code is to be computed. For each pixel on the circular neighborhood of size *P* and radius *R*, a *LBP* bit is assigned. This results in a *P* bit binary *LBP* code which is achieved by thresholding the intensity value of each pixel on the circular neighborhood at the value of the central pixel. If the value of the neighboring pixel is less than the central pixel, its *LBP* bit code is assigned to be zero. Otherwise the *LBP* bit code is assigned to be unity. This procedure is illustrated in fig 3.

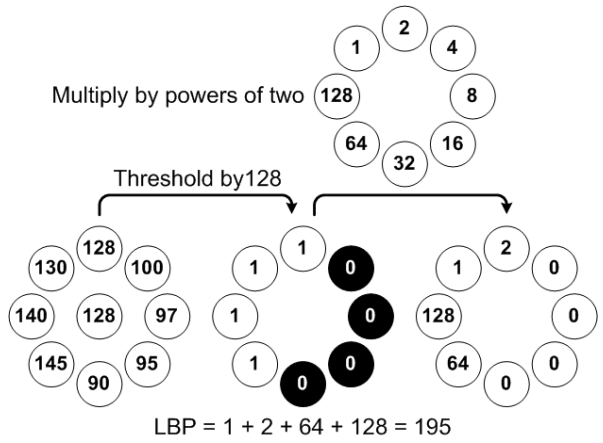

Fig 3- *LBP* code

This procedure is performed for every pixel in the area (for example rectangular window). This results in a *LBP* image. Then the histogram of *LBP* codes for this window is computed and considered as a texture descriptor for this area.

As discussed above, *LBP* is defined for a circular neighborhood of size *P*. If *P* does not divide 360°, for those samples which are not at the center of pixels the gray values are estimated by interpolation.

4.2 Rotation invariance

The *LBP* patterns obtained in the previous step are not rotation invariant. This is because a rotation in the texture results in a circular shift in the *LBP* histogram. To make *LBP* rotation invariant, one idea is to iteratively apply a circular bit-wise right shift on the binary code and choose the minimum code. This is further explained in fig 4. Since this procedure alters *LBP* codes in the pixel level, it makes *LBP* descriptors both locally and globally rotation invariant. We desire a texture descriptor to be globally rotation invariant. However, local rotation invariance is undesired as it disregards local orientation of texture. This issue will later be further discussed in section 4.6.

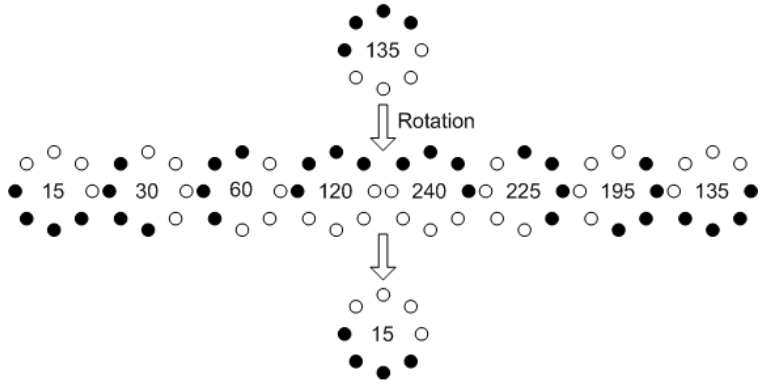

Fig 4- Rotation invariant *LBP* by rotating neighborhoods to their minimum value

4.3 Compactness

It is a reasonable idea to give more importance to those *LBP* patterns which are seen in the real word more frequently. Ojala et al. (2002) showed that more than 94% of *LBP* patterns seen in the real world are *"uniform"*. By *uniform LBP codes*, they mean those binary patterns that have only up to two transitions from zero to one or vice versa. *Non-uniform LBP codes*, however, have more than two transitions from zero-bits to one-bits or vice versa. Fig 5 demonstrates this definition.

