Tools and Techniques for Color Image Retrieval

John R. Smith and Shih-Fu Chang
Columbia University Department of Electrical Engineering and
Center for Telecommunications Research
New York, N.Y. 10027

Abstract

The growth of digital image and video archives is increasing the need for tools that effectively filter and efficiently search through large amounts of visual data. Towards this goal we propose a technique by which the color content of images and videos is automatically extracted to form a class of meta-data that is easily indexed. The color indexing algorithm uses the back-projection of binary color sets to extract color regions from images. This technique provides for both the automated extraction of regions and representation of their color content. It overcomes some of the problems with color histogram techniques such as high-dimensional feature vectors, spatial localization, indexing and distance computation. We present the binary color set back-projection technique and discuss its implementation in the *VisualSEEk* content-based image/video retrieval system for the *World Wide Web*. We also evaluate the retrieval effectiveness of the color set back-projection method and compare its performance to other color image retrieval methods.

Keywords

color image and video databases, color extraction, content-based searching, image features

I. Introduction

New tools and techniques are needed for effective on-line searching and retrieval of images and video. Color indexing is one process by which the images and videos in the database are retrieved on the basis of their color content. A color indexing system requires that several important objectives are satisfied, namely: automated extraction of color, efficient indexing and effective retrieval. In practice it has been difficult to design a system that simultaneously meets all of these goals. In this paper we propose and evaluate a new system for color indexing that provides for automated extraction of local color regions, efficient indexing and excellent query performance. The system uses the back-projection of binary color sets to identify salient color regions in images and video. By way of iteration over a large number of color sets, localized and arbitrarily shaped color regions are extracted. These are indexed directly by color set value which enables very fast indexing of the collection. Furthermore, information extracted from the regions such as the size, shape and position enables a rich variety of queries that specify not only color content but also the spatial relationships and composition of color regions.

II. General Approach

There are two classes of techniques for color indexing: indexing by (1) *global* color distribution and (2) *local* or region color. An important distinction between these techniques is that indexing by global distribution enables only whole images to be compared while regional indexing enables matching between localized regions within images. Both techniques are very useful for retrieval of images and videos but are suited for different types of queries.

Color indexing by global distribution is most useful when user provides a sample image for the query. For example, if the user is interested in finding panoramic shots of a football game between two teams, then by providing one sample of the image others can be found that match the query image. Color indexing by global color distribution works well in this case because the user is not concerned with the positions of colored objects or regions in the images. However, when the user is interested in finding things within images the global color distribution does not provide the means for resolving the spatially localized color regions from the global distribution.

On the other hand, color indexing by localized or regional color provides for partial or sub-image matching between images. For example, if the user is interested in finding all images of sunsets where the sun is setting in the upper left part of the image, then regional indexing enables this query to be answered. Localized or regional color indexing is

This work was supported in part by the National Science Foundation under a CAREER award (IRI-9501266), IBM under a 1995 Research Partnership (Faculty Development) Award, and sponsors of the ADVENT project of Columbia University.

generally more difficult because it requires the effective extraction and representation of local regions. In both cases a system is needed for the automated extraction and efficient representation of color so that the query system provides for effective image and video retrieval. In this paper we present a technique that applies to both global and localized color region extraction and indexing.

The feature extraction system consists of two stages: (1) extraction of regions and (2) extraction of region properties as shown in Figure 1. An automated system performs both stages without human assistance. It is typically stage 1 that is either manual or semi-automatic when the system is not fully automated. The representation of region color is often more robust when the region extraction is not automated.

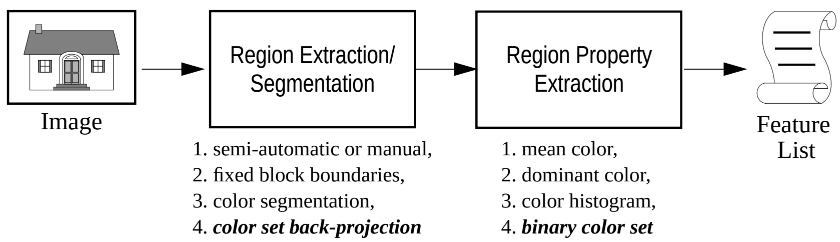

Fig. 1. General approach for color image feature extraction.

A. Region Extraction/Segmentation

The goal of the region extraction system is to obtain the spatial boundaries of the regions that will be of most interest to the user. The process of region extraction differs from image segmentation. Segmentation corresponds to a complete partitioning of the image such that each image point is assigned to one segment. With region extraction an image point may be assigned to many regions or to none. Conceptually, this is more desirable than segmentation because it supports an object-oriented representation of image content. For example, an image point corresponding to the wheel of a car can simultaneously belong to the two different regions that encapsulate respectively the wheel and the car as a whole.

There are several techniques for region extraction. The least complex method involves (1) *manual* or *semi-automated* extraction. In this process the images are evaluated by people and the pertinent information is confirmed or identified visually. This is extremely tedious and time-consuming for large image and video databases. Another procedure for region extraction utilizes a (2) *fixed block segmentation* of the images. By representing color content of small blocks independently there is greater likelihood that matches between regions can be obtained. However, it is difficult to pick the scale at which images should be best blocked. A third technique involves (3) *color segmentation*. There have been several techniques recently proposed for this such as color pairs [CLP94] and foreground object color extraction tools [HCP95].

We propose a new technique which partly employs the color histogram (4) *back-projection* developed by Swain and Ballard for matching images [SB91][SC95]. The basic idea behind the back-projection algorithm is that the most likely location of a spatially localized color histogram within an image is found by the back-projection onto the image of the quotient of the query histogram and the image histogram. More specifically, given query histogram $g[m]$ and image histogram $h[m]$, let $s[m] = \min(g[m]/h[m], 1)$. Then replace each point in the image by the corresponding confidence score $B[m,n] = s[I[m,n]]$. After convolving $B[m,n]$ with a blurring mask, the location of the peak value corresponds to the most likely location of the model histogram within the image. In small image retrieval applications this computation is performed at the time of query to find objects within images [SB91][EM95]. However, for a large collection it is not feasible to compute the back-projection on the fly. A faster color indexing method is needed.

