Reducing the Dimension of Color Features using a Naïve Bayesian Classifier

Sun-Mi Park
Graduate School of EECS
Kyungpook National Univ.
Daegu, Korea
disvogue@gmail.com

Ku-Jin Kim
(corresponding author)
Dept. of Computer Engineering
Kyungpook National Univ.
Daegu, Korea
kujinkim@yahoo.com

Abstract—**Color histograms are usually used as the color feature vectors for classifying the color of objects in images. We reduce the dimension of the feature vector by a factor of about 30 by using a naïve Bayesian classifier, and use the resulting feature vectors with a support vector machine to recognize vehicle colors. Experiments show that the recognition rate is close to that achieved with the original large feature vectors, while recognition time is reduced by a factor of more than 30. We also show that our method outperforms principal component analysis.**

Keywords-component; color histogram; dimension reduction; Bayesian classifier

I. INTRODUCTION

Color is one of the most characteristic features of an object in an image, or of an entire image. The detection, tracking, and recognition of objects in an image has many applications in areas such as human-computer interaction, ubiquitous computing, and robot vision; and in these applications the colors of objects are an important denominator of their characteristics. The color information in an image is also widely used in content-based image retrieval, image classification, and image analysis [1-2].

Histograms are often used to extract the color features in an image. There are various methods to construct color histograms, but the most basic is to count the number of pixels whose colors lie in each partition of a two- or three-dimensional color space. In the case of an RGB image, two or three channels are selected from red, green and blue, and a two- or three-dimensional array is constructed with indices that correspond to quantized channel values.

There has been quite a lot of research on image classification using color histograms [3-4]. A color histogram is often used to construct a feature vector to represent color information. If a more precise representation of color information is required, the color space must be partitioned more finely, and the result is a feature vector of higher dimension. High-dimensional feature vectors cause algorithms which involve learning processes to run more slowly, and the requirement for training data is increased. This has motivated research on reducing the dimensionality of feature vectors. PCA (principal component analysis) [5] is widely used for this purpose, especially in pattern recognition.

In this paper, we propose a method of reducing the dimensionality of the color features used to recognize the colors of vehicles from an image of a road scene. Rather than using the color histograms themselves as feature vectors, we construct class histograms using a naive Bayesian classifier [5]. Experimental results show that this method of reducing the color space does not affect the success-rate of a recognition process, but the time required for recognition is reduced by a factor of 36, when the dimensionality of the feature vector is reduced by a factor of 32. We also show that the success rate of our algorithm is higher than that of principal component analysis.

The remainder of this paper is organized as follows. Related work is discussed in Section 2. We consider the color recognition problem and explain the brute-force algorithm in Section 3. In Section 4, we discuss methods of dimension reduction based on PCA and present a naive Bayesian classifier. We present our experimental results in Section 5, and conclude this paper in Section 6.

II. RELATED WORK

Bayesian classifiers have been used to identify the pixels in an image that correspond to skin or a part of face, and to recognize other types of object in color images. To our knowledge, there have been no previous attempts to use Bayesian classifiers to reduce the dimensionality of feature vectors.

Color histograms have been used for many applications [1-4]. Qiu et al. [1] used PCA to reduce the dimension of various kinds of histograms. They classified a large collection of color images based on content, by constructing several types of histograms, from which they obtained lower-dimensional feature vectors using PCA. Then they used a support vector machine (SVM) to process the feature vectors.

Sural et al. [2] analyzed the way in which changes of hue, saturation, and intensity contribute to visual recognition. They constructed feature vector for individual images by regarding hue or intensity values as dominant properties depending on saturation values. This method of feature extraction was applied to color histograms and used for content-based image retrieval.

Swain and Ballard [3] used color histograms to index an image database containing multicolored objects. They showed that a color histogram is a stable feature which allows different objects to be recognized. They also developed a way to recognize an object by computing the intersection of two histograms.

Chapelle et al. [4] proposed an image classification method that uses color histograms as feature vectors, and showed that color histograms have a reasonable performance in this task. An SVM with various kernels was used for classifying the images.

