Multimedia Systems (2005) 10(6): 499–512
DOI 10.1007/s00530-005-0178-3

REGULAR PAPER

Giang P. Nguyen · Marcel Worring

Relevance feedback based saliency adaptation in CBIR

Published online: 31 May 2005

Abstract Content-based image retrieval (CBIR) has been under investigation for a long time, with many systems built to meet different application demands. However, in all systems there is still a gap between user expectations and system retrieval capabilities. Therefore, user interaction is an essential component of any CBIR system. Interaction up to now has mostly focused on changing global image features or similarities between images. We consider the interaction with salient details in an image, i.e., points, lines, and regions. Interactive salient detail definition goes further than summarizing the image into a set of salient details. We aim to dynamically update the user- and context-dependent definition of saliency based on relevance feedback. To that end, we propose an interaction framework for salient details from the perspective of the user. A number of instantiations of the framework are presented. Finally, we apply our approach for query refinement in a detail-based image retrieval system with salient points and regions. Experimental results prove the effectiveness of adapting the saliency from user feedback in the retrieval process.

Keywords Image retrieval · Image segmentation · Query definition · Salient details

Categories and subject descriptors

H.3.3 [**Information storage and retrieval:**] Information search and retrieval—*query formulation, relevance feedback*

General terms
Image retrieval

G. P. Nguyen (✉) · M. Worring
Intelligent Sensory Information Systems, Faculty of Science, University of Amsterdam, Kruislaan 403, 1098SJ Amsterdam, The Netherlands
E-mail: {giangnp,worring}@science.uva.nl

1 Introduction

Content-based image retrieval (CBIR) has been under investigation for a long time, and many systems have been built to cater to the varying demands posed by different applications. An extensive review can be found in [35]. Despite the large number of systems developed, a gap remains between user expectations and system retrieval capabilities. The main reason for this is the semantic gap in [35] defined as:

> "*The semantic gap is the lack of coincidence between the information that one can extract from the visual data and the interpretation that the same data have for the user in a given situation.*"

The gap can be limited by using even more advanced features [28, 35], by introduction of ontologies for structuring the possible user queries [33], or by using additional resources like associated textual information [2]. However, as the interpretation is subjective, a gap will remain. Thus, user interaction is an essential part of any practical CBIR system.

In [35], a framework for interactive CBIR called query space is defined capturing all of the state-of-the-art interaction mechanisms. The query space is defined as $\mathcal{Q} = \{I_\mathcal{Q}, F_\mathcal{Q}, S_\mathcal{Q}, Z_\mathcal{Q}\}$. The first component is the set of active images $I_\mathcal{Q}$, the second is a selection of features $F_\mathcal{Q}$, the third is a similarity function $S_\mathcal{Q}$ used to compare images in the database, and the last one is a set of symbolic labels $Z_\mathcal{Q}$ with an associated probability for each image. The retrieval process in query space consists of five main steps. The first step is *initialization* in which the system initiates a query space. A query is then defined in the *specification* step. In the *visualization* step, retrieved results of the query are mapped into an n-dimensional display space. In the *feedback* step the user gives relevance feedback RF_i by changing one or more components in query space $\mathcal{Q}$:

$$\left\{I^i_\mathcal{Q}, F^i_\mathcal{Q}, S^i_\mathcal{Q}, Z^i_\mathcal{Q}\right\} \overset{\mathrm{RF}_i}{\rightarrow} \left\{I^{i+1}_\mathcal{Q}, F^{i+1}_\mathcal{Q}, S^{i+1}_\mathcal{Q}, Z^{i+1}_\mathcal{Q}\right\}. \quad (1)$$

The final results are returned in the *output* step.

Existing interactive systems mostly focus on changing I, S, or Z. For example, Vendrig focuses on changing I [39]. In [27], positive and negative examples are used in the similarity function S to update the weighting of different features. In [31], the user changes the relative position of the images in visualization space to update the similarity. Minka [23] learns labels in Z like grass, and brick from the user feedback. In [20], the user redefines the label of the object obtained by the system such as names of people or special type of animal.

Few systems support user feedback on the features. QBIC [9] has the user select features such as color, texture, and shape. The Pictoseek system [10] allows users to decide between different color spaces like RGB, HSI, and rgb. However, these selections have to be made beforehand by the user; they cannot be changed during interaction except by starting all over again. Dynamic feature selection was proposed in [27], but only for global features.

An alternative to global features is salient details to represent the image content where salient details can be points [12, 21, 34], lines [18, 26], or regions [7, 25, 40].

These methods are focused on automatically summarizing the image into a set of salient details. More precisely, they use a fixed definition of saliency. However, the saliency is user and context dependent and thus should be defined by the user through interaction.

Therefore, we aim at an interactive definition of saliency. To that end, we need to analyze existing detail detection methods and summarize them into a unifying framework. The framework is presented in order to find out which elements are involved in changing the output results. These are called *tunable parameters*. The main idea of our method is to tune those parameters to best fit the user feedback. From there we present an application of our approach in CBIR. Thus our approach follows the pattern:

$$\left\{I_{\mathcal{Q}}, F^{i}_{\mathcal{Q}}, S_{\mathcal{Q}}, Z_{\mathcal{Q}}\right\} \overset{\mathrm{RF}_i}{\rightarrow} \left\{I_{\mathcal{Q}}, F^{i+1}_{\mathcal{Q}}, S_{\mathcal{Q}}, Z_{\mathcal{Q}}\right\} . \quad (2)$$

To follow this pattern the system is composed of an offline stage to precompute candidate salient details and an interactive stage in which the user does the interactive retrieval. It yields a general framework for interactive retrieval based on salient details. Thus it can be used to create an add-on to existing methods rather than replacing them.

This paper is organized as follows. Section 2 describes the *offline stage*. This section is also a review of existing salient detail detectors in order to find the tunable parameters for the *interactive stage*. Different sets of candidate salient details are extracted by changing those parameters. In Sect. 3, this second stage of our framework is presented with methods to tune the parameters based on user feedback. Section 4 shows results to demonstrate the new approach.

2 Offline salient detail detection

Although the salient detail detectors in the literature follow different approaches, every method can be divided into five main steps: image processing, detail detection, feature computation, saliency computation, and selection based on significance. We will now define these steps more precisely.

Image processing is the step where both the input I and output I^* are one or more images. Image processing removes irrelevant information like noise or enhances specific image content like edges or contrast. An image processing operator is denoted by $e_{\vec{\sigma}}$, where $\vec{\sigma}$ is an element in the space Σ of parameters steering the process. Thus we denote the image processing step by

$$I^* = e_{\vec{\sigma}}(I). \quad (3)$$

Detail detection is the process whereby the image is decomposed into details $\mathcal{D}$, where the details can be regions, lines, or points. By adjusting the set of parameters $\vec{\omega}$ in parameter space Ω, different sets of details are obtained. The detail detection step is given as

$$\mathcal{D} = p_{\vec{\omega}}(I^*) , \quad (4)$$

where $p_{\vec{\omega}}$ is a specific segmentation method.

Feature computation calculates the set of feature values $\mathcal{F}$ over all detected details. A feature can be any description of a detail, e.g., based on color, texture, shape, or size. We denote feature computation for a set of details $\mathcal{D}$ as

$$\mathcal{F} = f_{\vec{\lambda}}(\mathcal{D}) , \quad (5)$$

where $\vec{\lambda}$ is a set of parameters in parameter space Λ to compute the feature f.

Saliency computation calculates how salient the features of a detail are relative to other details within its spatial context. Thus it needs to define a local area $\pi \in \Pi$ in which feature values of other details are considered. In the literature most methods consider π to cover effectively the whole image, thus ignoring local spatial context. We define it in the more general form where within the context defined by π this step typically uses an operator l to compute the local extrema. It could further enhance the saliency of the detail by normalizing the saliency inside the context. An example of this is shown in Fig. 2. We denote it by $l_{\vec{\pi}}(\mathcal{D})$, where $\vec{\pi}$ defines the interval or area used in determining the saliency of a certain detail. This step will select details standing out from the other details in the set $\mathcal{D}$. The general form of this step is

$$\mathcal{L} = l_{\vec{\pi}}(\mathcal{F}) . \quad (6)$$

Selection based on significance is the step that finally selects details $\hat{\mathcal{D}}$ based on the saliency values computed in the previous step. It reveals the most salient parts of the image, e.g., by defining a given threshold on the saliency values or by restricting the number of output salient details. The selection function is denoted by $g_{\vec{\gamma}}$ with $\vec{\gamma} \in \Gamma$. It gives

