Int J Multimed Info Retr (2012) 1:103–114
DOI 10.1007/s13735-012-0011-7

REGULAR PAPER

An efficient framework for location-based scene matching in image databases

Xu Chen · Madirakshi Das · Alexander Loui

Received: 15 October 2011 / Revised: 3 January 2012 / Accepted: 15 February 2012 / Published online: 24 March 2012

Abstract SIFT-based methods have been widely used for scene matching of photos taken at particular locations or places of interest. These methods are typically very time consuming due to the large number and high dimensionality of features used, making them unfeasible for use in consumer image collections containing a large number of images where computational power is limited and a fast response is desired. Considerable computational savings can be realized if images containing signature elements of particular locations can be automatically identified from the large number of images and only these representative images used for scene matching. We propose an efficient framework incorporating a set of discriminative image features that effectively enables us to select representative images for fast location-based scene matching. These image features are used for classifying images into good or bad candidates for scene matching, using different classification approaches. Furthermore, the image features created from our framework can facilitate the process of using sub-images for location-based scene matching with SIFT features. The experimental results demonstrate the effectiveness of our approach compared with the traditional SIFT-, PCA-SIFT-, and SURF-based approaches by reducing the computational time by an order of magnitude.

X. Chen (✉)
Department of EECS, University of Michigan, Ann Arbor, USA
e-mail: xhen@umich.edu

M. Das · A. Loui
Kodak Research Lab, Eastman Kodak Company, Rochester, USA
e-mail: madirakshi.das@kodak.com

A. Loui
e-mail: alexander.loui@kodak.com

Keywords Scene matching · SIFT · Clustering · Image search and retrieval · Face detection · Occlusion · Blur · Classification

1 Introduction

Scene matching refers to the process of matching a region in one image with the corresponding region in another image where both image regions are part of the same scene. Since most digital media currently being captured and the billions of earlier digital images taken before the availability of GPS lack detailed location information, scene matching plays an important role in determining location information. In the absence of GPS information, the location at which a photograph was captured can be described in terms of the stationary background [5]. Earlier work on scene matching involved computing correlation between images [8], local invariant features [20], and spatial intensity gradients of the images [16]. In [20], Schaffalitzky et al. present an effective approach for matching shots that are images of the same 3D scene in a film. The wide baseline method represents each frame by a set of viewpoint invariant local features. However, in addition to being very computationally intensive, these methods cannot handle the large variations in scale, lighting, and pose encountered in consumer images.

More recently, there has been a lot of work on matching feature-rich complex scenes using scale-invariant features (SIFT) [15] and faster feature extractions such as PCA-SIFT and Speeded Up Robust Features (SURF) [1,11]. Furthermore, semantic information has been considered in scene matching [13,17,19,23]. In the work [19], a combination of *query-by-visual-example* (QBVE) and *semantic retrieval*

(SR) has been adopted for image retrieval, where images are labeled with respect to a vocabulary of visual concepts, as is customary in SR. In [23], a color SIFT descriptor for scene matching is proposed to explore the invariant properties and distinctiveness with respect to photometric transformations. In [18], Quattoni and Torralba proposed a prototype that can successfully combine both local and global discriminative information. However, these techniques have been mainly used to match and register the entire scene for every image, which is a time-consuming process when a large image database is involved. To avoid comparing each pair of images in the database to detect scene matches, a method for selecting a few representative images that can be matched reliably based on their image characteristics is very desirable.

The criteria used for choosing representative images from a group of images (or a keyframe from a segment of video) are often tied to image quality [12] and the best similarity with other images in the group. The main difference between our work and Ke et al. [12] is that the work in [12] mainly focuses on the determination of the quality of photos, while our work addresses the challenging problem of selection of good images for the purpose of scene matching. Practically, a good-quality photo may not be the best candidate for scene matching. For instance, photos that contain large homogenous regions such as water, grass, and sky could be of high quality but they are usually poor for scene matching since they lack specific features that could determine the locations. Also, photographs of people, which constitute a large portion of consumer image databases, are not good candidates for scene matching if the people in the picture are occluding the background, or if the background is plain, whereas these characteristics usually result in a good quality portrait. Also, the complexity of various scenes limits the usefulness of applying traditional scene matching with SIFT features directly. These factors motivated our effort to produce an effective framework for fast selection of good candidate images for location-based scene matching.

In this work, we address these issues by proposing an efficient framework that explores image features that evaluate images in terms of their value for scene matching in consumer image collections. The proposed framework identifies images with good matching opportunities for SIFT and SIFT-like features. Typically, good images for this application contain distinctive elements with complex edge structures. Since our goal is computational savings, the selection process needs to be extremely fast when compared with the scene matching task, and our image features are designed following this criterion. These image features are incorporated into the framework for selecting representative images appropriate for the scene matching task. Further reduction in computation time is achieved by extracting sub-images containing regions of interest for scene matching. As far as we know, this is the first time that a generalized framework for selection of representative images for consumer image collection is proposed, which is motivated by the appropriateness of the images for scene matching.

The rest of the paper is organized as follows: in Sect. 2, we present a framework for selection of representative images. In Sect. 3, we investigate the feature representation for our framework. Given the extracted features, in Sect. 4 we discuss the classification algorithms relying on the extracted features. The comparison of the performance of our approach with traditional approaches is demonstrated in Sect. 5. We give a brief summary and conclusion in Sect. 6.

2 Our framework

A large number of images in consumer image collections are not suitable for scene matching. Earlier work on consumer collections [5] observed that only about 10% of the images present in co-located events can actually be matched using scene matching techniques. The main reasons for this are (1) The background elements that can be used for matching images captured at the same location are mostly occluded by the people in the images. (2) The images are blurry due to focusing problems or camera and/or object motion, resulting in failure of SIFT feature point detection. (3) The images contain few meaningful edges and specific objects, e.g., images with natural scenes, or generic objects such as cabinets and furniture common to many locations. Our goal is to select the best candidate images for successful SIFT-based scene matching, while eliminating the images with the above-mentioned problems.

An event clustering algorithm described in [14] is used to segment a user's collection into events and sub-events using temporal and color histogram information, where events are very likely to have been captured at the same location because of their temporal proximity. Our approach to selecting the best images for SIFT-based scene matching can be applied to select a few representative images from each event, thus greatly reducing the number of images that need to be matched in the collection.

We formulate the problem of selecting representative images as a fast binary classification problem—the separation of good representative images from unsuitable images in consumer image collections, using features that can discriminate between the two classes. This allows us to filter out a large fraction of the images in a collection, resulting in a framework in which scene matching can be performed efficiently. Figure 1 shows a block diagram of our approach. The output of our method is a shortlist of images that are suitable for SIFT-based scene matching, and sub-image regions corresponding to these images where scene matching should be performed.