III. Color Recognition Using a Brute-Force Method

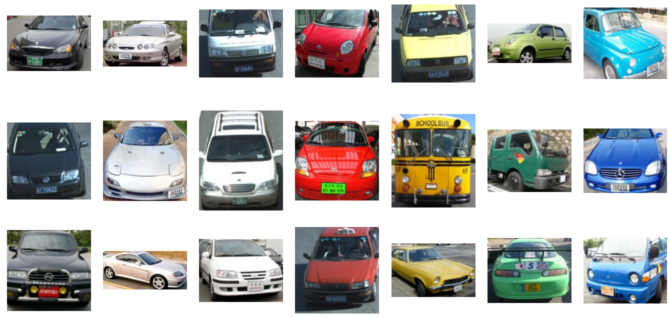
We now consider the problem of vehicle color recognition and present a brute-force method to construct feature vectors. The input images that we wish to process are mainly captured outdoors, so that the colors vary with the weather, especially the amount of sunlight and the properties of surfaces, particularly reflectivity. A changing environment can lead to very different images of the same object. If we had details of the environment and the surface properties of the object at the time that an image was acquired, it would be relatively easy to reconstruct the original color of the object. But this information is not usually available. Our algorithm must deal with this situation, but we do assume that vehicle colors are not so distorted that they would not be recognizable by a human viewer.

The color of a vehicle can usually be determined most effectively from its hood area. We would like to be able to segment the hood from the input image, so as to eliminate distracting colors from surrounding areas, such as the license plate, radiator grill, windshield, and headlamps. But this is difficult without the vehicle geometry, the relative position of the camera and the vehicle, and the camera parameters. We therefore have to deal with an input image which contains the whole vehicle rather than the hood alone. There are algorithms to segment the whole vehicle in a road image, which we can apply to incoming images, so that our algorithm receives a segmented vehicle image, although they may contain parts of the background.

We consider seven classes of vehicle colors, consisting of the three achromatic colors black, silver, and white and the four chromatic colors red, yellow, blue, and green. Fig.1 shows some examples of input images corresponding to each color. A brute-force method of constructing feature vectors to recognize vehicle colors is as follows:

1. Convert the input image to the HSI (hue saturation intensity) color model.
2. Compose hue-saturation, hue-intensity, and saturation-intensity histograms.
3. Compose three vectors by arranging the elements of the three histograms in row-major order.
4. Construct feature vectors by combining the three vectors.
5. Recognize vehicle colors by using the feature vectors with an SVM.

We use the HSI model because chromatic colors are differentiated by hue and saturations values, while achromatic colors are differentiated by saturation and intensity values.



Black Silver White Red Yellow Green Blue

Figure 1. Example input images for vehicle color recognition.

The dimension of a feature vector constructed from the color histogram is determined by the number of bins in the histogram. When hue, saturation and intensity are each divided into n ranges, the histogram consists of n^3 bins. To reduce the dimensionality that we have to deal with, we use two-dimensional histograms that are generated by projecting the three-dimensional color data on to a two-dimensional color plane. We decompose hue, saturation and intensity into 16 ranges each, and then construct two-dimensional histograms on the HS, HI, and SI planes. We will refer to these histograms as h^{HS}, h^{HI} and h^{SI}, respectively.

Each two-dimensional histogram consists of $16 \times 16 = 256$ bins, and we add one more bin to store the number of pixels that satisfy $r = g = b$, which have an undefined hue, and also an undefined saturation if $r = g = b = 0$. We can represent each histogram as a vector by listing it in row-major order. To construct a feature vector from two histograms, we represent each of them as a vector, and then combine the two vectors. The feature vectors for color recognition are combinations of three histograms. Table I lists the feature vectors and the corresponding color histograms.

TABLE I. Feature Vectors Constructed by the Brute-Force Method.

Feature vector (dimension)	V^{HS} (257)	V^{HI} (257)	V^{SI} (257)	$V^{HS,SI}$ (514)	$V^{HI,SI}$ (514)	$V^{HS,HI}$ (514)	$V^{HS,HI,SI}$ (771)
Color histogram	h^{HS}	h^{HI}	h^{SI}	h^{HS} & h^{SI}	h^{HI} & h^{SI}	h^{HS} & h^{HI}	h^{HS} & h^{HI} & h^{SI}

IV. Reducing the Dimensionality of Feature Vectors

A. The PCA method

Principal component analysis (PCA) is often used in pattern recognition to reduce the dimension of a feature vector by projecting high-dimensional data on to a low-dimensional subspace. Suppose that a feature vector has d dimensions, and that the subspace of the feature space consists of n basis vectors.