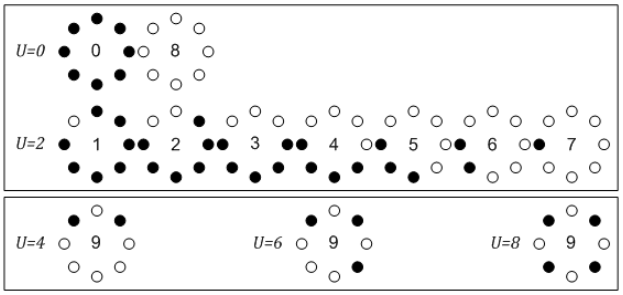

Fig 5- examples of uniform and non-uniform patterns. The corresponding code for each uniform rotation invariant pattern is equal to the number of ones in the pattern. All other patterns assigned to a single code.

All non-uniform *LBP* patterns are assigned a single *LBP* code. Therefore, the assigned code to uniform rotation invariant *LBP* patterns can be simply the number of one-bits in the pattern. We refer to this way of encoding *LBP* patterns that takes advantage of dividing *LBP* patterns to uniform and non-uniform ones as "LBP^{riu}". This makes the *LBP* code to be more compact and removes extra nearly zero bins in the *LBP* histogram. Although, it reduces the number of bins in the histogram, it increases the efficiency of *LBP* descriptor and elevates the need for histogram quantization.

4.4 Scale Invariance

LBP operator can be performed in different scales. This means by varying the parameters *P* and *R*, we can have different *LBP* patterns which encode texture information in different scales. Histograms of such *LBP* operators are combined to have a multi-resolution *LBP* descriptor. In fact, if the metrics used to compare *LBP* histograms have additive property, then the histograms can be safely concatenated to model the joint distribution of "assumingly" independent texture events (Mäenpää, 2003).

4.5 Adding Contrast to *LBP*

Although contrast is an important quality of texture, *LBP* operator ignores the values of gray level differences. Adding contrast information can improve segmentation accuracy. Indeed, texture can be considered as a two-dimensional phenomenon which is characterized by two qualities i.e. spatial structure or pattern and contrast (Ojala and Pietikäinen, 1999). Pattern is independent from grayscale but contrast is not; in addition, contrast does not change with rotation however pattern does. In view of that, these measures complement each other usefully.

Local contrast could be measured in a circular symmetric neighborhood like *LBP* (Ojala et al., 2002):

$$VAR_{P,R} = \frac{1}{P}\sum_{p=0}^{P-1}(g_p - \mu)^2 \qquad \mu = \frac{1}{p}\sum_{p=0}^{P-1} g_p \tag{9}$$

Where g_p refers to gray level of neighboring pixels. Although there is no any restriction for calculating *LBP* and *VAR* in different neighborhoods and choosing different (P,R), usually $P_1 = P_2$ and $R_1 = R_2$ is chosen.

Joint distribution of *LBP* and *VAR* is a powerful tool for texture analysis. But according to the equation (9) $VAR_{P,R}$ is a continuous quantity and should be quantized. In order to achieve optimum resolution, a comprehensive and numerous set of training images is needed to estimate the range of $VAR_{P,R}$ variations. Then for dividing this range into *N* equal parts, threshold values should be calculated. However, quantization with this method has three limitations. First, a training stage is necessary to determine threshold values. Second, because different classes may have very different contrast, quantization depends on training samples. Finally, third limitation is that finding optimal *N* based on feature vector size and the discrimination of the classes is difficult. If the number of quantization levels (*N*) is small, $VAR_{P,R}$ will be useless and classes will not separated correctly. Conversely, choosing large *N* causes histogram instability and increasing size of the feature vector.

To overcome these limitations Guo et al. (2010) proposed a simple yet effective method for combining contrast to local *LBP* histograms. They use $VAR_{P,R}$ as an adaptive weight for calculating the histogram of *LBP*. This feature which is named *LBPV* is calculated as follows (Guo et al., 2010):

$$LBPV_{P,R}(k)\Big|_{k\in[0,K]} = \sum_x \sum_y w(LBP_{P,R}(x,y),k)$$
$$w(LBP_{P,R}(x,y),k) = \begin{cases} VAR_{P,R}(x,y), & LBP_{P,R}(x,y) = k \\ 0 & otherwise \end{cases} \quad (10)$$

Consequently, *LBPV* is a simple representation of *LBP* / *VAR* two-dimensional distribution which is smaller and does not require quantization of *VAR* values.

