PERGAMON

Pattern Recognition 35 (2002) 1687–1704

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

www.elsevier.com/locate/patcog

Illumination color covariant locale-based visual object retrieval

Mark S. Drew *,[1], Ze-Nian Li, Zinovi Tauber

*School of Computing Science, Simon Fraser University, Vancouver, BC, Canada V5A 1S6**

Received 5 July 2001

Abstract

Search by object model — finding an object inside a target image — is a desirable and yet difficult mechanism for querying multi-media data. An added difficulty is that objects can be photographed under different lighting conditions. While human vision has color constancy, an *invariant* processing, presumably, here we seek only *covariant* processing and look to recover such lighting change. Making use of feature-consistent *locales* in an image we develop a scene partition by localization, rather than by image segmentation. A diagonal model for illumination change and a voting scheme in chromaticity space provide a candidate set of lighting change coefficients for covariant image transformation. For each pair of coefficients, *Elastic Correlation*, a form of correlation of locale colors, is performed along with a least squares minimization for pose estimation. Since the rotation, scale and translation parameters are thus estimated, we can apply an efficient process of texture support and shape verification. Tests on an image and video database of about 1500 images show an average recall and precision of over 70%.

Keywords: Color; Elastic correlation; Illumination invariance; Image segmentation; Locales; Object recognition; Search by object model

1. Introduction

Content-based image and video retrieval [1–3] depends on feature extraction, and typically multiple feature matching results are combined using a linear constraint model [4]. Types of visual features searched for include image color histograms [5], image texture [6–9], shape [10], motion trajectories in videos [3], and so on.

Illumination change can severely hinder color- and even texture-based search methods. Drew et al. [11] used a *diagonal model* for illumination change. Motivated by this model, images are illumination normalized by dividing each RGB color channel by the mean over the entire image. The normalized images are converted to chromaticity space, wavelet filtered, and DCT compressed to a 36-vector. This type of metric is meant to be illumination color *invariant*. This makes sense for whole-image comparisons, but here we wish to seek instances of a model within an image — a search by object model — so image normalization is ruled out. Instead, we recognize that pixel values for the object sought may change drastically and attempt to explicitly recover the lighting color change, and the pixel color shift — we desire an illumination color *covariant* method rather than an invariant one.

Healey and Wang [9] represented color texture using spatial correlation functions within and between sensor bands. Using the correlation functions, a correlation matrix is defined, and it was shown that two correlation matrices are related by a linear transform. After singular value decomposition (SVD), a second texture correlation matrix was projected to compute basis correspondence.

* Corresponding author. Tel.: +1-604-291-4682; fax: +1-604-291-3045.

E-mail addresses: mark@cs.sfu.ca (M.S. Drew), li@cs.sfu.ca (Z.-N. Li), zinovi@cs.sfu.ca (Z. Tauber).

[1] http://www.cs.sfu.ca.

0031-3203/02/$22.00
PII: S0031-3203(01)00163-7

Jain et al. [12] used deformable templates to retrieve images. Deformation transformations with associated distributions were defined for prototype templates, and using Bayesian probability a multi-resolution parameter search was used to minimize the objective function. Such search method is object-based since it tries to identify a particular object description irrespective of its surroundings. Unfortunately, the database has to either consist of manually extracted object boundaries, or there is an exhaustive search for the template in an image; this is infeasible for an image and video database in real-time.

In order to achieve reasonable search speed, some form of object segmentation is required so that object features can be retrieved during preprocessing (off-line) and stored in the database for matching. It is likely that a perfect image segmentation is impossible to attain [13,14]. Content-based retrieval can be achieved with a more relaxed form of image segmentation [14], with fuzzy similarity measure. The Blob-world system [15] allows object query by the user selecting an image region (query-by-example). Image segmentation uses polarity, anisotropy and normalized texture contrast for each pixel, and color in $L^*a^*b^*$ space. Pixels are grouped assuming a mixture of Gaussians model, and the maximum likelihood parameters are determined using the expectation–maximization algorithm for K regions. The method depends on an intermediate complete image segmentation step. The precision/recall curves reported on average range from 20–80% precision for recall less than 10%, and drop to 5–20% precision for recall of 20%. In the VideoQ system [3], the user sketches a query object with color, texture, shape and trajectory parameters (query-by-sketch). For a video, an initial frame is color quantized in CIE-LUV color space, and color segmented regions are projected to the next frame using estimated motion parameters. Color segmentation is accomplished by joining adjacent regions of close colors. Regions are searched and matched using the color, the three Tamura texture parameters, the principle components of the shape and the motion trails. The algorithm uses temporal consistency to refine the object segmentation. For test queries, the precision ranges from 20–100% for recall less than 50%, and is around 25% for higher recall.

In our C-BIRD (content-based image retrieval from digital libraries) system [16], we employ the query-by-example regime. Image features are coarsely localized into *locales* for the purpose of object-based retrieval. Localization is not segmentation: locales can be overlapped and/or non-connected, and the set of all locales does not have to include all image pixels. Locales are constructed using pixel blocks called *tiles*, and yet they contain pixel-level color, geometry and texture statistics. Object-based image retrieval is accomplished via real-time localization of the user-selected object model, and matching to the image locales stored in the database. For color covariance, illumination change can be recovered using a least square error (LSE) minimization for illuminant chromaticity shift under a given locale assignment [17]. For this approach, it is necessary to enumerate every possible assignment of model locales to image locales. Evaluating the feasibility of all the possible assignments takes significant processing time, and is not adequate for online search engines.

In order to reduce the number of assignments, we first obtain a set of estimates for chromaticity shift (or *chroma shift* for short) from an image illumination to a model illumination using a voting scheme in the chroma shift space. We then apply a form of chromaticity correlation called *elastic correlation* that evaluates the chroma shift estimates for generating an assignment. The very few assignments that pass this strict preselection screen are passed to a pose estimation screen and search speed is improved dramatically. Pose estimation is followed by histogram intersection on the locale texture, and finally a generalized Hough transform (GHT) [18] in the target image.

In Section 2 we describe the process of feature localization, and in Section 3 describe our search method. Section 4 presents a technique to compensate for illumination change so as to carry out a color covariant search, and Section 5 shows results of object retrieval from the C-BIRD database both without and with illumination covariant search, and also shows results for recovery of lighting change.

2. Locale-based object representation

2.1. Feature localization vs. image segmentation

For image segmentation (cf. [19]): If R is a segmented region,

1. R is usually connected; all pixels in R are *connected* (8-connected or 4-connected).
2. $R_i \cap R_j = \phi,\ i \neq j$; regions are *disjoint*.
3. $\bigcup_{i=1}^{n} R_i = I$, where I is the entire image; the segmentation is *complete*.

Object retrieval algorithms based on image segmentation permit imprecise regions by allowing a tolerance on the region matching measure. This accounts for small imprecision in the segmentation, but not for over-/under-segmentation, which can be attributed to the pixel-level approach. This works only for simplified images where object pixels have statistics that are position invariant.

We argue that a more effective and attainable process than image segmentation is a coarse localization of image features based on proximity and compactness.

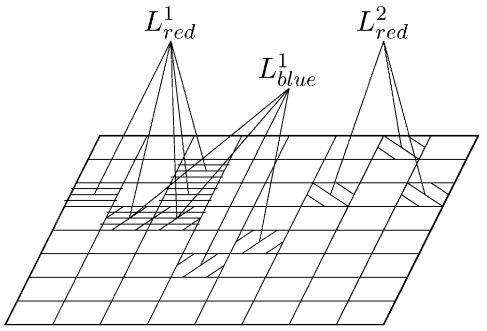


Fig. 1. Locales for localization.

Definition 1. A *locale* $\mathscr{L}_f$ is a local enclosure of feature f.

A locale $\mathscr{L}_f$ uses blocks of pixels called *tiles* as its positioning units, and has the following descriptors:

1. Envelope $\mathscr{L}_f$ — a set of tiles representing the locality of $\mathscr{L}_f$.
2. Geometric parameters — mass $M(\mathscr{L}_f) =$ count of the pixels having feature f, centroid $\mathbf{C}(\mathscr{L}_f) = \sum_{i=1}^{M(\mathscr{L}_f)} \mathbf{P}_i / M(\mathscr{L}_f)$, and eccentricity $E(\mathscr{L}_f) = \sum_{i=1}^{M(\mathscr{L}_f)} ||\mathbf{P}_i - \mathbf{C}(\mathscr{L}_f)||^2 / M(\mathscr{L}_f)$.
3. Color, texture, and shape parameters of the locale. For example, locale chromaticity color and locale texture histogram.

Initially, an image is subdivided into square tiles (e.g., 8×8 or 16×16). While pixel is the building unit for image segmentation, tile is the building unit for feature localization. Tiles group pixels with similar features within their extent, and are said to have feature f if enough pixels in them have feature f (e.g., 10%). Tiles are necessary for good estimation of initial object-level statistics and representation of multiple features *at the same location*. However, locale geometric parameters are measured in pixels, not tiles. This preserves feature granularity. Hence, feature localization is not merely a reduced-resolution variation on image segmentation.

After a feature localization process the following can be true:

1. $\exists f : \mathscr{L}_f$ is *not connected*.
2. $\exists f \exists\ g : \mathscr{L}_f \cap \mathscr{L}_g \neq \phi,\ f \neq g$; locales are *non-disjoint*.
3. $\bigcup_f \mathscr{L}_f \neq I$, *non-completeness*; not all image pixels are represented.

Fig. 1 shows a sketch of two locales for color red, and one locale for color blue. The links represent an association with an envelop, which demonstrates that locales do not have to be connected, disjoint or complete, yet colors are still localized.