Starting with a training data set, we compute the mean vector μ, the covariance matrix $\sum$, and its eigenvalues and eigenvectors. We then find the highest n eigenvalues and use the corresponding n eigenvectors as basis vectors of the subspace. We can now construct a $d \times n$ matrix A from column vectors that consist of the chosen eigenvectors. Then we can convert a d–dimensional feature vector X to an n–dimensional vector Y :

$$Y = A^T (X - \mu) . \qquad (1)$$

We will now see how this method applies to reducing the dimension of the feature vectors constructed by the brute-force method, as shown in Table I. Consider the feature vector V^{HS}, which has a dimension of 257, and let us assume that the corresponding set of data contains k feature vectors. Then we can represent the training data set for V^{HS} as a $257 \times k$ matrix C, where the element c_{ij} of C is the i th feature value of the j th training data. If the mean of the column vectors in C is μ, and C_μ is a matrix constructed by repeating this average column k -times, the covariance matrix $\sum$ is derived as follows:

$$\Sigma = (C - C_\mu)(C - C_\mu)^T . \qquad (2)$$

We then compute the eigenvalues and eigenvectors of the 257×257 matrix $\sum$, and select the n eigenvectors whose corresponding eigenvalues are the highest, from which we can construct the $257 \times n$ matrix A. We can then compute an n–dimensional feature vector Y using Equation (1). The dimensions of the other feature vectors in Table I can be reduced by a similar process.

B. A naïve Bayesian classifier

A naive Bayesian classifier uses a statistical approach based on Bayes' theorem. If X is a feature vector and w_j is a class, then the probability that X is in w_j is $p(w_j \mid X)$, as given as follows:

$$p(w_j|X) = \frac{p(X|w_j)p(w_j)}{p(X)} \quad , \quad (3)$$

where $p(w_j \mid X)$ is a posterior probability and $p(w_j)$ is the prior probability of the sample data being in w_j. The probability $p(X)$ is the same as $\sum_j p(X|w_j)p(w_j)$, and $p(X \mid w_j)$ is the likelihood of w_j for X.

If we want to classify X into either w_1, w_2, w_3,…, or w_n, we compute the posterior probability $p(w_j \mid X)$ for each class w_j, and X is put into class w_i which has the highest probability $p(w_i \mid X)$. In Equation (3), $p(X)$ is independent of $p(w_j \mid X)$. If the same amount of training data is provided for each class, the prior probability has the property that $p(w_j) = p(w_k)$ for all j and k. Therefore, the posterior probability $p(w_j \mid X)$ is proportional to the likelihood $p(X \mid w_j)$. If $w_i = \arg\max_j p(X \mid w_j)$, which is the likelihood of class w_i, is the highest, then we can put X into class w_i. Consequently, we can classify X by estimating the likelihood $p(X \mid w_j)$. In our approach, the likelihood is estimated as the Gaussian density of a d–dimensional random vector X :

$$p(X \mid w_j) = \frac{1}{(2\pi)^{d/2}|\Sigma_j|^{1/2}} \exp\left[-\frac{1}{2}(X - U_j)^t \Sigma_j^{-1}(X - U_j) \right] , \quad (4)$$

where U_j is the mean vector and $\sum_j$ is the covariance matrix of the sample feature vectors that are included in the class w_j.

We can use a naive Bayesian classifier to construct a class histogram, which we can then use instead of the color histogram in reducing the dimensionality of a feature vector, as follows:

1. For each pixel in the vehicle image,

   a) Generate three vectors X^{HS}, X^{HI} and X^{SI} from its HSI values (h, s, i) as follows:

   $$X^{HS} = [h,s] \quad X^{HI} = [h,i] \quad X^{SI} = [s,i] \ . \qquad (6)$$

   b) Classify each of X^{HS}, X^{HI} and X^{SI} into the appropriate vehicle color class.

2. Construct the class histograms for the result of Step 1, which are arrays whose indices correspond to each color class.
3. Count the number of pixels in each color class corresponding to each index.