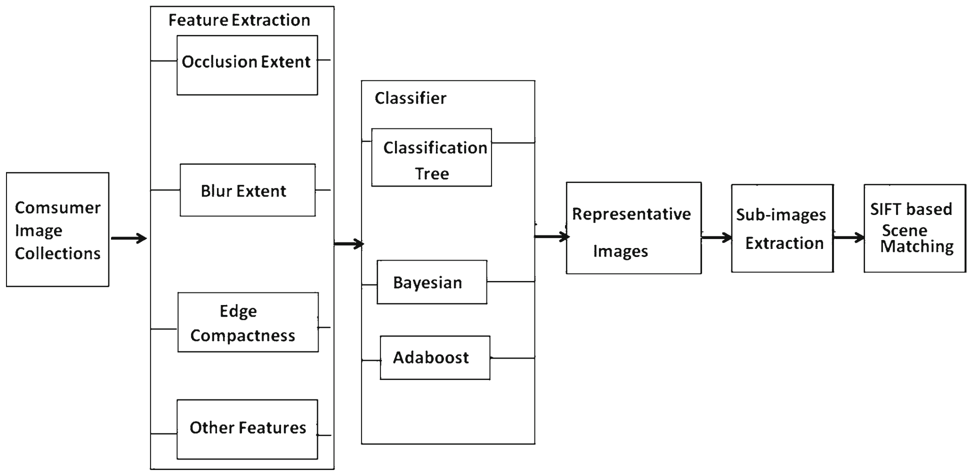

Fig. 1 Block diagram of our framework

3 Relevant feature extraction

This section describes the main features we have developed to distinguish between good candidate images for scene matching and other images in the collection.

3.1 Occlusion extent

Based on evidence gathered from a large number of consumer images, it can be seen that the occlusion of objects of interest due to the presence of people is an important factor in determining whether the image is a good candidate image for scene matching. Obviously, the higher the extent of the occlusion, the smaller the probability that the image can be matched with other images from the same scene by using unique objects present in the scene. Typically, a large fraction of consumer images contain people. To measure the occlusion extent, we determine the approximate positions of people in the images using face detection. Specifically, we estimate the position of people from the position and size of the facial circle (the center and the radius) output by a face detector. We use the face detection algorithm by Jones et al. [9] due to its faster implementation, utilizing integral images and a cascade of classifiers. It should be noted that human detection approaches that could yield higher accuracy in detecting people in images in some domains, while they may not achieve the speed of [9]. However, for consumer image collections, in many cases the torsos are occluded by other people or objects in the scene, while the faces are most likely unobstructed in the photos. Therefore, the face detector has better detection performance as compared with human detectors in our application. We rely on bagging [2] to reduce the false detection and improve the accuracy of face detection. Particularly, in three runs of the detector using different parameters, if the detection results satisfy $|x(i) - x(j)| < 5, |y(i) - y(j)| < 5, |r(i) - r(j)| < 5$, where $i, j = 1, 2, 3, i \neq j$, $x(i), y(i)$ are the coordinates of the center of the face, and $r(i)$ is the radius of the circle, we consider the detection to be correct. We choose the regions with the highest vote as the regions where faces are detected. Subsequently, we estimate the area of the regions of the human body by making reasonable assumptions that (1) the width of the shoulder is approximately twice as long as the width of the head and (2) the length of the occluding part of the body in the image is approximately four times as long as the length of the head. We obtain these priors by averaging the results over 3,000 images with different postures of people, including standing and sitting.

We compute the occlusion extent k_1 as

$$k_1 = \frac{P}{Q}, \tag{1}$$

where P is the total number of occluded pixels and Q is the total number of pixels in the image. Figure 2 illustrates the occlusion extent for some sample consumer images. It should be noted that there could be some situations where people are overlapping with each other, which could make our estimates slightly larger than the true value. However, from the experimental results, it does not have much influence on our classification results. Furthermore, this problem can be solved by first sorting the human face by x and y coordinates and then determining the overlap regions of adjacent human faces.

3.2 Compactness of edges

The success of scene matching is highly dependent on the presence of spatially compact regions with dense edge structures corresponding to unique objects in the image. Therefore, we propose a feature that estimates the number of regions of interest for scene matching in any candidate image. The larger the number of such regions contained in the image, the higher the probability that the image is a good candidate for scene matching. In this paper, we consider the edge

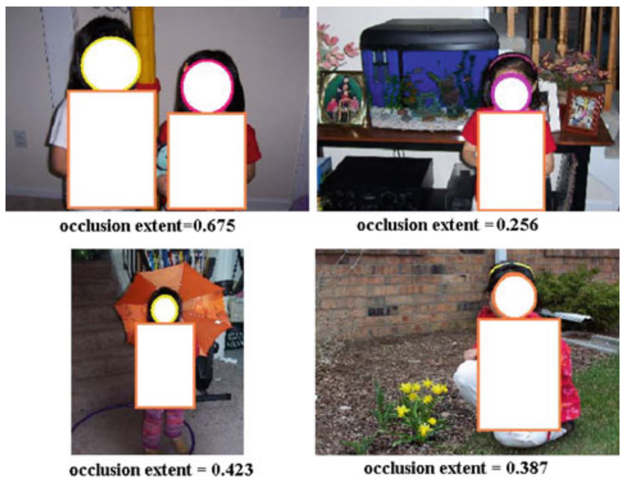

Fig. 2 Visual illustration of occlusion extent for some example images from consumer collections

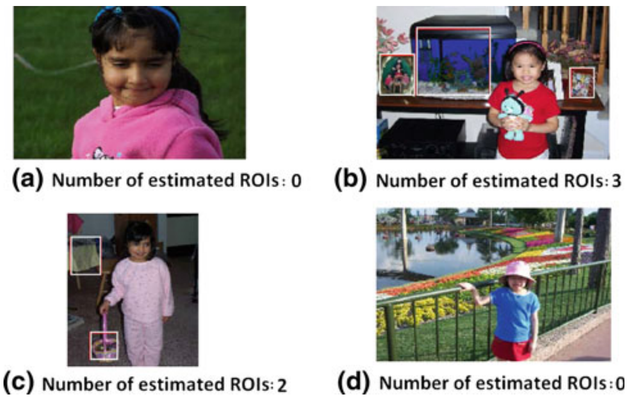
(a) Number of estimated ROIs: 0
(b) Number of estimated ROIs: 3
(c) Number of estimated ROIs: 2
(d) Number of estimated ROIs: 0

Fig. 3 Visual illustration of the regions of interest for scene matching estimated in some example images from consumer collections

information [7,24] as the most salient information for determining regions of interest for scene matching. This choice enables a fast implementation—the edge clustering method has also been shown to be an efficient method for fast object detection [4]. In this work, we evaluate the number of possible regions of interest by first clustering the edges spatially and then computing the variance of each cluster. If the variance of each cluster is smaller than a threshold, we consider the cluster to be a region of interest. The steps in this process are shown in Fig. 4. Since the edges of textured regions such as grass are usually not compact enough (such as in Fig. 3a), these regions can be filtered out by thresholding based on the variance of the cluster.