4.6 Local Binary Pattern Histogram Fourier (*LBP-HF*)

Ahonen et al. (2009) introduced a rotation invariant texture descriptor named Local Binary Pattern Histogram Fourier (*LBP_HF*) which is derived from the magnitude of discrete Fourier transform of uniform *LBP* histograms. Unlike the earlier local rotation invariant features discussed in section 4.2, the rotation invariance of *LBP-HF* descriptor is attained globally. This means that the descriptor is invariant against rotations of the whole image but at the same time, if only some parts of the image are rotated the descriptor will differ. Details on how to calculate the descriptor are given in (Guo et al., 2010).

4.7 Evaluating texture descriptors and similarity measures

In this section we evaluate different variations of *LBP* and determine the most suitable distance measure for each descriptor.

The following texture descriptors were applied to the training images:

- $LBP^{riu}_{8,1}$ (applied on gray scale image)
- $LBP^{riu}_{(8,1)+(16,2)}$ (Multi-scale *LBP* applied on gray scale image)
- $LBP^{riu}_{(8,1)+(16,2)}$ (applied on RGB channels)
- $LBP^{riu}_{(8,1)+(16,2)}$ (applied on HS channels)
- $LBPV$ (applied on gray scale image)
- $LBP-HF$ (applied on gray scale image)

In the case of *LBPV* and *LBP-HF* descriptors, histograms of "*uniform LBP*" codes were computed in (8,1) and (16,2) neighborhoods to achieve scale invariance.

To choose the most suitable descriptor among the above descriptors, we have evaluated them in identifying the right label for a test set of single class homogenous textured images (275 single class test images; that is 55 images per class) using *K-NN* classifier. For each texture descriptor, a number of different distance metrics are examined and the most effective one is selected. These metrics include *L1-Norm*, *L2-Norm* (Nixon and Aguado, 2008), *histogram intersection* (Cha, 2001) and *Chi square* distance.

The results of applying each texture descriptor with the abovementioned distances are given in table 4.

Table 4-comparison between classification error rates for different versions of *LBP* with 4 distance metrics using *k-NN* classifier.

Texture Feature	Feature vector length	L1-norm	L2-norm	Chi-square	Histogram intersection
$LBP^{riu}_{(8,1)}$	10	20.4%	19.6%	18.5%	20.4%
$LBP^{riu}_{(8,1)+(16,2)}$ (Multi scale)	28	14.9%	14.9%	13.8%	14.9%
$LBP^{riu}_{(8,1)+(16,2)}$ (RGB channels)	84	14.2%	13.1%	13.1%	14.2%
$LBP^{riu}_{(8,1)+(16,2)}$ (HS channels)	56	21.1%	21.4%	19.6%	21.1%
$LBPV$	302	22.5%	31.3%	22.2%	22.5%
$LBP-HF$	176	8%	**6.9%**	7.6%	33.4%

As shown in table 4, the combination of *LBP-HF* with *L2-Norm* distance metric outperforms the other cases. *LBP-HF* is invariant against global rotations of input image. This characteristic is very important in segmentation of aerial images, however, the length of the feature vector should also be considered.

It is important to note that *K-NN* classifier can be replaced with any other classifier or machine learning approach such as Random forests, AdaBoost or *SVM*. Since *K-NN* has been sufficiently capable of classifying *LBP-HF* descriptors, we avoided the use of more complex classifiers. As a result, the success of algorithm can be attributed to the algorithm itself and not the complexity of the classifier.