2.2. Dominant color enhancement

To localize on color, we first remove noise and blurring by restoring colors smoothed out during image acquisition. First, the image is converted from RGB color space to a chromaticity-luminance color space. For a pixel with color (R, G, B), we define

$$I = R + G + B, \quad r = R/I, \quad g = G/I, \tag{1}$$

where the luminance I is separated from the chromaticity (r, g).

Prior to the creation of tiles, image pixels are classified as having either *dominant color* or *transitional color*. Pixels are classified dominant or transitional by examining their neighborhood.

Definition 2. *Dominant colors* are pixel colors that do not lie on a slope of color change in their pixel neighborhood. *Transitional colors* do.

If a pixel does not have sufficient number of neighbors with similar color values within a threshold, it is considered noise and also classified as transitional.

Enhancing the uniformity of the dominant colors is accomplished by smoothing the dominant pixels only, using a 5×5 average filter, with the exception that only dominant pixels that have similar color are averaged.

2.3. Tile generation

Tiles have a *tile feature list* of all the color features associated with a tile and their geometrical statistics. On the first pass, dominant pixels are added to the tile feature list. For each pixel added, if the color is close to a feature on the list within the luminance-chromaticity thresholds, then the color and the geometrical statistics for the feature are updated. Otherwise, a new color feature is added to the list. This feature list is referred to as the *dominant feature list*.

On the second pass, all transitional colors are added to the dominant feature list without modifying the color, yet updating the geometrical statistics. To determine which dominant feature list node the transitional pixel should merge to, we examine the neighborhood of the transitional pixel and find the closest color that is well represented in the neighborhood. If an associated dominant color does not exist, it is necessary to create a second *transitional feature list* and add the transitional color to it.

The dominant color (r_i, g_i, I_i) that is taken on by a transitional pixel *tp* having color (r, g, I) satisfies the following minimization:

$$\min_{i=1}^{nc} \left\| \left(\frac{r}{g}\right) - \left(\frac{r_i}{g_i}\right) \right\| \Big/ F(r_i, g_i, I_i). \tag{2}$$

The parameter *nc* is the number of non-similar colors in the neighborhood of the *tp*. Similar colors are aver-

aged to generate the (r_i, g_i, I_i) colors. $F(r_i, g_i, I_i)$ is the frequency of the ith average color, or in other words it is the number of similar colors averaged to generate color i. The color that minimizes this equation is the best compromise for dominant color selection for *tp* in terms of color similarity and number of similar colors in the neighborhood. The neighborhood size was chosen to be 5×5 in our implementation.

When all pixels have been added to the tiles, the dominant and transitional color feature lists are merged. If a transitional list node is close in color to a dominant list node, the geometrical statistics for the merged node are updated, but only the color from the dominant list is preserved. Otherwise, the nodes from both lists are just concatenated onto the joint list.

2.4. Locale growing

Locales are generated using a dynamic 4×4 overlapped pyramid linking procedure [20]. On each level parent nodes compete for inclusion of child nodes in a fair competition. Image tiles are the bottom-level child nodes of the pyramid, and locales are generated for the entire image when the competition propagates to the top level. The top-level pyramid node has a list of color features with associated envelopes (collections of tiles) and geometrical statistics.

Competition on each level is initialized by using a 2×2 non-overlapped linkage structure where four child nodes are linked with a single parent node. Merging of child nodes onto a parent node is accomplished via merging their color feature lists and updating the statistics if necessary. As each child node *cn* is merged to the parent node *pn*, the color feature list of the child node *c* is merged to the color feature list of the parent node *p*. Node *j* in *c* is merged to node *i* in *p* with similar color if the merged statistics have $E(p[i]) < \tau$, where τ is a threshold normalized against $M(p[i])$. Otherwise, a new color feature node is created on *p*. The initialization proceeds as in procedure `LocalesInit`.

LocalesInit: Pseudo-code for Linkage Initialization

```
Let c[n_x][n_y] be the 2D array of child nodes.
Let p[n_x/2][n_y/2] be the 2D array of parent nodes.
For each child node c[i][j] do
  Let cn = c[i][j] and pn = p[i/2][j/2].
  For each node cn_p in the feature list of cn do
    Find node pn_q in the feature list of pn that
      has similar color.
    If the merged eccentricity of cn_p and pn_q has
      E < τ then
      Merge cn_p and pn_q.
    If pn_q doesn't exist or E >= τ then
      Add cn_p to the start of the feature list
        of pn.
End Procedure
```

After the pyramid linkage initialization the competition begins. Since a 4×4 overlapped pyramid structure is used, there are four parents competing for linkage with the child, one of which is already linked to it. Node *j* in *c* will merge to node *i* in *p* if the colors are similar, the merged statistics have $E(p[i]) < \tau$, and the Euclidean distance $d(i,j) = ||\mathbf{C}(p[i]) - \mathbf{C}(c[j])||$ is the minimum over all possible nodes from all four parents. According to the spatial continuity principle, the geometrically closest parent node to the child node is most likely to be the locale of the child node. However, we still make sure that the merged locale's eccentricity can describe a plausible color localization. All the geometrical statistics updates are done in parallel, so that at the end of this competition cycle the parent–child linkage might not be optimal. Therefore, the competition cycle is iterated until there is no more change in parent–child linkage. This process is illustrated by the pseudo-code in procedure `EnvelopeGrowing`.

EnvelopeGrowing: Procedure for Locale Creation

```
Let c[n_x][n_y] be the 2D array of child nodes.
Let p[n_x/2][n_y/2] be the 2D array of parent nodes.
Repeat until parent--child linkage does not
change anymore
  For each child node c[i][j] do
    Let cn = c[i][j] and pn ∈ p[(i ± 1)/2][(j ± 1)/2]
    For each node cn_p in the feature list of cn do
      Find node pn_q in the feature lists of pn
        that has similar color
        and minimizes the distance ||C(cn_p) −
          C(pn_q)||
      If the merged eccentricity of cn_p and pn_q
        has E < τ then
        Swap the linkage of cn_p to its parent
          to pn_q.
        Update the associated geometrical
        statistics.
In the parent feature list p remove empty nodes.
Go up a level in the pyramid and repeat the
  procedure
End Procedure
```

After the pyramidal linking is done locales having small mass are removed since small locales are not accurate enough, and are likely either an insignificant part of an object, or noise. Locales are also sorted according to decreasing mass size in order to increase the efficiency of the search.

The color update equation for parent locale *j* and child locale *i* at iteration $k+1$ is

$$(r_j^{(k+1)}, g_j^{(k+1)}, I_j^{(k+1)})^{\mathrm{T}} = \frac{(r_j^{(k)}, g_j^{(k)}, I_j^{(k)})^{\mathrm{T}} M_j^{(k)} + (r_i^{(k)}, g_i^{(k)}, I_i^{(k)})^{\mathrm{T}} M_i^{(k)}}{M_j^{(k)} + M_i^{(k)}} \quad (3)$$

and the update equations for the geometrical statistics are

$$M_j^{(k+1)} = M_j^{(k)} + M_i^{(k)}, \tag{4}$$

$$\mathbf{C}_j^{(k+1)} = \frac{\mathbf{C}_j^{(k)} M_j^{(k)} + \mathbf{C}_i^{(k)} M_i^{(k)}}{M_j^{(k+1)}}, \tag{5}$$

$$E_j^{(k+1)} = \frac{(E_j^{(k)} + C_{x,j}^{(k)^2} + C_{y,j}^{(k)^2}) M_j^{(k)} + (E_i^{(k)} + C_{x,i}^{(k)^2} + C_{y,i}^{(k)^2}) M_i^{(k)}}{M_j^{(k+1)}} - C_{x,j}^{(k+1)^2} - C_{y,j}^{(k+1)^2}. \tag{6}$$

The updates are done using the intermediate sums $\sum_{i=1}^{M^{(k)}} \mathbf{P}_i$ for centroid update and $\sum_{i=1}^{M^{(k)}} P_{x,i}^2 + P_{y,i}^2$ for eccentricity update. Note that both removing and adding nodes are possible using these equations with the same ease.

The competition criterion of geometric proximity guarantees termination of the linking procedure since at every cycle the overall distance between centroids decreases. In practice, most images require less than ten competition cycles to converge at every pyramid level. Though the competition criterion is geometric proximity, feature merging is based on eccentricity. This is preferable to using proximity since we can model the allowable shape of a feature much better using eccentricity than by using centroid distances.

2.5. Texture analysis

Every locale is also associated with a locale-based texture histogram. We construct a 2D texture histogram based on *directionality* ϕ and *edge separation* ξ. We use a locale-dependent threshold in generating the edgemap by examining the histogram of the locale edge magnitudes.

Edge separation ξ is also measured separately for each locale using the locale-based edgemap. Each edge pixel i inside a locale measures the distance along its gradient ψ_i to the nearest pixel j inside the locale envelope having $\psi_j \approx \psi_i$ within 15°. If such a pixel j does not exist, then the separation is considered infinite.

We use a locale-based 2D texture histogram of size 193 pixels $\times$ 180 degrees, where separation value $\xi = 193$ is reserved for a separation of infinity. The texture histogram is smoothed using a Gaussian filter and subsampled to size 8×7, and then normalized.

2.6. Locales in video

Video segmentation proceeds via the algorithm in Ref. [11], utilizing an illumination invariant image histogram and a hierarchical procedure with dynamic transition thresholds for efficiently and robustly detecting cuts and some other types of gradual transitions. Keyframes generated are then handled as any other static images, and treated in a color covariant fashion.