If the number of color classes is n, we can construct an n–dimensional feature vector by arranging the elements in the class histogram in row-major order. There are eight classes: black, silver, white, red, yellow, green, blue, and undefined. The undefined class is for the pixels whose H or S values are not defined. We classify each pixel on the three color planes HS, HI, and SI using Bayesian classifiers, estimating the likelihoods with a Gaussian density function.

Starting with sample pixels for each color class w_j, we can compute a mean vector and a covariance matrix from the set of X^{HS} vectors, and repeat the process for X^{HI} and X^{SI}. Then we can derive the Gaussian densities $p^{HS}(X \mid w_j)$, $p^{HI}(X \mid w_j)$ and $p^{SI}(X \mid w_j)$.

When we classify the pixel into a class on a color plane, the corresponding Gaussian density is used. To classify one pixel into a class on the HS plane, we compute the vector X^{HS} for the pixel, and then classify the pixel into class w_i, where $w_i = \arg\max_j p^{HS}(X^{HS} \mid w_j)$. A similar method is used on the other two color planes.

It would take too long to compute the Gaussian density for every pixel, so we partition each color plane into a 256×256 grid, and use a matrix to obtain the color coordinates for each box in the grid from the H, S, and I values of the sample data. The algorithm for this preprocessing step is as follows:

1. Select a number of hood images for each of the seven colors from the vehicle images. These images are chosen to show the apparent vehicle color.
2. Generate three vectors X^{HS}, X^{HI} and X^{SI} from the H, S, and I values of the pixels in the sample data.
3. For each color class w_j, compute the posterior probabilities for the HS, HI and SI color planes by entering values for Σ_j and U_j in Equation (4), where Σ_j is the covariance matrix and U_j is the mean vector of X^{HS}, X^{HI} and X^{SI}.
4. For each color class w_j, derive Gaussian densities from X^{HS}, X^{HI} and X^{SI}, which we denote by $p^{HS}(X \mid w_j)$, $p^{HI}(X \mid w_j)$ and $p^{SI}(X \mid w_j)$ respectively.
5. Compose three 256×256 matrices M^{HS}, M^{HI} and M^{SI}, which partition colors on the HS, HI, and SI planes, respectively. If h is the hue and s is the saturation, then $M^{HS}[h,s]$ contains the class w_i, where $w_i = \arg\max_j p^{HS}(X \mid w_j)$ and $X = [h,s]$. M^{HI} and M^{SI} are composed in a similar way.

We use the same number of samples for each color class; thus, if $w_i = \arg\max_j p(X \mid w_j)$, then w_i also satisfies $w_i = \arg\max_j p(w_j \mid X)$. Fig. 2 shows how the seven vehicle colors are partitioned by the matrices M^{HS}, M^{HI} and M^{SI}. For instance, if a pixel in the input image has the values $h = 250$, $s = 90$, $i = 128$, then its X^{HS}, X^{HI} and X^{SI} will be classified as red, red, and green, respectively.

Fig. 3 shows how the pixels of an input image are classified into one of the eight classes on the HS plane. In Fig. 3(b), some black, silver, and white pixels are wrongly classified because the feature vector X^{HS} does not include intensity.

When the class histograms generated in the HS, HI and SI color planes are ch^{HS}, ch^{HI}, and ch^{SI}, respectively, each histogram can be represented as an eight-dimensional vector. By combining these vectors, we can generate more feature vectors: V^{HS}, V^{HI}, V^{SI}, $V^{HS,SI}$, $V^{HI,SI}$, $V^{HS,HI}$, and $V^{HS,HI,SI}$.