To remove the small variations in pixel values and false edges due to illumination differences and textured regions, we pre-process the images with Gaussian filters to smooth the images. Subsequently, Canny detectors are applied to the smoothed images. We subtract the edges contributed by people in the images from the total edge map by combining

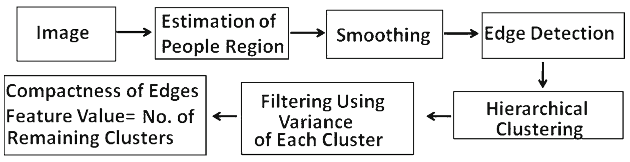
Image
Estimation of People Region
Smoothing
Edge Detection
Hierarchical Clustering
Filtering Using Variance of Each Cluster
Compactness of Edges Feature Value= No. of Remaining Clusters

Fig. 4 Steps in the extraction of compactness of edges feature

the results from face detection and people detection discussed earlier. By doing this, it is more likely that the remaining edges are mostly from non-people objects in the image.

Once we obtain the edge map for objects in the image, we utilize hierarchical clustering [10], where agglomerative methods proceed by a series of fusions of the n sub-block of edges into objects where each of the sub-block of edges is of size 3×3. This method can handle the uncertain number of regions in the images (unlike K-means clustering [6] where the number of regions need to be pre-specified). The detailed hierarchical clustering algorithm is as follows:

1. Initialize $c = n$, $D_i = \{x_i\}$ where c is the current number of clusters, n is the number of sub-block of edges, D_i represents the ith cluster and x_i represents the ith sample which belongs to the cluster D_i.
2. Decrease c by 1 through finding nearest clusters, say, D_i and D_j, merge D_i and D_j.
3. Repeat Step 2 until the distance between nearest clusters $d_{mean}(D_i, D_j) = \|m_i - m_j\|$ (where m_i and m_j are mean value vectors for the clusters D_i and D_j) is larger than the threshold of the average distance between objects in images learned from the training dataset, which is about one fourth of the length of the image. Return the current number of clusters and the corresponding clusters c, for regions of interest.

We compute the feature compactness of edges, k_2, as the number of estimated regions of interest, which is defined explicitly as

$$k_2 = |c : \mathbf{Var}(c) < threshold|, \quad (2)$$

where c is the number of clusters detected by hierarchical clustering, the outside || denotes cardinality, and $\mathbf{Var}(c)$ represents the variance of the cluster c. As shown in Fig. 3 (where regions of interest are highlighted with bounding boxes), in most of the cases, our method based on compactness of edges can identify regions of interest that are relevant for scene matching. For instance, in Fig. 3a, where a person is featured against a grassy background, we correctly determine that there is no specific region fitting our matching needs. In Fig. 3c, there are two regions identified: table and basket. In Fig. 3d, although the background is complicated, the use of our feature enables us to eliminate the false positives and correctly identify that there are no regions of interest in this

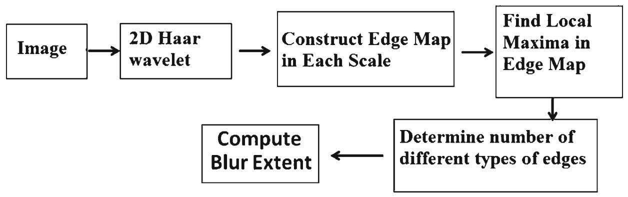
Image
2D Haar wavelet
Construct Edge Map in Each Scale
Find Local Maxima in Edge Map
Determine number of different types of edges
Compute Blur Extent

Fig. 5 Steps in the extraction of the blur extent feature

image. The objects that are present (flowers, lake, vegetation) are not suitable for SIFT-based scene matching. These objects are filtered out at different steps of our method. The strength of edges from flowers is weak and they are removed during the preprocessing stage when smoothing with Gaussian filters. The variances of the edge locations for the fence, lake or trees are larger than the threshold. Therefore, they are also removed in the process of computing the compactness of edges feature.

3.3 Blur extent

Some of the consumer images are of poor quality due to lack of focus and motion blur, and not suitable as candidate images for scene matching. As we know, edges can be generally classified into four types: Dirac-structure, Astep-structure, Gstep-structure, and Roof-structure. We rely on the approach described in Tong et al. [22] to determine the blur extent of images. Typically, if the blur occurs, both of the Gstep-structure and Roof-structure tend to lose their sharpness as described in [22]. Figure 5 shows the steps in determining the types of edges. The detailed steps are provided as follows:

1. Perform Harr wavelet transform to the original image with decomposition level 3, which results in a hierarchical pyramid structure.
2. Construct the edge map in each scale by

$$Emap_i(m,n) = \sqrt{LH_i^2 + HL_i^2 + HH_i^2}, \tag{3}$$

   where $i = 1, 2, 3$; m and n are the coordinates of rows and columns in the edge map.
3. Partition the edge maps and find local maxima in each window, with window sizes of 2×2, 4×4, and 8×8, respectively.
4. We denote the results from step (3) as E_{max_i} $(i = 1, 2, 3)$. We compute the percentage of Gstep-structure and Roof-structure edges to describe the blur extent. Typically, if $E_{max_1}(m,n) > threshold$ or $E_{max_2}(m,n) > threshold$ or $E_{max_3}(m,n) > threshold$, the edge at (m,n) is a Dirac-structure or Astep-structure edge.
5. If $E_{max_1}(m,n) < E_{max_2}(m,n) < E_{max_3}(m,n)$, (m,n) has a Roof-structure or Gstep-structure edge.

   If $E_{max_2}(m,n) > E_{max_1}(m,n)$ and $E_{max_2}(m,n) > E_{max_3}(m,n)$, (m,n) has a Roof- structure edge.