3. Visual object retrieval

3.1. Model selection and object matching

C-BIRD provides a user interface to the end-user through the web using a JAVA applet. The applet provides the functionality necessary to crop a sample object from an image and submit it as a search query (query-by-example). A user selects a thumbnail image to display the full image, and then uses the available selection tools to crop out a portion of the image as a sample query object. An image region can be cropped by outlining it with the primitive shapes rectangle and ellipse, by flooding a region using a magic wand tool, by roughly specifying its boundary using an active contour, and by drawing on the region itself with a selection brush. The primitive shapes can be scaled, translated and rotated, and multiple tools can be combined together using the Boolean operations union, intersection, or exclusion. The user can also control parameters such as flooding thresholds, brush size and active contour curvature. Fig. 2 shows an example image with a region selected.

The object search method recovers 2D rigid object translation, scale, and rotation, as well as illumination change. C-BIRD also allows a combination search where an object search can be combined with other simpler search types. In that case, the searches are executed according to decreasing speed, and since object search is the most complex search available, it is executed last and only on the search results so far by the other search types.

The object image selected by the user is sent to the server for matching against the locales database. The localization of the submitted model object is considered

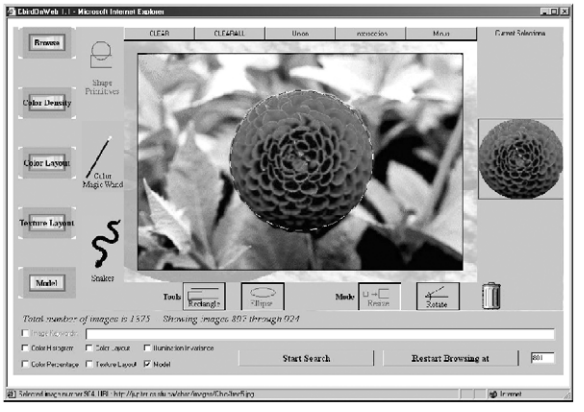


Fig. 2. C-BIRD interface showing object selection using an ellipse primitive.

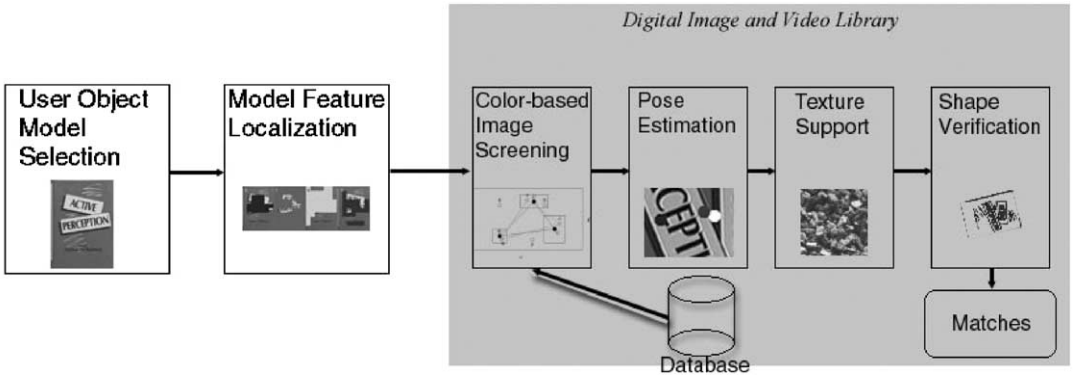


Fig. 3. Block diagram of object matching steps.

the appropriate localization for the object, so that image locales need to be found that have a one-to-one correspondence with model locales. We call such a correspondence an *assignment*. A locale assignment has to pass several screening tests to verify an object match. Screening tests are applied in order of increasing complexity and dependence on previous tests. The sequence of steps during an object matching process is shown in Fig. 3: (a) user object model selection and model feature localization, (b) color-based screening test, (c) pose estimation, (d) texture support, and (e) shape verification.

The object match measure Q is formulated as follows:

$$Q = n \sum_{i=1}^{m} c_i Q_i, \tag{7}$$

where n is the number of locales in the assignment, m is the number of screening tests considered for the measure, Q_i is the fitness value of the assignment to screening test i, and c_i are weights that correspond to the importance of the fitness value of each screening test. The c_i's can be arbitrary; they do not have to sum up to 1. Care has to be taken to normalize the Q_i values to be in the range of [0...1] so that they all have the same numerical meaning.

Locales with higher mass statistically have smaller percentage of localization error; the features are better defined and small errors averaged out, so we have higher confidence in locales with large mass. Similarly, assignments with many model locales are preferable to few model locales, since the cumulative locale mass is larger and the errors average out. We try to assign as many locales as possible first, then compute the match measure and check the error using a tight threshold. We remove or change locales in the assignment as necessary until we obtain a match. At that point, it is very probable we obtained the best match measure possible so there is no need to try other assignments. In this case, we do not have to check all possible permutations of locale assignments. In the worst case, when the object model is not present in the search image, we have to test all assignments to determine there is no match. The image locales in the database and the object model locales are sorted according to decreasing mass size.

3.2. Illumination color covariant screening

The first screening test is the color-based locale assignment filtering (Step (b)). Without any illumination changes, the color screening process is relatively trivial. Each model locale is allowed to be assigned only to database image locales that have a similar color, where color similarity is measured as an L_1 distance between colors, and the maximum allowed distance between the r, g, I is predefined.

Not all images in a digital library, however, are taken under the same illumination conditions, and the same object in different pictures could have substantially different colors. The color-based screening test would require assignments of locales taken under different illuminants. In this section, we discuss how to recover and compensate for illumination changes.

3.2.1. Diagonal model for illumination change

Assuming a Lambertian model, the pixel values R_j^x for RGB channel j and pixel position x are given by

$$R_j^x = (\mathbf{a}^x \cdot \mathbf{n}^x) \int E(\lambda) S^x(\lambda) q_j(\lambda)\, \mathrm{d}\lambda, \quad j = 1, \ldots, 3. \tag{8}$$

The shading factor $(\mathbf{a}^x \cdot \mathbf{n}^x)$ is the inner product of the light-source vector a^x incident at pixel x and the surface normal $\mathbf{n}$ at pixel x, $E(\lambda)$ is the illuminant energy distribution, $S^x(\lambda)$ is the spectral reflectance response of the surface, and $q_j(\lambda)$ is the camera sensitivity in the jth color channel, with integration over the visible.

The chromaticity equations in Eq. (1) normalize the light source intensity and remove the color dependence on the light source direction by canceling the term $(\mathbf{a}^x \cdot \mathbf{n}^x)$. A simplified working model of illumination change

is the *diagonal model* [11]. Given color (R', G', B') under model illumination and color (R, G, B) under image illumination we have

$$\mathrm{diag}(\alpha, \beta, \gamma)(R, G, B)^{\mathrm{T}} = (R', G', B')^{\mathrm{T}}. \tag{9}$$

In chromaticity space (r, g), the equations are no longer linear. Dividing Eq. (9) by γ, we define

$$\tilde{\alpha} = \frac{\alpha}{\gamma} = \frac{R'B}{RB'} = \frac{r'b}{rb'}, \qquad \tilde{\beta} = \frac{\beta}{\gamma} = \frac{G'B}{GB'} = \frac{g'b}{gb'} \tag{10}$$

so that

$$r' = \frac{R'}{R' + G' + B'} = \frac{\alpha R}{\alpha R + \beta G + \gamma B}. \tag{11}$$

Multiplying and dividing by $\gamma(R + G + B)$ results in

$$r' = \frac{\tilde{\alpha} r}{\tilde{\alpha} r + \tilde{\beta} g + b} = \frac{\tilde{\alpha} r}{(\tilde{\alpha} - 1)r + (\tilde{\beta} - 1)g + 1}. \tag{12}$$

and similarly for g', with numerator $\tilde{\beta} g$. It follows that we only need to recover the two *chroma shift* parameters $\tilde{\alpha}$ and $\tilde{\beta}$ in order to derive r' and g' from r and g.

In Ref. [17] we used a least squares minimization of the color differences between the assigned model locales and the corresponding chroma shifted image locales to find the chroma shift parameters $\tilde{\alpha}$ and $\tilde{\beta}$. This method requires an assignment of model to target locales in order to estimate the chroma shift parameters, as does the pose estimation screening test. Since the color-based screening test without illumination change was the major constraint for generating feasible assignments, requiring an assignment before screening will potentially increase the number of feasible assignments by an exponential amount. Computationally, the matching is prohibitive for some images, and too slow for most others to be feasible as a search method for media databases. Instead, here we base illumination change recovery on a type of voting algorithm, akin to a Hough transform.

3.2.2. Chromaticity voting

The purpose of applying chromaticity voting is to obtain a small set of candidate chroma shift parameters. Every pair of shift parameters will effectively induce an illumination covariant color constraint on locale correspondences, and so will be equivalent in discriminating power to the color-based screening test without illumination change considerations.

The 2D voting space is $\Phi\{\tilde{\alpha}, \tilde{\beta}\}$, where the domain is defined in the range 0.1–10.0.

$$\Phi = \{\tilde{\alpha}, \tilde{\beta} \,|\, 0.1 \leqslant \tilde{\alpha}, \tilde{\beta} \leqslant 10.0\}. \tag{13}$$

Each image locale is matched with each model locale to calculate an $\tilde{\alpha}, \tilde{\beta}$ pair using Eq. (10). This is used to index the target cell in Φ space to cast a vote. Due to the division in the calculation of $(\tilde{\alpha}, \tilde{\beta})$ in Eqs. (10) and (12), a logarithmic scale for the voting space axes is employed. The logarithmic scale helps keep the granularity of the votes. Using $\log_{10}$, we define the voting array dimensions as -1.0 to 1.0, and the voting space is discretized into 50×50 bins for creating the voting array.