4.8 Color and texture fusion

We tested our semantic segmentation algorithm on aerial images using the combination of color and texture descriptors. In view of the fact that it is unlikely that both color and texture descriptors make the same mistake, they can correct each other. As shown in table 4 extracting *LBP* descriptors of color channels is

not an efficient way for using color information. Color and texture features fusion can be carried out by simply connecting two feature vectors to each other. The other option is to classify color and texture separately and then evaluate the accuracy of each classifier to make final decision. This way, it is possible to set the parameter for each classifier independently in order to achieve the optimum result.

To each area of the test image, the color descriptor (a vector of size 96 constructed from 32-bin-histogram of R, G and B channels) and the texture descriptor of our choice, (here *LBP-HF*) are applied. The decision of using *LBP-HF* among all different variations of *LBP* is based on the evaluation study explained in section 4.74.7. The experiments show that the most successful combination is *LBP-HF* and *L2-Norm*. The pool of *LBP-HF* descriptors for the samples of the five classes accompanied with their semantic label is fed to a *K-NN* classifier. For the color classifier the similarity measure is set to *histogram intersection*. For *K-NN* classifier the maximum number of agreed votes among *K* neighboring samples in the feature space is used as a measure for accuracy. The process of how to make final decision is shown in fig 6. The *fusion* weights of *color and texture* are assigned equal values.

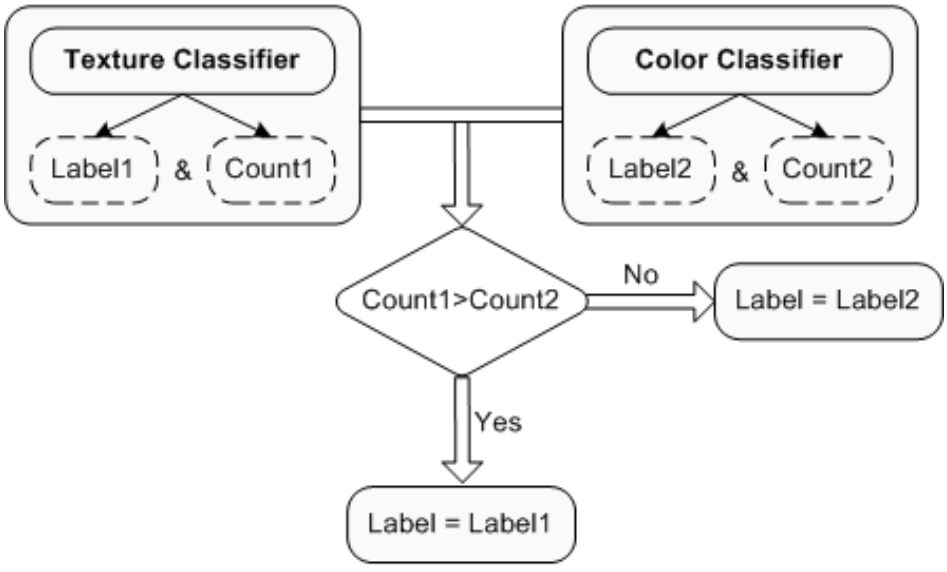

Fig 6- the fusion of color and texture descriptors in classifier level. Label and Count are the final label and vote count of each classifier respectively.

5 Results and future directions

As fig 7 shows, in spite of our limitations for gathering an efficient texture database, *LBP-HF* enhanced the segmentation results. In order to display segmentation result graphically buildings are shown with white, road with gray, water with blue, grass with green and trees with dark green.

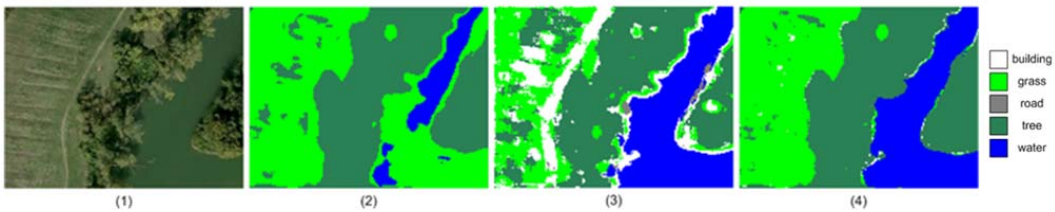

Fig 7- (1) original image (2) segmentation result using RGB histograms (3) segmentation result using LBP-HF (4) segmentation result using color and texture fusion. The texture classifier compensates the color classifier weakness in classifying water and the color classifier corrects texture classifier error in classification of grass and tree classes.