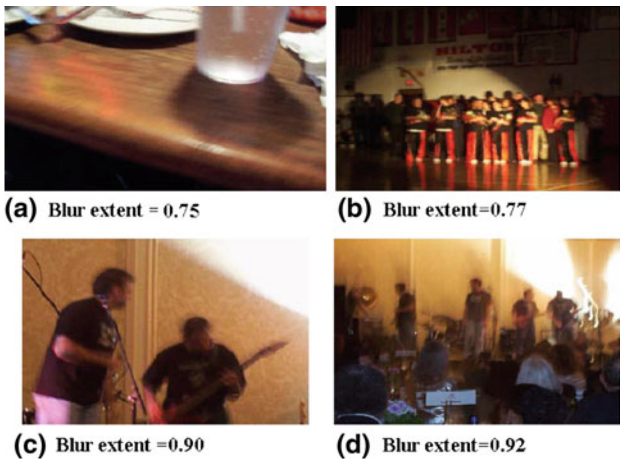

Fig. 6 Visual illustration of the blur extent for some example images from consumer collections

The blur extent k_3 is computed as

$$k_3 = \frac{N_1}{N_2}, \tag{4}$$

where N_1 denotes the sum of the number of Gstep-structure and Roof-structure edges, N_2 denotes the total number of edges. Figure 6 provides visual illustration of the blur extent for some example images from consumer image collections. Figure 6a and b show blur caused by camera motion, where the proposed method computes a blur extent of 0.75 and 0.77, respectively. For two out-of-focus, blurred images shown in Fig. 6c and d, the proposed method also effectively computed the blur extent as 0.90 and 0.92, respectively.

There are other features that could further improve the accuracy of selections, such as contrast, brightness, color histogram, and the camera focal length. However, the overall improvement is not very significant since these affect only a small portion of images. For example, brightness is important when the pictures are taken during the night. Therefore, in our experiments, we mainly present results using the three most discriminative features described in detail in this section.

4 Classification algorithms

We explore a number of classification strategies for grouping images into two categories—good candidates for scene matching and poor candidates for scene matching—based on the features we have described in the previous section.

4.1 Naive Bayesian framework

Given the list of features, k, one way of integrating them into one unified framework is to use the naive Bayesian framework [12]. Let us assume we consider the first three most discriminative features (it can be easily extended to the case of utilizing more features). The overall quality metric according to Bayes rule is defined as

$$k_{all} = \frac{P(good|k_1, k_2, k_3)}{P(bad|k_1, k_2, k_3)} = \frac{P(k_1, k_2, k_3|good)P(good)}{P(k_1, k_2, k_3|bad)P(bad)}. \quad (5)$$

Assuming independence of the features given the class,

$$k_{all} = \frac{P(k_1|good)P(k_2|good)P(k_3|good)P(good)}{P(k_1|bad)P(k_2|bad)P(k_3|bad)P(bad)}. \quad (6)$$

We can choose equal numbers of good and bad candidate images, so that $P(good)$ and $P(bad)$ can be dropped from the equations.

4.2 Adaptive boosting classifier

The goal of boosting is to improve the accuracy of any given learning algorithm. In boosting, a classifier is first created that has greater-than-average accuracy on the training dataset, and then new component classifiers are added to form an ensemble whose joint decision rule has arbitrarily high accuracy on the training set. In such a case, the classification performance is said to have been "boosted". There are a number of variations on basic boosting. The most popular, AdaBoost (from "adaptive boosting" [21]), can also be effectively used in our problem since AdaBoost is adaptive in the sense that it allows the designer to continue adding weak learners until some desired low training error has been achieved. In AdaBoost, each training pattern receives a weight that determines its probability of being selected for a training set for an individual component classifier. If a training pattern is accurately classified, then its chance of being used again in a subsequent component classifier is reduced. On the contrary, if the pattern is not accurately classified, then its chance of being used again is raised. The major steps of the AdaBoost algorithm are outlined as follows:

1. We begin by initializing $D = \{x^1, y_1, \ldots, x^n, y_n\}$ where x^i are patterns and y_n are their labels in D, the maximum number of iterations k_{max}, and the weights $W_1(i) = 1/n, \forall i = 1, \ldots, n$.
2. In each iteration, we train weak learner C_k using D sampled according to $W_k(i)$. If we denote E_k as the training error of C_k measured on D using $W_k(i)$, the updated weight is determined by

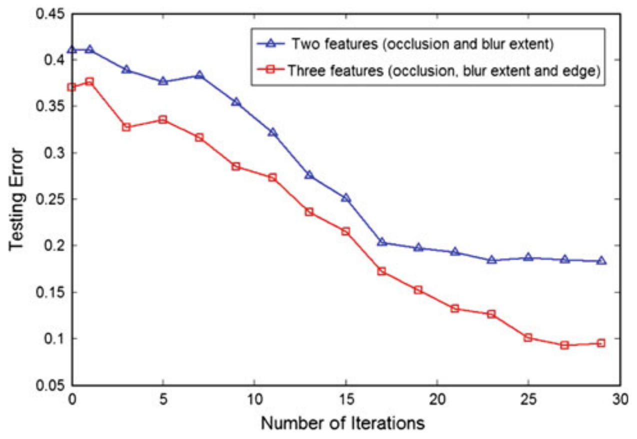

Fig. 7 Classification error with AdaBoost relying on two features (occlusion and blur extent) and three features (occlusion, blur extent, and compactness of edges)

$$\begin{aligned} W_{k+1}(i) &= \frac{W_k(i)}{Z_k} \times e^{-\alpha_k}, \\ &\quad if\ h_k(x^i) = y_i (CorrectlyClassified) \\ W_{k+1}(i) &= \frac{W_k(i)}{Z_k} \times e^{\alpha_k}, \\ &\quad if\ h_k(x^i) \neq y_i (IncorrectlyClassified), \end{aligned} \quad (7)$$

where the error for classifier C_k is determined with respect to the distribution $W_k(i)$ on which it was trained. Z_k is a normalizing constant computed to ensure that $W_k(i)$ represents a true distribution and $h_k(x^i)$ is the category label (+1 or -1) given to pattern x^i by component classifier C_k. The final classification decision of x^i is based on a discriminant function that is merely the weighted sums of the outputs given by the component classifiers:

$$g(x) = \left[\sum_{k=1}^{k_{max}} \alpha_k h_k(x)\right]. \quad (8)$$

The classification decision for this binary case is then $sgn[g(x)]$, where the sign function is defined as $sgn[g(x)] = -1$, when $g(x) < 0, sgn[g(x)] = 1$ when $g(x) > 0$. AdaBoost provides a good way for us to assign proper weights to these different features. In our case, it is entirely possible that the image that has less occlusion is heavily blurred. The weak classifiers used are constructed from single features using a Bayesian classifier based on a unitary Gaussian model. Figure 7 shows the performance when using AdaBoost with two and three features. It can be seen from Fig. 7 that by incorporating more features, the classification based on the AdaBoost algorithm is able to improve the classification accuracy significantly.