Complete confidence cannot be put in the calculated $\tilde{\alpha}, \tilde{\beta}$ even for two correctly matched locales, due to the diagonal model approximation as well as noise affecting the locale colors. Therefore, a Gaussian voting is utilized whereby every $\tilde{\alpha}, \tilde{\beta}$ pair will add its corresponding normalized Gaussian probability to cells in the voting array in its neighborhood.

For the same reasons, locales with small color values are numerically less stable for calculating $\tilde{\alpha}, \tilde{\beta}$. Accordingly, the standard deviation σ of the 2D Gaussian voting function is increased in the dimension in which the color value r' or g' is reduced. This makes the Gaussian function more flat and so the vote distribution over its neighborhood is more significant. We choose a Gaussian mask size of 5×5 since it is efficient to apply and it has adequate smoothing for the voting array granularity. The variance for the mask is $\sigma^2 = 2$ for an exact cutoff, hence we vary the standard deviation σ from 1.0 to 5.0 depending on color values, and normalize the mask so that no votes are lost. Since the errors in calculating $\tilde{\alpha}, \tilde{\beta}$ using Eq. (10) can be also caused by small values of R, G, B which yield large values of r, g, it is not enough to merely check the size of the chromaticity values, rather we need to consider the RGB values which are recoverable using Eq. (1).

It is possible to have several image locales with similar chromaticity values. During the voting procedure these image locales will contribute large values when matched to any model locale by voting for the same cell. Due to the fact that there may be only few image locales corresponding to model locales but many image locales with similar chromaticity, it would be error prone to simply take the peak values. Therefore, the voting array is actually composed of n 2D voting arrays, where n is the number of model locales. Each model locale has a corresponding 2D voting array. An image locale j matched to a model locale i would only vote to voting array i. In each of the voting arrays, votes are not allowed to be accumulated, but rather the maximum vote is kept. We chose to only keep the maximum since ideally the correct match will generate the maximum vote for the cell, and all other votes to the cell should not contribute as they are of a wrong match. This assures at most one best vote for each model locale at each position of the voting array. Since the votes are the Gaussian distribution values, they are fractional numbers and not merely ones or zeros; hence they are not biased. The n voting arrays are finally added together to form a global 2D voting array for all model locales.

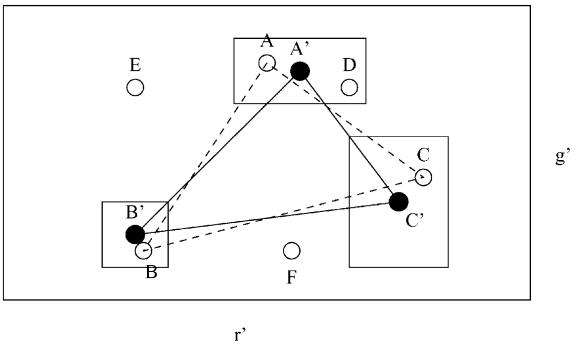


Fig. 4. Elastic correlation in $\Omega\{r', g'\}$.

The coordinates of $\Phi\{\tilde{\alpha}_p, \tilde{\beta}_p\}$ that have a high score yield a set of candidate chroma shift values $(\tilde{\alpha}_p, \tilde{\beta}_p)$. Typically, the highest score corresponds to the best $\tilde{\alpha}, \tilde{\beta}$ estimate for the image chroma shift to the model. In case there is more than one candidate $(\tilde{\alpha}_p, \tilde{\beta}_p)$, they will all be forwarded to the next step for further examination. An $(\tilde{\alpha}_p, \tilde{\beta}_p)$ pair is considered a candidate if its score is at least 70% of the highest peak score. The highest peak chroma shift is not always correct in cases where the number of model locales and their color values are small, resulting in a lower cumulative vote for the correct chroma shift, while coincidentally other locales with high color value may vote for an incorrect chroma shift that results in a higher vote.

3.2.3. Elastic correlation

We can evaluate the feasibility of having an assignment of image locales to model locales using the estimated chroma shift parameters by a type of *elastic correlation*. This computes the probability that there can be a correct assignment, and returns the set of possible assignments. Having a candidate set of chroma shift parameters $(\tilde{\alpha}_p, \tilde{\beta}_p)$, each pair is successively utilized for computing the elastic correlation measure. If the measure is high enough (in our experiments higher than 80%), then we test the possible assignments returned by the elastic correlation process for object matching using pose estimation, texture support and shape verification. Fig. 4 shows the elastic correlation process applied in the model chromaticity space $\Omega\{r', g'\}$: the model image has three locale colors located at A′, B′ and C′. All the image locale colors, A–F, are shifted to the model illuminant using Eq. (12) and the current $(\tilde{\alpha}_p, \tilde{\beta}_p)$ values. Although the locales (A', B', C') and (A,B,C) are supposed to be matching entities, they do not appear exactly at the same location. Instead of a rigid template matching (or correlation) method, we employ the elastic correlation technique in which the nodes A–C are allowed to be located at the vicinity of A', B', C', respectively.

Define a window in the chromaticity space $\Omega\{r', g'\}$ around each model locale chromaticity. Every such window is assigned a weight equal to the percentage of the corresponding locale mass out of the total locale mass for the model. In the pose objective function, we use weights w_i that are simply the L_1 norm of mass $M_i(\mathscr{L}'_f)$, divided by the total mass. Here, we use similar weights but here the L_1 normalization is done over all model locales, not only the ones in the assignment. The weights of all the windows containing a shifted image locale chromaticity are added to produce the correlation measure. Also a set of candidate assignments is realized by allowing model locales to be assigned only to image locales contained in their window. Typically, not many candidate assignments exist, and the chroma shift parameters $(\tilde{\alpha}_p, \tilde{\beta}_p)$ that yield the best correlation measure are the best estimate for the chromaticity shift from the image to the model illuminant. The elastic correlation is also insensitive to partial object occlusions, locales that split or merge, and noise.

The correlation process used here is made even more *elastic* by allowing non-fixed window sizes. As in chromaticity voting in the $\Phi\{\tilde{\alpha}_p, \tilde{\beta}_p\}$ space, the window size in $\Omega\{r', g'\}$ is made larger on the dimension where r' or g' values are small in order to compensate for the potential error in applying Eq. (12) to small color values. This of course implies that the shape of the window is often not square. We have varied the window size on each dimension from 0.04 to 0.08.

Also, care must be taken not to consider the same image locale for two overlapping model windows. This can be achieved by not allowing image locales to be assigned to more than one window. The difficulty is in selecting the overlapping window an image locale should be assigned to. The brute force method is to test all possible ways, and keep the highest correlation score. We have used this since our implementation of the search method lends itself easily to those tests. However, we are investigating a computational geometry solution.

In some ways the elastic correlation is similar to a deformable template matching [12]. In Fig. 4, if we consider the template to be the triangle $A'B'C'$ of model locales, then it is deformed to triangle ABC to match the image locales. Though, generally a deformable template has a global model (possibly with local variation in deformation parameters), the elastic correlation model locale window sizes are completely independent from each other. In practice template matching is applied only on the image space domain, while elastic correlation is applied in the chromaticity color space domain.

3.3. Estimation of image object pose

From matched color locales, we define a *mass-ratio* factor *mr* as follows:

$$mr = \log(M(\mathscr{L}_f)/M(\mathscr{L}'_f)) = \log(M(\mathscr{L}_f)) - \log(M(\mathscr{L}'_f)). \quad (14)$$

This *mr* factor is given in terms of the ratio of the mass of the target locale $M(\mathscr{L}_f)$ to the mass of the corresponding model locale $M(\mathscr{L}'_f)$. Since for model locales smaller than target locales $M(\mathscr{L}_f)/M(\mathscr{L}'_f)$ changes more significantly over the assignment than for model locales larger than target locales, it is not possible to set a constant threshold directly on the ratio similarity among assigned locales. Taking the logarithm, the division turns to subtraction, and a constant threshold is now possible to derive. It is required that the *mr* factors be similar for all corresponding locales. This constraint requires that an *mr* factor already be known before we can apply it. However, it still does not need an assignment and therefore also helps to reduce the combinatorial nature of this problem. *mr* factors are averaged over all the locales already assigned. Since image locales are sorted in decreasing order, if for a given model locale the *mr* factor with a target locale is less than the average *mr* factor so far then we can say that model locale cannot be matched (at least under the current assignment), since all the next target locales will only generate lower *mr* factors, and will never be similar to the average. The *mr* factors, together with the sorting, efficiently, enforce bounds on the assignments that can be generated. We call all assignments that are allowable *as feasible assignments.*

Our pose estimation method (Step (c)) uses geometrical relationships between locales for establishing pose parameters. For that reason it has to be performed on a feasible locale assignment. Locale spatial relationships are represented by relationships between their centroids. The number of assigned locales is allowed to be as few as two, which is enough geometry information to drive estimation of a rigid body 2D displacement model with four parameters to recover: x, y translation, rotation $\mathbf{R}$, and scale s [17].

We obtain both the best pose parameters for an assignment and the minimization objective value, which is an indication of how well the locales assignment fit using our displacement model. If the error is within a small threshold, then the pose estimate is accepted.

Now, we also verify that the scale and the mass ratio factors are related according to

$$mr \approx \log(s^2). \quad (15)$$

Clearly, if the model mass uniformly grows by a factor of s^2, then all distances within it grow by a factor of s.