We extend the back-projection to the retrieval from large databases by precomputing for all images in the database the back-projections with predefined color sets. By processing these back-projections ahead of time, the system returns the best matches directly and without new computation at the time of the query. More specifically, we modify the back-projection algorithm so that it back-projects binary color sets onto the images. Instead of blurring the back-projected images we use morphological filtering to identify the color regions. This process is described in greater detail in later sections.

B. Region Feature Extraction

Once the image regions are identified, each region is characterized and represented using a feature set. The goal of color representation is to accurately capture the salient color characteristics of the region in a low-dimensional vector. Two choices for simple color features are (1) *mean color* and (2) *dominant* color of the region. They both use only a 3-D vector to represent the color of the region. The comparison measurement is fairly easy to compute but the discrimination is inadequate. An alternative is to use (3) *color histograms*. A color histogram is a high-dimensional feature vector typically having greater than 100 dimensions and the comparison of histograms is computationally intensive. They are best suited for representation of global color rather than local color regions because of storage requirements and the large number of computations required at query time.

We propose a new approach for representation of color content which is well matched to the binary color set back-projection region extraction. We represent region color using a (4) *binary color set* that selects only those colors which are sufficiently present in the region. Since the color sets are binary vectors, they can be indexed efficiently, i.e., using a binary tree, rather than requiring substantial computations to compare histograms at query time. The binary set representation of color regions and indexing is described in detail in the following sections.

C. Color Image Retrieval Systems

Recently, several systems have appeared for the content-based retrieval of images. The *QBIC* system [FFN+93] provides for the retrieval of images by color, texture and shape. It supports two classes of color content – global and local. In the *QBIC* system the extraction of local regions is handled manually by requiring a person to draw region boundaries using a mouse. Both the global and local color information is represented by mean color and color histogram. The *QBIC* system also uses a quadratic distance metric for comparing histograms. Because the quadratic distance measure is very computationally intensive, the mean color distance is used as a pre-filter for color queries [HSE+95].

D. VisualSEEk

We are currently developing *VisualSEEk*: a content-based image/video retrieval system for the *World Wide Web*. *VisualSEEk* supports query by *color* and *texture* and *spatial-layout*. Other features such as *shape*, *motion* and *embedded text* will be incorporated in the near future. The system (see Figure 2) provides the user with a highly functional and platform independent *Java* user-interface which collects the user's query. The query is communicated to the *VisualSEEk* server on the *World Wide Web* through the Common Gateway Interface (CGI). The server answers the user's query by accessing the extracted meta-data that describes the images and videos in the archive.

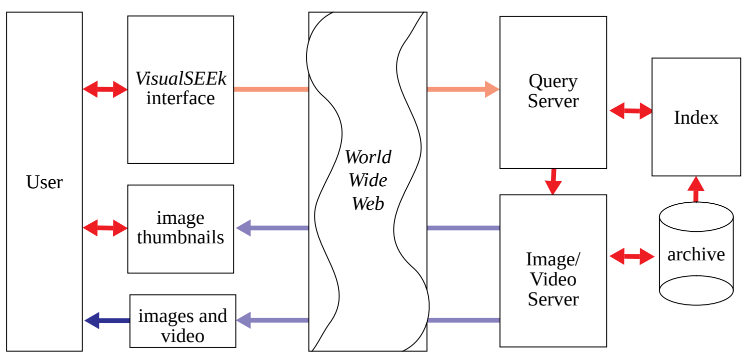

Fig. 2. *VisualSEEk* external system operation.

The *VisualSEEk* system emphasizes several research goals – automated extraction of localized regions and features, efficient representation of features, preservation of spatial properties, extraction from compressed data [Cha95][CS95] and fast indexing and retrieval. The color indexing component of the system uses the binary color set back-projection algorithm to extract color regions. The user may search for images using both global and local features. For a local color region query the user indicates the locations of the color regions by positioning regions on the query grid (see Figure 3(a)). The returned images can be used for a global query whereby the user selects one image and uses the *give me more* feature of *VisualSEEk* to retrieve images that best match the selected one in a specified way (see Figure 3(b)).

III. Color Sets

Color sets provide an alternative to color histograms for representation of the color content of images and regions. The following paragraphs define color sets and explain their relationship to color histograms.

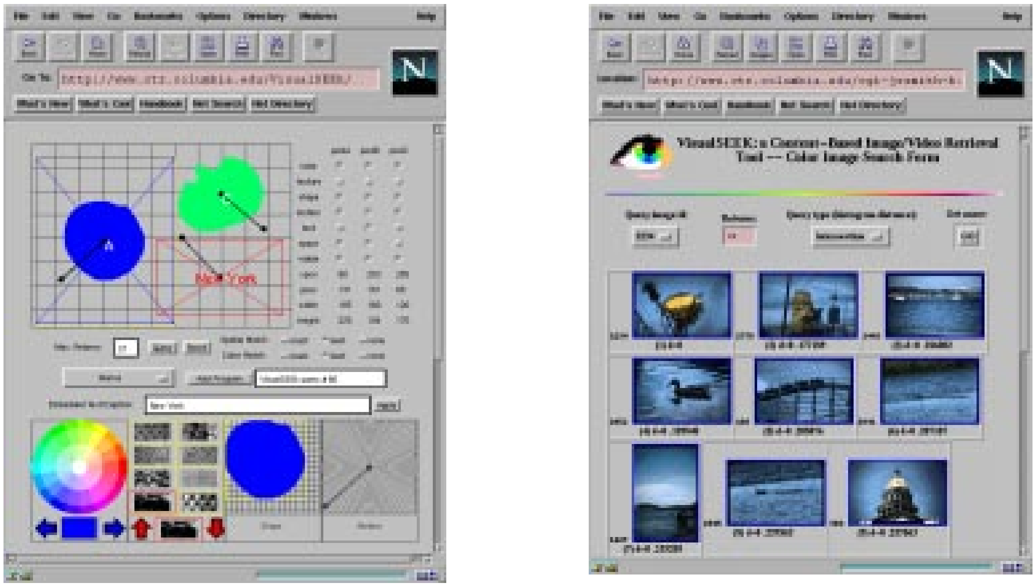

Fig. 3. (a) *VisualSEEk* Java user interface and (b) query returns with *give me more* feature of *VisualSEEk*.