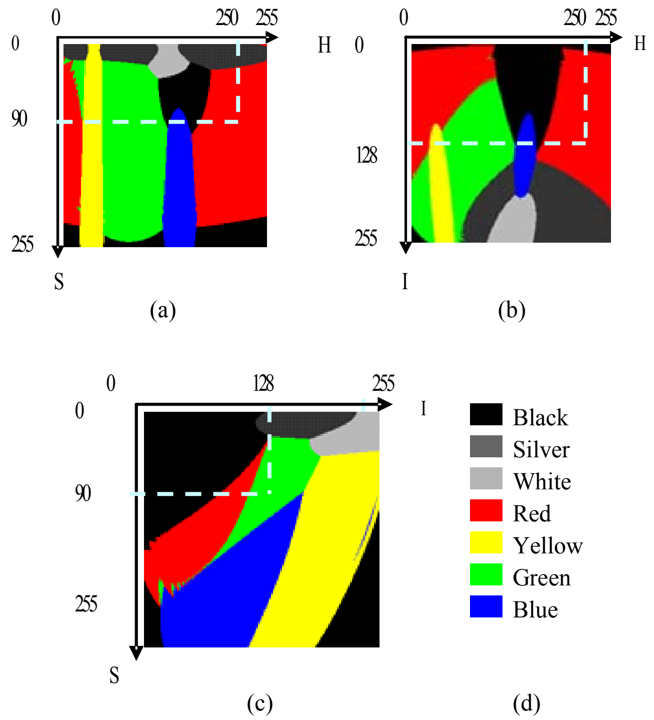


Figure 2. Color planes with their color classes (a) M^{HS}, (b) M^{HI}, (c) M^{SI}, and (d) the key.

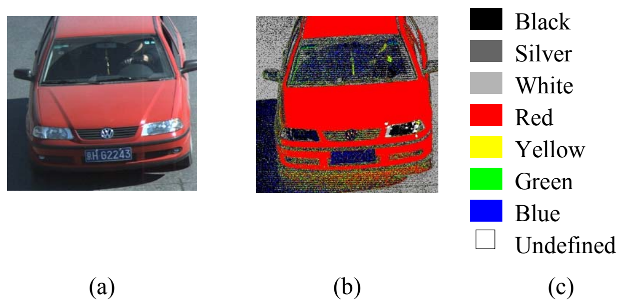


Figure 3. (a) Input image, (b) pixels classified into eight classes on the HS color plane, and (c) the key.

V. REDUCING THE DIMENSIONALITY OF FEATURE VECTORS

Vehicle images were downloaded from various internet sites, and the actual vehicle colors were determined from the original annotation or by inspection. Our data set consisted of 700 images, 100 of each color. Half the images were used as a training data set. The remaining 350 images were used as a test data set. To achieve a stable result, we split the original 700 images at random into ten sets of learning and test data for each color. The successful recognition rates presented in this section are averaged over those 10 data sets.

The experiments were performed on a 3GHz Pentium IV PC and 1GByte of RAM. After computing the feature vectors, an SVM was used for classification. We used $\mathrm{SVM}^{multiclass}$[6], which performs multi-class classification with a linear kernel.

A. *Brute-force method*

We partitioned the HS, HI and SI color planes into 16×16 areas, and generated the h^{HS}, h^{HI} and h^{SI} histograms as 16×16 matrices. Then, we generated three vectors by arranging the elements in each matrix in row-major order. By combining those three vectors, we generated the feature vectors V^{HS}, V^{HI}, V^{SI}, $V^{HS,HI}$, $V^{HS,SI}$, $V^{HI,SI}$ and $V^{HS,HI,SI}$ (See Table I).

Table II shows the rate of successful recognition for each feature vector. When the feature vector $V^{HS,HI,SI}$ is used, the success rate is the highest. Table III shows the success rate for each color. While the success rates for black, blue, and green are lower than 91%, the rate for all the other colors is above 95%. The relatively dark colors are more affected by other dark colors in the adjoining windshield, radiator grill, and the shadow cast by the vehicle on the road.

TABLE II. RECOGNITION RATE FOR THE BRUTE-FORCE METHOD.

Feature vector (dimension)	V^{HS} (257)	V^{HI} (257)	V^{SI} (257)	$V^{HS,HI}$ (514)
Success rate (%)	90.7	88.3	68.7	92.7

TABLE III. RECOGNITION RATE FOR EACH COLOR WITH THE FEATURE VECTOR $V^{HS,HI,SI}$.

Color	Black	Silver	White	Red
Success rate (%)	88.2	96.2	95.0	96.0
Color	Yellow	Green	Blue	Average
Success rate (%)	95.8	91.0	90.6	93.3

B. *Color recognition after dimension reduction*

We compared the performance of the Bayesian classifier with the PCA method. Both methods reduce the dimensions of the feature vectors used by the brute-force method, which are 257, 514, and 771, to 8, 16, and 24 respectively. Table IV shows the rate of successful recognition by PCA, and Table V shows how the naive Bayesian classifier performed.