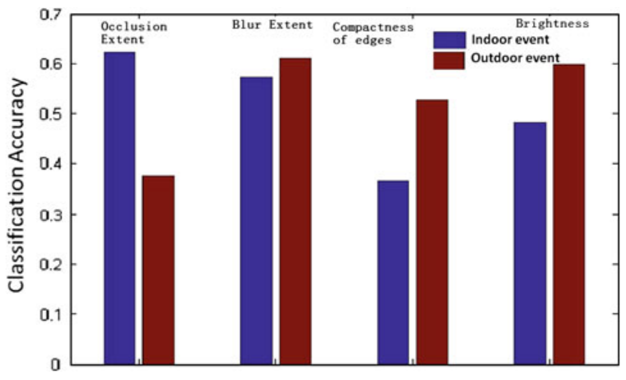

Fig. 8 Comparison of classification accuracy with single feature for indoor and outdoor event

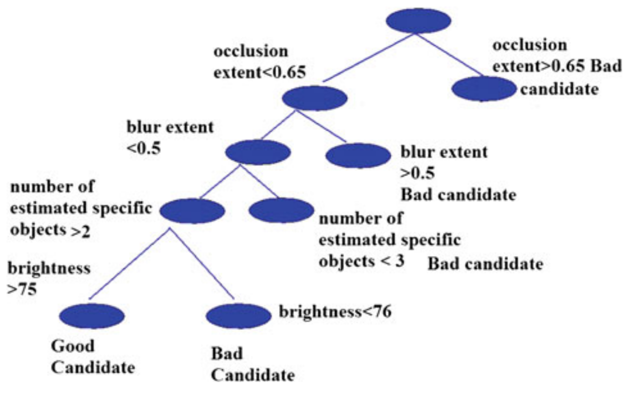
occlusion extent<0.65
occlusion extent>0.65 Bad candidate
blur extent <0.5
blur extent >0.5 Bad candidate
number of estimated specific objects >2
number of estimated specific objects < 3 Bad candidate
brightness >75
brightness<76
Good Candidate
Bad Candidate

Fig. 9 Classification tree used for identifying good candidate images for scene matching

4.3 Classification tree-based approach

Another classifier that can be used in this problem is a rule-based classification tree [3]. We adapt the widely used CART (classification and regression trees) method. CART provides a general framework that can be instantiated in various ways to produce different decision trees. The fundamental principle underlying tree creation is that decisions that lead to a simple, compact tree with few nodes is preferred. Therefore, we seek a property query T at each node that makes the data reaching the immediate descendent nodes as "pure" as possible. There are various mathematical measures of impurity; in this paper we define the impurity as the entropy impurity (information impurity):

$$i(N) = -\sum_{j} P(\omega_j) \log_2 P(\omega_j), \qquad (9)$$

where $P(\omega_j)$ is the fraction of patterns at node N that are in category ω_j. By the property of entropy, if all the patterns are of the same category, the impurity is 0. Given a partial tree down to node N, we choose the value for the property test T, which decreases the impurity as much as possible. The decrease in impurity is defined by

$$\Delta i(N) = i(N) - P_L i(N_L) - (1 - P_L) i(N_R), \qquad (10)$$

where N_L and N_R are the left and right descendent nodes, $i(N_L)$ and $i(N_R)$ are their impurities, and P_L is the fraction of patterns at node N that will go to N_L when property query T is used. Therefore, the "best" query value s is the choice for T that maximizes $\Delta i(T)$. For simplicity, in this paper, we generally restrict our discussion to the monothetic tree where each query is based on a single property. Figure 8 shows the relative strengths of the features when applied to indoor and outdoor images. It is not surprising that the occlusion extent plays the most important role for indoor events (as shown in Fig. 8). For outdoor events, blur extent dominates over other features, which can be mainly attributed to moving objects in the outdoor events. Figure 9 shows the structure of the classification tree constructed based on experiments using 2,200 images. The thresholds in Fig. 9 are determined by maximization of the drop in impurity. Therefore, this method can save significant computational time by eliminating bad candidates early while maintaining high classification accuracy.

5 Experimental results

5.1 Selection of ground truth

We collected the training and test data by using the software described in [5], which performs the core step of determining whether there is a scene match between two images. The matching is based on SIFT features, subject to a set of constraints that reduce false matches. Positive examples for our application are images that match at least one other image captured at the same scene, and negative examples are images that could not be matched to any other image from the same scene as the negative example. We selected 3,000 positive example images (images producing matches) and 3,000 negative example images (images that could not be matched with other images). The testing database contains images from consumer collections, mainly depicting different people-based activities such as family get-togethers, playing ballgames, etc., where there are around 35% blurred images and around 80% images with complex edges from people or unique objects such as building, or pictures on the wall. For the testing stage, fivefold cross validation is used. For each test, 20% of the images are used as test images and the other 80% serve as training images. In the naive Bayesian framework, we assume each feature has a Gaussian distribution and the mean and variance can be computed during the training stage.

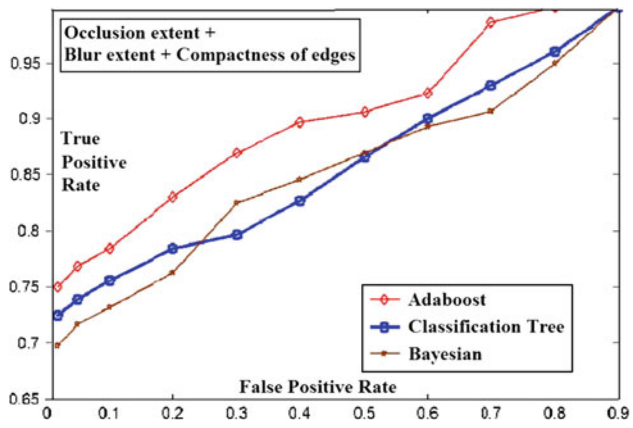

Fig. 10 Comparison of the classification accuracy of naive Bayesian classifier, adaptive boosting, and classification tree for the combination of different features

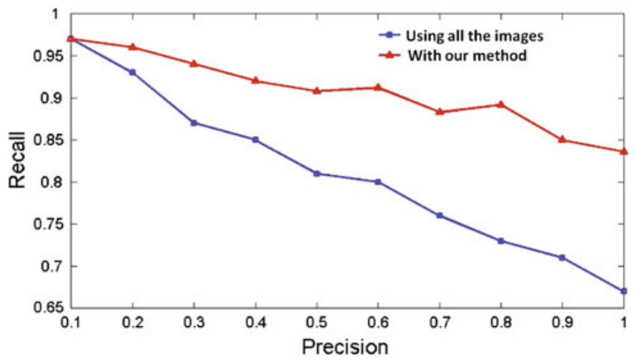

Fig. 11 Precision and recall curves for the matching process using all images (blue), and using the selected good candidate images with the proposed AdaBoost classifier

5.2 Simulation results and comparison

Figure 10 shows the classification performance with the naive Bayesian classifier, adaptive boosting, and classification tree for the combination of the three features. From Fig. 10, it can be seen that the AdaBoost algorithm with Bayesian classifiers provides the best performance when the prior information is known. When specific prior information about the appropriate form of classifier is lacking, the tree-based approach can yield classifiers with accuracy comparable to other methods. Figure 11 shows the precision and recall curves for the matching process, where the blue curve is the result of matching using all the images and the red curve represents the matching result with our method. Figure 12 provides visual illustrations of the SIFT-based scene-matching results using selected good representative images with our system, which demonstrates the effectiveness of our approach. Figure 13 shows more examples of good representative images with the proposed approaches. In the 25 good representative images chosen by our system, only 2 are false positives. The ones with bounding boxes in Fig. 13 are false positives. These false positives are mainly due to the presence of non-frontal faces, since the face detector is not very effective in this situation. We also show some of the images that are not suitable for scene matching with SIFT features in Fig. 14. All of them are classified as bad candidates by our algorithm.