3.4. Texture support

The locale texture histogram is *not* invariant to object transformation; therefore, texture matching has to be done after the pose parameters are estimated. However, since we assume an angle-preserving 2D displacement model, it is easy to register the model texture histogram to the image texture histogram. Let $s\mathbf{R}\mathbf{H}'_i$ denote the model texture histogram $\mathbf{H}'_i$ adjusted by scale s and rotation $\mathbf{R}$. The directionality axis coordinate in the texture histogram depends only on edge orientation, so we must add the estimated rotation angle to the model directionality coordinates. The indexed angle is adjusted to lie in the range 0–180°. Similarly, the texture separation axis coordinate depends only on the object scale, so we multiply the model separation coordinates by the estimated scale s. If the indexed separation is larger than 192 then it is considered infinity. The texture counts in histogram cells are normalized, so that the scale of the object does not have to be considered to compensate for more edge pixels in larger objects.

Texture matching is carried out via a fast texture histogram intersection approach similar to color histogram intersection [5]. For every locale correspondence in the assignment, their texture histograms are used for texture support if they contain sufficient texture information. Having corresponding texture histograms for both model and database image locales, the texture histogram intersection measure v is obtained by taking the sum of the minimum corresponding cells

$$v = \sum_{i=1}^{56} \min\{\mathbf{H}_i, s\mathbf{R}\mathbf{H}'_i\}. \quad (16)$$

If v is greater than a threshold for all assigned locales, then the locale textures were matched successfully.

3.5. Shape verification

The final match verification process (Step (e)) is shape verification by the method of GHT [18]. The GHT is robust with respect to noise and occlusion [21]. Performing a full GHT search for all possible rotation, scale and translation parameters are computationally very expensive and inaccurate. Such a search is not feasible for large databases. However, after performing pose estimation we already know the pose parameters, and we can apply them to the model reference point to find the estimated reference point in the database image. Hence, the GHT search reduces to a mere confirmation that the number of votes in a small neighborhood around the reference point is indicative of a match. This GHT matching approach takes only few seconds for a typical search. The reference point we choose is the model center since it minimizes voting error for errors in edge gradient measurements.

Fig. 5. (a): Sample model image. (b): Sample database image containing the model book.

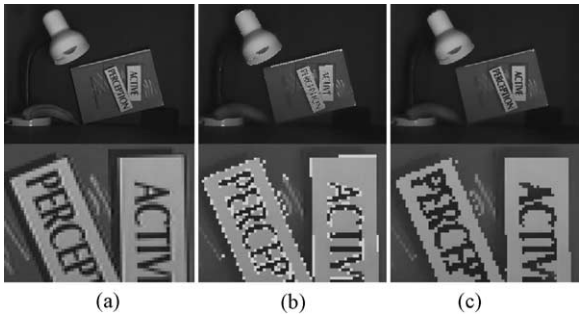


Fig. 6. Smoothing using dominant colors. (a) Original image not smoothed. (b) Smoothed image with transitional colors shown in light gray. (c) Smoothed image with transitional colors shown in the replacement dominant colors (if possible).

Once we have shape verification, the image is reported as a match, and its match measure Q returned, if Q is large enough. After obtaining match measures Q_i for all images in the database, the Q_i measures are sorted according to decreasing value. The number of matches can further be restricted to the top k if necessary.

4. Experimental results

Our C-BIRD database contains about 1500 images and several video clips. The following results were obtained using a Pentium III 500 MHz with 256 MB of memory and an ATA33 hard-drive.

4.1. Localization results

Both feature localization and object search details are illustrated using a sample object model image, Fig. 5(a), and a sample database image containing the model object under different illumination and pose, Fig. 5(b). The first step in the extraction of locales is image smoothing using dominant colors and color redefinition of transitional pixels during the tile creation process. Fig. 6 presents the original image and two stages of the color enhancement process. In Fig. 6(a) the original image is shown. Fig. 6(b) shows the smoothed image using dominant colors only, and the transitional pixels in gray. Almost all pixels in a gradient color region have been classified as transitional. In Fig. 6(c) transitional colors have taken on dominant color values that will be used in the tile creation process. Of particular interest is the fact that most transitional pixels have in fact correctly assumed the purple color of the letters in the title, rather than the whiteish color of the background, despite the overwhelming number of white pixels in the neighborhood in comparison with purple ones. This is a testament to the fact that Eq. (2) is powerful enough for such a discrimination. Tiles are generated using the dominant color list and a transitional color list for pixels that have not assumed a dominant color.

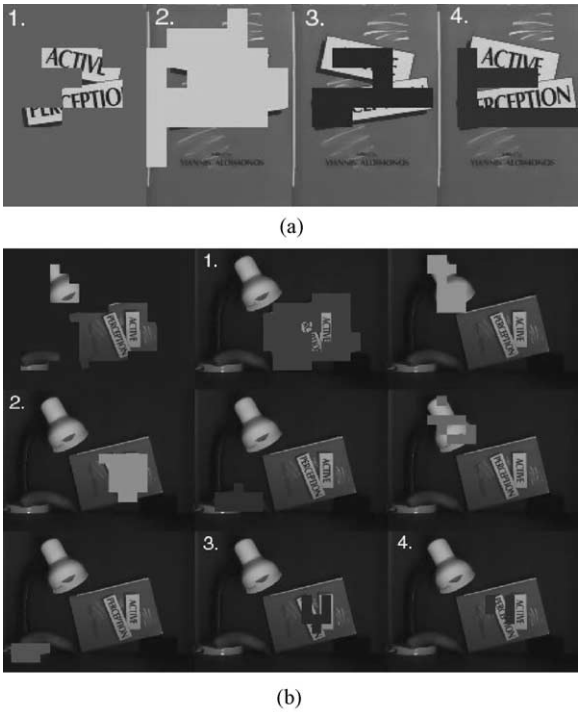


Fig. 7. (a) Color locales for the model image. (b) Color locales for a database image.

After creating tiles, locales are extracted. Fig. 7 depicts the color localization. Locales are shown by their envelope in the locale color. At the end of the localization process the pyramid's top node contains the locales list. Although locales can overlap, localized features respect object separation and represent the multiple conceptual features humans attribute to an image area. Locales do not have to be connected, but as demonstrated in Fig. 7(b), even nearby locales with similar colors, such as the white locale of the book and the white locale of the lamp (photographed under bluish fluorescent light),

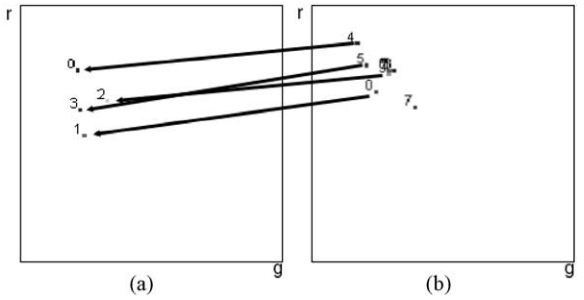


Fig. 8. (a) Locale chromaticities for the model image. (b) Locale chromaticities for the database image.

will not normally merge when they belong to different feature positions. The hierarchical approach enables the eccentricity to be calculated with respect to the feature centroid regardless of the seed position of feature merging. This is because on lower levels of the pyramid, tiles have to merge to geometrically closer parents.

Localization time depends on scene complexity. Our object model image 5 has size about 320×240, and its localization takes 1.9 s. The test image 5 has size 640×480 and its localization takes 6.1 s. Typical times for localization of user selected objects are around 1.5 s. Half of the processing time is taken for separation-map generation and about a fifth each is spent on the edge-map generation and dominant color smoothing. Performing the feature merging for all hierarchy levels takes less than 10 ms. It has also recently been attempted to reduce localization time through various optimizations, with about five-fold speed improvement. We would be able to increase localization speed significantly more through use of hardware acceleration.

4.2. Analysis of object matching results

We illustrate illumination covariant object search using the model image in Fig. 5(a) that is taken under tungsten illumination, and the database image in Fig. 5(b) that is taken under fluorescent illumination.

The first task of the object search method is the estimation of the chroma shift parameters in order to enforce color covariance constraints on a locale assignment. In Fig. 8, the chromaticity axis origin is at the top left, and values go from 0 to 1. Arrows represent chromaticity shift for corresponding locales. When using the LSE method for chroma shift estimation, the correct assignment of locales is eventually generated since previous assignments do not generate an object match, and the estimated chroma shift is $(2.41, 1.67)$ with the objective value being 0.0035. This objective value indicates that the chroma shift values are a good estimate under this locale assignment.

Alternatively, using the chromaticity voting scheme, all image locales are paired with all model locales to vote

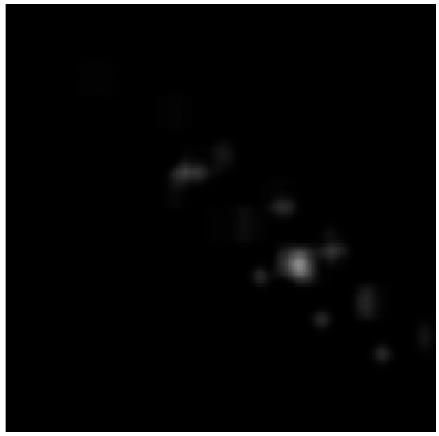


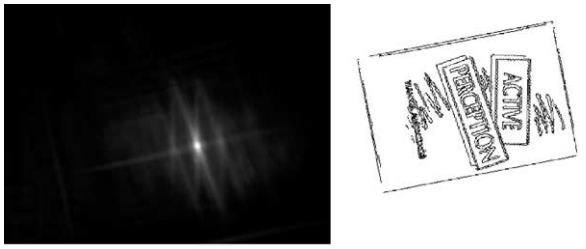
Fig. 9. Voting result in Φ space. Peaks are denoted by bright intensity values.

for $(\tilde{\alpha}, \tilde{\beta})$ positions in the voting array Φ. The voting array is presented in Fig. 9. The highest peak in the voting array is located at the coordinate $(34, 30)$ which corresponds to $(\tilde{\alpha}, \tilde{\beta})$ values of $(2.33, 1.75)$. In comparison, the average $(\tilde{\alpha}, \tilde{\beta})$ values when the locales are assigned manually is $(2.39, 1.68)$. This average is not the true illumination change, which is unknown in this experiment, yet it is a strong indication that the chroma shift parameters are correct. The chroma shift values are also very similar to those estimated using the LSE method.