Fig 8 shows a number of sample results.

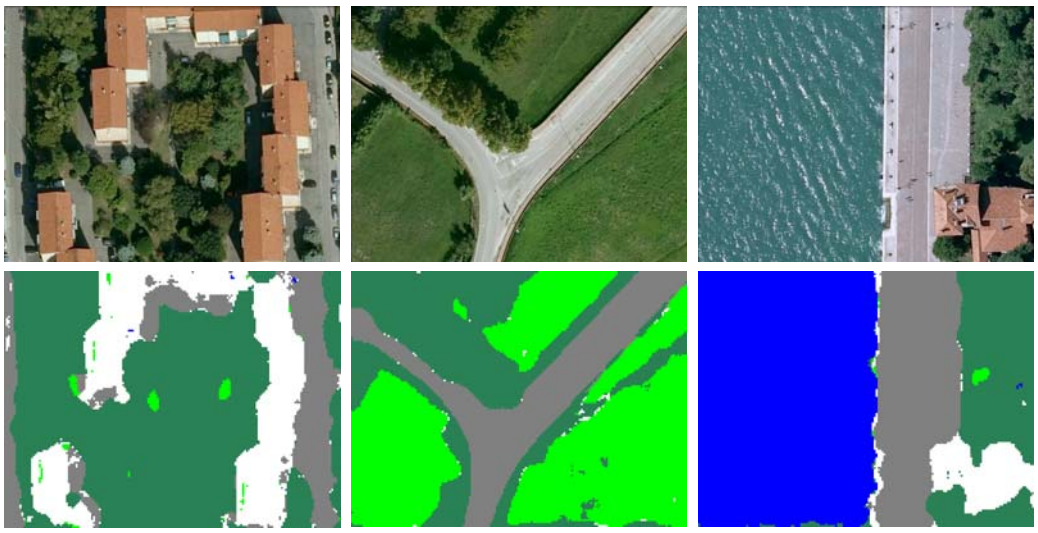

Fig 8- input image (top row) and segmentation result (bottom row).

The results obtained from testing the algorithm over 25 aerial images (considerably bigger test set than previous works outlined in Table 1) are summarized in table 5.

An important aspect of automatic site selection for Unmanned Aerial Vehicle (*UAV*) landing application is that some classes are safer to be misclassified to than other classes. For example, it is very dangerous to identify buildings as grass. This is because grass is a safe area for landing while buildings are unsafe areas. As a result, a more meaningful classification rate is computed as shown in the right half

of table 5. This classification is based on a two class labeling: safe (*water*, *grass*, and *tree*) and unsafe (*building*, *road*) areas.

Table 5- segmentation error rate over 25 aerial images using two *K-NN* classifiers for color and texture.

Confusion matrix						binary pixel-level labeling error			
						interclass		Intra-class	total
						Unsafe to safe	Safe to unsafe		
	building	grass	road	tree	water	4.74%	3.92%	5.24%	**13.89%**
building	83.4%	1.3%	6.7%	8.6%	0				
grass	0.7%	84.8%	5.9%	8.6%	0				
road	2.7%	4.1%	82.1%	10.4%	0.7%				
Tree	3.9%	2.6%	3.8%	87.5%	2.2%				
water	0	0	1.8%	4.9%	93.3%				

One of the major difficulties during experiments was the collection of ground truth for testifying experimental results. We collect ground truth by hand. Manually examining high resolution images was tiresome and inevitably involved imprecision. Our experimental results would not be comparable with previous works mainly because of basic differences existing between images in terms of resolution, lighting and scene complexity. More importantly, the number of images greatly affected classification accuracy. In addition, there are a variety of methods for evaluating results which makes direct comparisons difficult. The only conclusion from browsing similar case studies of automatic classification of aerial images is that our system performed well and confirms that the proposed system can be incorporated into industrial systems for the automated analysis of similar images.

