Proceedings of the 2003 IEEE/RSJ
Intl. Conference on Intelligent Robots and Systems
Las Vegas, Nevada · October 2003

Recognition of Traversable Areas for Mobile Robotic Navigation in Outdoor Environments

James C. Davidson and Seth A. Hutchinson
jcdavid@coltrane.ai.uiuc.edu, seth@uiuc.edu

Dept. of Electrical and Computer Engineering
The Beckman Institute for Advanced Science and Technology
University of Illinois at Urbana-Champaign
Urbana, IL USA

***Abstract*— In this paper we consider the problem of automatically determining whether regions in an outdoor environment can be traversed by a mobile robot. We propose a two-level classifier that uses data from a single color image to make this determination. At the low level, we have implemented three classifiers based on color histograms, directional filters and local binary patterns. The outputs of these low level classifiers are combined using a voting scheme that weights the results of each classifier using an estimate of its error probability. We present results from a large number of trials using a database of representative images acquired in real outdoor environments.**

I. Introduction

In recent years there has been rapid progress in the area of vision-based navigation by mobile robots (see [1] for a review of recent research). Much of this research is concerned with the two problems of localization and mapping, although there are visual servoing approaches that use an implicit map of the environment [2], [3]. The problem of map construction requires the ability to distinguish landmarks in the environment [4], [5]. When the dual problems of localization and mapping are treated simultaneously (the SLAM problem), a variety of techniques can be applied, all of which require identification of some sort of landmarks [6], [7]. In nearly all cases, recognized landmarks are used in conjunction with a global (possibly constructed incrementally) map.

None of these methods deal with determining whether an object is traversable or not. This problem is much different from the problems of localization or mapping, since it is inherently a local problem, and since landmark recognition is not required. In a sense, determining traversability can be treated as a binary classification problem, and this is the approach we take in the present paper.

Past work in determining traversability has focused largely on the geometric aspects of the objects that populate the robot's immediate workspace [8], [9], [10], [11], [12]. For example, objects larger than some predetermined size are typically considered to be obstacles. LIDAR or stereo vision are particularly useful for providing this kind of data. However, a purely geometric approach fails to exploit the fact that in natural environments many objects are not rigid, and can therefore be traversed by most robots. The bushes in Figure 1 are an example of this; since the bushes are flexible, many robots could easily roll over them. Since traversability is not purely a geometric property of an object, techniques based

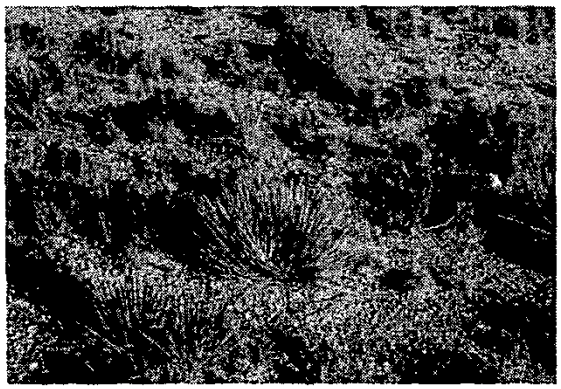

Fig. 1. An example of a traversable environment

on contours [13] or general shape [14] are not well suited to the problem. Because of this pose consistency methods like those based on [15] seemed to be of minimal use. Additionally, aspect graph based techniques [16] like [17] are not applicable because of the deformable nature of the traversable obstacles. This implies appearance based methods [18] do not yield good results either.

In this paper, we deal with the problem of determining a classification scheme that relies on data from a single color image to classify regions of the environment as either traversable or non-traversable. We use three single-feature classifiers together with a weighted voting scheme to make the classification. In particular, we investigate the use of color histograms, directional filters, and local binary patterns to derive three classifiers. The results of these are combined by a probabilistic voting classifier that weights the results of each of the individual elements by its correct classification rate.

The remainder of the paper is organized as follows. In Section 2, we present the three individual classifiers. Then in Section 3 we describe how the outputs of these classifiers are combined to reach a single classification. Experimental

0-7803-7860-1/03/$17.00 © 2003 IEEE

results on a large data set of images of natural environments are presented in Section IV.

II. Features

For this exercise only manually chosen features were evaluated. Manually chosen features, which are less rigid than learned features, were chosen because of the desire to extend this research to have a semi-supervised classification technique with a variable number of output classes. The classification technique was trained in an arid mountainous outdoor environment encompassing the high desert to woodland transition. The benefit to training a certain environment is that the number of classes and images in the database can be minimized, thus keeping computational time down because smaller databases can be searched more quickly. If the robot needs to automatically select the correct database for the given environment, the technique proposed in [19] that use the earth movers distance of an entire image to figure out the type of environment can be implemented. Conversely, if the type of environment is known ahead of time, the environment's database can be preloaded.

Feature extractors can be categorized using either global or local methods. Global methods are methods that work on the image as a whole. Global methods are sensitive to noise and fluctuations in scale, lighting conditions, pose, and rotation; all of which are prevalent in outdoor environments. In general most global methods were avoided in this research. However, the global method classifying color is sensitive only to lighting conditions and therefore was deemed reliable enough for use for classification.