In Table 1, we show the distribution of good and bad candidate images among the different scene classes using the classification tree, naive Bayesian classifier, and AdaBoost for classification, respectively. As shown in Table 1, the scene "BeachFun" has the largest percentage of bad images for matching purposes since it contains large areas of water. We compare the performance of our framework with SIFT [15], PCA-SIFT [11], and SURF [1] in terms of computational time. Like SIFT, PCA-SIFT [11] descriptors encode the

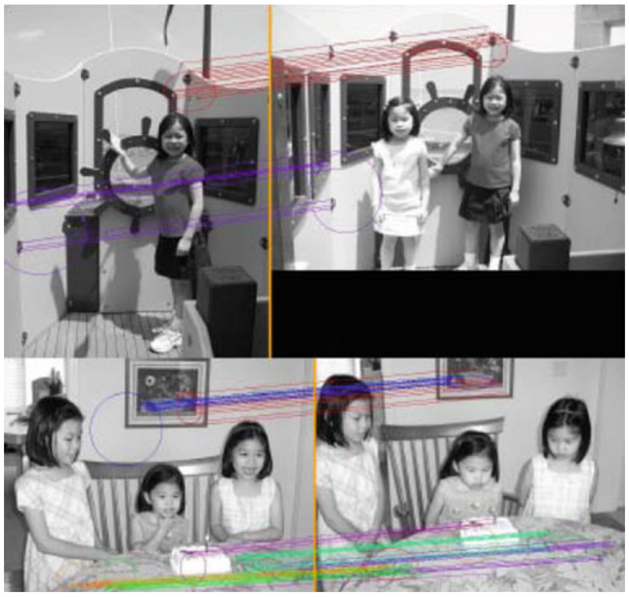

Fig. 12 Examples of successful scene matching with our selected candidate images

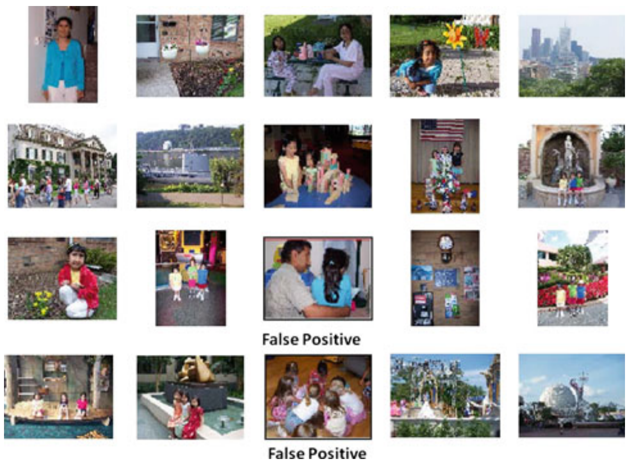

Fig. 13 Good representative images for fast scene matching with SIFT features, with false positives labeled

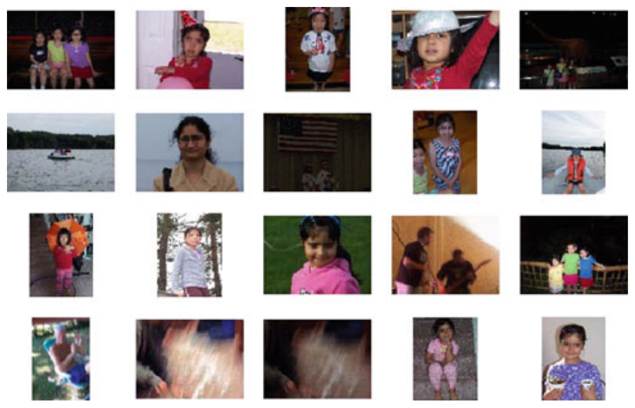
Fig. 14 Some of the images that are not suitable for fast scene matching with SIFT features, detected by our algorithm

salient aspects of the image gradient in the feature point's neighborhood. Instead of using SIFT's smoothed weighted histograms, PCA-SIFT applies Principal Components Analysis (PCA) to the normalized gradient patch. It is demonstrated that the PCA-based local descriptors are more distinctive, more robust to image deformations, and more compact than the standard SIFT representation. SURF (Speeded Up Robust Features) [1] is a faster performing scale- and rotation-invariant interest point detector and descriptor. SURF approximates or even outperforms previously proposed schemes with respect to repeatability, distinctiveness, and robustness, yet can be computed and compared much faster, which is achieved by (1) relying on integral images for image convolutions; (2) building on the strengths of the leading existing detectors and descriptors (using a Hessian matrix-based measure for the detector, and a distribution-based descriptor); and (3) simplifying these methods to the essential steps.

Table 2 shows the experimental results for finding scene matches for a collection of 3,000 images covering 46 events from consumer image collections. The precision and recall curves are evaluated in Fig. 15 with the proposed methods and other methods (SIFT, PCA-SIFT and SURF), where we demonstrate that the proposed methods significantly outperform other methods. We decompose the total computational time T_{total} of our methods into two terms:

$$T_{total} = T_{select} + T_{match}, \tag{11}$$

where T_{select} denotes the computational time of selecting good candidates and T_{match} denotes the computational time of SIFT matching of selected images. We utilize the software developed in [5] to evaluate the performance of this method using actual consumer images where using a novel clustering algorithm with intelligent filtering steps, consumer images with cluttered background and common objects can be matched effectively. The details for the software can be found in [5].