A. Color Histogram

A color histogram denotes the joint probabilities of the intensities of the three color channels. The color histogram is defined as

$$h_{R,G,B}[r,b,g] = N \cdot Prob\{R = r, G = g, B = b\} \tag{1}$$

where R, G and B are the three color channels and N is the number of pixels in the image. The color histogram is computed by discretizing the colors within the image and counting the number of pixels of each color. Since the number of colors is finite, it is usually more convenient to transform the three channel histogram into a single variable histogram. Given an RGB image, one transform is given by $m = r + N_r g + N_r N_g b$, where N_r, N_g and N_b are the number of bins for colors red, blue and green, respectively. This gives the single variable histogram

$$h[m] = N \cdot Prob\{M = m\}. \tag{2}$$

B. Color Set Notation

B.1 Transformation

The color set representation is defined as follows: given the color triple (r, g, b) in the 3-D RGB color space and the transform T between RGB and a generic color space denoted XYZ, for each (r, g, b) let the triple (x, y, z) represent the transformed color such that

$$(x, y, z) = T(r, g, b). \tag{3}$$

B.2 Quantization

Let Q_M be a vector quantizer function that maps a triple (x, y, z) to one of M bins. Then m, where $m \in \{0 \ldots M-1\}$, is the index of color (x, y, z) assigned by the quantizer function and is given by

$$m = Q_M(x, y, z). \tag{4}$$

B.3 Binary Color Space

Let $\mathcal{B}^M$ be an M dimensional binary space that corresponds to the index produced by Q_M such that each index value m corresponds to one axis in $\mathcal{B}^M$.

Definition 1: Color Set, $\hat{c}$. A color set is a binary vector in $\mathcal{B}^M$. A color set corresponds to a selection of colors from the quantized color space.

For example, let T transform RGB to HSV and let Q_M where $M = 8$ *vector quantize* the HSV color space to 2 hues, 2 saturations and 2 values. Q_M assigns a unique index m to each quantized HSV color. Then $\mathcal{B}^8$ is the 8-dimensional binary space whereby each axis in $\mathcal{B}^8$ corresponds one of the quantized HSV colors. Then a color set $\hat{c}$ contains a selection from the 8 colors. If the color set corresponds to a unit length binary vector then one color is selected. If a color set has more than one non-zero value then several colors are selected. For example, the color set $\hat{c} = (10010100)$ corresponds to the selection of three colors, $m = 7, m = 4$ and $m = 2$, from the quantized HSV color space.

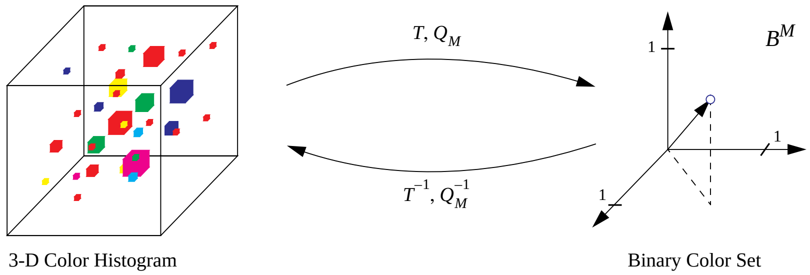

Fig. 4. Transformation between 3-D color histogram and binary color set.

C. Color Representation

In order for color sets to provide a useful characterization of region color, several objectives must be met. Firstly, each color in the binary color space $\mathcal{B}^M$ must be visually distinguishable from the others. Secondly, the binary color space $\mathcal{B}^M$ must satisfactorily include all distinguishable colors. The condition is usually application specific. For example, medical and satellite images have quite different sets of distinguishable colors. The transformation T and quantization Q_M are designed to produce a representation that meets these objectives.

C.1 Color Space

The first design parameter is the transformation T by which the new color space is reached from the RGB space. The RGB space has the major deficiency of not being perceptually uniform. Other color spaces such as $CIE-LAB$, $CIE-LUV$ and $Munsell$ offer improved perceptual uniformity [WS82]. In general they represent with equal emphasis the three color variants that characterize color: *hue*, *lightness* and *saturation*. This separation is attractive because color image processing performed independently on the color channels does not introduce false colors (hues) [Rus95]. Furthermore, it is easier to compensate for artifacts and color distortions. For example, lighting and shading artifacts are typically isolated to the lightness channel. However, these color spaces are often inconvenient due to the necessary non-linearity in forward and reverse transformations with RGB space. We utilize a tractable form of the *hue*, *lightness* and *saturation* transform from RGB to HSV that has the above mentioned characteristics and is non-linear but easily invertible. The transformation T from RGB to HSV is accomplished through the following equations [Hun89]: let the color triple (r, g, b) define a color in RGB space and let (h, s, v) be the transformed triple in HSV color space. For $r, g, b \in [0 \ldots 1]$ then T gives $h \in [0 \ldots 6]$ and $s, v \in [0 \ldots 1]$ by

$$\mathbf{v} = \max(r, g, b), \qquad \mathbf{s} = \frac{\mathbf{v} - \min(r, b, g)}{\mathbf{v}}$$

$$\text{let } \acute{r} = \frac{\mathbf{v} - r}{\mathbf{v} - \min(r, b, g)}, \qquad \acute{g} = \frac{\mathbf{v} - g}{\mathbf{v} - \min(r, b, g)}, \qquad \acute{b} = \frac{\mathbf{v} - b}{\mathbf{v} - \min(r, b, g)}$$

$$\mathbf{h} = \begin{cases} 5 + \acute{b} & \text{if } r = \max(r, g, b) \text{ and } g = \min(r, b, g) \\ 1 - \acute{g} & \text{if } r = \max(r, g, b) \text{ and } g \neq \min(r, b, g) \\ 1 + \acute{r} & \text{if } g = \max(r, g, b) \text{ and } b = \min(r, b, g) \\ 3 - \acute{b} & \text{if } g = \max(r, g, b) \text{ and } b \neq \min(r, b, g) \\ 3 + \acute{g} & \text{if } b = \max(r, g, b) \text{ and } r = \min(r, b, g) \\ 5 - \acute{r} & \text{otherwise} \end{cases} \tag{5}$$

C.2 Quantization

The next design parameter is the quantization Q_M of the color space. The HSV color space is cylindrical, as illustrated in Figure 5. The long axis represents *value*: blackness to whiteness. Distance from the axis represents *saturation*: amount of color present. The angle around the axis is the *hue*: tint or tone. Since hue represents the most significant characteristic of the color, it requires the most fine quantization. In the hue circle the primaries red, green and blue are separated by 120 degrees. A circular quantization at 20 degree steps sufficiently separates the hues such that the three primaries and yellow, magenta and cyan are represented each with three sub-divisions. Saturation and value are each quantized to three levels yielding greater perceptual tolerance along these dimensions.