$$\hat{\mathcal{D}} = g_{\vec{\gamma}}(\mathcal{L}) . \quad (7)$$

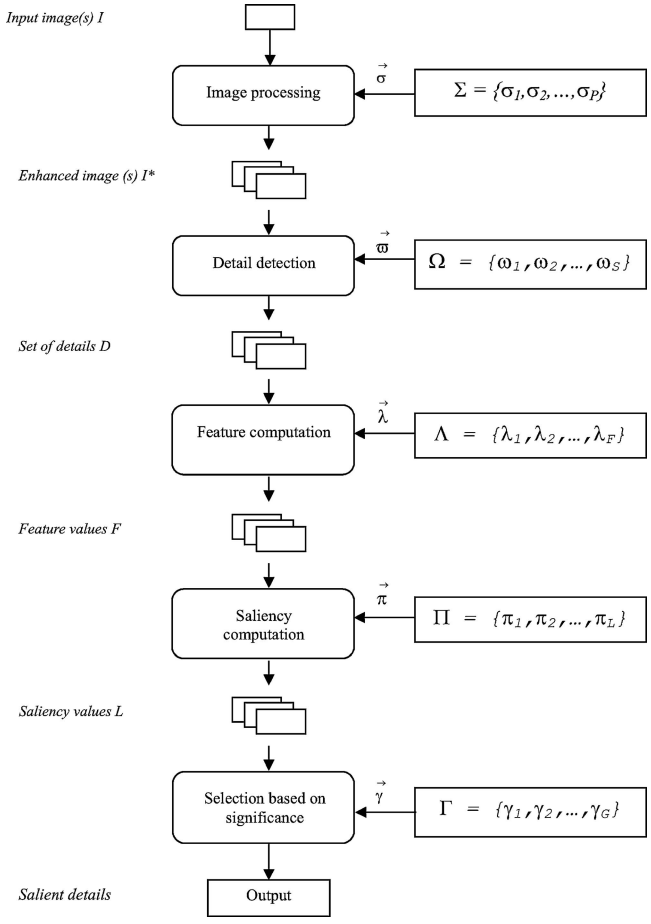


Fig. 1 Framework for offline salient detail detectors

Going through the general scheme, the whole process of automatic salient detail detection can be described as

$$\hat{\mathcal{D}} = g_{\vec{\gamma}} \circ l_{\vec{\pi}} \circ f_{\vec{\lambda}} \circ p_{\vec{\omega}} \circ e_{\vec{\sigma}}(I)\,, \tag{8}$$

where I is the input image and $\hat{\mathcal{D}}$ is the final output set of salient details, $\hat{\mathcal{D}} \subseteq \mathcal{D}$.

As stated, each step has its own set of parameters. So for each salient detail detector we have a set of parameters $\theta = (\vec{\sigma}, \vec{\omega}, \vec{\lambda}, \vec{\pi}, \vec{\gamma})$. By changing any of the parameters in θ, we get a different set of salient details.

Figure 1 shows the model for the salient detail extraction process. Each step is described in more detail in the sequel by considering how it is defined in existing methods.

2.1 Image processing

Image processing enhances specific image characteristics. Examples are smoothing, gradient computation, or enhancement of linelike structures. These methods all involve a scale parameter [1, 14, 18, 36].

Recognizing that the proper scale is dependent on the user need, scale-based methods consider a range of parameter values. Different scale spaces have been defined. Sebe et al. [34] use the wavelet transform to represent image variations at different scales. Another similar approach considers the input image(s) at different resolutions, but with different features. As an example in [7], luminance, color, and texture at several scales are computed at every position in the image. The strength of inhomogeneities of luminance, color, and textures are used as indicators of edge evidence. An accumulated edge evidence map (AEEM) for different scales and different local measures is then created. Another example is in [13], where Itti builds a scale space where for every pixel in the image he computes color, intensity, and orientation. Along the same line, Salah et al. [30] use line orientations as the features at different scales. In [41], Walker uses normalized Gaussian derivative kernels with different scales to construct the differential structure of the image.

In summary, the process of salient detail detection, at this step, represents the input image I as one or more images in which specific characteristics are emphasized, but with scale as the most important parameter.

2.2 Detail detection

Detail detection involves the segmentation of images into a set of details, namely, points, lines, or regions. This step may use statistical classification [41], edge detection [12], region detection [7], or a combination of these techniques [15]. Each method has its own parameters.

Regarding point detection, Schmid [32] gives a comprehensive overview. Examples of methods segmenting images into points are also given in [12]. The author first segments the image into lines and then finds the "cotermination" of pairs of lines under some constraints.

Also in [12], Qasim uses a perceptual grouping algorithm to segment the image into line-based details such as: line segments, L-junctions, and U-junctions. Based on the intensity image, pixels are grouped into "line-support" regions if they have a similar gradient orientation. Other examples are in [1, 18].

For region detection, a typical example is the method in Blobworld [5]. After a set of features are computed at each pixel at a selected scale, the system then groups pixels into regions using the EM algorithm. Hoang in [11] uses k-means clustering to group pixels in the image feature space to find homogeneous regions. In [6], Cinque et al. segment the image into regions using a region-growing algorithm. Each time a pixel is added to a region, its four nearest neighbors that have not been processed are considered. A threshold on the color difference is used to decide whether a point belongs to the region or not. In the reference, they also merge two nearby regions if the difference between their mean colors is less than a threshold. These two thresholds determine the region segmentation results.

2.3 Feature computation

Features can be any description of the details. Examples are the color histogram of a region, the length of a line, or the curvature at a point on a curve. Changing the features used and steering their computation gives another possibility for changing the final set of salient details.

In [22], where the image with candidate points is represented in a scale space, the scale invariance of points is computed as feature. In [34], for every point extracted in the previous step using wavelets, the feature computed at each point is the sum of the coefficients along the trace from the coarse level to the finer one. This method thus depends on the wavelet function selected.

Another approach to computing features of points is described in [41]. Several vectors of invariants are formed at each candidate point in the image. These vectors build a multivariate distribution. At each point, they estimate the local density in a distribution by summing the contribution from a mixture of Gaussians. The density estimated for each vector of invariants is used as a feature at the appropriate scale.

In the case of lines, [1] uses Gestalt principles, which calculate the following features of curves: length of curve, total curvature or energy, and amount of fragmentation. In [18], the authors use the expected number of closed contours that pass through an edge as its feature.

In [12], the length of a line is used. In [26], the author presents some features of edges such as lifetime, which is the time that an edge persists before disappearing during the blurring.

Feature computation for regions is presented in [7, 13, 20, 25]. Cinque [6] simply uses the mean color of a region. In [7], Dimai computes the strength of inhomogeneities of luminance, color, and texture as the feature of the region considered.

The feature computation step is very important for providing good candidates for the next step. This step depends on the segmentation step as it needs correctly segmented details. The most important parameter is the choice of the right feature to compute for each detail.

2.4 Saliency computation

For humans, saliency is always defined relative to a neighborhood; thus saliency should be computed locally. However, most of the methods compute saliency as global saliency based on the feature values. This leads to missing local salient details that in the whole image may not seem significant but within a local area become salient. An example of locally defined saliency is given in Fig. 2.

It follows that the typical parameter for this step is Π determining the area or the interval where the saliency is considered [22, 32]. For example, in [32], the authors first apply the Harris detector for extracting corner points. They then use relatively small circles around each corner to compute the saliency of that point based on variance under rotations.

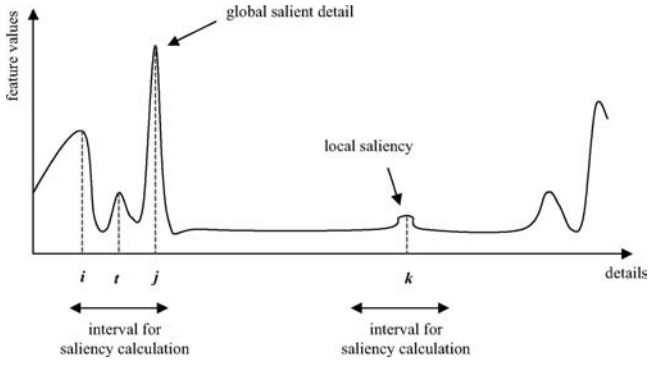


Fig. 2 Example of local saliency. For global saliency computation, detail k will not be considered salient. Within the interval, the local maxima detector returns k as a salient detail since, compared to its neighbor, it exhibits a significant change in feature value

In the special case where Π denotes the whole image, it boils down to global saliency computation.

Hence, in the output of this step, for each candidate detail d, the saliency value is $l_{\vec{\pi}}(d)$.