In Table 2, the first three rows are the computational time and accuracy with the traditional approaches such as SIFT, PCA-SIFT, and SURF. The fourth row represents the computational time and accuracy obtained using SIFT with the modified matching process described in [5]. The last three rows are the computational time and accuracy with our selec-

Table 1 The distribution of good and bad candidate images among different scene classes using the classification tree, naive Bayesian classifier, and AdaBoost classifiers, respectively

Event classes	BeachFun	Graduation	Wedding	Birthday party
Percentage of good images	3%/5%/5%	7%/8%/7%	11%/16%/15%	9%/11%/9%
Event classes	Urban tour	YardPark	Family time	Dining
Percentage of good images	26%/30%/27%	16%/19%/18%	9%/11%/11%	19%/21%/21%

Table 2 Comparison of search time and matching accuracy for traditional methods and our methods

	Total time (s)	Selection time (s)	Matching time (s)	Accuracy (%)
SIFT	7.425×10^4	0	7.425×10^4	75.2
PCA-SIFT	4.35×10^4	0	4.35×10^4	72.5
SURF	2.37×10^4	0	2.37×10^4	68.3
SIFT with modified matching process [5]	1.56×10^4	0	1.56×10^4	79.7
Classification Tree+SIFT	1,288.6	732.5	556.1	85.9
Bayesian Classifier+SIFT	2,536.5	1,952.6	583.9	83.5
AdaBoost+SIFT	2,876.2	2,234.9	641.3	92.6

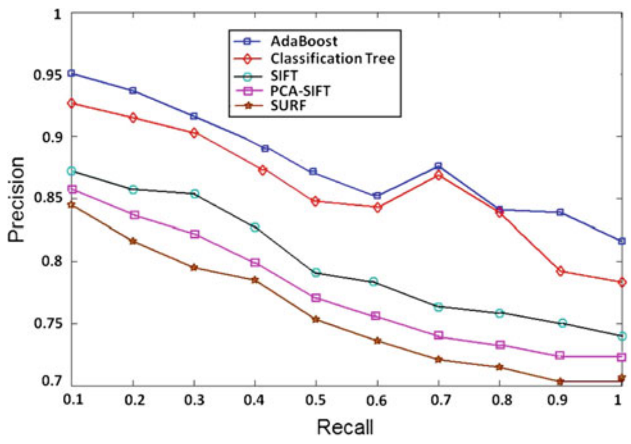

Fig. 15 Comparison of precision and recall curves for traditional methods and our methods

tion strategy using different classification methods. With the classification tree, 376 representative images are selected out of the 3,000 images, 457 representative images are selected using naive Bayesian classifiers, and 429 representative images are selected using AdaBoost, which corresponds to an average reduction 86, 82, and 83% in SIFT features, respectively. It can be seen that our approach can save computational time by an order of magnitude for scene matching. Compared with the result using modified matching process [5], the proposed approach achieves significant improvement which can be mainly attributed to the selection of representative images. Moreover, the proposed approach can achieve high accuracy with average precision 92.6% in matching with AdaBoost, which is significantly better than using SIFT (75.2%), PCA-SIFT (72.5%), SURF (68.3%), and the modified matching process [5] (79.7%), where we compute the classification accuracy by comparing the selected good candidates to the ground truth (example images that produced correct scene matches). Since human segments are deleted from the good matching candidates before the actual SIFT matching, the matching time reduction of the methods utilizing the proposed pre-processing comes from two source: (1) reduced number of matching candidate, and (2) reduced amount of image pixels after human deleted. In the experiment, we identified that 78% of time savings comes from the reduced number of candidates to be matched, and 22% of the time savings comes from the reduced number of image pixels remaining after deleting the pixels corresponding to humans in the image.

5.3 Further speed improvements using sub-images

In order to further speed up the process of scene matching, we propose to use sub-images for scene matching as the last step of the proposed algorithm. Since the computational time of SIFT feature matching depends on the size of the input images, down-sampling could be one way to further speed up the scene matching with SIFT features. However, down-sampling might affect the quality of the images and therefore performance accuracy cannot be guaranteed. Here we propose to use the idea of sub-images for scene matching with SIFT features. If we could use only a portion of the image that contains the most important information for scene matching, we could reduce the search space and further reduce the computational time for scene matching. However, in practice, not every candidate is suitable for segmentation into sub-images. For example, images with features that are spatially concentrated tend to be easier to segment. On the contrary, images with the features that are distributed may not be easy to segment. The computational savings due to the use of sub-images is reported separately since the extent of savings is data dependant. We propose the following scheme to evaluate whether an image is suitable for segmentation or not:

1. Once we obtain the edge map from the Canny detector from the previous feature extraction process, we compute the histogram of the edges in each image horizontally and vertically. In each dimension, we use $N = 20$ bins for computing the histogram, where the horizontal or vertical axis of the histogram represents the value of the summation of edge map in rows or columns.
2. For each image, we record the size of the smallest window that covers 80% edges from 20 bins both horizontally and vertically. We denote them as L_1 and L_2.
3. We compute $L = L_1 L_2$ and sort L for all of the images. We select the images that satisfy $L < L_0$ and consider them suitable for segmentation into sub-images, where L_0 is the threshold and L_0 is set to be 80 in the experiment due to its best performance.

If an image is good for segmentation, we use the bounding box (computed from step 3) to split the image into sub-images after removal of human regions in the image. Otherwise, we use the whole image for scene matching. We further measure the accuracy of scene matching by the total number of corresponding feature point matches. The accuracy is a number between 0 and 1, which is computed as the ratio between the number of corresponding feature points using sub-images and the number of corresponding feature points when using the entire images. From our experiments, the value of threshold L_0 selected is quite important in determining computational time. If the threshold is too large, the computational time savings will be insignificant. If the threshold is too small, the accuracy of scene matching will decrease due to loss of meaningful features.

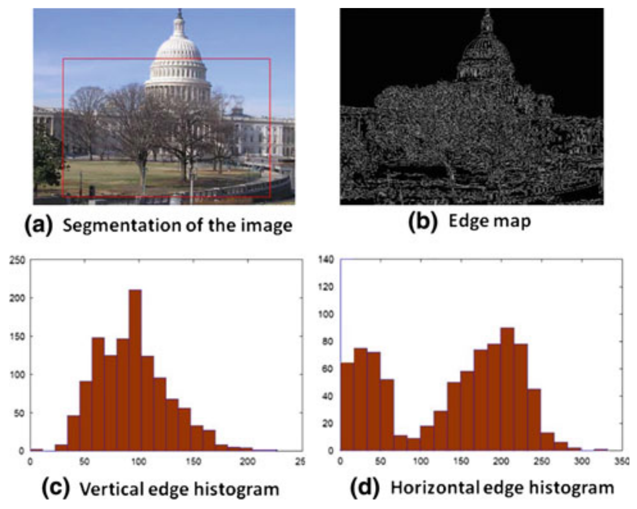

Fig. 16 Selecting a suitable sub-image for speeding up scene matching: **a** Segmentation of the image covering 80% edges. **b** Edge map by Canny edge detector. **c** Vertical edge histogram of the image. **d** Horizontal edge histogram of the image

We experimentally determine the threshold and then use the sub-images for scene matching to further reduce the computational time. Figure 16 shows an example of a sub-image selected from an image considered suitable for segmentation, where Fig. 16 provides the edge map by Canny edge detector, and Fig. 16c and d illustrates the vertical and horizontal edge histograms of the image, respectively. Table 3 demonstrates the tradeoff between accuracy, time savings, and the percentage in the number of SIFT features using sub-images for scene matching with different thresholds. The time taken for segmentation has been included in the computational time. The AdaBoost classifier is used to determine good candidate images. segmentation with a threshold of 80 provides better accuracy (0.946) than using the whole image (0.926) as shown in Table 3. Therefore, we select 80 as the threshold in our experiments, and we find that using a single sub-image with this threshold could further save about 16.7% the computational time in processing our database of 3,000 images when compared with using the entire images for scene matching. Meanwhile, it still maintains high matching accuracy. We show two examples of using sub-images for scene matching in Fig. 17.