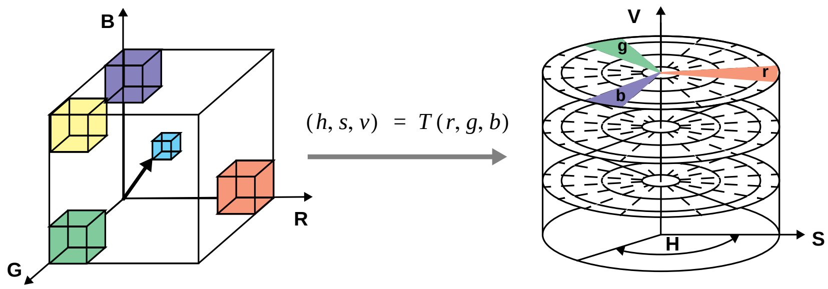

Fig. 5. Transformation T from RGB to HSV and quantization gives 18 *hues*, 3 *saturations*, 3 *values* and 4 *grays* = 166 colors.

IV. Color Set Back-Projection

The extraction system for color regions and their features has four stages: (1) transformation, quantization and filtering of the image, (2) back-projection of binary color sets onto the image, (3) region labeling and thresholding and (4) region feature extraction (see Figure 6). The goal of system is to reduce the color gamut and the insignificant color information in the image and emphasize the prominent color regions.

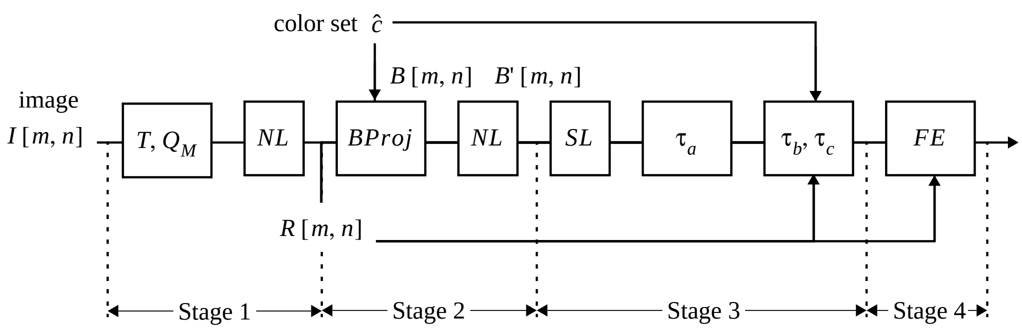

Fig. 6. The color region and feature extraction system, T = color space transformation, Q_M = color space vector quantization, $BProj$ = back-projection, NL = non-linear filter, SL = sequential labeling, τ = thresholding and FE = feature extraction.

A. Stage 1 – Color Processing

As illustrated in Figure 7, in the first stage T and Q_M reduce and reorganize the color space. The color images are first low-pass filtered and subsampled to approximately 128 x 128 pixels. The insignificant detail is further reduced in the image by non-linear filtering. This processing is accomplished using a median filter on each of the HSV channels of the subsampled image. The non-linear median filtering in HSV space eliminates outliers and emphasizes prominent color regions while preserving boundary information and avoiding the introduction of false hues.

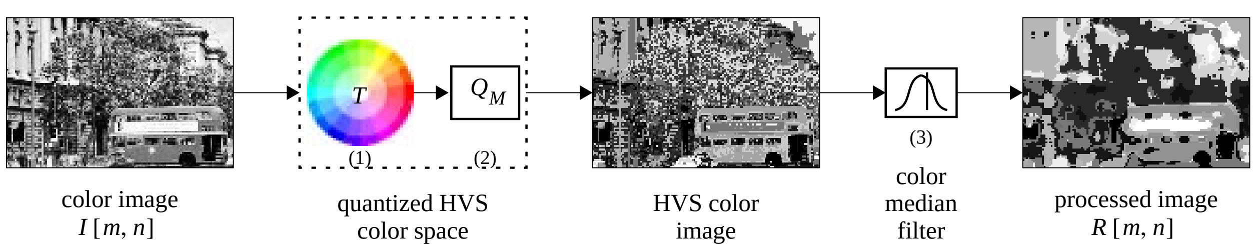

Fig. 7. First stage of color region extraction: (1) transformation T of RGB color image $I[m,n]$ to HSV color space, (2) quantization Q_M and (3) color median filtering. The processed color image $R[m,n]$ has dominant color regions emphasized.

B. Stage 2 – Color Set Back-Projection

The next stage involves the extraction of the color regions from the images and is illustrated in Figure 8. This is done by systematically iterating over color sets for the image. The back-projection of the binary color set onto the image produces a bi-level image $B[m,n]$. The levels correspond to the selected and un-selected pixels for the specified color set.

After back-projection the bi-level image is filtered morphologically to (1) eliminate spots and (2) connect neighboring regions. The binary color set back-projection algorithm is given as follows: given binary color set $\hat{c}$ and processed image $R[m,n]$:

1. Generate the bi-level image $B[m,n]$ from back-projection of color set $\hat{c}$: let $k = R[m,n]$ then $B[m,n] = c[k]$.
2. Filter $B[m,n]$ with the 5×5 box median filter $U[m,n]$ to produce $B'[m,n]$. Since $B[m,n]$ is a bi-level image, the median statistic is equivalent to the majority computation.