The success rate of the naive Bayesian classifier is higher than that of the PCA method in all cases. And when the reduced feature vectors are of dimension 24, the success rate of the naive Bayesian classifier is 92.74%, which is only 0.5% lower than that of the brute-force method.

Table VI compares the computation times for the brute-force and naive Bayesian classifiers. The run-time of the SVM is very dependent on the dimensionality of the feature vectors: the reduction in size of the feature vectors is largely responsible for the naive Bayesian classifier, compared to the brute-force method. For instance, the dimension of the feature vector $V^{HS,HI,SI}$ is reduced from 771 to 24, cutting the computation time by a factor of 36.

TABLE IV. RESULTS OF DIMENSION REDUCTION BASED ON THE PCA METHOD.

Feature vector (dimension)	V^{HS} (8)	V^{HI} (8)	V^{SI} (8)	$V^{HS,HI}$ (16)
Success rate (%)	83.31	79.37	52.29	89.43
Feature vector (dimension)	$V^{HS,SI}$ (16)	$V^{HI,SI}$ (16)	$V^{HS,HI,SI}$ (24)	
Success rate (%)	89.83	89.80	91.86	

TABLE V. RESULTS OF DIMENSION REDUCTION BASED ON THE NAÏVE BAYESIAN CLASSIFIER.

Feature vector (dimension)	V^{HS} (8)	V^{HI} (8)	V^{SI} (8)	$V^{HS,HI}$ (16)
Success rate (%)	90.06	84.49	66.00	91.68
Feature vector (dimension)	$V^{HS,SI}$ (16)	$V^{HI,SI}$ (16)	$V^{HS,HI,SI}$ (24)	
Success rate (%)	92.37	90.85	92.74	

TABLE VI. 그림 1 COMPUTATION TIME FOR COLOR RECOGNITION(MS).

Feature vector (dimension)	**Brute-force method**			
Computing time	$V^{HS,HI}$ (514)	$V^{HS,SI}$ (514)	$V^{HI,SI}$ (514)	$V^{HS,HI,SI}$ (771)
Input image	2	2	2	2
Generate a feature vector	71	70	72	71
Save the feature vector	3	4	4	5
Run SVM (including file read/write)	3730	4627	4502	7755
Total time	3806	4703	4580	7833
Feature vector (dimension)	**Naive Bayesian classifier**			
Computing time	$V^{HS,HI}$ (16)	$V^{HS,SI}$ (16)	$V^{HI,SI}$ (16)	$V^{HS,HI,SI}$ (24)
Input image	2	2	2	3
Generate a feature vector	73	73	74	89
Save the feature vector	0	0	0	0
Run SVM (including file read/write)	105	113	109	121
Total time	180	188	185	213

VI. CONCLUSIONS

We use a naive Bayesian classifier to reduce the dimension of feature vectors in color recognition, and applied our technique to vehicle images. Our method performs better performance than PCA-based dimension reduction, which achieving the recognition rate close to that of the brute-force approach, which is very slow. Our classifier could also be applied to applications such as content-based image retrieval, object recognition, and other image-processing problems.

REFERENCES

[1] G. Qiu, X. Feng, and J. Fang, "Compressing histogram representations for automatic colour photo categorization," Pattern Recognition, 37(11): 2177-2193, 2004.

[2] S. Sural, G. Qian, and S. Pramanik, "Segmentation and histogram generation using the HSV color space for image retrieval," Proc. of IEEE International Conference on Image Processing, 2: 589-592, 2002.

[3] M. J. Swain and D. H. Ballard, "Color indexing," International Journal of Computer Vision, 7(1): 11-32, 1991.

[4] O. Chapelle, P. Haffner, and V. N. Vapnik, "Support vector machines for histogram-based image classification," IEEE Trans. on Neural Networks, 10(5): 1055-1064, 1999.

[5] R. O. Duda, P. E. Hart, and D. G. Stork, Pattern Classification, 2nd ed., 2001.

[6] http://www.cs.cornell.edu/People/tj/svm_light/svm_multiclass.html