2.5 Selection based on significance

For describing an image one should select a limited set composed of the most relevant salient details. Given saliency based on feature values, we classify existing methods in the following two types. The first type employs a simple way of selection by using a threshold on the number of details. This ignores the varying complexity of different images. The second type is more natural using a threshold δ on the saliency values [3, 6, 12]. For example, in [12], a threshold is used to select the longest lines. In general, this approach can be represented as follows:

$$\hat{\mathcal{D}} = \{\, d \subset \mathcal{D} \mid l_{\vec{\pi}}(d) > \delta, \delta \in \mathbb{R} \,\}. \tag{9}$$

The following is the special case requiring no parameters where only the most salient detail will be returned:

$$\hat{\mathcal{D}} = \underset{d \in \mathcal{D}}{\operatorname{argmax}}\, l_{\vec{\pi}}(d). \tag{10}$$

3 Interacting with salient details

In the *interactive stage*, the aim is to take the offline data as the basis and find the details that are salient from the user's perspective. At this point we should make a distinction between salient details for which visual evidence is present in the image and details for which this is not the case. The latter can only be found by the system using prior knowledge on the shape [1]. We focus on the former case and hence make the assumption that for a user-desired salient detail the evidence can be found for some parameter setting. However, due to the "*semantic gap*," the system cannot define automatic methods for setting those parameters in a general domain. Support by the user to find the most appropriate parameters is required.

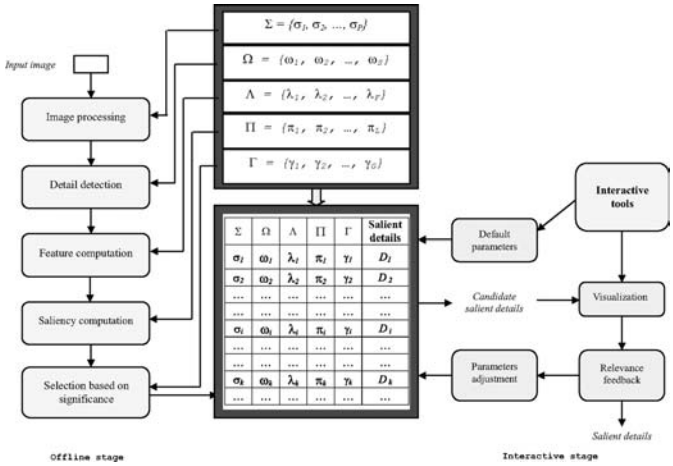


Fig. 3 Interaction framework with salient details

3.1 Processing steps

In current systems, changing the features, if possible at all, is done by manipulating the parameters and visualizing the result. This is acceptable for a computer vision expert, but being able to control the system in such a manner is not feasible for the end user. Therefore, rather than having the user manipulate the parameters directly, we let the user give relevance feedback on the results obtained. Based on this user interaction, the system then has to select the optimal parameters accordingly.

Thus, the *interactive stage* consists of four stages: initialization of the result, visualization of the result using default parameters, relevance feedback, and parameter adjustment, which will now be described. Figure 3 presents the whole framework composed of the two stages.

Initialization This step starts the interactive stage with default parameters θ_0 as suggested by the original methods. Those values are sent to the database, and the system returns the default salient detail set $\hat{\mathcal{D}}^0$.

Visualization In this step, the details and their properties are visualized to help the user in giving useful relevance feedback. In the simplest case, points and lines are shown with their positions, and regions are shown with their boundaries overlaid on the image. A more advanced method would also show why these details are considered more salient than the others.

Relevance feedback By interacting with the objects visualized, the user gives relevance feedback RF_i to the system. A number of possibilities are indicating positive/negative examples, denoting a degree of ir/relevance with respect to the target, changing positions, or indicating region of interest, merging, or splitting [17, 19, 24, 29]. The general form of this step is

$$\hat{\mathcal{D}}_{\theta^i} \xrightarrow{\mathrm{RF}_i} \hat{\mathcal{D}}_{\theta^{i+1}}\,, \tag{11}$$

where $\hat{\mathcal{D}}_{\theta^i}$ is the current set of salient details and $\hat{\mathcal{D}}_{\theta^{i+1}}$ is the set returned after the relevance feedback.

Parameter adjustment Given the user feedback RF_i, the system should find a set of parameters $\hat{\theta}^{i+1}$ optimizing some criterion. Let us consider an example in the case of points. When the user selects a region of interest, the new set of salient points should contain as many points as possible in the interest region while limiting the number of points in other regions. A general form for this is given by

$$\hat{\theta}^{i+1} = \underset{\theta}{\operatorname{argmax}}\ \mathcal{E}_{\mathrm{RF}_i}(\hat{\mathcal{D}}_{\theta^{i+1}})\,, \tag{12}$$

where $\hat{\mathcal{D}}_{\theta^{i+1}}$ is the set of salient details returned with parameter θ^{i+1} and $\mathcal{E}_{\mathrm{RF}_i}$ is an evaluation function on the output salient details. The choice for this function depends on the specific application. A general criterion is that the new set of salient details should be closer to the user-expected result than the previous one.

To avoid having the user become disoriented by large changes, a constraint $\mathcal{C}$ is added. In other words, the constraint assures a smooth path to the optimal query. For example, a constraint could be that the size of the new set of salient details should not be much larger than the previous set. The general constraint is represented as

$$\mathcal{C} = \mathcal{C}(\hat{\mathcal{D}}_{\theta^i}\,, \hat{\mathcal{D}}_{\theta^{i+1}})\,. \tag{13}$$

From 12 and 13 this parameter adjustment is described as

$$\hat{\theta}^{i+1} = \underset{\theta}{\operatorname{argmax}}\ \mathcal{E}_{\mathrm{RF}_i}(\hat{\mathcal{D}}_{\theta^{i+1}}\,|\mathcal{C})\,. \tag{14}$$

After *parameter adjustment*, the system finds the optimal set of parameters. These values will be sent to the database, after which a new set of candidates is returned. The process is repeated until the user accepts the results.

3.2 Offline computation

We have shown that salient details can be computed using a five-step process where each step is steered by a set of parameters. For interaction it is not feasible to do all the computation at run time. Therefore, for a given range of the parameter values we precompute the salient details and store them in a database in the offline computation step. At each step, we determine the most important parameter while keeping the other parameters to their default values. Computing details beforehand leads to the problem of storing the offline data. As only the resulting details have to be stored, the storage requirements are moderate compared to the space required for the image itself. For example, in the case of salient points, only their coordinates are stored for each parameter setting. In the case of salient regions, the storage is also moderated, as for each parameter setting an identification image is stored where the value corresponds to a region. As the number of regions is limited, the identification image of an RGB image is on average 3 KB only (in

PNG format with the size of an image is 384 × 256 or 256 × 384).

Assume the setting for a parameter is 10 different values. In the worst case, when the method employs each of the five steps per parameter, the size of Σ is then 10^5. Each segmented image has size 3 KB, so for example with 10,000 images, total storage is 3 GB. It seems that the setting for each parameter is unlimited. In practice, those parameters should not be too different from the default ones since we meant to adjust them to suit a specific query. As in our experimental results, the setting takes the default parameters as a standard point then decreases and/or increases within a certain range to get the new set of parameters. Hence, this will limit the explosion of possible combinations between parameters.

In what follows the precomputed data for a given input image are called the *offline data* for that image. The size of the *offline data* is $O(N_{\text{it}})$:

$$N_{\text{it}} = (\|\Sigma\| \times \|\Omega\| \times \|\Lambda\| \times \|\Pi\| \times \|\Gamma\|)\,,$$

where the $\|\cdot\|$ denotes the size of a set.

When the dataset grows beyond 10,000 images, we should consider the use of special data structures to make the access to the offline data more efficient. This depends on the type of relevance feedback given by the user, the specific function chosen in Eq. 12, and the constraint function chosen in Eq. 13. With a carefully chosen constraint it is feasible to obtain a considerable pruning of the search region in parameter space. That is, for every image and for every parameter setting we can precompute which other parameter settings fall within the constraint. Links to those parameter settings can be stored in an index. Only if the initial query image is chosen outside the dataset do we have to go through the whole parameter space.

4 Instantiations of the framework

In the previous section a general framework for interacting with salient features was defined. To show its validity, we now introduce three example instantiations of the framework to show how it can be used to add an interaction step to existing methods. Each example illustrates different aspects of the framework to show how it can be used to add an interaction step to existing methods. These examples are respectively based on points, lines, and regions. Finally, we show how to apply the latter as the query redefinition step in CBIR.

4.1 Interacting with salient points

For offline salient point detection, the Harris detector is used [16]. The Harris detector has a set of parameters $\theta = \theta(\sigma, r, t)$, where σ is the standard deviation of the smoothing Gaussian used in the image processing step, r is the mask radius considered for local maxima in the detail detection step, and t is the threshold for selecting output salient points.

The *offline stage* as described above stores an archive of possible sets of salient points by computing results for $\sigma = \{1.0, 2.0, 3.0, 4.0\}$, $r = \{1.0, 2.0, 3.0, 4.0\}$, and $t = \{100, 200, 500, 800, 1000, 1100, 1200, 1500, 1800, 2000, 2500\}$. The default set of parameters is given by $\theta^0 = (1.0, 1000, 1.0)$ [16]. In this example, the relevance feedback is the selection of a region of interest.

Given an input image with a set of points extracted by the initialization step, the user then selects a region of interest $\mathcal{T}$ by drawing a rectangle with points inside taken as positive examples. This relevance feedback is used to search through the offline database to find the optimal result.