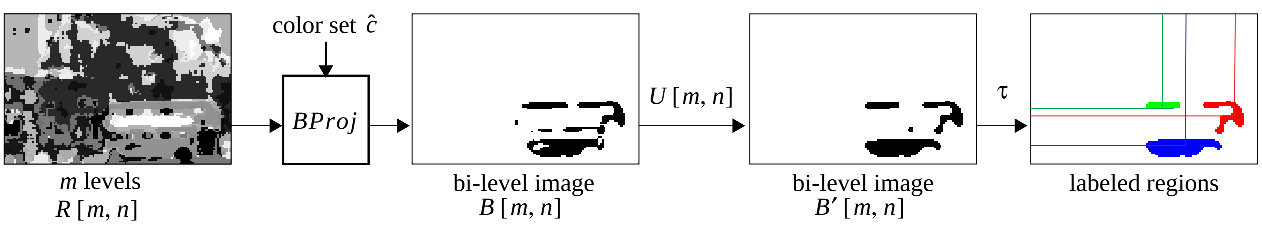

Fig. 8. Second and third stages of color region extraction: (1) back-projection of color set $\hat{c}$ (in this example $\hat{c}$ has one color selected) onto processed image $R[m,n]$ produces bi-level image $B[m,n]$, (2) filtering with $U[m,n]$ produces $B'[m,n]$ and (3) thresholding determines surviving regions.

C. Stage 3 – Back-Projection Thresholding

After the back-projection with the color set, the filtered image $B'[m,n]$ is analyzed. The first step in this analysis is a sequential labeling algorithm applied to $B'[m,n]$ that assigns each isolated region a unique *non-zero* label. Each uniquely labeled region is analyzed in regard to several thresholds. The motivation for these thresholds is to extract regions that meet minimum size constraints and provide a satisfactory representation of the colors in the color set $\hat{c}$ that produced the $B'[m,n]$. If a region does not meet all the thresholds then it is assigned the *zero* label. The thresholds are described next and are summarized in Table I. These threshold values are appropriate for the full size color images and are rescaled to correspond to the subsampled image size.

The threshold τ_a corresponds to region size. The region must contain more than $\tau_a = 32^2$ pixels to be significant. This value allows for sufficiently small regions to be extracted. The second threshold τ_b corresponds to the region's absolute representation of each color in the color set. If a selected color from the color set does not contribute at least $\tau_b = 32^2$ pixels to the region then it is not added. The third threshold τ_c corresponds to the region's relative representation of each selected color in the color set. It requires that each selected color from the color set contributes at least $\tau_c = 20\%$ of the region area.

$H[m]$ = Image histogram, $\hat{c}$ = binary color set, $L[m]$ = local region histogram.

threshold	*value*	*condition*	*description*
τ_a	1024	$\sum_n L[n] \geq \tau_a$	region area
τ_b	1024	$c[m] = 1 \rightarrow L[m] \geq \tau_b$	absolute contribution of color m to region
τ_c	0.2	$c[m] = 1 \rightarrow L[m] \geq \tau_c \sum_n L[n]$	relative contribution of color m to region
τ_0	1024	$c[m] = 1 \rightarrow H[m] \geq \tau_0$	global histogram
τ_1	1024	$c[m] = 1$ and $H[m] \geq \tau_0 \rightarrow H[m] - \sum_n L_n[m] \geq \tau_1$	global histogram residue

TABLE I

Color region and region extraction thresholds for 128 x 128 color image.

If an extracted color region does not pass one of the above thresholds then it will not be indexed by that color set. If a region is rejected because one of the selected colors from the color set is not sufficiently represented in the region, then the region may still be extracted using a reduced color set that omits the under-represented color. Enforcing the color set thresholds prevents the unnecessary and redundant extraction of multiple-color regions.

D. Iteration over Color Sets

To help prune the iteration over the great number of possible color sets we utilize two additional thresholds in the iteration system. The algorithm back-projects only those color sets $\hat{c}$ that meet minimum support constraints. The color set $\hat{c}$ is back-projected onto the image only if thresholds are met that predict whether the back-projection will produce new and desirable regions.

D.1 Threshold τ_0

The threshold τ_0 corresponds to the image's support for each color in the color set. A color set $\hat{c}$ used for the back-projection only if for m such that $c[m] = 1$ there are at least τ_0 pixels in the image of color m. In other words, if color m is selected in the color set then color m must be present in the global histogram with a support of τ_0 samples.

D.2 Threshold τ_1

The threshold τ_1 corresponds to the un-allocated colors from the global histogram. After a color set is back-projected and regions are extracted, the colors from the extracted regions are subtracted from the global image histogram. A color set $\hat{c}$ is used for the back-projection only if for m such that $c[m] = 1$ there are at least τ_1 pixels in the global residue histogram of the color m. In other words, if a color set includes color m then τ_0 samples of color m must be still be un-allocated to previously extracted regions. This threshold works well when the color sets are explored in order of increasing magnitude. For example, single color – color sets are explored first, followed by two color – color sets, etc. This gives desirable results because color regions are extracted in order of dominance.

If τ_0 is not met then $\hat{c}$ contains colors that are not represented sufficiently by any region. The back-projection of this $\hat{c}$ and all its super-sets are non-generative. If τ_0 is met while τ_1 is not met then at query time a color region containing all of the colors in $\hat{c}$ can be reconstructed using subsets of $\hat{c}$ and spatial composition. Therefore, exploration of $\hat{c}$ and its super-sets generates redundant information. The color set iteration procedure is summarized as follows:

1. Single color – color sets
   (a) Given the image histogram $H[m]$ find all $\acute{m} = m$ where $H[m] \geq \tau_0$.
   (b) For each $\acute{m}$ construct unit length $\hat{c}$ where $c[k] = 1$ for $k = \acute{m}$ and $c[k] = 0$ otherwise. Back-project each $\hat{c}$ onto image $R[m, n]$ and extract regions. For each region n record the local region histogram $L_n[m]$.
   (c) Compute residue histogram $H_r[m] = H[m] - \sum_n L_n[m]$.
2. Two color – color sets
   (a) Find all $\acute{l} = l$ and $\acute{m} = m$ where $l \neq m$, $H[l] \geq \tau_0$, $H[m] \geq \tau_0$ and $H_r[l] \geq \tau_1$, $H_r[m] \geq \tau_1$.
   (b) For each pair $\acute{l}$ and $\acute{m}$ construct $\hat{c}$ where $c[k] = 1$ for $k = \acute{l}$ or $k = \acute{m}$ and $c[k] = 0$ otherwise. Back-project each $\hat{c}$ onto image $R[m, n]$ and extract regions. For each region n record the local region histogram $L_n[m]$.
   (c) Update residue histogram $H_r[m] = H_r[m] - \sum_n L_n[m]$.
3. Three color – color sets, etc., repeat above.