In contrast, local methods only analyze a small region of the image. Local methods were used to analyze the image for underlying features (basic components of the objects shape). Texture analysis techniques were used to find such features. However, before texture analysis techniques could be used, the images needed to be segmented into tiles of texture patterns. This was accomplished by using a key texture selection technique that found a certain subset of tiles best representing the image as a whole. In [20] the color histogram is computed locally along with the LBP texture, but [21] suggests that the color histogram need not be local, so global histograms were used.

1) **Global Techniques:** *Color Histogram*

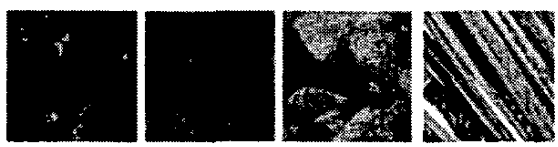

Fig. 2. Example of Differing Colors

Color is a natural choice as an identifiable and characteristic feature, as seen in Figure 2. Color histogram comparison between images is done by first converting the image from RGB space to Hue Saturation Value (HSV) space. This transformation is done because HSV is more robust to changes in lighting conditions than RGB. Once the image color space is converted, a 3 dimensional histogram is calculated for each component (H, S, V). This allows the unknown object's histogram to be compared with the known histograms in the class database. This comparison is done using the Histogram Intersection technique [21], which has been shown to be invariant to translation, rotation, and scale. It also performs well under pose changes and occlusion. The Histogram Intersection technique is based on calculating the intersection between two different histograms. Where the intersections between histograms Model (M) and Reference (I) is defined as:

$$\sum min(M_j, I_j) \tag{1}$$

with j equal to the number of bins in the histogram. This result represents the number of pixels that are shared between the model and reference image. The results can then be normalized by dividing the above equation by:

$$H(I, M) = \frac{\sum min(M_j, I_j)}{\sum M_j} \tag{2}$$

which can be thought of as the area of the image. If the areas of the reference image and model image are the same, or:

$$\sum M_i = \sum I_i \tag{3}$$

then this method becomes a representation of the city-block metric or 1-norm method of calculating the distance between two elements. Because of the motivation for images to be the same scale and to obtain a distance measurement, for the implementation in this project all reference images are scaled to the model histogram size. Then the class associated with the maximum intersection value is assigned to the test image.

Fig. 3. Example of inter-class differences

2) **Local Techniques:** *Texture Analysis*

The key to a successful classification system relies on a good texture analysis technique. This is primarily due to the variations between similar objects as can be seen in Figure 3.

Even though variations commonly occur in parameters such as shape and scale changes, the textures within the shape stay relatively unchanged. A significant portion of the available literature on texture analysis uses various forms of wavelet filters, most notably, the popular Gabor Filter technique.

However, as suggested in [22] most of these wavelet routines perform poorly. In addition to poor performance, wavelet based techniques are computationally expensive and, therefore, were not used in this exercise.

In the analysis of texture, two major approaches were taken. The first was to apply a set of image directional filters to the extracted windows. The second employed a technique called Local Binary Patterns (LBP) [23].

Directional Filter

Directional filters operate in a specified direction by performing an approximation of the derivative in that direction. While vertical and horizontal filters are popular and a natural part of some filter techniques, such as the Sobel edge detector, they are perpendicular to each other. Therefore, many features that may be present at other angles are not detected by these techniques. Directional filters can provide quite a bit of information about the images if enough directions are chosen, thus it has become common [24] to use a grouping of directional filters at different angles to analyze and classify images. The difficulty with choosing the number of filters or more precisely the angle between filters comes down to time, more filters require more computational time and give more information. Because the eventual goal is to create a real time system the number filters was chosen to be either 16 segments or 11.25 degrees of separation. As will be seen this number produced satisfactory results.

For this research a simple 3x3 Sobel kernel was used. The angular direction of the filter was computed from two basic directional filters (vertical and horizontal):

$$D_{horiz} = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} D_{vert} = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad (4)$$

Then these matrices were then multiplied by the angle θ and added together. So that,

$$D_\theta = D_{horiz} sin(\theta) + D_{vert} cos(\theta). \quad (5)$$

Figure 4 shows the directional filter output. The mean and standard deviation for each of the filter outputs is then joined together to create a point represented in a 16x2, or 32, dimensional space. To analyze the data Principle Component Analysis (PCA) was exploited to project the data to three dimensions. Then classification was performed using the K-nearest neighbor method.

Local Binary Patterns

The motivation for implementing two forms of texture analysis was to attempt to obtain techniques that operated in significantly different ways in order to compare the techniques. Also, the hope was that the data they shared would be orthogonal or non-overlapping in nature and thus complementary when used together in a classification basis system.