The updating step searches for a θ^{i+1} satisfying Eq. 12. In this experiment, we aim at finding many points inside the region of interest, and few points outside. Hence, the system finds θ^{i+1} such that

$$\theta^{i+1} = \underset{\theta}{\text{argmax}} \left(\frac{\|P_{\text{in}}(\theta)\|}{\|P_{\text{in}}(\theta)\| + \|P_{\text{out}}(\theta)\|} \right),$$

where

$$\begin{cases} P_{\text{in}}(\theta) &= \mathcal{T} \cap \hat{\mathcal{D}}_\theta\,, \\ P_{\text{out}}(\theta) &= \hat{\mathcal{D}}_\theta \setminus \mathcal{T}\,. \end{cases}$$

To assure that points are gradually added to $\mathcal{T}$, we use the constraint

$$\mathcal{C} = \{(1-\epsilon)\|\hat{\mathcal{D}}_{\theta^i}\| < \|\hat{\mathcal{D}}_{\theta^{i+1}}\| < (1+\epsilon)\|\hat{\mathcal{D}}_{\theta^i}\|\}\,,$$

with $\epsilon = 0.2$. This means that within the constraint $\mathcal{C}$, we find a parameter set θ that returns the maximum value of $\frac{P_{\text{in}}}{P_{\text{in}}+P_{\text{out}}}$. This is illustrated in Fig. 4.

We now show some results of using the method thus defined. In the first example, we consider the image depicted in Fig. 5a. With the default set of parameters, the Harris point detector returns 463 points, mostly lying along the borders. In case the user wants to have points inside the

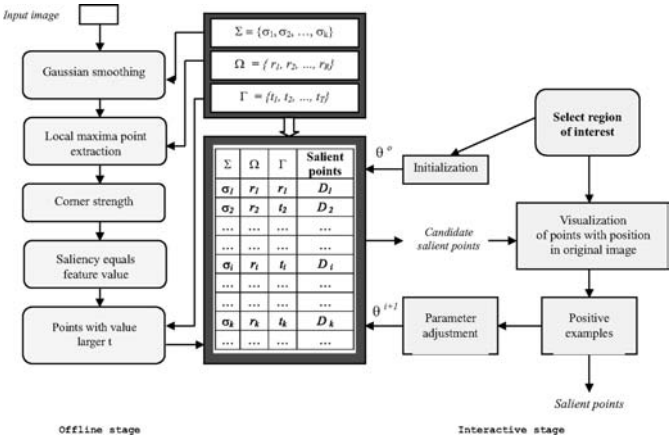


Fig. 4 Instantiation of the framework for the Harris point detector

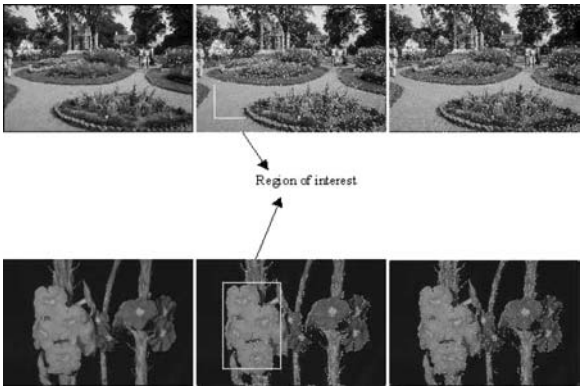


Fig. 5 Examples of experimental results. The *left* images are query images. The *middle* ones are images with default automatically detected salient points superimposed; the *rectangles* show the regions of interest with positive examples given by the user. The *right* images are the images returned after user feedback

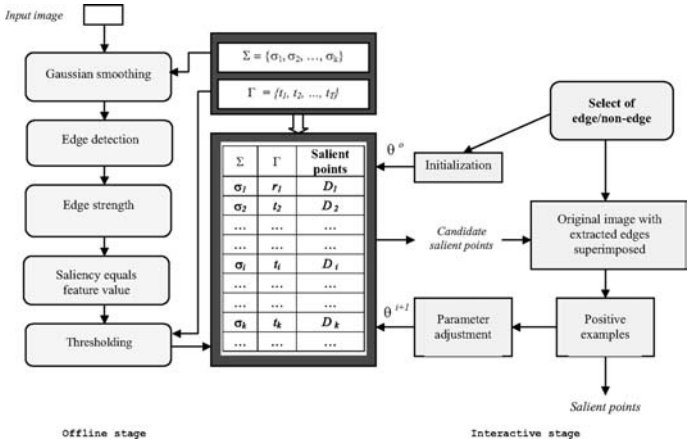


Fig. 6 Instantiation of the framework for Canny edge detection

petal area, the default parameter set fails to return the set of salient points inside that region. Within the region of interest, the number of positive points is 128/463. Based on the user feedback, the system finds the new values for parameters that return the total number of 475 points, with more points in the region of interest, namely, 162/475.

The second example is similar. The default parameter set returns 1904 points with 39 positive points. The system updates the parameters, and 1568 points are found. The final set of salient points (107/1568) is still able to cover the main corners (borders) and able to representing the user-selected region.

4.2 Interacting with salient lines

In this section, we show an instantiation of the framework for salient line extraction.

In this example, we employ the Canny edge detector [4]. In the image processing step, the input image is smoothed using a Gaussian mask. Gradient magnitude and edge direction are then computed at each pixel to extract edges. In the final step, Canny uses a threshold over the average strength of candidate edges, which will yield a set of salient edges.

The output is given to the user to provide feedback. The final image should contain as many details as possible that are salient from the user's perspective. The instantiation is shown in Fig. 6.

From the above we collect the set of parameters for this method as $\theta = \theta(\sigma, t)$, where σ is the standard deviation of the Gaussian and t is the threshold to select whether or not there is an edge point. Default θ^0 is set to (1.0, 3.0) as proposed in [8]. The *offline stage* stores archive results for parameters $\sigma = \{1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4, 4.5, 5\}$ and $t = \{1.0, 2.0, 2.5, 3, 3.5, 4, 5, 6, 6.5, 7\}$.

In this experiment, we let the user give feedback by selecting the line that they want to have removed. Here we make the assumption that these edges occur because the level of detail is too high; hence too many edges of limited length are present. Therefore, the length l_0 of the selected line is computed. The new θ^{i+1} will be searched through the *offline data* such that the new set of salient lines contains as few lines as possible with length smaller than l_0. Thus,

$$\theta^{i+1} = \underset{\theta}{\operatorname{argmax}} \left(\frac{\|\mathcal{D}_p(\theta)\|}{\|\mathcal{D}_p(\theta)\| + \|\mathcal{D}_n(\theta)\|} \right),$$

where

$$\begin{cases} \mathcal{D}_p(\theta) = \{d : \|d\| > l_0\}, \\ \mathcal{D}_n(\theta) = \{d : \|d\| \leq l_0\}. \end{cases}$$

The constraint is

$$\mathcal{C} = \left\{ \|(1-\epsilon)\|\hat{\mathcal{D}}_{\theta^i}\| \leq \|\hat{\mathcal{D}}_{\theta^{i+1}}\| \leq (1+\epsilon)\|\hat{\mathcal{D}}_{\theta^i}\| \right\},$$

where we use $\epsilon = 0.1$.

To start the system, we begin with the smallest value of σ and t to obtain all possible edges. Each time the unwanted lines are removed. This is illustrated in Fig. 7. We observe that the image after user feedback has fewer petty lines compare to the result using default parameters. Of course, this experiment can also be done in the case where the user looks for more-detailed edges. The user then gives feedback by pointing at regions in the image where edges should be found but are not present. This is quite similar to the case of finding points of interest as presented in Sect. 4.1.