Next, the elastic correlation process confirms that the chroma shift parameters discovered by the highest peak do correspond to a feasible assignment. The correlation value of 1 is also higher than the correlation values of any other peaks in the voting array. The assignment of image locales to model locales indicated by the elastic correlation is the same as shown in Fig. 7. Notice that in this locale correspondence, locales are assigned in sorted order of both model and image locales. This is typically the case and it speeds up the computation significantly. For a similar image to Fig. 5(b) that was photographed under tungsten illumination, the localization and assignment generated using a strict non-covariant color matching approach is identical, and so are most of the following results.

The pose estimation yields a rotation of 80.94°, a scale of 0.82 and a translation of $(230.96, 212.01)$ pixels. These values are consistent with our manual image measurements. The overall error as returned by the pose objective function is 4.59 pixels squared. This is the weighted sum of the displacement errors of all the model centroids to the image centroids.

The pose error is good enough to allow the assignment to be checked for texture support. Since the texture histograms are very coarse, we cannot expect a high match measure for object rotations and scales that are not



(a) (b)

Fig. 10. (a): GHT accumulator array image. (b): Drawing of the GHT template using the estimated pose.

integral multiples of texture histogram bin ranges. The minimum texture histogram intersection value obtained from all the locale correspondences in the assignment is 0.65. Our threshold for texture matching is lower than this, so the only screen test left to pass is match verification using GHT for shape matching.

Using the GHT table obtained for the model at the beginning of the search, voting is performed on a reduced resolution Hough transform accumulator array according to the discussion in Section 3.5. The voting array is shown is Fig. 10(a) and shows a clear peak. The correct reference point for the peak is (189, 141) and the position where the peak was actually detected is (190, 137). The GHT match measure Q_{GHT} returned accounts for most model edge pixels, and thus is good enough to declare this locale assignment a match. To further illustrate the match parameters correspondence to the database image, we make the search engine generate a reconstructed drawing of the template at the peak position using the GHT table only, in Fig. 10(b). Fig. 10 shows the reconstructed template using the estimated pose parameters.

4.3. Database search results

We performed a search of the full database (1500 images) with three model objects which we know exist in the database under different illuminations. To demonstrate the necessity and effectiveness of the illumination covariant search, we first present the search results using strict color distance restrictions, and then the results using our covariant color comparisons. Fig. 11 shows the search results for the pink book model in Fig. 5(a) without illumination change considerations. Fig. 11(a): before applying texture support; (b): with texture support but without shape verification; and (c): after shape verification. The precision is improved from 67% in phase (a) to 100% in phase (c). The recall is 62% and the texture and shape screens do not reduce it.

Table 1 summarizes the search results. Recall and precision are defined by

$$\text{recall} = \frac{CM}{A}, \qquad \text{precision} = \frac{CM}{CM + FM},$$

where CM is the number of correct matches, FM the number of false matches, and A the number of available (matchable) objects in the database. Color-based search alone can produce many false matches, but the subsequent stages of texture checking and shape verification by GHT reduce the number of false matches without compromising recall much. The images containing the pink book are under a variety of illuminations; also, some are gamma corrected and some are not. The green book images are also photographed under different illuminants, and only two match the model. This underlines the importance of a color covariant method. All the blue books have the same illumination, so they are matched easily.

Typical running times are less than 20 s without shape verification.[2] Database access time is fairly constant since

Table 1
Retrieval and timing results for object-based searches, no illumination change

Model object		Recall			Precision		Timing results (s)		
		A	CM	%	FM	%	Total	Matching	DB access
Blue book	Pose	5	5	100	4	56	16.378	0.031	15.984
	Pose + texture	5	5	100	3	63	16.056	0.038	15.632
	Pose + texture + shape	5	5	100	1	83	20.156	4.285	15.693
Pink book	Pose	13	8	62	4	67	16.286	0.038	15.898
	Pose + texture	13	8	62	1	89	16.339	0.055	15.924
	Pose + texture + shape	13	8	62	0	100	38.290	22.336	15.657
Green book	Pose	7	2	29	8	20	16.364	0.038	15.934
	Pose + texture	7	2	29	5	29	16.474	0.039	16.109
	Pose + texture + shape	7	2	29	2	50	27.775	11.580	15.878

[2]The demo site for C-BIRD, containing a Java applet, can be found at "http://jupiter.cs.sfu.ca/cbird".

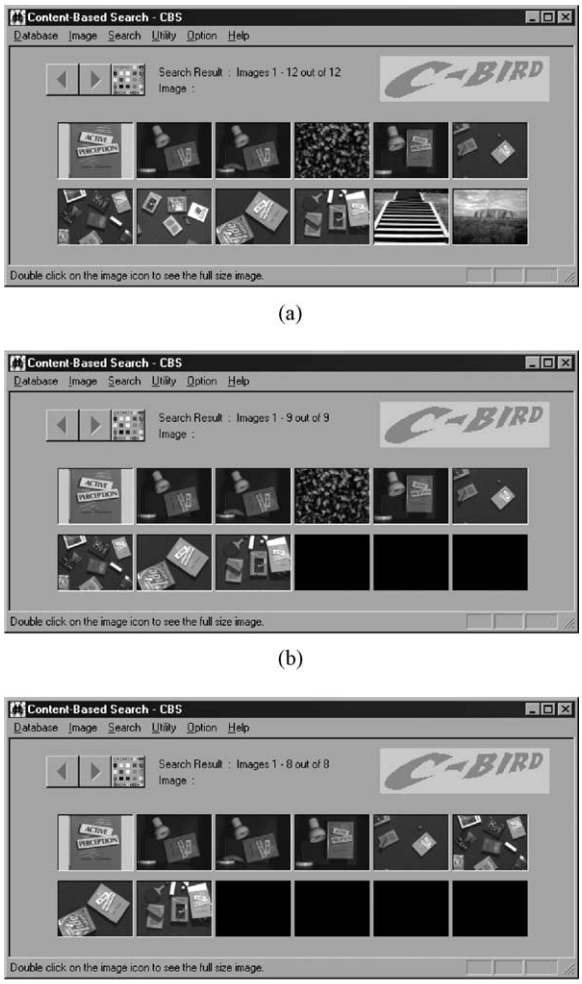


Fig. 11. Search results for the pink book model image. (a) Shows search results using pose estimation only. (b) Shows search results using pose estimation and texture support. (c) Shows search results using the GHT shape verification.

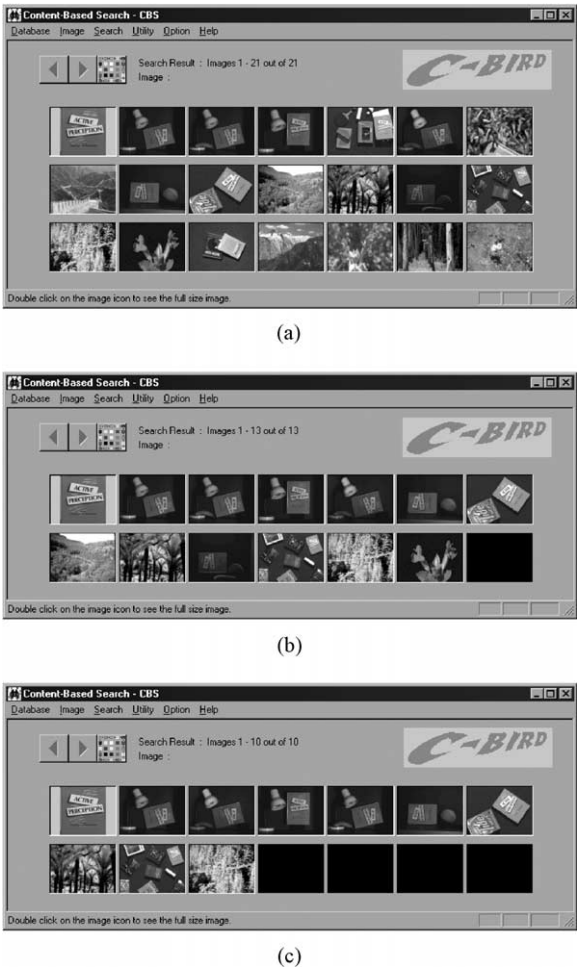


Fig. 12. Search result for the pink book model with illumination change support. (a) Shows search results using pose estimation only. (b) Shows search results using pose estimation and texture support. (c) Shows search results using the GHT shape verification.

the texture histograms are loaded for all locales, and image edgemaps are loaded only when necessary. Most time is spent accessing the large amount of data in the texture histograms. When search using shape verification is performed, note that the expense of the voting and peak allocation procedures of the GHT depends on the number of images returned in previous stages, and is performed more frequently for false matches where the number of assignments tested is larger.