As mentioned a significant portion of the current literature for texture analysis focuses on wavelet based techniques, along with Markov Random Fields. However, recently a new method was developed [23] claiming a higher correct match rate. The general premise of LBP is that certain local patterns make up the fundamental properties of an image texture. These patterns are generated from the local neighborhoods surrounding pixels [25] as seen in Figure 5. Beyond being algorithmically simple, the other benefits of this approach are that it is orientation independent and invariant to any monotonic transformation in the image intensity value. First to calculate the LBP value, the

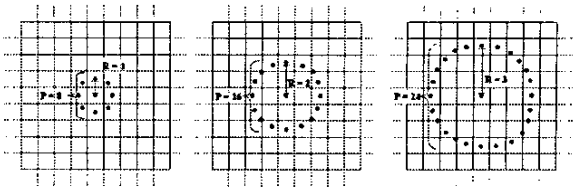

Fig. 5. : The sampling neighborhood for various sizes of the LBP Operator

texture operator, T, is defined in the local neighborhood of an image by the joint distribution, $t(g)$, of the intensity values of P, where $P > 1$, image pixels:

$$T = t(g_c, g_0, ..., g_{p-1}) \quad (6)$$

Where g_c is defined as the value of the center pixel of the neighborhood.

Accordingly, the g_p, for $p = 0...P-1$, represent the values of the equally space pixels that form a circular set of radius R, $R > 0$, around the center pixel. The difficulty of R and P not coinciding with the descritized image, or the pixel spacing, is dealt with by interpolating values between pixels to extract the values of points in P that fall between them. To achieve invariance to the scaling of the pixel's intensity value, only the signs of the differences between values are considered. Therefore, the equation for the texture, T, is modified to be:

$$T \approx t(s(g_0 - g_c), s(g_1 - g_c), ..., s(g_{p-1} - g_c)) \quad (7)$$

Where the function s(x) is defined as:

$$s(x) = \begin{cases} 1, x \geq 0 \\ 0, x < 0 \end{cases} \quad (8)$$

Now the number of uniform patterns, $LBP^{riu2}_{P,R}$ (where $riu2$ is an indication that this is variant of the LBP method that the finds uniform patterns) is computed by:

$$LBP^{riu2}_{P,R} = \begin{cases} \sum_{p=0}^{P-1} s(g_p - g_c) & if U(LBP_{P,R} \leq 2 \\ P+1 & \text{otherwise} \end{cases} \quad (9)$$

Where U, the uniformity measure, is the number of spatial transitions in binary value, switches in signs, in the image and is defined as:

$$U(LBP_{P,R}) = \begin{array}{l} |s(g_{p-1} - g_c) - s(g_0 - g_c)| + \\ \sum_{p=1}^{P-1} |s(g_p - g_c) - s(g_0 - g_c)| . \end{array} \quad (10)$$

When the value of U is at most two the pattern is considered uniform. Then a histogram is composed for each of the pixels in the image. The histogram is created from the set $0, ..P+1$ of bins, where the $P+1$ bin corresponds the the "others"

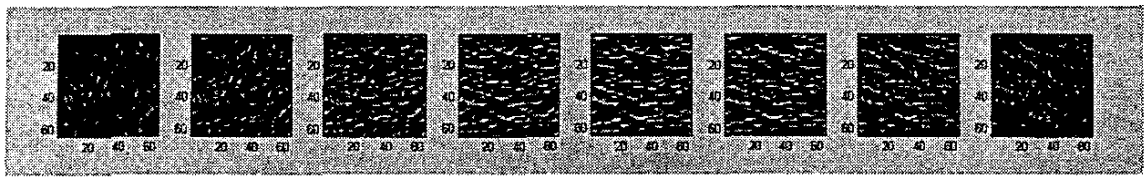

Fig. 4. Results of 8 segments directional filters.

bin and the uniform patterns. Therefore the number of bins for the method is $P + 2$. To measure the differences between textures through the histograms is used on each of the model histograms, a nonparametric test to avoid assumptions about feature distributions . This is accomplished by use of the G-statistic or log likelihood ratio. Given a test sample S and a model M the log likelihood is calculated as:

$$L(S, M) = \sum_{b=1}^{B} S_b \log M_b \tag{11}$$

This ratio is calculated for each of the models in the training set. Then the sample is assigned to the class of the model value, M, with the maximum ratio value with the sample.

Fig. 6. Image with corresponding Mask

3) Texture Selection: Because the texture techniques require small regions of the image a method was developed, which separates an image into tiles of a specified size. To accomplish this a mask, Figure 6, was used that outlined the segmented object to compare the areas occupied by sample data for each of the tiles. If the total occupied area was less than 90%, the tile was thrown away unless there were not enough tiles that satisfied the condition. The remaining tiles were then passed to the LBP technique to generate a histogram for each tile. For this method the LBP parameters P=8, $R = 1$; $P = 16$, $R = 2$; and $P = 24$, $R = 3$ were concatenated into one histogram. These concatenated histograms were then used to project points into a multidimensional space of 54 dimensions. With all the tiles in the image represented in this space, k-means clustering [26] was used to isolate the tiles that represented distinct clusters. K-means clustering segments a data space by assuming a Gaussian distribution of points. By specifying a number of distributions, or clusters, the technique chooses random mean values for each of these clusters and through several iterations refines the mean values of distribution by classifying all points relative to the clusters and then recalculating the mean of the points within the same class. This is done until some heuristic is satisfied. The most commonly used heuristic is convergence to stable values. Once the center of cluster is found, the point in the space that is closest to the mean value is selected as a key texture. K-means was chosen because it is simple and the number of clusters can be specified as an input parameter. This property was desired because of the want to limit the clusters to a small set of the total number of tiles.