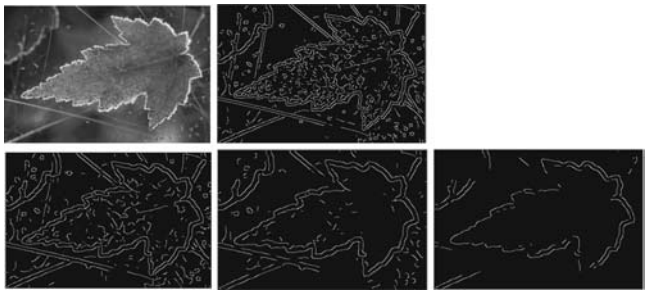


Fig. 7 Examples of experimental results with interactive salient line detection. The *first row* contains the original image and the edge image with default parameters. Some results after user interaction are shown in the *second row*

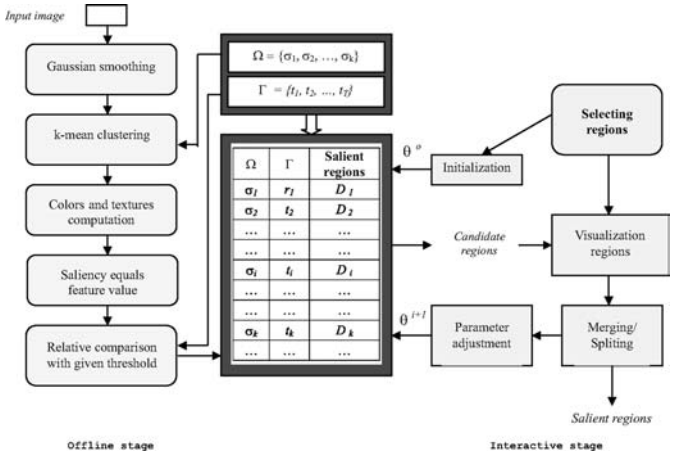


Fig. 8 An instantiation of the framework using Hoang's region segmentation

4.3 Interacting with salient regions

In this example, we work with the region segmentation method from [11]. In this reference, starting with Gaussian smoothing, the image is filtered using a set of Gabor filters. All pixels are represented in the feature space given by the Gabor results and the color. A k-means clustering method is then used to group pixels into regions. They collect the same color and texture features from the previous step to find the similarities between extracted regions. Regions with similarities larger than a given threshold are then merged to give the final salient regions. The instantiation is shown in Fig. 8.

Different parameters are used in the process; we select two of them as tunable parameters: the scale σ used in the Gabor filters computation and the similarity threshold t. Thus we collect a set of parameters for this method as $\theta = \theta(\sigma, t)$. The remaining parameters for the Gabor filter, namely, the specific frequencies and orientations, are taken following the guideline in [11]. The default values are $\theta^0 = (\{4, 3.5, 2.95, 2.35, 1.75\}, 7.5)$. In the experiment, we take $t \in [4.0, 8.0]$ with step 0.5 and $\sigma \in [1.0, 6.0]$ with step 0.05 and apply them to the method.

The resulting salient regions are visualized. The user gives feedback by asking the system to split or merge regions. When the user asks for a global split or merge, the system searches for a $\hat{\theta}$ that returns $\hat{\mathcal{D}}'_\theta$ with a number of regions larger or smaller respectively than the previous result $\hat{\mathcal{D}}$, i.e.,

$$\theta^{i+1} = \underset{\theta}{\text{argmax}} \left(\| \hat{\mathcal{D}}_\theta \| \right),$$

$$\text{with} \begin{cases} \mathcal{C} = \{ \| \hat{\mathcal{D}}_{\theta^i} \| < \| \hat{\mathcal{D}}_{\theta^{i+1}} \| < (1+\epsilon) \| \hat{\mathcal{D}}_{\theta^i} \|, \ \epsilon \in \mathbb{R}^+ \}, \\ \mathcal{C} = \{ \| \hat{\mathcal{D}}_{\theta^i} \| > \| \hat{\mathcal{D}}_{\theta^{i+1}} \| > (1-\epsilon) \| \hat{\mathcal{D}}_{\theta^i} \|, \ \epsilon \in \mathbb{R}^+ \}. \end{cases}$$

The new θ^{t+1} is returned with the maximum number of regions or minimum, in the case of splitting or merging respectively, within the constraint $\mathcal{C}$.

In our experiment, we simply select ϵ such that the number of salient regions will increase/decrease 20% in the new set. If $\epsilon * \| \hat{\mathcal{D}}_{\theta^i} \| < 1$, then the constraint will be ± 1 region, respectively. Therefore,

$$\epsilon = \max \left(0.2, \frac{1}{\| \hat{\mathcal{D}}_{\theta^i} \|} \right).$$

In case of a local split, the system tries to keep the outer boundary of the selected region and introduces inner boundaries. With local merging of regions, the system keeps the outer boundary and removes inner ones. This leads to general criteria for selecting a set of parameter θ, which will now be explained.

Assume the current set of salient regions is $\hat{\mathcal{D}} = \{\hat{d}_1, \ldots, \hat{d}_k\}$. The user decides to merge two regions $\hat{d}_i$ and $\hat{d}_j$. The system returns $\hat{\mathcal{D}}'_\theta = \{\hat{d}'_1, \ldots, \hat{d}'_t\}$ such that

$$\exists m : \ \hat{d}'_m \simeq \{\hat{d}_i \cup \hat{d}_j\},$$

where $\simeq$ denotes that the new region $\hat{d}'_m$ approximates the combination of the two regions $\hat{d}_i$ and $\hat{d}_j$ with an error $\tau \in \mathbb{R}^+$.

Therefore, let $\mathcal{A}(\hat{d})$ denote the area of $\hat{d}$; then the evaluation function will be

$$\mathcal{E} = ((\mathcal{A}(\hat{d}'_m) \cup \mathcal{A}(\{\hat{d}_i \cup \hat{d}_j\})) \setminus (\mathcal{A}(\hat{d}'_m) \cap \mathcal{A}(\{\hat{d}_i \cup \hat{d}_j\}))),$$

with constraint $\mathcal{C} = \{\| \hat{\mathcal{D}}_{\theta^{i+1}} \| = \| \hat{\mathcal{D}}_{\theta^i} \| - 1\}$.

In the similar case for local splitting, we have $\hat{d}_m \simeq \{\hat{d}'_i \cup \hat{d}'_j\}$. Those formulas can easily be generalized for n regions. Repeating this process, the user will be able to find the meaningful segmentations. Figures 9 and 10 show some results of interactive segmentation of images.

The results depend on what the user is looking for, since the definition of "object" is at the semantic level. In the first example, the salient region represents the "woman." In the second example, the user looks for the region of the sun.

Fig. 9 Example of an interactive segmentation result. The *left* image is the query image. The *middle* one is the segmented image with default parameters. The *right* image is the result after user-based splitting/merging process. The default set of candidate salient regions is shown to the user, and a global merging command is given. The system then returns the segmented image with a smaller number of regions than before

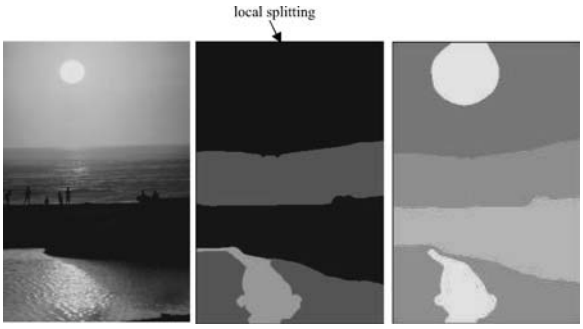


Fig. 10 Example of an interactive segmentation result. In this case, a local splitting command is given to the default set. A local search is given within the selected region to find more regions in it, and thus the parameter θ is locally defined

4.4 Content-based image retrieval with query refinement

In this section, we present two applications of the proposed framework for CBIR using salient points and salient regions. We leave out the application with salient lines as it is in the middle of points and regions.

Some available examples of the use of salient points in CBIR are presented in [32, 34], for lines in [12], and for regions in [5]. For CBIR with salient details an appropriate similarity function has to be defined. For instance, Hausdorff or Chamfer distance would be an appropriate choice to measure the similarity between images based on comparing the locations of salient points. With sets of lines, the similarity can be based on the comparison between two sets of lines on either the curvatures of lines or their shapes, or we can view lines as the border between two regions and compare their features. With regions, features can be shapes, colors, or textures.

In the previous sections, we used the framework to find the optimal parameters for one image. Now let us consider how to apply this for searching. If we perform t steps, we have (from Eq. 2)

$$\{I_Q, F_Q^0, S_Q, Z_Q\} \overset{\mathrm{RF}_i}{\rightarrow} \{I_Q, F_Q^t, S_Q, Z_Q\}.$$

Thus at every step the whole query space is updated as the current parameters are applied to the whole dataset. So after initialization we iteratively search for a good query. The number of iterations during the user interaction depends on the value of ϵ in the constraint function $\mathcal{C}$. If ϵ is set to a small value, the system will take several steps to reach the final result.

4.4.1 Examples

For the experiment, a dataset of 1100 Corel images is used, which consists of different scenes and objects. The initial images can be selected by the user from a set of internal or external examples. For simplicity, we work on a single image at a time. Working with multiple examples can be done by combining user feedback on each image.

The system is based on the query details defined by default parameters and tuned parameters after relevance feedback as described in Sects. 4.3 and 4.1.

If salient points are used, for each image in the database, a set of salient points is extracted. To compute the feature, we use [37] to obtain color moments at each color channel. In our experiment, we use HSI color space. First, the color histogram HSI is computed at each salient point in a 3×3 neighborhood. Those values are summed up and normalized by the number of salient points in the images. Next, three color moments – mean, variance, and skewness – are computed for each color channel. Hence, each image is then represented by a vector of 9 values. $\mathcal{L}_1$ is used as a similarity function between feature vectors.

