# FACE AND OBJECT RECOGNITION AND DETECTION USING COLOUR VECTOR QUANTISATION

*Christian J. Walder and Brian C. Lovell*

Intelligent Real-Time Imaging and Sensing Group,
School of Information Technology and Electrical Engineering,
The University of Queensland, Australia, 4072
{walder, lovell}@itee.uq.edu.au

**ABSTRACT**

In this paper we present an approach to face and object detection and recognition based on an extension of the content-based image retrieval method of Lu and Teng (1999). The method applies vector quantisation (VQ) compression to the image stream and uses Mahalonobis weighted Euclidean distance between VQ histograms as the measure of image similarity. This distance measure retains both colour and spatial feature information but has the useful property of being relatively insensitive to changes in scale and rotation. The method is applied to real images for face recognition and face detection applications. Tracking and object detection can be coded relatively efficiently due to the data reduction afforded by VQ compression of the data stream. Additional computational efficiency is obtained through a variation of the tree structured fast VQ algorithm also presented here.

## 1. MOTIVATION

In recent years considerable advances have been made in face recognition technology. In 1998 it was reported that computers can recognize faces better than humans [6], however this result refers to the classification of well segmented face images. The recognition of faces in a cluttered image under varied pose is a more difficult problem. The accurate extraction of a face from a cluttered image is widely considered to be the most difficult and important step in the face recognition process and is the subject of extensive research [5].

Many approaches have been taken in the design of face detection systems. Two distinct types of approach are those that use color and those that use spatial and edge information. A common spatial approach to the face detection and recognition problems is the principal components analysis method, often referred to as eigenfaces [7] in the face recognition context. This is a template matching approach that compares a template of the target to each part of the image. A major shortcoming of this technique is its sensitivity to rotation and scaling. Indeed, Cendrillon[8] found that the eigenface method was effective over only a $12°$ and 12% range of rotation and scale respectively, while Lemieux and Parizeau [10] found the acceptable range to be only $5°$ and 5%. By way of contrast, the simple colour histogram method is both rotation and scale invariant, but also ignores vital spatial information essential for reliable recognition [12]. This paper presents a method which incorporates a mixture of both color and spatial information by moving from a scalar to a vector quantiser. We show that the proposed technique preserves the desirable property of insensitivity to scale and rotation of the simple colour histogram but adds spatial sensitivity. The motivation behind this work is to provide an additional analysis method which has different properties from both the purely color and purely spatial methods to provide another useful mode of analysis.

## 2. BACKGROUND THEORY

### 2.1. Vector Quantisation

Vector Quantisation (VQ) is an extension of scalar quantisation to higher dimensions. VQ maps each $n$-dimensional input vector to the nearest of a set of $B = \{m_i \in \Re^n : i = 1 \dots k\}$ $n$-dimensional output vectors. The set $B$ is usually referred to as the *code-book* and the $k$ elements ($m_i$) of the code-book are referred to as *code-vectors*. For an input vector, $x \in \Re^n$, the output is the index $w$ (the "winner") of the code-vector nearest to $x$. By "nearest" we usually mean the closest vector in a Euclidean distance sense, thus $\|x - m_w\| \leq \|x - m_i\|, i = 1 \dots k$. Several algorithms exist for creating a code-book that will quantise a given data set with low distortion. The algorithm we chose for this study is the Self Organizing Map (SOM) algorithm [2].

### 2.2. VQ Based Image compression

VQ can be used to achieve lossy image compression with an arbitrary compression ratio [3]. The VQ based image compression is achieved by representing blocks of pixels as vectors, quantising them to the nearest codevector, and then representing these blocks by the corresponding index into the code-book. The compression process for an image $I$ is as follows:

1. A VQ code-book is generated from a suitable set of training images using the SOM code-book generating algorithm. Each code-vector is associated with a unique index in the code-book.

2. The image $I$ is divided into disjoint blocks of $k$ by $k$ pixels. Each pixel is represented by an RGB triple.

3. The vectors representing each of these blocks are then quantised using the code-book of step 1. The compressed form of the image consists of the indices of the best matching code-vector for each of the blocks in the image.

4. The input to the decoder is then the matrix of the index numbers of step 3, as well as the code-book of step 1. The decoder retrieves the code vector corresponding to each stored index number, and replaces the corresponding block in the reconstructed image with the code-vector.