E. Stage 4 – Region Feature Extraction

After the analysis of $B'[m, n]$ in regard to the above mentioned thresholds the features are extracted from all remaining *non-zero* labeled regions. For each of these regions the generative color set $\hat{c}$ is recorded and the region area and location are measured. The region location is represented by the minimum bounding rectangle (MBR) that encloses the region. The features are added to the feature table which compiles the meta-data to be used to answer the color region queries.

Color indexing using the binary color sets is accomplished by indexing the color region feature table on the color set attribute. Since this field contains a binary vector, a binary tree is an appropriate index structure. In a straight-forward color set query such as *find all images that have a region with precisely this color set*, the matches are found by looking up the color set value in the index and reading off the images. However, more interesting queries specify several color regions and their spatial layout.

The *VisualSEEk* system presently supports queries that include up to three color regions and their spatial positions. The system allows the user to select priorities for matching color and spatial positions to the items in the database. *VisualSEEk* allows the user to specify whether the color matches are to be *exact* match or *best* match. Exact match means the color sets must be exactly equal for a match. For best match the system also examines nearby color sets and assigns matches a color distance so that the system returns items in order of color set distance. *VisualSEEk* also allows the user to specify whether the spatial locations in the query specify are *exact* or *best* locations or *none*. For best match the system assigns a spatial distance depending on the drift from the locations specified in the query. In the case of *none* the system ignores the location parameters. The system also allows the user to specify the maximum number of images $\mathcal{N}$ to be returned by the query.

F. Color Set Distance

The distance between color sets utilizes the *walk* distance in HSV color space to determine the distance between individual colors. For the 166 colors in the binary color space $\mathcal{B}^{166}$ a table with entries d_{m_0,m_1} is constructed that records the distance between all individual colors. That distance is computed as follows: given two colors corresponding to axes m_0 and m_1 in $\mathcal{B}^{166}$ retrieve the HSV values through the inverse of vector quantization Q_M. This gives $(\tilde{h}_i, \tilde{s}_i, \tilde{v}_i) = Q_M^{-1}(m_i)$ for $i = 0, 1$, where $\tilde{h}_i \in \{0 \ldots 18\}$, $\tilde{s}_i \in \{0, 1, 2\}$ and $\tilde{v}_i \in \{0, 1, 2\}$. Then the distance between the colors corresponding to m_0 and m_1 is given by

$$d_{m_0,m_1} = \min(|\tilde{h}_0 - \tilde{h}_1|, 18 - |\tilde{h}_0 - \tilde{h}_1|) + |\tilde{s}_0 - \tilde{s}_1| + |\tilde{v}_0 - \tilde{v}_1|. \tag{6}$$

The total distance between two unit length (single color) color sets $\hat{c}_0$ and $\hat{c}_1$ is given by

$$d_c(\hat{c}_0, \hat{c}_1) = \sum_{m_0=0}^{M-1} \sum_{m_1=0}^{M-1} c_0[m_0] \cdot d_{m_0,m_1} \cdot c_1[m_1]. \tag{7}$$

The *VisualSEEk* system currently supports the assignment of a single color (unit length $\hat{c}$) to each region in the query. The assignment in the query of multiple colors to each region and the more general formula for distance between any two non-zero color sets has not yet been implemented in the system.

G. Query Color Set Distance and Back-off

To answer a color set query that does not specify exact match, a color set distance *back-off* is used. The idea is to return first the zero distance items $d_c = 0$ then the $d_c = 1$ items, etc., until $\mathcal{N}$ images are found. For the $d_c = 0$ matches the specified color set is used verbatim. For the $d_c = 1$ matches the group of color sets that are within distance *one* from the query color set are found. Then each of these color sets is looked-up in the index and the matches are read off. This is repeated for the $d_c = 2$ matches, etc., until $\mathcal{N}$ images are retrieved. By using this method we avoid an exhaustive search of all regions in the database for each specified region in the query.

When several color regions are specified in the query the individual color region distances are summed. This computes the overall query color distance value D_c for each image match using $D_c = \sum_{k=0}^{K-1} d_c^k$ where K is the number of regions in the query. The current implementation requires that matches are found for all specified regions in the query in order for an image to be considered as a match.

H. Spatial Layout Distance and Back-off

The matches in spatial layout are found by assigning each matched color region from an image in the database to one of three spatial classes with corresponding spatial distances: (1) *encapsulation*: $d_s = 0$, query region completely completely encloses the image region, (2) *intersection*: $d_s = 1$, query region intersects with the border of the image region, (3) *disjoin*: $d_s = 2$, query region does not overlap with the image region. The overall image spatial distance D_s is computed by summing the individual region's spatial distance scores using $D_s = \sum_{k=0}^{K-1} d_s^k$ where K is the number of regions in the query. The query server assigns a total match score to each image match with K regions. The images are returned to the user in lowest match score order. The match score D_{match} is given by

$$D_{match} = D_c + D_s = \sum_{i=0}^{K-1} d_c^i + \sum_{j=0}^{K-1} d_s^j = \sum_{k=0}^{K-1} (d_c^k + d_s^k). \tag{8}$$

V. Sample Image Queries

The color set region extraction and representation method is evaluated by conducting a variety of image retrieval experiments. The color set method is compared to other color histogram approaches for color indexing. The query examples show how both global and local color indexing complement each other when integrated into a complete system for color image retrieval.

A. Retrieval Effectiveness

Two metrics for retrieval effectiveness are *recall* and *precision*. Recall signifies the proportion of relevant images in the database that are retrieved in response to a query. Precision is the proportion of the retrieved images that are relevant to the query [Jon81]. More precisely, let A be the set of relevant items, let B be the set of retrieved items and a, b, c and d are given in Figure 9 then recall and precision are defined as the following conditional probabilities:

$$\text{recall} \;= P(B|A) = \frac{P(A \cup B)}{P(A)} = \frac{a}{a+c} \;\text{ and }\; \text{precision} \;= P(A|B) = \frac{P(A \cup B)}{P(B)} = \frac{a}{a+b}. \qquad (9)$$

As indicated by the formulas the measures of recall and precision require that the relevance to the query of each item in the database is established ahead of time. For the experiments this was done for each query by subjectively assigning one of three values to each image in the database: relevant = 1, partially or marginally relevant = 0.5 and not relevant = 0. The relevance value corresponds to the probability that the item is relevant to the query.