III. Classification

Once the features are extracted, they are passed to the classifier. The primary purpose of this classifier is to create the decision boundary that both minimizes the error while not over specializing the decision region. Over specialization may cause the error for the training set may be small but, because the region fits so tightly around the data, any new data may be misclassified. Poor classification may also result from an inaccurate decision region based on an inadequate training set.

Because of the high accuracy of each of the individual methods, a complicated classification algorithm was not used to meld the results. Instead a probabilistic voting algorithm was used in which each of the three methods cast its vote for the class it thought it belonged to. Each of the votes was then weighted by an individual score that was based on the percent of its correct classification of the test data. Then a percentage of the total votes cast verses the votes per class was computed and the class with the highest vote percentage, or probability, was assigned to the test image. A benefit to this approach, although not technically complicated, is that it gives a certainty measure. If all the individual classifiers agree then it is very likely that the chosen class is correct. However, if they all disagree and one class *wins* by a very small margin, then caution should be taken in accepting the results as valid.

IV. Results

To tackle the problem of distinguishing negotiable and non-negotiable objects this work adds an additional level of intelligence to the navigation hierarchy. This level acts as a classifier for objects detected in the robot's path. Once an object is segmented from the image, it is relatively easy to implement a system to classify the object and determine and its possible hindrances for use in a path planning technique.

The process of identifying objects is done in four stages. The first stage retrieves the image. The images come from still images generated by a consumer level digital camera with a resolution of 1280x960 pixels. A human observer, wandering through the outdoor environment, acquires images

of relevant objects. For this system a total of eleven classes of objects were used to test the system. They were: arrow brush, sage brush, rabbit brush, yucca plant, decorative tree 1, decorative tree 2, juniper tree, pine tree, prickly pear cactus, russian thistle, and medium rocks. For each class the number of images per class is listed in Table I. The second stage

TABLE I
Image Database

Class	1	2	3	4	5	6	7	8	9	10	11
# Images	13	13	9	8	10	9	7	3	4	2	4

segments the retrieved images into likely objects. As automatic segmentation was not addressed in this paper, the images were segmented manually. Manual segmentation was done by first by cropping the total image to the size of the object of interest. Then a border around the object was used to identify a mask image defining the region of interest. The third stage uses feature extraction techniques to analyze the segmented objects. The fourth, and final stage takes the data from stage 3 and classifies the unknown objects. Figure 7 illustrates the flow of a complete system for this approach.

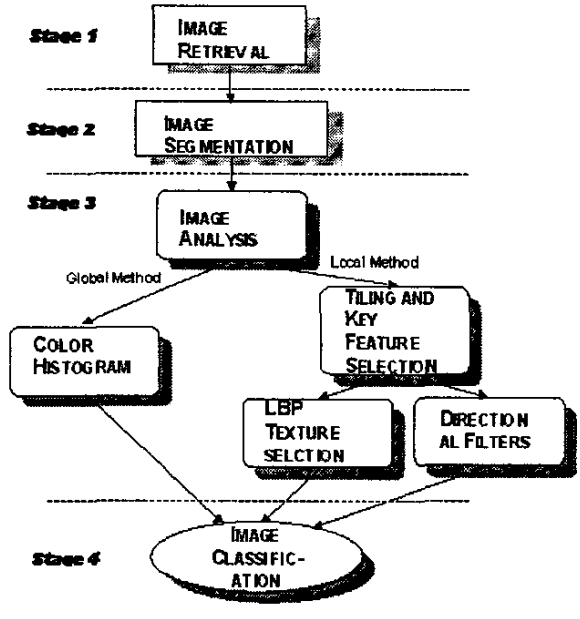

Fig. 7. Flow chart of method development

For analysis, the aforementioned methods for classification were first tested separately to obtain individual results. Then the methods were coupled together and used to form the complete object recognition system. The system was then tested in the same manner as the individual methods to verify the results of the system against the individual methods. In each case half of each of the images per class were used to train and the other half were used to test. Each test was iterated ten times to verify the stability and precision of the system. In addition, for several of the tests other parameters, like tile size and the number of neighbors to use for K-nearest neighbor technique, were changed in order to ascertain the optimal parameters for each component.