For query region specification and refinement we follow the method of Sect. 4.3. However, in this case, as features of the region, we take hue and saturation as they are simple and effective for the Corel dataset. With hue and saturation features, the function S_Q is based on histogram intersection [38]. The similarity value is defined by the best matching region of an image to the query one. Finally, returned images are ranked based on similarities.

As described in the previous section, we select $\epsilon = 0.2$. Then for each query, it takes four to five iterations to reach the user's desired result, i.e., $t = 4$ or 5.

An illustration of how the system works is given in Fig. 11. In this particular example, it follows that with default parameters, the query is almost the whole image, and hence contains both red (flowers) and green (leaves). The system then returns images that contain regions with similar color distributions. The user shows that he/she is only interested in the red region; the results after adaptation therefore show all images with red regions also if they are on a

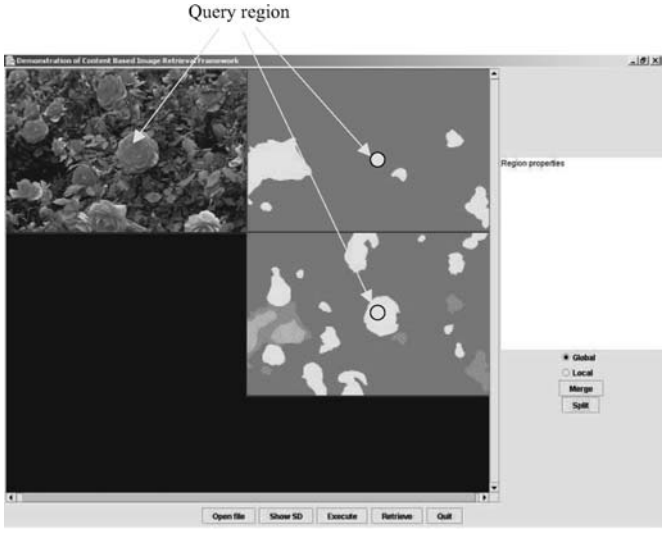
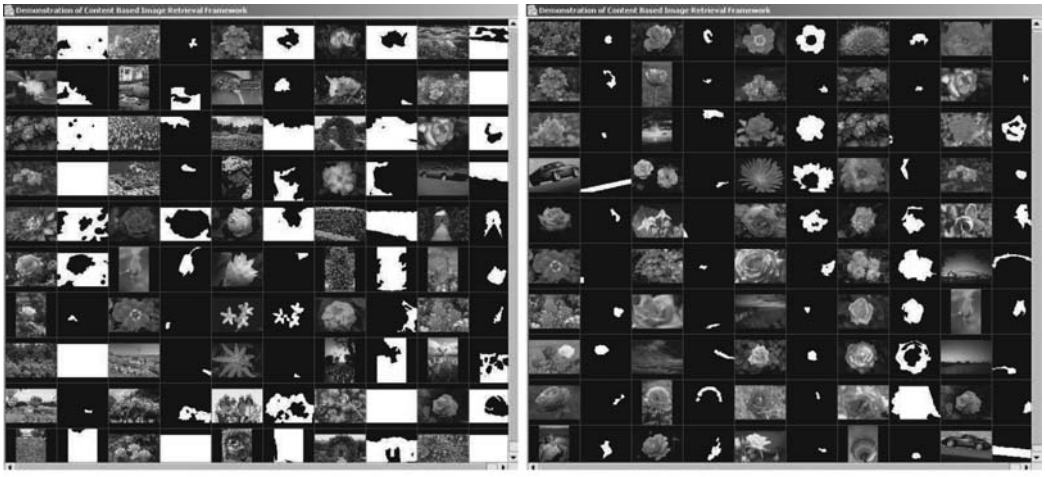
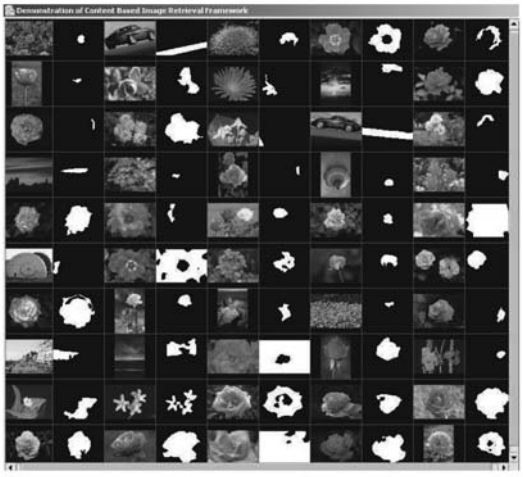


Fig. 11 Example of a query region definition. The *upper left corner* is the query image, and the *upper right corner* is the image with query region using default parameters. The *lower right corner* is the image with redefined query region. The *circle* denotes the point clicked by the user



(a) (b)



(c)

Fig. 12 **a** Result images using region-based search where the query region is extracted using default parameters. To the *right* of each picture the region found is indicated. **b** The same image but now using the updated parameters. **c** Result images using region-based search where the query region is extracted using updated parameters but without applying those updated parameters to the whole dataset

different background. In Fig. 12, we show the results with the corresponding retrieved regions similar to the query region. It is observed that the result with updated parameters (Fig. 12a) is closer to the desired results than the default (Fig. 12b) since retrieved regions are indeed similar red regions. In Fig. 12c, the updated parameters are only applied to the query image but not to the whole dataset. Since the new parameters define a smaller scale, the results show that images returned have red regions extracted at a large scale but miss ones that only exist at a smaller scale. When applying the updated parameters to the whole dataset, the regions at the larger scale are subdivided into smaller regions, hence those contain smaller red regions, which are now retrieved by the system.

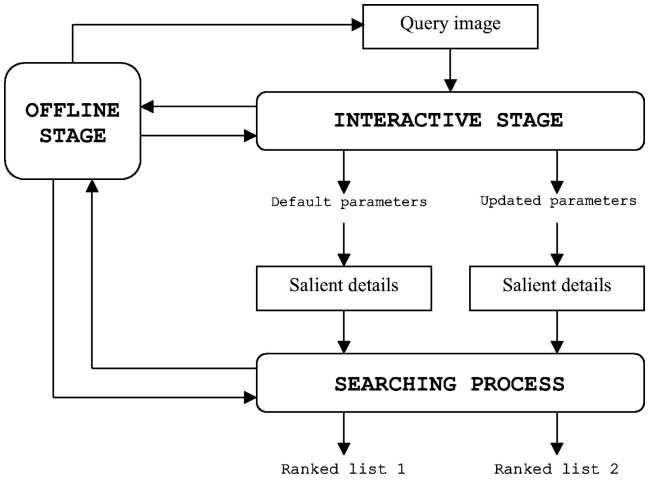
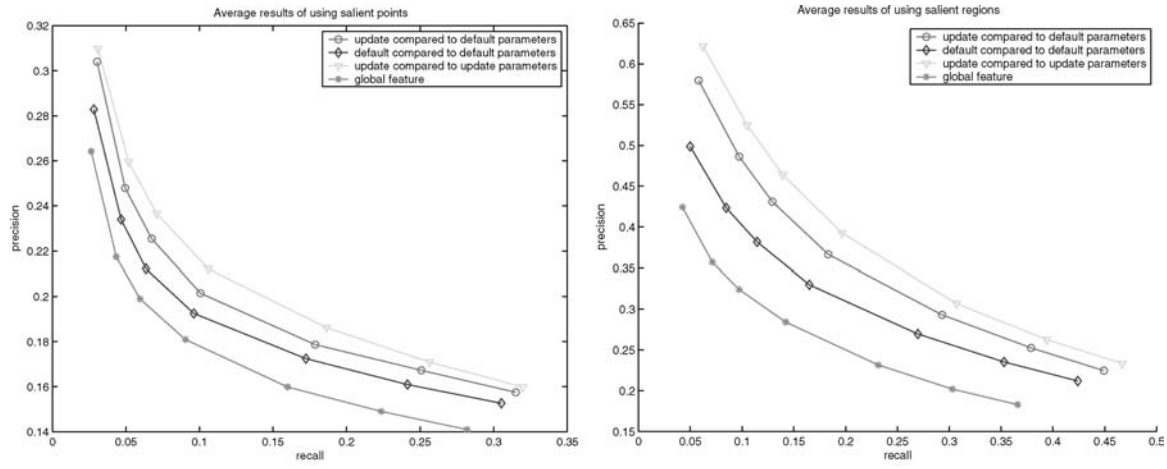


Fig. 13 Experimental setup



(a) Using salient points (b) Using salient regions

Fig. 14 Average recall and precision comparison of 1100 Corel images

4.4.2 Experiments

As stated earlier, finding the user's desired results is a subjective problem, hence it is not easy to evaluate or compare our system to existing ones. However, we believe that using updated parameters, the user will be presented with the choice to select the right details, which are missing using the default parameters, for a query search. To see whether salient details perform better than global features, and to see whether it pays off to optimize the parameters for each specific query, we compute the following comparisons:

(i) Default query detail compared to details with default parameters in offline data.
(ii) Updated query detail compared to details with default parameters in offline data.
(iii) Updated query detail compared to details with updated parameters in offline data.
(iv) Using global features (i.e., features are computed for the whole image).