## 3. IMAGE CLASSIFICATION

The "feature vector" used for the purpose of image classification is the histogram of the VQ-compressed image representation described in Section 2.2. This is similar to what Lu and Teng [1] used in the domain of content based image retrieval, but we use a single code-book including colour information whereas Lu and Teng use a separate code-book for each color component. This difference is significant because it incorporates more spatial information pertaining to intensity variations in the different color components. The code-book used for the compression is generated using the SOM algorithm, the training data being taken from example images of the target (for example a person's face).

For a VQ compression scheme using a block size of $m$ by $m$ pixels, a 3D image matrix $I$ of dimensions $mh$ by $mw$ by 3 ($m, h, w \in \mathcal{Z}$) will be reduced in size by VQ-compression to a 2D matrix $J$ of size $h$ by $w$, the elements of $J$ being the index numbers of the corresponding code-vectors resulting from the compression process. Denoting the code-book $B = \{c_1, c_2, \ldots, c_k \in \Re^{3m^2}\}$, the histogram $h = \{h_1, h_2, \ldots, h_k\}$ is then computed where $h_i$ is the number of occurrences of code-vector index $i$ in $J$. This histogram is herein referred to as the "VQ-histogram". For classification purposes, we use several training images of each class of image to be classified (e.g., several training images of each person's face) to determine the intraclass variation of the feature vector. Because some histogram bins are more significant than others for classification purposes, we use the Mahalanobis distance between the VQ-histograms as follows.

We use the distance

$$d(D, h) = \sum_{i=1}^{k} \frac{\|m_i - h_i\|}{\sigma_i} \tag{1}$$

where $h = \{h_1, h_2, \ldots, h_k\}$ is the $k$-bin VQ-histogram being compared to the set $D$ of VQ-histograms of a given class of image (e.g., a particular person's face). The set $D = \{h^1, h^2, \ldots, h^n\}$ contains the $n$ histograms of a particular class, where $h^j = \{h^j_1, h^j_2, \ldots, h^j_k\}$ is the $j$-th histogram in the set. $m = \{m_1, m_2, \ldots, m_k\}$ and $\sigma = \{\sigma_1, \sigma_2, \ldots, \sigma_k\}$ are the mean and variance of the $h^j$ in $D$ across $j = 1 \ldots n$. That is, $m_i = \sum_{j=1}^{k} \frac{h^j_i}{k}$, etc. The motivation for the Mahalanobis distance is that the contribution of each bin to this distance should be weighted inversely to the variance of that bin across $D$. This is because high variance is indicative of an unreliable descriptor.

## 4. OBJECT DETECTION

The object detection system is an extension of the image classification system presented in Section 3. The object detection system is based on the "template matching" approach, whereby a "window" is swept around the image and at each point some similarity measure is calculated between the windowed region and the target. The similarity metric used here is exactly that of the image classification system - the Mahalanobis distance of the VQ-histograms.

Before training (calculation of the Mahalanobis means and variances) commences, the training images are scaled to match the size of the window used — a parameter selected by the user. For optimal performance, it would seem that the size of the window should be similar to the expected size of the target in the image to be scanned. Despite this expectation, the system proves to be quite robust to scale changes even with a fixed window size (see Section 6).

We suggest that this insensitivity to scale is probably due to the regular nature of most images of interest which have reasonably large regions of featureless areas of similar colour. A simple way to understand this point is to consider laying a ceramic tile mosaic of an image such as a persons face. Regardless of the size of the face, as long as the tiles were small in size compared to the face the tiler would still use the same set of tile colours in the same proportions. It would largely be the run lengths of same colour tiles that would change with scale. Thus the VQ-histogram measure is relatively insensitive to scale over a wide range.

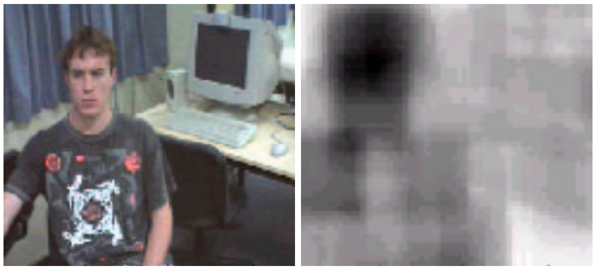

Figure 1: An example of the intermediate steps in the face detection process. The second image is a VQ-histogram distance map of the first image, for the face target.