A. *Individual Elements*

1) Key Feature Window Selection: Because this method relies on the LBP algorithm to select which windows of the tiled image carried the most information, refer to the section of the results for the LBP method for a thorough analysis. To get an idea of the key texture selection performance, tests were done purely on a visual basis. A fraction of the whole image database was passed to the technique and then compared to the whole image. This comparison was the designer's simple evaluation of how well the selected textures represented the image in question. It is important for the reader to understand that only a crude approximation of the performance could be done this way. Figure 8 illustrates an example of tiles chosen for some the images. While for some classes it is difficult to

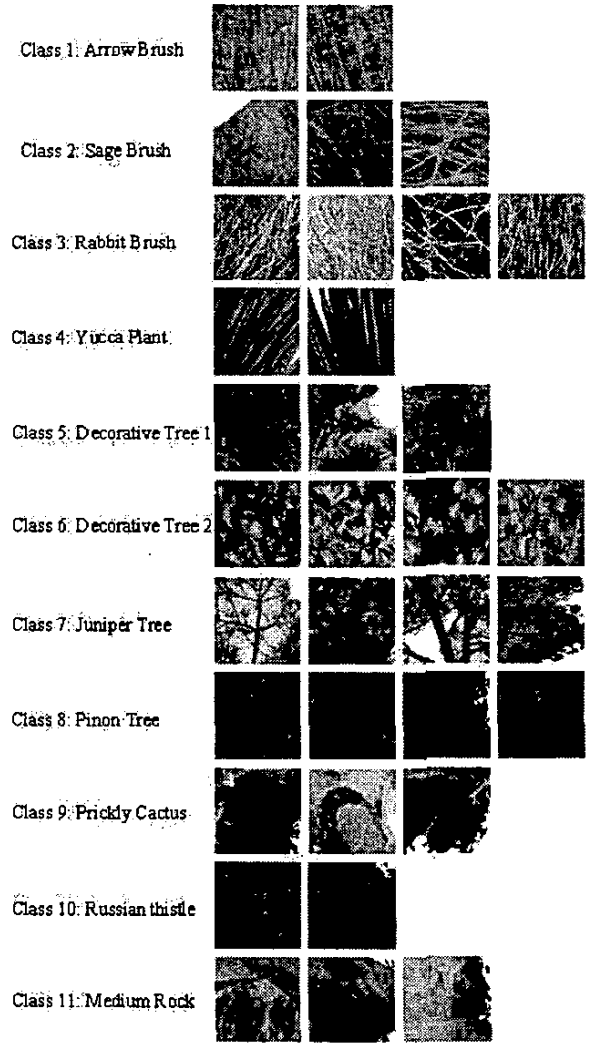

Fig. 8. Key Tiles

see how each texture differs from the others within its class, there are two in particular, class 2 and class 3, for which, the texture variance is clearly discernible. It is important that the key texture selection method choose the textures that represent the most disparate tiles within the image. From visual inspection, which is by no means a robust analysis, this algorithm works adequately. The results of the above window extraction method were then fed into the texture analysis techniques.

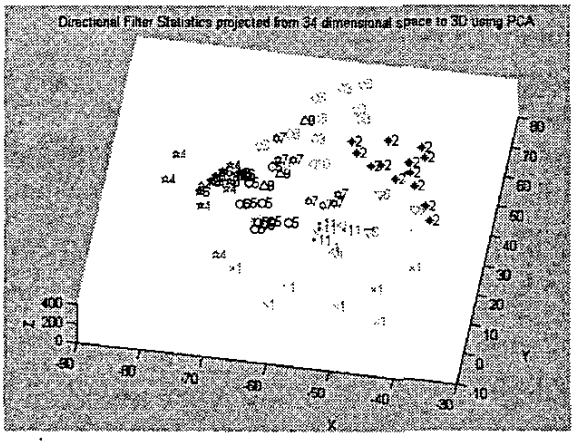

Fig. 9. Visualization by PCA of Directional Filters distribution between classes using 16 directional filters for a tile size of 80x80 pixels.

2) Directional Filter: The first step in testing the directional filter method was to visualize a small subset of the data giving a general idea of the technique's relative effectiveness. Since the filtering technique uses the mean and standard deviation of each directional filtered image, each image is represented in a 32 hyper dimensional space. In order to visualize this data, PCA was used to project from the 32 dimensional space to a 3 dimensional space for visualization purposes. Figure 9 illustrates the results for the directional filter. Figure 9 also shows that each class is relatively well defined, in that there is good clustering within class and minimal class overlap. This can be seen in the sample of the results in Tables II, III. These results were obtained by performing ten iterations of the classification system. The classification consisted running the k-nearest neighbor algorithm with differing values of k. Also, the tile size was varied to compare results as a function of the tile area.

3) LBP: Like the Directional Filter technique, the data was first visualized, Figure 10. While the separation is not ideal, there does seem to be some distinction between classes. However, because the LBP method used Log-Likelihood as its distance metric, the figure does not represent the actual distance between classes and therefore should be regarded with some skepticism. To acquire actual results, the tiles from the texture selection method were passed to the LBP method for comparison. Each of the testing tiles was compared against all of the training tiles for each image and each class by calculating a distance measure. Then the class of the training