The comparison between total error rates in table 3 and table 5 demonstrates that adding texture information to semantic segmentation has improved average segmentation accuracy by 2.87%. Intra-class misclassification accounts for 37.7% of the total error. Regarding the fact that there is no any meaningful distinction between safe classes or unsafe classes, this part of error can be discarded. Using the proposed method we could detect unsafe sites with 95.3% and safe sites with 96.1% accuracy (Note that we compute the error through pixel by pixel matching). The specificity and sensitivity indices of the method for detection of unsafe sites are 91.13% and 94.28% respectively. The results of detection of

unsafe sites obtained by the proposed method are given on a Pie chart in Fig. 9 to be compared graphically.

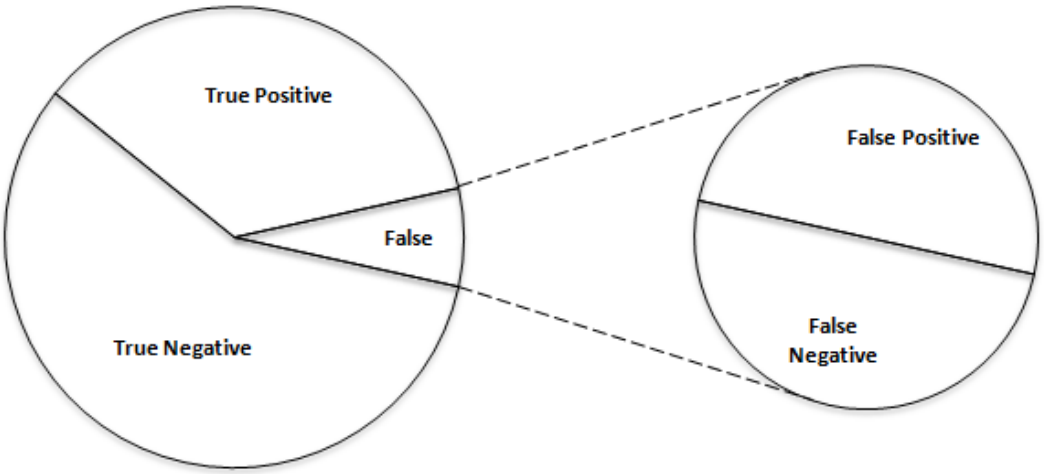

Fig 9- unsafe regions detection results obtained by the proposed method over a database of 25 multi-class images

This paper presents a semantic segmentation algorithm for aerial image understanding. Because of the nature of aerial images, features must be invariant to rotation and scale. In this paper two separate *K-NN* classifiers recognize image texture and color. Local texture characteristics of the gray image and local color histograms are calculated and classified individually. Final segmentation is obtained by evaluating the certainty with each classifier (color or texture) independently. Employing two classifiers is motivated by this observation that it is unlikely that both classifiers make mistake in the same case. Therefore, it is possible to correct errors made by each other. In aerial images, like most natural images neighboring pixels usually have similar characteristics. Based on this quality, calculations can be reduced and better segmentation can be obtained. Accordingly, classification result of each feature vector is assigned to a group of adjacent pixels (a patch) instead of labeling each pixel. Proposed method, which is applicable to any type of color-texture images, is simple and rapid. The other outcome of comparison between obtained results is that proposed method for fusing color and texture information in classifier level is more efficient than applying *LBP* operator on color channels. The main advantage of the proposed method is that it facilitates using different clues and fusing them. Although, the histogram of RGB channels was a distinctive descriptor for our database, we used a texture descriptor to show how different descriptors can be fused easily. The error rate of the proposed algorithm is small and segmentation results for aerial

images are visually acceptable. The majority of the segmentation error pertains to intra-class misclassification such as when grass mixes up with tree. Because we do not consider specific distinction between the two risk classes or three safe classes, this part of segmentation error is not very important. In addition, the comparison of the obtained results by similar works given in introduction section confirms the effectiveness of the proposed method especially that the number of the images which the algorithm is investigated on them is considerably more. However, basic differences existing between these works in terms of image databases and the methods for reporting results make direct comparisons difficult.
In our future work, we will develop the proposed algorithm for automatic landing site selection for *UAV* forced landing. In the test stage we would use a super-pixel partitioning algorithm instead of rectangular windows.