For that purpose, we built a model for the search task, which will be described as follows.

We used the predefined categories from Corel. The ten categories are cars, surfing, sunset, flowers, roses, seasons, flower beds, balloons, summer, and winter. The size of each category is 100 images, except the category "roses," which contains 200 images. Each image in the dataset is subsequently used as a query. Salient details are extracted using either default or updated parameters. With salient points, the feature vector of the query image is compared to all feature vectors in the database at the same parameter setting. In the case of region-based search, the comparison is more complicated. Each region in the image is selected as the query region. The search process looks into the offline data and compares the query with all extracted regions at the same parameter setting. The similarity is the maximum value of all regions in one image compared to the query region. Hence, for each query region, we get a ranked list of images based on the similarity values (Fig. 13). We then compute the recall and precision. The final rank list of a query image is the one with the highest recall and precision values. The number of retrieved images in the experiments is 10, 20, 30, 50, 100, 150, and 200 images. In Fig. 14, we show the average results over the whole set of 1100 images. Figures 15 and 16 show in detail the average precision and recall of each category using salient regions and points, respectively.

The results show that salient details generally perform better than global features. This especially holds for region-based search when finding red flowers, cars, and sun. Since the searched objects are specific, the user easily figures out the salient regions for the search. In the case of a general search, for example searching for seasons, it is hard to point out which regions are most representative for the term "season." Besides, we show that finding the appropriate query details (ii) leads to an improvement of the search. However, updating the whole dataset with the new updated parameters (iii) will give the best results. The experiments also prove that for each category there is an optimal parameter set for the search different from the default one.

5 Conclusion

In this paper, we have proposed a user-based framework for interacting with salient details. Using this framework we have identified that existing salient detail detections can be classified into the following five steps: image processing, detail detection, feature computation, saliency computation, and selection based on significance. Tunable parameters at each step are then found, and from there we present efficient methods for updating those parameters via user relevance feedback. The instantiations of the framework show that adapting saliency of details can get closer to saliency from a user's perspective. Moreover, based on a set of 1100