The efficiency of the algorithm benefits from the fact that the image only needs to be VQ compressed once per frame using the fast VQ algorithm of Section 5. Following this step, the histograms are computed on windows in this compressed format, which decreases computational load. Additionally, the histogram of a region $A$ can be calculated quickly if the histogram of an overlapping region $B$ is known by taking the VQ-histogram of $B$, subtracting the VQ-histogram of $B \cap \bar{A}$ and adding the histogram of $A \cap \bar{B}$.

After sweeping this window around the image and computing the Mahalanobis distance at each point, these distances are stored in a two dimensional distance map. An example distance map is depicted in Figure 1.

A user-defined threshold is applied to the distance map and the centroids of any clusters determined. We use the "subtractive clustering" [4] algorithm because it has the facility to locate multiple clusters without knowing in advance how many to expect.

## 5. FAST VQ ALGORITHM

A well known speedup to the standard vector quantisation algorithm is the Tree Structured VQ (TSVQ) algorithm [3]. This involves generating a VQ code-book from a set of input vectors while forming a balanced tree structure to allow for fast searching. Using this tree structure, the winning code-vector can be approximated very quickly in comparison to the slower (but 100% accurate) "exhaustive" search method. For the system being presented in this paper, however, it was desirable to retain the SOM code-book generating algorithm but then to speed up the quantisation process through a tree indexing scheme. To achieve this the TSVQ algorithm was adapted to produce a tree structure on an existing code-book, as presented below.

In the following, the $I_n$ are the indices of $C^*$ that fall in the Voronoi region[1] of node $n$ with code-book $C_n$. The algorithm constructs a tree in which the $j$th node will contain a code-book $C_j$ and a list of code-vector indices $I_j$ corresponding to code-vectors in $C^*$. Terminal nodes will have code-books of less than $s$ code-vectors, and non-terminal nodes will have $m$ code-vectors.

*Given:* A code-book $C^*$ of $n$ code-vectors to be transformed into an $m$-ary TSVQ structure, and a maximum terminal node code-book size of $s$.

1. Set the index $i$ to 1 (the root node index) and set $I_1 = \{1, 2, \ldots, n\}$ (these are the indices corresponding to the code-vectors in $C^*$).

2. If the number of indices in $I_i$ is greater than $s$ then:

   - Perform the SOM VQ code-book generating algorithm on those code-vectors in $C^*$ corresponding to indices in $I_i$. Do this with the number of output code-vectors constrained to $m$, and set $C_i$ to the resultant code-book. Go to step 3.

   Otherwise:

   - Set $C_i$ to contain all the code-vectors in $C^*$ corresponding to index numbers in $I_i$. (Do not go to step 3).

3. For $j$ in $\{1, 2, \ldots, m\}$ do the following:

   - Create a child node with a previously unused index, $k$, and with $I_i$ as its parent. Set $I_k$ to the indices in $I_i$ that correspond to the code-vectors in $C^*$ that lie in the Voronoi region of the $j$th code-vector of $C_i$. Go to step 2 with $i = k$ (but $i$ stays the same here).

Determining the output code-vector for a given input vector is then a standard tree search using the TSVQ tree structure produced above, as detailed in [3]. Interestingly, increasing the size of the leaf node code-books decreases the efficiency of the algorithm, but reduces the probability of a non-minimum quantisation distortion (the proof of this is straightforward). This property would allow useful tradeoffs in many applications.

## 6. OBJECT DETECTION SYSTEM EXPERIMENTAL RESULTS

Although many performance metrics exist for object tracking systems (e.g. tracking length measure [11]), these metrics are not absolute. Since, for this system, good conditions can allow a *perfect* tracking length measure to be attained, this type of measurement is not used. Instead, several examples are given of the output. These examples appear in

[1] The Voronoi region of a code-vector $c$ is the region in which all (input) vectors are quantised to $c$.