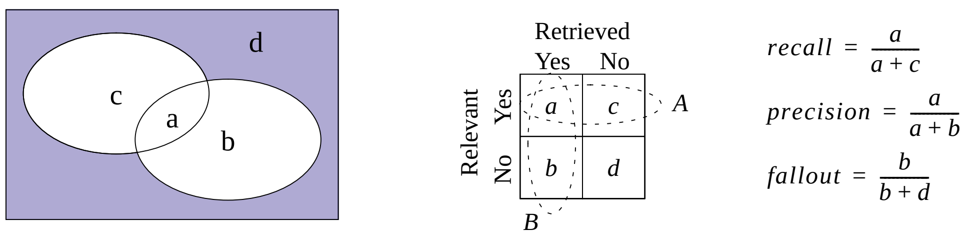

Fig. 9. Retrieval effectiveness: $a + b + c + d$ = full database, $a + c$ = relevant items and $a + b$ = retrieved items.

B. Color Retrieval Techniques

Several color indexing methods that use color histograms were also evaluated to provide comparison for the color sets method. Four image retrieval techniques were used in the queries: (1) euclidean histogram distance, (2) histogram intersection, (3) quadratic histogram distance and (4) color sets.

B.1 Histogram Euclidean Distance

The euclidean distance between two color histograms h and g is given by

$$d_E(h,g) = \sum_{m=0}^{M-1} (h[m] - g[m])^2 \qquad (10)$$

In this distance formula there is only comparison between the identical bins in the respective histograms. Two different bins may represent perceptually similar colors but are not compared cross-wise. All bins contribute equally to the distance.

B.2 Histogram Intersection

The color histogram intersection was proposed for color image retrieval in [SB91]. The intersection of histograms h and g is given by:

$$d_I(h,g) = \frac{\sum_{m=0}^{M-1} \min(h[m], g[m])}{\min(\sum_{m_0=0}^{M-1} h[m_0], \sum_{m_1=0}^{M-1} g[m_1])} \qquad (11)$$

Colors not present in the user's query image do not contribute to the intersection. This formulation differs from that proposed by Swain and Ballard [SB91] in order to make it a true distance metric. The intersection formula in [SB91] is not symmetric in h and g and therefore is not a distance metric.

B.3 Histogram Quadratic Distance

The color histogram quadratic distance is used by the *QBIC* system [FFN+93]. The quadratic distance between histograms h and g is given by:

$$d_Q(h,g) = \sum_{m_0=0}^{M-1} \sum_{m_1=0}^{M-1} (h[m_0] - g[m_0]) \cdot a_{m_0,m_1} \cdot (h[m_1] - g[m_1]) = -\sum_{m_0=0}^{M-1} \sum_{m_1=0}^{M-1} (h[m_0] - g[m_0]) \cdot d_{m_0,m_1} \cdot (h[m_1] - g[m_1]). \qquad (12)$$

The quadratic distance formula uses the cross-correlation a_{m_0,m_1} between histogram bins based on the perceptual similarity of the colors m_0 and m_1. One appropriate value for the cross-correlation is given by $a_{m_0,m_1} = 1 - d_{m_0,m_1}$ where d_{m_0,m_1} is the distance between color m_0 and m_1 normalized with respect to the maximum distance. In this case the histogram quadratic distance formula reduces to the form on the right [HSE+95].

C. Sample Queries

We evaluated the retrieval effectiveness of the color indexing approaches on a test set of 3100 color images. The test set includes color images from a variety of subjects such as cities, nature, animals and transportation. For each query the retrieval effectiveness using recall and precision was recorded. The overall query process is illustrated in Figure 10. At each round in the query process the user can (1) initiate a new color set query using the interface tools or (2) use one of the retrieved images as a seed image. After L rounds the user zeros in on the desired image.

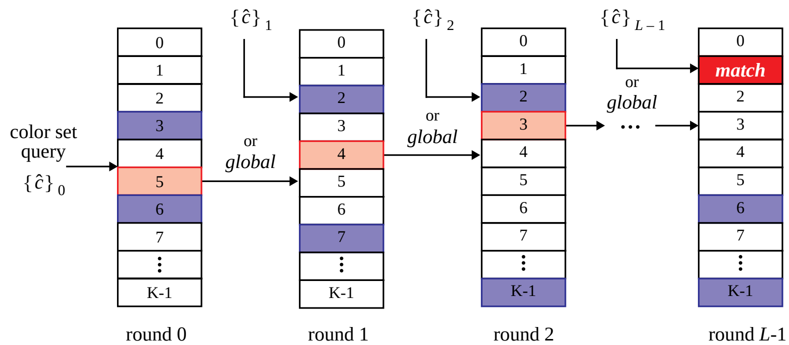

Fig. 10. Query paths for image queries in *VisualSEEk*, $\mathcal{N} = 3$. Starting from color set query $\{\hat{c}\}_0$ constructed using interface tools, *round 0* yields $\mathcal{N}$ images. After viewing the matches the user makes decision to either (1) formulate a new color set query $\{\hat{c}\}_1$ or (2) use a returned image for a *give me more* color query to find similar images. The process is repeated until desired image is found at round $L-1$.

C.1 Query 1 - *Purple* Regions

The goal of this query is to retrieve all images that contain a significant *purple* region somewhere in the image. Prior to the trials each of the 3100 images was inspected and assigned a relevance to the purple region query. A three-value relevance was used. The query results are shown in Figure 11(a). The color set indexing method out-performed all others in both retrieval effectiveness and query response time. Furthermore, the color set method was able to retrieve all 97 purple region images with an overall precision of 0.625. For the color histogram queries a seed image was picked from the collection by the user that was most "typical" of the desired images. The color set query was formulated in *VisualSEEk* by simply drawing one purple region and by selecting *best* match in color and *none* for match in space.