TABLE II

Directional Filter Mean Values (%)Tile Size vs. Class

		16	32	64	96	128	160
Class 1	K=1	66.25	89.20	96.67	100.00	100.00	100.00
	3	77.64	83.61	97.08	100.00	100.00	100.00
Class 2	1	53.75	61.67	69.58	83.05	87.77	94.57
	3	54.17	61.67	62.50	81.14	76.09	84.86
Class 3	1	40.63	37.50	69.50	72.08	94.67	97.14
	3	36.88	36.88	35.88	46.17	80.46	86.95
Class 4	1	32.50	57.50	72.36	89.38	100.00	100.00
	3	22.50	35.00	41.65	80.00	100.00	100.00
Class 5	1	44.50	77.50	100.00	100.00	100.00	100.00
	3	43.00	77.50	98.50	100.00	100.00	100.00
Class 6	1	61.88	83.13	97.50	100.00	100.00	100.00
	3	60.63	81.25	99.38	100.00	100.00	100.00
Class 7	1	19.17	17.50	82.50	93.79	96.27	93.80
	3	6.67	10.00	62.50	77.73	89.76	84.67
Class 8	1	20.00	20.00	67.50	77.50	86.67	78.33
	3	20.00	5.00	45.00	65.83	51.67	65.00
Class 9	1	58.75	30.54	80.48	81.00	100.00	100.00
	3	31.79	9.05	29.46	66.67	100.00	100.00
Class 10	1	9.17	86.67	100.00	100.00	100.00	100.00
	3	0.00	0.00	0.00	0.00	30.00	100.00
Class 11	1	41.25	79.29	86.67	100.00	100.00	100.00
	3	30.00	74.29	71.67	100.00	100.00	100.00

TABLE III

Directional Filter Standard Deviation Values (%) Tile Size vs. Class

		16	32	64	96	128	160
Class 1	K=1	11.00	6.13	5.83	0.00	0.00	0.00
	3	10.19	11.63	4.41	0.00	0.00	0.00
Class 2	1	8.44	5.12	8.11	7.14	5.42	6.59
	3	9.21	10.54	5.89	8.98	12.29	10.04
Class 3	1	7.37	14.13	10.94	6.96	8.78	6.02
	3	13.96	8.56	9.88	12.44	10.49	9.92
Class 4	1	8.23	8.74	17.39	6.62	0.00	0.00
	3	12.22	10.70	9.65	12.77	0.00	0.00
Class 5	1	11.65	10.34	0.00	0.00	0.00	0.00
	3	5.37	8.25	4.74	0.00	0.00	0.00
Class 6	1	12.31	10.23	3.23	0.00	0.00	0.00
	3	11.04	11.41	1.98	0.00	0.00	0.00
Class 7	1	7.91	7.30	12.70	11.34	4.82	9.21
	3	5.27	8.61	11.28	16.64	11.20	6.73
Class 8	1	15.81	25.82	12.08	21.89	17.21	15.32
	3	15.81	10.54	15.81	20.20	12.30	19.95
Class 9	1	12.47	15.18	6.40	12.28	0.00	0.00
	3	9.74	10.34	15.18	23.04	0.00	0.00
Class 10	1	14.93	17.21	0.00	0.00	0.00	0.00
	3	0.00	0.00	0.00	0.00	48.30	0.00
Class 11	1	11.86	13.02	17.21	0.00	0.00	0.00
	3	8.74	8.84	8.05	0.00	0.00	0.00

sample that corresponded to the smallest distance measure between it and the test tile was then selected to as the most likely class. The test was performed 10 times for several different tile sizes. The results were then scrutinized to find the best tile size. Below, Tables IV and V depict a portion of the results according to the tile size along with the iterations averages and standard deviation per class.

4) Color Histogram: Unlike the two texture approaches discussed above, the color histogram did not use the results of the texture selection technique. Instead, it calculated the histogram information from the entire image. To test the classification accuracy of the Color Histogram technique, the

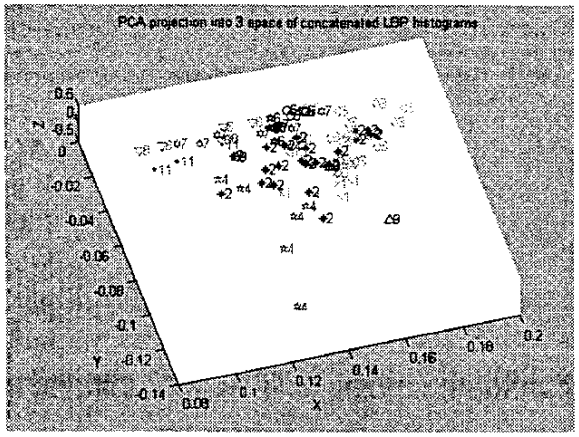

Fig. 10. LBP Histogram for each class. Represented in 3D subspace.

TABLE IV

LBP Mean Values (%) Tile Size vs. Class

	16	32	64	96	128	160	Avg.
1	57.50	77.50	91.77	94.42	100.00	100.00	90.45
2	28.75	48.33	77.50	87.50	98.30	100.00	79.20
3	45.63	48.13	71.25	88.42	89.67	91.71	73.90
4	48.75	76.25	82.50	90.70	100.00	100.00	86.00
5	59.50	81.00	100.00	100.00	100.00	100.00	92.49
6	65.63	76.25	94.33	100.00	100.00	100.00	93.62
7	18.33	48.33	70.00	95.83	100.00	90.46	75.62
8	5.00	35.00	60.69	87.50	70.00	89.67	84.08
9	54.46	73.57	84.45	81.10	100.00	98.00	84.08
10	35.00	58.33	100.00	87.14	93.33	96.00	84.08
11	19.31	34.98	75.44	88.27	84.21	86.61	70.13
Avg.	39.80	59.79	82.54	90.99	94.14	95.68	

entire data set was split, as mentioned before, into two roughly equal sets of images per class: training and testing. When an odd number of images existed the training set contained the additional image. The testing was done by comparing the test image's known class to the class found by the Histogram Intersection method. Ten iterations were run on the method to verify the results. To obtain the optimal histogram size, several different bin sizes were tested. To obtain the optimal histogram size, several different bin sizes were tested. Some of the results for the tests are listed in Tables VI and VII.

color histogram were varied from each individual test with the same parameters to ascertain how the complete system performed.)