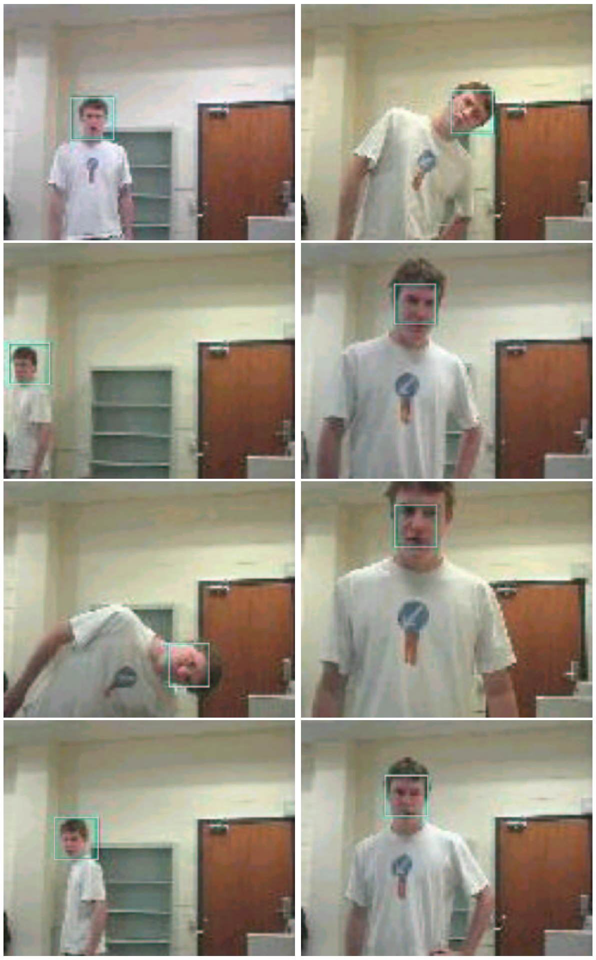

Figure 2: An example of the object detection system output. The sequence demonstrates the capacity of the system to handle rotation and scaling of the target.

Figure 2, where the square box represents the position of the target (the shown person's face) as determined by the object detection system. Additional examples of tracking multiple targets are available on the IRIS web site [9]. These results were obtained using 352 by 288 pixel images, which were processed at approximately 3 frames per second on a Pentium III 600MHz PC.

### 6.1. VQ-Histogram as a Feature Vector

It should be stressed that the VQ-histogram does not bear enough information to replace, say, the eigenface method of face recognition. The method does, however, have some suitable properties for object detection — in particular tolerance to rotation and scale changes. The VQ-histogram is largely a color based method, but the technique does incorporate some spatial information.

## 7. REFERENCES


[1] G. Lu and S. Teng, *A Novel Image Retrieval Technique Based on Vector Quantization*, Proc. Int'l Conf. Computational Intelligence for Modeling, Control and Automation, Vienna, Austria, 1999.

[2] T. Kohonen, *Self-Organizing Maps*, Second Extended Edition, Springer Series in Information sciences, Vol. 2, No. 3, 1994.

[3] A. Gersho and R. Gray, *Vector Quantization and Signal compression*, Kluwer International Series in Engineering and Computer Science, Norwell Massachesetts, 1991, pp410-423.

[4] S. Chiu, Fuzzy Model Identification Based on Cluster Estimation, Journal of Intelligent & Fuzzy Systems, Vol. 2, No. 3, 1994.

[5] R.L. Hsu, M. Adbel-Mottaleb and A.K. Jain, *Face Detection in color Images*, to appear in IEEE Trans. PAMI, May 2002.

[6] W.Barrett, *A Survey of Face Recognition Algorithms and Testing Results*, Conf. Record of the Thirty-First Conf. on Signals, Systems and Computers, IEEE Computer Society Press, Los Alamitos, California, 1998, pp. 301-305.

[7] Turk, M., Pentland, A., *Face Recognition Using Eigenfaces*,, IEEE Conference on Computer Vision and Pattern Recognition, 1992.

[8] R. Cendrillon and B. Lovell, *Real-time face recognition using Eigenfaces*, Proc. of the SPIE International Conference on Visual Communications and Image Processing 2000, Perth, Australia, June 2000.

[9] IRIS, http://www.itee.uq.edu.au/˜iris/

[10] A. Lemieux and M. Parizeau, *Experiments on Eigenfaces Robustness*, Proc. of the International Conference, on Pattern Recognition (ICPR), Qubec, August 11-15, 2002.

[11] *2D-Object Tracking Based on Projection-Histograms* FORWISS-Knowledge Processing Research Group, Bavarian Research Center for Knowledge Base Systems, Am Weichselgarten, Denmark.

[12] T. W. Yoo and I. S. Oh, *A Method for Real-Time Face Region Tracking Based on Color Histogram*, Proceedings of the SPIE, Acquisition, Tracking and