We performed the same searches of the C-BIRD database again, only this time using chromaticity voting and elastic correlation. We expect to have similar precision and running time, yet higher recall, since the images under different illuminants will be retrieved as well.

Fig. 12 shows search results for the model book object in Fig. 5(a). The retrieval results are very similar to those using the LSE criterion for chromaticity estimation. Fig. 12(a) illustrates the search results using the chromaticity voting and elastic correlation for generating assignments for pose estimation, (b) illustrates the results with added texture support but without shape verification, and (c) illustrates the search results after shape verification.

Table 2 summarizes the results for the three object searches used in Section 3. A search for those objects using the LSE method for chromaticity shift takes over an hour for all images. The results using chromaticity voting and elastic correlations have timing results increased by only one second over a search with no support for illumination change.

Table 2
Retrieval and timing results for object-based searches under illumination change

Model object		Recall			Precision		Timing results (s)		
		A	*CM*	%	*FM*	%	Total	Matching	DB access
Blue book	Pose	5	5	100	5	50	17.380	1.184	15.885
	Pose + texture	5	5	100	1	83	17.096	1.162	15.674
	Pose + texture + shape	5	5	100	0	100	19.905	3.942	15.700
Pink book	Pose	13	10	77	11	48	17.605	1.194	16.024
	Pose + texture	13	9	69	4	69	17.297	1.220	15.817
	Pose + texture + shape	13	8	62	2	80	48.481	32.159	15.957
Green book	Pose	7	5	71	4	56	17.568	1.321	15.890
	Pose + texture	7	5	71	3	63	17.808	1.350	15.848
	Pose + texture + shape	7	5	71	1	83	34.586	18.439	15.813

The recall and precision measures are relatively high for all the objects tested. For the pink book images, we see from Fig. 12 that some pink books under different illuminations were retrieved, while they were not in Fig. 11. Yet, the search with illumination change estimation had problems with the gamma corrected books. This explains the mixed results. The green book had retrieved most of the books under different illuminations. This helped improve recall and precision significantly. These results show that illumination change recovery is important for image retrieval. For images with no illumination change, such as the blue book images, both approaches perform equally well.

4.4. Recovery of lighting change

To verify chroma shift and pose estimation procedures using data in which we have high confidence, we used beach ball images from Ref. [22] to test lighting change recovery using chromaticity voting and elastic correlation methods. The ball is captured under five different illuminations which were carefully calibrated. Comparing each of the images against the four other remaining images, we can explicitly calculate, $\tilde{\alpha}$, $\tilde{\beta}$ using the estimated scene illumination values. Those $\tilde{\alpha}$, $\tilde{\beta}$ values are considered correct for comparison purposes. The illuminants and their chromaticities are given in Table 3.

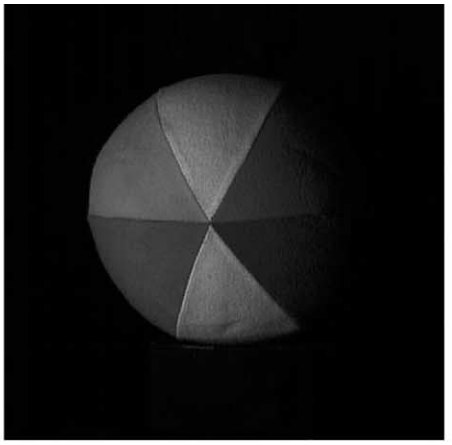
Table 3
Illuminant chromaticity values

Light name	*r*	*g*	*b*
Macbeth 5000 K+ 3202 Filter (Mb + f)	0.08	0.18	0.74
Macbeth 5000 K (Mb)	0.182	0.266	0.552
Sylvania Cool White (Syl)	0.266	0.26	0.474
Philips Ultralume (Ulm)	0.327	0.33	0.343
Halogen (Hal)	0.398	0.31	0.292

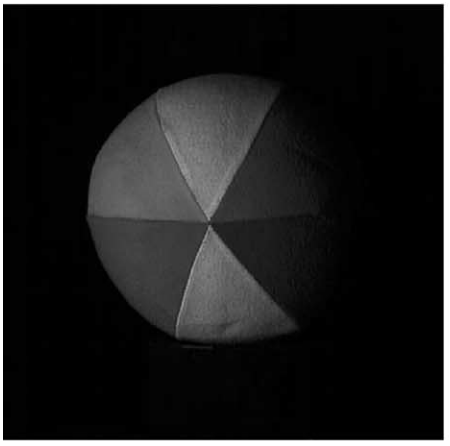
Fig. 13 shows pseudo-color images (c) and (e) of the chroma shift parameters $\tilde{\alpha}$, $\tilde{\beta}$ for the illumination change from a sample model ball in (a) to the sample balls on the left. A red color in the pseudo-images on the right corresponds to $\tilde{\alpha}$ value, and a green color corresponds to $\tilde{\beta}$ value. In particular, if an image were matched to itself, the pseudo-color would be yellow (equal contributions of r and g for every pixel) everywhere as is the case for image (c). Some pixels have a black pseudo-color if the chromaticity or shift parameters are uncertain (or undefined) for them. The psuedo-color image in (e) shows that chroma shift values on the same patch are fairly constant despite heavy shading, yet pseudo-colors between patches differ, though not by much. This is typical of most images, while the chroma shift estimation variance between patches depends on the illuminants mapped to each other.

We have also created five new images by randomly selecting target images from the C-BIRD database, and placing one of the beach balls into the target image, forming a composite image. The balls are placed into a random image with a known rotation (e.g., 33°), scaling (e.g., 0.58) and translation (e.g., 121, 81). The ten beach-ball images are inserted into the C-BIRD database. Each time we use one of the original five beach ball images as a model image to test whether we can find the correct match with the right lighting change ($\tilde{\alpha}$, $\tilde{\beta}$), rotation, and scale. Fig. 14 shows one of the search results in which all ten beach-ball images are retrieved together with two false targets from the C-BIRD database.

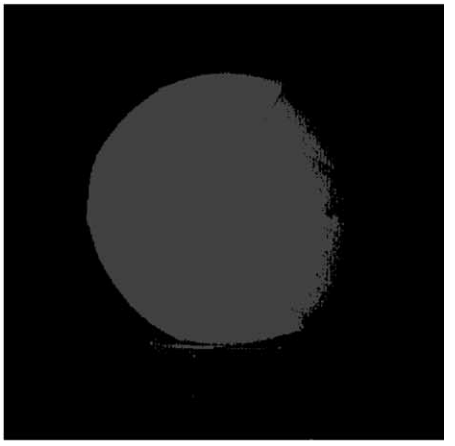
Fig. 13. Sample beach-ball images and their associated chroma shift pseudo-colors. (a) Beach ball image under Syl. light; (b) Beach ball image under Syl. light; (c) Psuedo-color image for (a) and (b); (d) Beach ball image under Ulm. light; (e) Pseudo color image for (a) and (d).



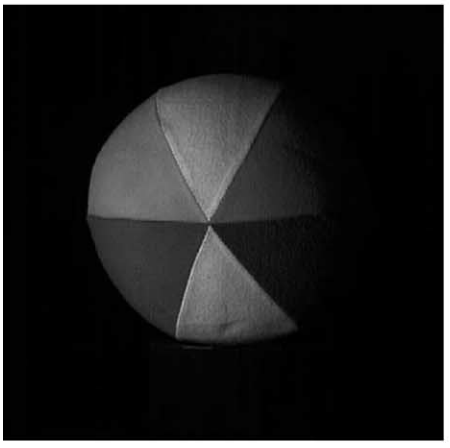
(a)



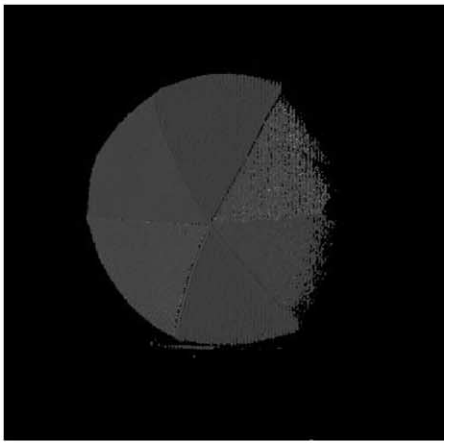
(b)



(c)



(d)



(e)

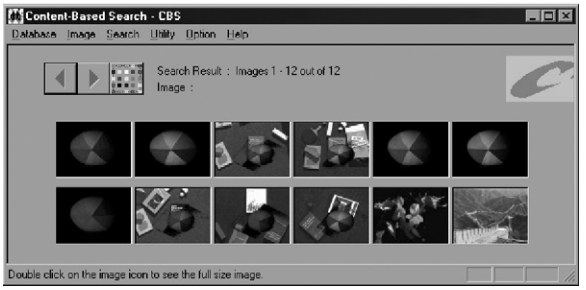


Fig. 14. Search using a beach ball under Sylvania Cool White illuminant as a model.

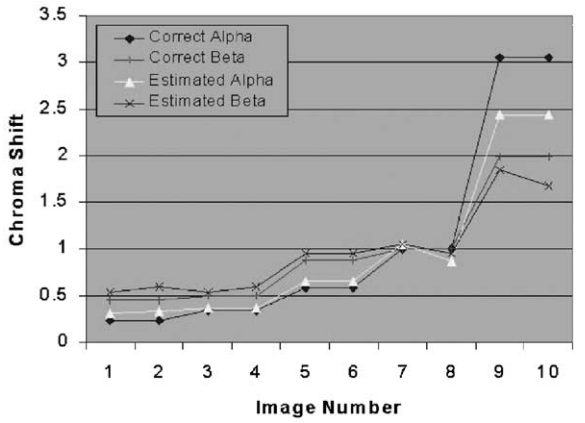


Fig. 15. Correct and recovered $\tilde{\alpha}_s$ and $\tilde{\beta}_s$.