B. Complete System

The complete system was run with the same parameters as the other tests; the tile size, number of nearest neighbors to use for the directional filter, and the size of the color histogram color histogram were varied from each individual test with the same parameters to ascertain how the complete system performed.) Even with the simple voting algorithm most of the results show perfect classification. Because, at first, the system seemed to be performing too well, several of the tests were monitored to verify that the results were correct.

TABLE V

LBP Standard Deviation Values (%) Tile Size vs. Class

	16	32	64	96	128	160	Avg.
1	14.14	7.40	7.12	5.92	0.00	0.00	4.71
2	6.93	6.86	9.86	5.89	2.20	0.00	4.41
3	8.36	8.86	9.86	5.16	4.33	7.28	8.13
4	15.53	13.11	8.74	3.24	0.00	0.00	5.67
5	9.85	11.25	0.00	0.00	0.00	0.00	2.87
6	13.58	12.77	9.17	0.00	0.00	0.00	3.55
7	10.24	15.62	18.51	8.10	0.00	7.01	9.00
8	10.54	12.91	22.50	17.68	33.15	13.72	15.16
9	13.94	22.60	10.67	12.13	0.00	6.32	8.72
10	12.91	8.78	0.00	17.99	21.08	12.65	12.80
11	11.67	12.56	8.23	4.40	6.48	5.36	7.62
Avg.	11.61	12.07	9.51	7.32	6.11	4.76	

TABLE VI

Color Histogram Mean Values (%) Number of Bins vs. Class

	2	8	16	32	48	64	Avg.
1	48.33	90.00	100.00	100.00	90.00	88.33	91.00
2	63.33	90.00	100.00	96.67	96.67	95.00	93.11
3	22.50	77.	47.50	47.50	67.50	70.00	57.17
4	50.00	55.00	47.50	62.50	60.00	55.00	53.83
5	100.00	98.00	100.00	100.00	100.00	100.00	98.13
6	75.00	75.00	72.50	85.00	72.50	87.50	77.50
7	46.67	76.67	53.33	40.00	53.33	50.00	51.33
8	0.00	0.00	0.00	0.00	0.00	0.00	0.00
9	40.00	65.00	85.00	85.00	70.00	75.00	71.67
10	10.00	100.00	100.00	100.00	100.00	100.00	91.33
11	10.00	60.00	100.00	100.00	100.00	100.00	89.33
Avg.	2.35	71.56	73.26	74.24	73.64	74.62	

V. Conclusion

Each individual method faired remarkably well in the tests. The Histogram Intersection method average peaked with a 76.67% correct classification rate. Illumination changes and inherent color variations within each class are the probable culprits for misclassificatione in this technique. Still, with such a high success rate, the Color Intersection method proved to be effective in outdoor environment classification. The Directional Filters' performance provided the most surprising results with a 96.72% maximum average performance for all the classes. Like the Directional Filters, the LBP algorithm,

TABLE VII

Color Histogram Standard Deviation Values (%) Number of Bins vs. Class

	2	8	16	32	48	64	Avg.
1	19.95	8.61	0.00	0.00	8.61	8.05	6.82
2	20.49	11.65	0.00	7.03	7.03	8.05	6.94
3	21.89	18.45	14.19	21.89	23.72	28.38	23.83
4	20.41	28.38	18.45	21.25	21.08	22.97	22.47
5	0.00	6.32	0.00	0.00	0.00	0.00	1.83
6	26.35	23.57	36.23	21.08	24.86	21.25	24.15
7	28.11	38.65	23.31	21.08	28.11	28.33	29.49
8	0.00	0.00	0.00	0.00	0.00	0.00	0.00
9	51.64	24.15	24.15	24.15	25.82	26.35	27.20
10	31.62	0.00	0.00	0.00	0.00	0.00	5.55
11	21.08	39.44	0.00	0.00	0.00	0.00	5.76
Avg.	21.96	18.11	10.58	10.59	12.66	13.03	

classified remarkably well with a peak 93.62% classification accuracy. To further enhance the LBP technique the full extent of the algorithm needs to be implemented. Currently the VAR function is not used to scale the LBP function. This should be done in future applications as its addition, as suggested in [10], produces a more accurate and robust method. Also, techniques to obtain scale independent results need to be investigated.

With near perfect scores for many of the methods results, the system out-performs each of the individual components. In fact only class 8, pine tree, did not obtain perfect classification. If further tests are performed with the system, additional and more complicated techniques for grouping the methods together should be investigated.