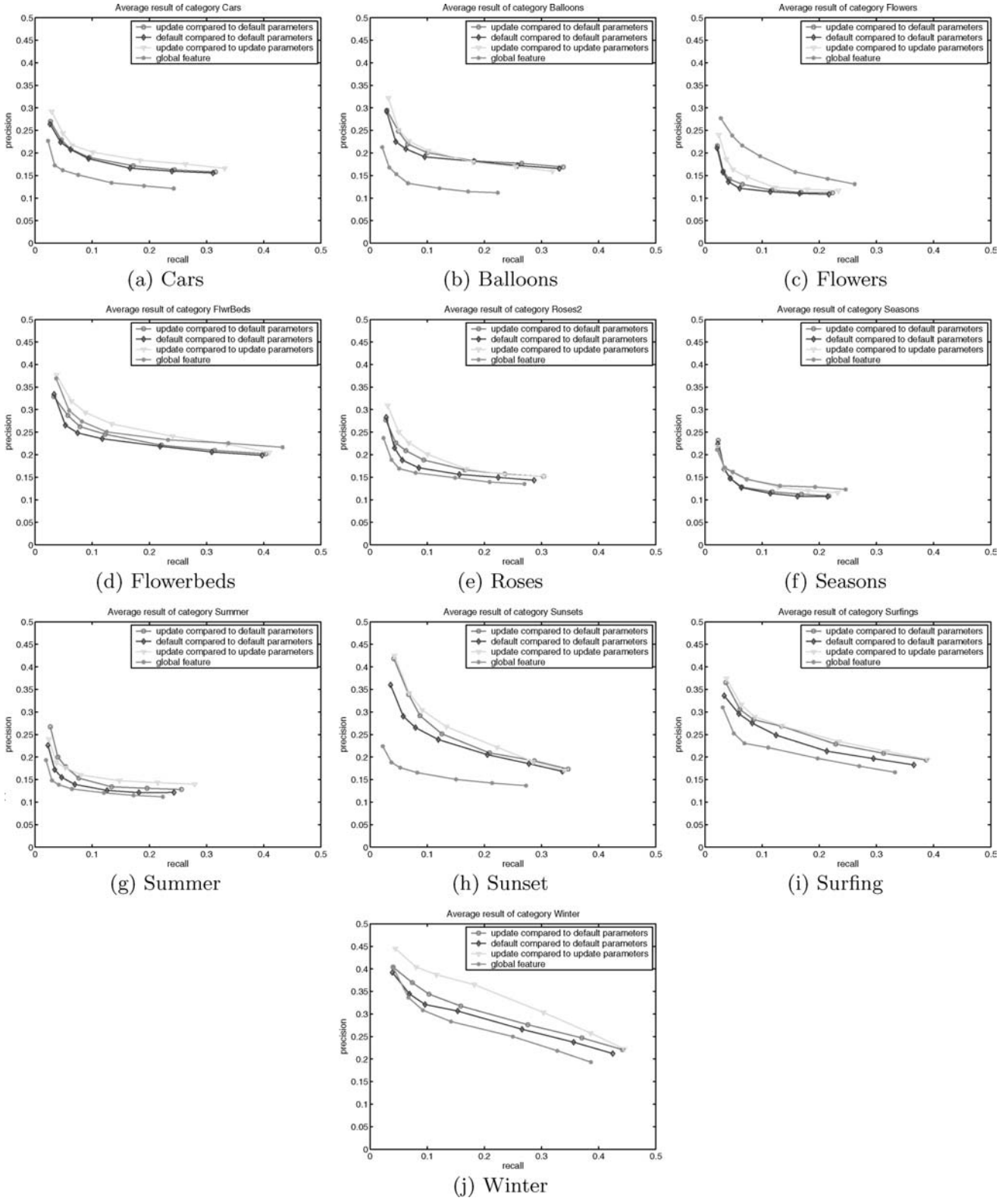


Fig. 15 Recall and precision comparison using salient points

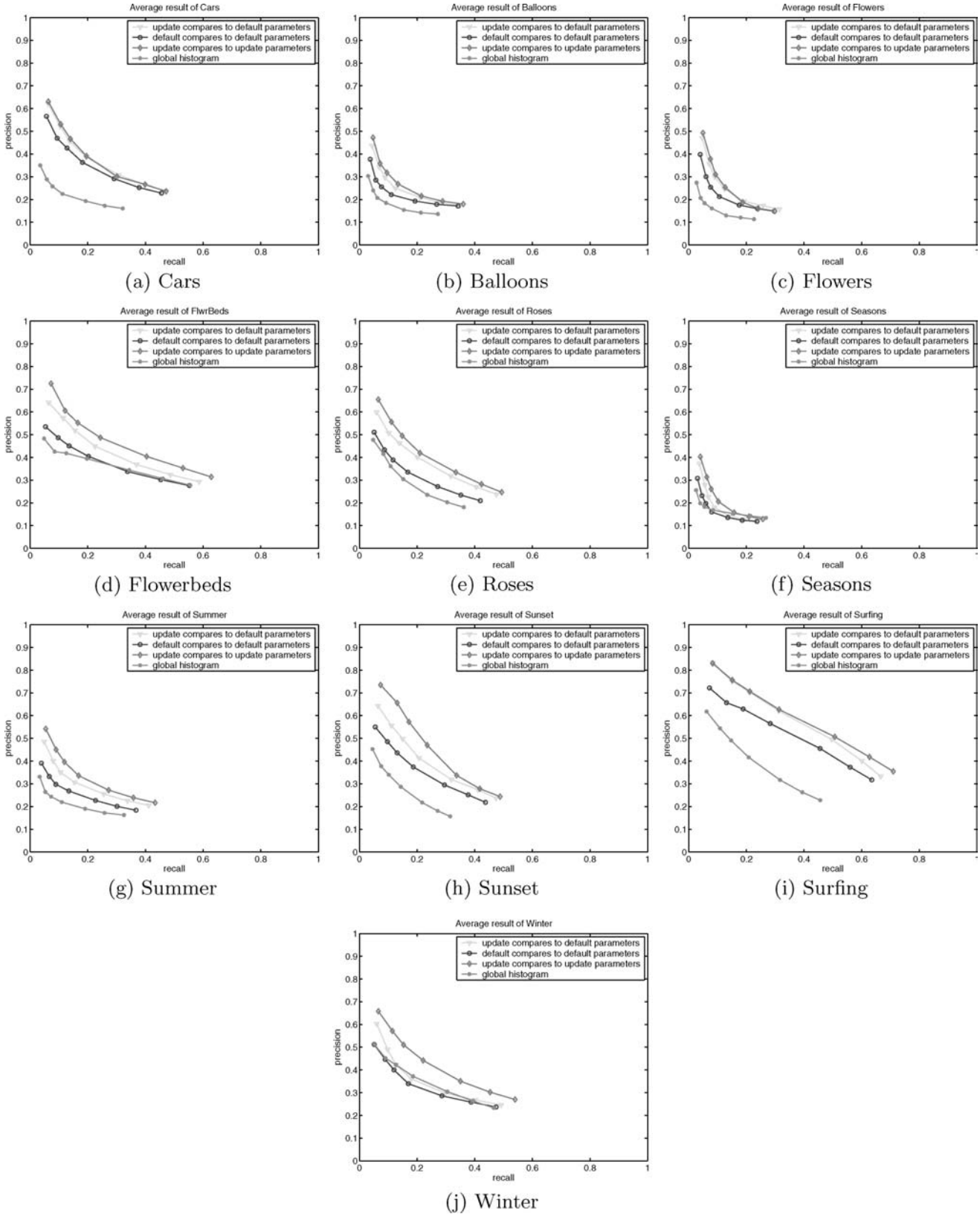


Fig. 16 Recall and precision comparison using salient regions

images from the Corel collection, the potential of applying the framework to the image retrieval system is illustrated. Experimental results first prove our theory that using salient details perform better than global features, especially using salient regions. Secondly, it is demonstrated that the use of parameter optimization to update the query space allows for remarkable improvements in retrieval performance. Extending the proposed framework so that it can deal with larger datasets (e.g., more than 100000 images) is an interesting topic for future work. For such a large dataset, a number of issues such as indexing for interaction and search as well as storing data must be studied thoroughly.

Acknowledgements The authors would like to thank the anonymous reviewers for their valuable suggestions and comments.

References

1. Alter, T., Basri, R.: Extracting salient curves from images: an analysis of saliency network. Int. J. Comput. Vis. **27**(1), 51–69 (1998)
2. Barnard, K., Duygulu, P., Guru, R., Gabbur, P., Forsyth, D.: The effects of segmentation and feature choice in a translation model of object recognition. In: International Conference on Pattern Recognition (2003)
3. Bres, S., Jolion, J.: Detection of interest points for image indexation. In: 3rd International Conference on Visual Information Systems, pp. 427–434 (1999)
4. Canny, J.: A computational approach to edge detection. IEEE Trans. Pattern Anal. Mach. Intell. **8**(6), 679–698 (1986)
5. Carson, C., Belongie, S., Greenspan, H., Malik, J.: Blobworld: image segmentation using expectation-maximization and its application to image querying. IEEE Trans. Pattern Anal. Mach. Intell. **24**(8), 1026–1038 (2002)
6. Cinque, L., Lecca, F., Levialdi, S., Tanimoto, S.: Retrieval of images using rich region descriptions. In: 14th International Conference on Pattern Recognition, vol. 1 (1998)
7. Dimai, A.: Unsupervised extraction of salient region-descriptors for content based image retrieval. In: Proceedings of the 10th International Conference on Image Analysis and Processing (1998)
8. Fleck, M.: Some defects in finite-difference edge finders. IEEE Trans. Pattern Anal. Mach. Intell. **14**(3), 337–345 (1992)
9. Flickner, M., Sawhney, H., Niblack, W., Ashley, J., Huang, Q., Dom, B.: Query by image and video content: the qbic system. IEEE Comput. **28**(9), 23–32 (1995)
10. Gevers, T., Smeulders, A.: Pictoseek: combining color and shape invariant features for image retrieval. IEEE Trans. Image Process. **9**(1), 102–119 (2000)
11. Hoang, M., Geusbroek, J., Smeulders, A.: Color texture measurement and segmentation. In: Signal Processing (2003)
12. Iqbal, Q., Aggarwal, J.: Retrieval by classification of images containing large manmade objects using perceptual grouping. Pattern Recognit. **35**, 1463–1479 (2001)
13. Itti, L., Koch, C., Niebur, E.: A model of saliency-based visual attention for rapid scene analysis. IEEE Trans. Pattern Anal. Mach. Intell. **20**(11), 1254–1259 (1998)
14. Jacobs, C., Finkelstein, A., Salesin, D.: Fast multiresolution image querying. In: Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques, pp. 277–286 (1995)
15. Kam, A.: A general multiscale scheme for unsupervised image segmentation. Phd thesis, University of Cambridge (2000)
16. Kovesi, P.: http://www.csse.uwa.edu.au/∼pk/research/matlabfns/
17. MacArthur, S., Brodley, C., Kak, A., Broderick, L.: Interactive content-based image retrieval using relevance feedback. Comput. Vis. Image Understand. **88**(2), 55–75 (2002)
18. Mahamud, S., Williams, L., Thornber, K., Xu, K.: Segmentation of multiple salient closed contours from real images. In: IEEE Trans. Pattern Anal. Mach. Intell. **25**(4), 891–897 (1999)
19. Marques, O., Costa, F.M., Furht, B.: Content-based image search and retrieval using relevance feedback: the muse project. In: Proceedings of the International Association of Science and Technology for Development (2000)
20. Martinez, A., Serra, J.: A new approach to object-related image retrieval. J. Vis. Lang. Comput. **11**, 345–363 (2000)
21. Mikolajczyk, K., Schmid, C.: Indexing based on scale invariant interest points. In: International Conference on Computer Vision, pp. 525–531 (2001)
22. Mikolajczyk, K., Schmid, C.: An affine invariant interest point detector. In: European Conference on Computer Vision, pp. 128–142 (2002)
23. Minka, T., Picard, R.: Interactive learning using a 'society of models'. In: International Conference on Computer Vision and Pattern Recognition (1996)
24. Muller, H., Muller, W., Maillet, S., Pun, T., Squire, D.: Strategies for positive and negative relevance feedback in image retrieval. In: Proceedings of the 15th International Conference on Pattern Recognition (2000)
25. Pauwels, E., Frederix, G.: Fiding salient regions in images—nonparametric clustering for image segmentation and grouping. Comput. Vis. Image Understand. **75**(1/2), 73–85 (1999)
26. Rosin, P.: Edges: saliency measures and automatic thresholding. Mach. Vis. Appl. **9**, 139–159 (1997)
27. Rui, Y., Huang, T.: Optimizing learning in image retrieval. In: International Conference on Computer Vision and Pattern Recognition (2000)
28. Rui, Y., Huang, T., Chang, S.: Image retrieval: current techniques, promising directions and open issues. J. Vis. Commun. Image Represent. **10**, 39–62 (1999)
29. Rui, Y., Huang, T., Mehrotra, S.: Relevance feedback: a power tool for interactive content-based image retrieval. IEEE Trans. Circuits Video Technol. **8**(5), 644–655 (1998)
30. Salah, A., Alpaydin, E., Akarun, L.: A selective attention-based method for visual pattern recognition with application to handwritten digit recognition and face recognition. IEEE Trans. Pattern Anal. Mach. Intell. **24**(3), 420–425 (2002)
31. Santini, S., Gupta, A., Jain, R.: Emergent semantics through interaction in image databases. In: IEEE Transactions on Knowledge and Data Engineering (2001)
32. Schmid, C., Mohr, R.: Local grayvalue invariants for image retrieval. IEEE Trans. Pattern Anal. Mach. Intell. **15**(5), 530–535 (1997)
33. Schreiber, A., Dubbeldam, B., Wielemaker, J., Wielinga, B.: Ontology-based photo annotation. In: IEEE Intelligent Systems (2001)
34. Sebe, N., Tian, Q., Loupias, E., Lew, M., Huang, T.: Salient points for content based image retrieval. In: International Conference on Computer Vision and Pattern Recognition (2001)
35. Smeulders, A., Worring, M., Santini, S., Gupta, A., Jain, R.: Content-based image retrieval at the end of early years. IEEE Trans. Pattern Anal. Mach. Intell. **22**(12), 1349–1380 (2000)
36. Stollnitz, E., DeRose, T., Salesin, D.: Wavelets for computer graphics: a primer, part 2. IEEE Comput. Graph. Appl. **15**(4), 75–85 (1995)
37. Stricker, M., Orengo, M.: Similarity of color images. Proc. SPIE Stor. Retrieval Image Video Databases **2420**, 381–392 (1995)
38. Swain, M., Ballard, D.: Color indexing. Int. J. Comput. Vis. **7**(1), 11–32 (1991)
39. Vendrig, J., Worring, M., Smeulders, A.: Filter image browsing: interactive image retrieval by using database overviews. Multimedia Tools Appl. **15**(1), 83–103 (2001)
40. Walker, K., Cootes, T., Taylor, C.: Locating salient object features. In: 9th British Machine Vision Conference **2**, 557–566 (1998)
41. Walker, K., Cootes, T.F., Taylor, C.J.: Locating salient facial features using image invariants. In: 3rd International Conference on Automatic Face and Gesture Recognition, pp. 242–247 (1998)