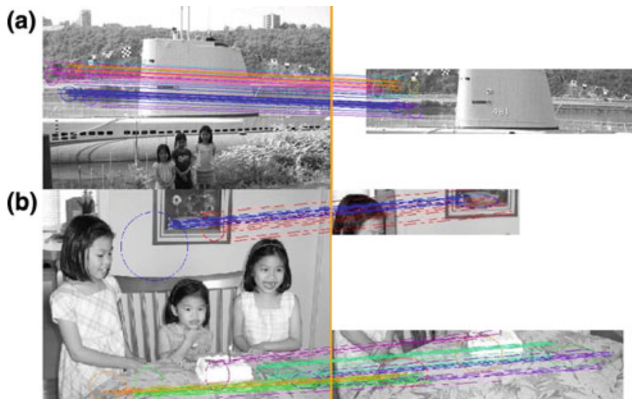

Fig. 17 Examples of using sub-images for scene matching with AdaBoost+SIFT

6 Conclusion

In this paper, we present an efficient framework for selecting good candidate images for scene matching using SIFT-like features. We propose a set of discriminative image features that are relevant to successful scene matching including occlusion extent, compactness of edges, and blur extent. The AdaBoost algorithm based on Bayesian classifiers and the classification tree method are proposed to integrate these features, creating a binary classifier to determine if an image is a good or bad candidate for scene matching. When prior information about distribution of images is known, the AdaBoost algorithm tends to provide better performance. When the prior information is lacking, the classification based on a tree structure is able to yield comparable performance. The proposed approach is an order of magnitude faster than scene matching using the traditional approach of comparing images pair-wise on large consumer image databases. Further improvement in speed is achieved when scene matching is performed on sub-images extracted from images that contain compact regions of interest.

The experimental results demonstrate the superiority of our approach in terms of saving significant computational time while achieving high accuracy. Future work, designed to further improve the accuracy of selection of candidate images, will focus on detection of non-frontal faces for better accuracy of feature computation and the use of event information to guide the choice of candidate images.

Table 3 Tradeoff between accuracy, time savings, and the reduction in the number of SIFT features when using sub-images for scene matching

Threshold	35	40	45	50	55	60	65	70	75	80
Accuracy	0.75	0.78	0.803	0.827	0.851	0.857	0.863	0.892	0.91	0.946
Time savings (%)	36	34.5	31.2	29.7	26.8	24.5	22.3	21.7	18.2	16.7
Reduction in no. of features (%)	23	20.5	17.2	15.8	13.7	11.6	10.3	9.7	8.6	8.2

References

1. Bay H, Ess A, Tuytelaars T, Gool LV (2008) Surf: speeded up robust features. Comput Vis Image Understand (CVIU) 110:346–359
2. Breiman L (1996) Bagging predictors. Mach Learn 24:123–140
3. Chou P (1991) Optimal partitioning for classfication and regression tree. IEEE Trans Pattern Anal Mach Intell 13:340–354
4. Cui W, Zhou H, Qu H, Wong P, Li X (2008) Geometry-based edge clustering for graph visualization. IEEE Trans Vis Comput Graph 14:1277–1284
5. Das M, Farmer J, Gallagher A, Loui A (2008) Event-based location matching for consumer image collections. In: Proceedings of the international conference on image and video retreival (CIVR)
6. Duda RO, Hart PE, Stork DG (2001) Pattern classification. Wiley, New York
7. Gao J, Yang J (2001) An adaptive algorithm for text detection from natural scenes. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)
8. Hall E, Davies D, Casey M (1980) The selection of critical subsets for signal, image and scene matching. IEEE Trans Pattern Anal Mach Intell 2(4):313–322
9. Jones M, Viola P (2003) Face recognition using boosted local features. In: Proceedings of the IEEE conference on computer vision
10. Karypis G, Han E, Kumar V (1999) Chameleon: hierarchical clustering using dynamic modeling. IEEE Comp 32:68–75
11. Ke Y, Sukthankar R (2004) Pca-sift: a more distinctive representation for local image descriptors. In: Proceedings of the IEEE conference on computer vision and, pattern recognition
12. Ke Y, Tang X, Jing F (2006) The design of high-level features for photo quality assessment. In: Proceedings of the IEEE conference on computer vision and, pattern recognition
13. Li L, Su H, Xing E, Fei-Fei L (2010) Object bank: a high-level image representation for scene classification and semantic feature sparsification. In: Proceedings of the neural information processing systems (NIPS)
14. Loui A, Savakis A (2003) Automated event clustering and quality screening of consumer pictures for digital albuming. In: Proceedings of the IEEE transactions on multimedia
15. Lowe D (2004) Distinctive image features from scale invariant features. Int J Comput Vis 60:91–110
16. Lucas BD, Kanade T (1981) An iterative image registration technique with an application to stereo vision. In: Proceedings of the imaging understanding workshop, pp 120–131
17. Maire M, Arbelaez P, Fowlkes C, Malik J (2008) Using contours to detect and localize junctions in natural images. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)
18. Quattoni A, Torralba A (2009) Recognizing indoor scenes. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)
19. Rasiwasia N, Moreno PL, Vasconcelos N (2007) Bridging the gap: query by semantic example. IEEE Trans Multimed 9:923–938
20. Schaffalitzky F, Zisserman A (2003) Automated location matching in movies. Comput Vis Image Understand Spec Isssue Video Retr Summ 92:236–264
21. Schapire RE, Singer Y (1999) Improving boosting algorithms using confidence-rated predictions. Mach Learn 37:297–336
22. Tong H, Li M, Zhang H, Zhang C (2004) Blur detection for digital images using wavelet transform. In: Proceedings of the IEEE international conference on multimedia and expo (ICME)
23. van de Sande KEA, Gevers T, Snoek CGM (2008) Evaluation of color descriptors for object and scene recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)
24. Wong R (1978) Sequential scene matching using edge features. In: Proceedings of the IEEE transactions on aerospace and electronic systems, vol 14