Fig. 15 graphs the chromaticity shift estimate curves in relation to the correct illuminant curves. It is apparent that when $\tilde{\alpha}$, $\tilde{\beta}$ are close to one (that is when the chromaticity change of the illuminants is moderate), the estimation errors are smaller than 10%. However, when the illuminant chromaticities are different, chroma shift computation errors increase and the diagonal model approximation degrades. Nevertheless, since the shift is non-linear, the potentially larger error for shift estimation does not translate to a much larger error in the shifted chromaticity values themselves. This further supports the logarithmic scale of the chromaticity voting space, and is also compatible with the elastic correlation that is performed in the chromaticity space.

Table 4 summarizes the five beach-ball searches performed in which each ball under a particular light is used as a model. The searches are performed on the entire database, and precision and recall values are shown. $\Delta\theta$ is the error in angles of the rotation estimation and Δs the error of scaling factors estimation. Both error values are computed as an average of the error values of all correct matches. The table shows a fairly high recall and precision with rotation estimate error of less than 1°, and scale error of less than 0.05. Obviously, these are very good estimates, and are much better than the matching thresholds.

Table 4
Quality of the five beach-ball searches

Light for model image	Precision	Recall	$\Delta\theta$	Δs
Sylvania Cool White	73%	80%	0.50	0.021
Macbeth 5000 K	83%	100%	0.40	0.019
Macbeth 5000 K + 3202	62%	80%	0.83	0.017
Philips Ultralume	78%	70%	0.30	0.018
Halogen	64%	70%	0.65	0.035

Some of the models have very high or very low chroma shift values when matched to some images. For example, when a Halogen illuminated ball is used as a model, and it is matched to a ball illuminated in a Macbeth 5000 K + 3202 Filter light booth, the chroma shift values are (12.608, 4.365), which is even higher than our voting space $\Phi\{\tilde{\alpha}, \tilde{\beta}\}$ dimensions. In this case, the images still matched since the estimated $\tilde{\alpha}$ was actually about 9 which *is* within the voting space dimensions, and the shifted chromaticities were accurate enough. This is an evidence to the appropriateness of the logarithmic scale for the chromaticity voting space. In general, though corresponding locales with larger chromaticity shift have a higher possibility of error, the object can still be matched.

5. Conclusion

This paper presents a color covariant search method by object model using multiple features (color, texture, shape, etc.). Database images and video sequences are preprocessed off-line for locale extraction. Locales are generated using a feature localization technique that uses compactness as a merging determiner and geometric proximity as a competition criterion.

The object search method starts with an attempt to limit the number of locale assignments by using locale colors as a constraint. Under illumination change, we use the techniques of chromaticity voting and elastic correlation. It is demonstrated that those techniques are equivalent in speed to using simple color similarity restrictions, and superior in retrieval recall without sacrificing precision. In recovering lighting change, we also yield a partial solution for the problem of color constancy. Subsequently, we employ a series of fast checks that perform pose estimation, texture support and shape verification, which lead to high reliability of the retrieved images.

One limitation of the current approach is that it relies on the diagonal model for illumination change. The diagonal model is only approximate, and seems to be more accurate when the illumination change is smaller. An

added difficulty is that some images have a linear-gamma while some are gamma-corrected. Although images with gamma correction can still match objects with a linear gamma, this is less accurate.

Also, localization is not always perfect, locales can be inaccurate or missing. The locale color is an average of all pixel colors in the locale, which might also be inaccurate and affect the color matching during the object search.

References

[1] J.R. Bach, C. Fuller, A. Gupta, The virage search engine: an open framework for image management, SPIE Storage and Retrieval for Image and Video Databases, Vol. IV, February 1996.

[2] M. Flickner, H. Sawhney, W. Niblack, Query by image and video content: the QBIC system, IEEE Comput. 28 (9) (1995) 23–32.

[3] S.-F. Chang, W. Chen, H.J. Meng, H. Sundaram, D. Zhong, A fully automatic content-based video search engine supporting multi-object spatio-temporal queries, IEEE Trans. Circuits Systems Video Technol. 8 (5) (1998) 602–615.

[4] H. Sundaram, S.-F. Chang, Efficient video sequence retrieval in large repositories, Proc. SPIE Storage and Retrieval for Image and Video Databases, Vol. VII, 1999.

[5] M.J. Swain, D.H. Ballard, Color indexing, Int. J. Comput. Vision 7 (1) (1991) 11–13.

[6] H. Tamura, S. Mori, T. Yamawaki, Texture features corresponding to visual perception, IEEE Trans. Systems, Man, Cybernet. 8 (6) (1978) 460–473.

[7] F. Liu, R.W. Picard, Periodicity, directionality, and randomness: world features for image modeling and retrieval, IEEE Trans. Pattern Anal. Mach. Intell. 18 (7) (1996) 722–733.

[8] A. Jain, G. Healey, A multiscale representation including opponent color features for texture recognition, IEEE Trans. Image Process. 7 (1) (1998) 124–128.

[9] G. Healey, L. Wang, The illumination-invariant recognition of color texture, IEEE International Conference on Computer Vision, Cambridge, MA, 1995.

[10] F. Mokhtarian, S. Abbasi, J. Kittler, Robust and efficient shape indexing through curvature scale space, Proceedings of British Machine Vision Conference, Edinburgh, 1996, pp. 53–62.

[11] M.S. Drew, J. Wei, Z.N. Li, Illumination-invariant image retrieval and video segmentation, Pattern Recognition 32 (1999) 1369–1388.

[12] A.K. Jain, Y. Zhong, S. Lakshmanan, Object matching using deformable templates, IEEE Trans. Pattern Anal. Mach. Intell. 18 (3) (1996) 267–278.

[13] D. Marr, Vision, W.H. Freeman, San Francisco, CA, 1982.

[14] P. Salembier, F. Marques, Region-based representations of image and video: segmentation tools for multimedia services, IEEE Trans. Circuits Systems Video Technol. 9 (8) (1999) 1147–1169.

[15] S. Belongie, C. Carson, H. Greenspan, J. Malik, Color- and texture-based image segmentation using expectation-maximization algorithm and its application to content-based image retrieval, International Conference of Computer Vision, Bombay, 1998, pp. 675–682.

[16] Z.N. Li, O.R. Zaïane, Z. Tauber, Illumination invariance and object model in content-based image and video retrieval, J. Vis. Commun. Image Rep. 10 (1999) 219–244.

[17] M.S. Drew, Z. Tauber, Z.N. Li, Feature localization and search by object model under illumination change, in: M.M. Yeung, Boon-Lock Yeo, C.A. Bouman (Eds.), Proceedings of IS& T/SPIE Symposium on Electronic Imaging, Storage and Retrieval for Media Databases, SPIE 3972, 2000, pp. 399–410.

[18] D. Ballard, Generalizing the hough transform to detect arbitrary shapes, Pattern Recognition 13 (2) (1981) 111–122.

[19] D.H. Ballard, C.M. Brown, Computer Vision, Prentice-Hall, Englewood Cliffs, NJ, 1982.

[20] T.H. Hong, A. Rosenfeld, Compact region extraction using weighted pixel linking in a pyramid, PAMI 6 (1984) 222–229.

[21] P. Gvozdjak, Z.N. Li, From nomad to explorer: active object recognition on mobile robots, Pattern Recognition 31 (6) (1998) 773–790.

[22] B.V. Funt, K. Barnard, L. Martin, Is machine colour constancy good enough?, in: Proceedings of ECCV98, 1998. http://www.cs.sfu.ca/colour/Images/eccv-98-db.html.

About the Author—MARK S. DREW is an Associate Professor in the School of Computing Science at Simon Fraser University. His undergraduate education was in Engineering Science and his subsequent education was in Theoretical Physics. He received the B.A.Sc. degree from the University of Toronto in 1970, and studied the Foundations of Quantum Mechanics for the M.Sc. degree in 1971 in the Mathematics Department there. He studied Field Theory at the University of British Columbia and received the Ph.D. in 1976. Combining work on Energy Systems and Computer Applications, he held an Industrial Postdoctoral Fellowship and subsequently an Industrial Research Fellowship in Industry until he joined SFU in 1982.

His interests lie in the fields of multimedia, computer vision especially focusing on color, photorealistic computer graphics, computer algorithms for color reproduction, and scientific applications of algebraic programming.

Dr. Drew is the holder of a US Patent in digital color processing and a Director of Lightseer Ltd., a UK color research firm.

About the Author—ZE-NIAN LI received the B.S. degree in Electrical Engineering from the University of Science and Technology of China in 1970, and the M.S. and Ph.D. degrees in Computer Sciences from the University of Wisconsin, Madison in 1981 and 1986, respectively. From 1970 to 1979 he was an Electronic Engineer in charge of digital and analogical system design. He was

an Assistant Professor at the University of Wisconsin, Milwaukee from 1986 to 1987. In 1988, he joined the School of Computing Science at Simon Fraser University, where he is currently a Professor and Director of Vision and Media Lab. His current research interests include multimedia, computer vision and pattern recognition.

About the Author—ZINOVI TAUBER is currently employed in R& D at Digital Accelerator Corp. in British Columbia, Canada. He received a B.Sc. in Mathematics and Computing in 1997 and the M.Sc. in Computing Science in 2000, both from Simon Fraser University. His research interests include multimedia, computer vision and computer graphics.