C.2 Query 2 - *Sunsets*

The goal of this query is to retrieve all images of *sunsets* which come in various flavors. A sunset image was defined to contain a glowing horizon with the sun still partially visible. Prior to the trials each of the 3100 images was inspected and assigned a relevance. The query results are shown in Figure 11(b). The results are interesting. Since the sunset images in the archive do not have a typical spatial layout of color regions, the color set retrieval precision was initially low, also returning many images of yellow and orange flowers. When a "typical" sunset image was used to seed the color histogram queries the retrieval effectiveness was significantly better. However, if the user desires that *all* sunset images are to be returned with the highest possible precision then the color set method is far superior. In order to return all 83 sunsets using color histograms nearly all (3100) images must be returned. However, using color sets all sunset images are found with a precision of 0.2 (415 returned images).

C.3 Query 3 - *Find the Image*

The goal of these two queries is to find particular images: (1) image with a *red sports car* and (2) image with a *British double decker bus* (see Figure 7). The user was shown the color images and asked to use the system to compose rounds of either (1) color set or (2) *give me more* color histogram queries to find them. The user was not given an appropriate initial seed image and had to pick from 21 randomly supplied images. The same set was supplied for all queries. In all query rounds the return number $\mathcal{N} = 21$ was used. At each round l, $\mathcal{M}_l$ out of $\mathcal{N}$ images are new matches not returned in rounds $0 \ldots l-1$. The trials are summarized below. The color set method does very well in these types of queries.

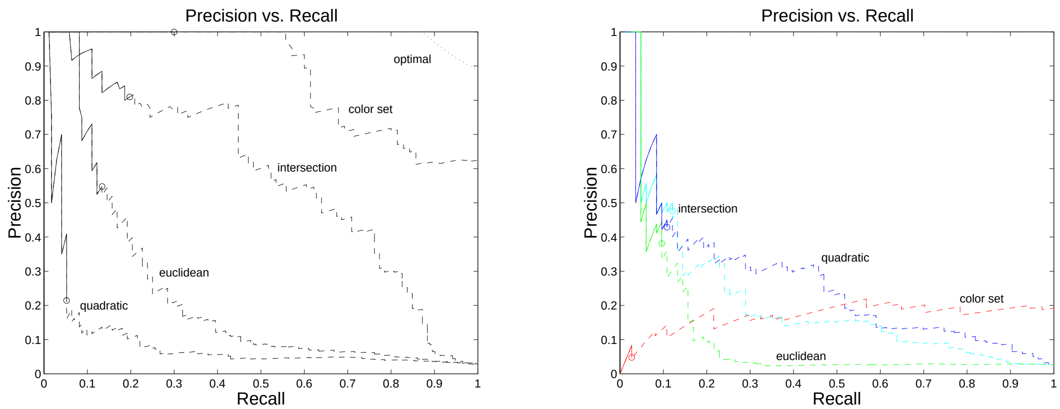

Fig. 11. Retrieval effectiveness (a) Query 1 – Purple regions, the symbol 'o' marks the recall and precision values at point for $\mathcal{N} = 21$ and (b) Query 2 – Sunsets.

The color histogram retrievals are largely dependent on the seed images. It takes more rounds to find the desired images using only *give me more* queries. The best overall performance combines both type of queries: local color set and global histogram. In this case the color set query provides the initial seed images to be used for subsequent histogram queries.

	histogram euclidean		*histogram intersection*		*color set*		*color set + histogram*	
image	bus	car	bus	car	bus	car	bus	car
rounds $= L$	8	5	9	5	**1**	3	**1**	**2**
unique returns $= \sum_{l=0}^{L-1} \mathcal{M}_l$	101	72	115	75	**21**	47	**21**	**35**

VI. Conclusion

We presented a system for automated extraction of local color regions which provides for efficient indexing and effective color image retrieval. The system uses the back-projection of binary color sets to identify salient color regions in images. Prior to queries a table of meta-data is created by the systematic iteration over color sets to extract localized color regions. The table is indexed directly by color set value which enables very fast indexing of the image collection. The color set approach also allows queries to include multiple regions and their spatial layout. We also described the implementation of the color set techniques in the *VisualSEEk* content-based image/video retrieval system. Some sample queries and initial evaluation results based on recall and precision were also presented.

References

[Cha95] S.-F. Chang. Compressed-domain techniques for image/video indexing and manipulation. In *I.E.E.E. International Conference on Image Processing*, October 1995.

[CLP94] T.-S. Chua, S.-K. Lim, and H.-K. Pung. Content-based retrieval of segmented images. In *ACM Multimedia 1994*, October 1994.

[CS95] S.-F. Chang and J. R. Smith. Extracting multi-dimensional signal features for content-based visual query. In *SPIE Symposium on Visual Communications and Image Processing*, May 1995. Best paper award.

[EM95] F. Ennesser and G. Medioni. Finding waldo, or focus of attention using local color information. *I.E.E.E. Transactions on Pattern Analysis and Machine Intelligence*, 17(8), August 1995.

[FFN+93] C. Faloutsos, M. Flickner, W. Niblack, D. Petovic, W. Equitz, and R. Barber. Efficient and effective querying by image content. *IBM RJ 9453 (83074)*, August 1993.

[HCP95] W. Hsu, T. S. Chua, and H. K. Pung. An integrated color-spatial approach to content-based image retrieval. In *ACM Multimedia 1995*, November 1995.

[HSE+95] J. Hafner, H. S. Sawhney, W. Equitz, M. Flickner, and W. Niblack. Efficient color histogram indexing for quadratic form distance functions. *I.E.E.E. Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, July 1995.

[Hun89] R. W. G. Hunt. *Measuring Color*. John Wiley and Sons, 1989.

[Jon81] K. S. Jones. *Information Retrieval Experiment*. Butterworth and Co., 1981.

[Rus95] J. C. Russ. *The Image Processing Handbook*. CRC Press, Boca Raton, 1995.

[SB91] M. J. Swain and D. H. Ballard. Color indexing. *International Journal of Computer Vision*, 7:1 1991.

[SC95] J. R. Smith and S.-F. Chang. Single color extraction and image query. In *I.E.E.E. International Conference on Image Processing*, October 1995.

[WS82] G. Wyszecki and W. S. Stiles. *Color Science: Concepts and Methods*. John Wiley and Sons, 1982.