The database as used contains four classes containing four or fewer images. For this case of the class with only two images, one image is used for training and the other is used for testing. This extremely small set of testing and training images is inadequate for a comprehensive test. Because of dataset limitations further testing should be done with an extended image database. Additionally, more classes need to be added to analyze how the breadth of classes affects the classification system.

Once these above mentioned objectives are met, the next logical objective is to implement an automatic segmentation routine. It may be possible to combine both the classification and segmentation into one step.

ACKNOWLEDGMENT

The authors would like to thank both John Harrington, Sandia National Laboratories, for his support of this project along with Joe Fogler, Sandia National Laboratories, for his helpful comments.

This work was sponsored by Sandia National Laboratories which is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under Contract DE-AC04-94AL85000.

REFERENCES

[1] G. N. DeSouza and A. C. Kak, "Vision for mobile robot navigation: A survey," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 24(2), pp. 237–267, 2002.

[2] R. Swain, M. Devy, and S. Hutchinson, "Sensor-based navigation in cluttered environments," *International Conference on Intelligent Robots and Systems*, pp. 1662–1669, 2001.

[3] H. Zhang and J. Ostrowski, "Visual motion planning for mobile robots," *Transactions on Robotics and Automation*, vol. 18(2), pp. 199–208, April 2002.

[4] C. Taylor and D. Kriegman, "Vision-based motion planning and exploration algorithms for mobile robots," *Transactions on Robotics and Automation*, vol. 14(3), pp. 417–427, 1998.

[5] Y. Takeuchi and M. Hebert, "Finding images of landmarks in video sequences," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR '98)*, June 1998.

[6] J. Guivant, E. Nebot, , and S. Baiker, "Autonomous navigation and map building using laser range sensors in outdoor applications," *Journal of Robotic Systems*, vol. 17 (10), pp. 565–583, 2000.

[7] S. Thrun, "Bayesian landmark learning for mobile robot localization," *Machine Learning*, vol. 33, no. 1, pp. 41–76, 1998.

[8] L. Matthies, "Toward stochastic modeling of obstacle detectability in passive stereo range imagery," *CVPR*, vol. 92, pp. 765–768.

[9] L. Matthies, A. Kelly, T. Litwin, and G. Tharp, "Obstacle detection for unmanned ground vehicles: a progress report," in *Proceedings of IEEE Intelligent Vehicles '95 Conference*, pp. 66 – 71, September 1995.

[10] S. Betge-Brezetz, R. Chatila, and M. Devy, "Natural scene understanding for mobile robot navigation," *IEEE International Conference on Robotics and Automation*, vol. 1, pp. 730–736, 1994.

[11] L. and E. Krotkov, "Natural terrain hazard detection with a laser rangefinder," *IEEE International Conference on Robotics and Automation*, pp. 968–973, 1997.

[12] S. L. et al, "Color indexing," in *To Appear in: International Journal of Computer Vision*, 2002.

[13] F. M. et al, "Silhoutte based isolated object recognition through curvature scale space," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 17, no. 5, pp. 539 – 544, 1995.

[14] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 24(3), pp. 509–522, 2002.

[15] D. Thompson and J. Mundy, "Three-dimensional model matching from an unconstrained viewpoint," *International Conference on Robotics and Automation*, pp. 208–220, 1987.

[16] J. Koenderink and A. V. Doorm, "An internal representation of a solid shape with respect to vision," *Biological Cybernetics*, vol. 32, pp. 211–216, 1979.

[17] C. M. Cyr and B. B. Kimia, "3d object recognition using shape similarity-based aspect graph," *International Conference on Computer Vision*, 2001.

[18] S. K. Murase and S. K. Nayar, "Visual learning and recognition of 3-d objects from appearance," *International Journal of Computer Vision*, vol. 14(1), pp. 5–24, 1995.

[19] R. M.-C. et al, "Building multi-level models: From landscapes to landmarks," *Conference on Robotics and Automation*, pp. 4346–4353, 2002.

[20] M. Pietikäinen, T. Mäenpää, and J. Viertola, "Color texture classification with color histograms and local binary patterns," *http://www.cee.hw.ac.uk/ texture2002/papers/ab002.pdf*, 2002.

[21] M. J. Swain and D. H. Ballard, "Color indexing," in *International Journal of Computer Vision*, vol. 7(1), pp. 11–32, 1991.

[22] T. Randen and J. Husoy, "Filtering for texture classification: A comparative study," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 21(4), pp. 291–310, 1999.

[23] M. Pietikainen, T. Maenpaa, and J. Vierola, "Multiresolution gray-scale and rotation invaritant texture classification with local binary patterns," *Transactions on Pattern Analysis and Machine Intelligence*, vol. 24(7)), pp. 971–987, 2002.

[24] J. Ponce and D. Forsyth, eds., *Computer Vision: A Modern Approach*. Upper Saddle River, NJ: Prentice Hall, 2001.

[25] A. Lucieer, "Local binary pattern (lbp) texture measure," *http://www.itc.nl/personal/lucieer/research/leicester/Local_Binary_Pattern.html*, October 2002.

[26] R. Duda, P. Hart, and D. Stork, eds., *Pattern Classification*. USA: John Wiley and Sons, 2001.