References

Ahonen, T., Matas, J., He, C. and Pietikäinen, M., 2009. Rotation Invariant Image Description with Local Binary Pattern Histogram Fourier Features. Image Analysis, SCIA 2009 Proceedings, Lecture Notes in Computer Science 5575, pp. 61-70.

Blaschke, T., 2010. Object based image analysis for remote sensing. ISPRS Journal of Photogrammetry and Remote Sensing, 65(1), pp. 2–16.

Breiman, L., 2001. Random Forests. Meachine Learning, 45(1), pp. 5–32.

Cha, S.H., 2001. Use of Distance Measures in Handwriting Analysis. PHD Dissertation, Graduate School of the State University of New York at Buffalo.

Guo, Z., Zhang, L., Zhang, D., 2010. Rotation Invariant Texture Classification Using *LBP* Variance (*LBPV*) with Global Matching. Pattern Recognition, 43, pp.706–719.

Hu, X., Tao, C.V., Prenzel, B., 2005. Automatic Segmentation of High-Resolution Satellite Imagery by Integrating Texture, Intensity, and Color Features. Photogrammetric Engineering & Remote Sensing, 71(12), pp. 1399–1406.

Kluckner, S., Mauthner, T., Roth, P.M., and Bischof, H., 2010. Semantic Classification in Aerial Imagery by Integrating Appearance and Height Information. Proceedings 9th Asian Conference on Computer Vision (*Computer Vision–ACCV 2009)*, Springer Berlin Heidelberg, pp. 477-488.

Mäenpää, T., 2003. The Local Binary Pattern Approach to Texture Analysis: Extensions and Applications, Dissertation, Oulun yliopisto.

Moosmann, F., Nowak, E., and Jurie, F., 2008. Randomized clustering forests for image classification. Pattern Analysis and Machine Intelligence, *IEEE Transactions on* 30.9, pp.1632-1646.

Nitze, I., Schulthess, U., Asche, H., 2012. Comparison of machine learning algorithms random forest, artificial neural network and support vector machine to maximum likelihood for supervised crop type classification. Proceedings of the 4th GEOBIA, Brazil.

Nixon, M.S. and Aguado, A.S., 2008. Feature Extraction and Image Processing, 2nd edition, Academic Press is an imprint of Elsevier, Oxford.

Ojala, T., and Pietikäinen, M., 1999. Unsupervised Texture Segmentation Using Feature Distributions. Pattern Recognition, 32, pp.477–486.

Ojala, T., Pietikäinen, M. and Mäenpää, T., 2002. Multiresolution Gray Scale and Rotation Invariant Texture Analysis with Local Binary Patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7), pp.971– 987.

Permuter, H., Francos, J., Jermyn, I., 2006. A study of Gaussian mixture models of color and texture features for image classification and segmentation. Pattern Recognition, 39, pp. 695–706.

Ripley, B.D., 2008. Pattern recognition and neural networks. Cambridge university press.

Steinwart, I., and Christmann, A., 2008. Support vector machines. Springerverlag, New York.

Taylor, J., Cristianini, N., 2000. Support Vector Machines and other kernel-based learning methods, Cambridge University Press.

Xu, S., Fang, T., Li, D., Wang, S., 2010. Object classification of aerial images with bag-of-visual words. IEEE Geosci. Remote Sens. Lett., 7, pp. 366-370.

Yang, Y., Newsam, S., 2008. Comparing SIFT descriptors and Gabor texture features for classification of remote sensed imagery. Proceedings of the 15th IEEE International Conference on Image Processing, *2008. ICIP 2008,* San Diego, USA, October 12-15, pp. 1852-1855.