Optimization of Interactive Visual-Similarity-Based Search

GIANG PHUONG NGUYEN and MARCEL WORRING
University of Amsterdam, The Netherlands

At one end of the spectrum, research in interactive content-based retrieval concentrates on machine learning methods for effective use of relevance feedback. On the other end, the information visualization community focuses on effective methods for conveying information to the user. What is lacking is research considering the information visualization and interactive retrieval as truly integrated parts of one content-based search system. In such an integrated system, there are many degrees of freedom like the similarity function, the number of images to display, the image size, different visualization modes, and possible feedback modes. To base the optimal values for all of those on user studies is unfeasible. We therefore develop search scenarios in which tasks and user actions are simulated. From there, the proposed scheme is optimized based on objective constraints and evaluation criteria. In such a manner, the degrees of freedom are reduced and the remaining degrees can be evaluated in user studies. In this article, we present a system that integrates advanced similarity based visualization with active learning. We have performed extensive experimentation on interactive category search with different image collections. The results using the proposed simulation scheme show that indeed the use of advanced visualization and active learning pays off in all of these datasets.

Categories and Subject Descriptors: H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms: Experimentation, Human Factors, Algorithms

Additional Key Words and Phrases: Interactive search, similarity based visualization, active learning

ACM Reference Format:

Nguyen, G. P. and Worring, M. 2008. Optimization of interactive visual-similarity-based search. ACM Trans. Multimedia Comput. Commun. Appl. 4, 1, Article 7 (January 2008), 23 pages. DOI = 10.1145/1324287.1324294 http://doi.acm.org/10.1145.1324287.1324294

1. INTRODUCTION

Research on interactive search mechanisms is currently split into two separate fields. In content-based retrieval, concentration is on machine learning methods for effective use of relevance feedback [Rui et al. 1998; Smeulders et al. 2000]. The information visualization community focuses on effective methods for conveying information to the user [Keim 2002; Shneiderman 1996; Bederson 2001]. What lacks is research considering the information visualization and interactive content based retrieval as truly integrated parts of one search system.

Features form the basis for content-based retrieval (CBR). In literature, a large number of features have been introduced to improve search performance. Some features are only proposed for specific tasks or specific domains, for example, medical images, or satellite images. These methods have limited

Authors' address: Intelligent Systems Lab Amsterdam, University of Amsterdam, Kruislaan 403, 1098 SJ Amsterdam, The Netherlands, email: {giangnp, M.Worring}@uva.nl.

use outside the application domain. General features, such as global color histograms, quite often fail because of the *semantic gap* between user expectation and system ability. Employing accompanying text, the semantic gap can be reduced either by using generic features such as keywords in the text [Naphade et al. 2002] or multi-modal concepts [Snoek et al. 2005]. Getting the required indices is labor expensive and time consuming. For instance, when text is not available; hence, processing a textual query requires annotating the whole collection, which is impractical for large collections.

When no suitable concept is present, many image retrieval systems use query by example, where examples are taken from inside or outside the collection. In reality, there are many search tasks where the user does not have any example to start with. Then, the user needs to explore the collection to find relevant examples. In both query by example and exploration, a good similarity function is essential.

Finding a suitable similarity function is strongly dependent on the chosen features. Systems in literature usually select the similarity function, commonly associated with the chosen features. For example, histogram intersection [Swain and Ballard 1991] is typically selected for comparing two color histograms. Features and similarity can be computed offline.

The interactive stage contains two main steps, namely visualization and relevance feedback, which are iterated [Smeulders et al. 2000]. The visualization step displays a selected set of images to the user. Based on that, the user judges how relevant those images are with respect to what she is looking for. The system learns from the user's feedback and repeats the above step.

Using learning is well-known in interactive CBR. Comprehensive overviews of techniques are presented in Rui et al. [1998], Smeulders et al. [2000], and Zhou and Huang [2003]. Recently, the use of support vector machines in learning has gained interest. It has proved to give the highest boost to the performance [Chen et al. 2001, 2005; Gosselin and Cord 2004; Guo et al. 2001; Manevitz and Yousef 2004; Tong and Chang 2001].

For effective interaction, a good learning method is not sufficient, an interface for communication between the user and the system is needed. Such an interface should not be judged on its aesthetic value alone, but more importantly on its effectiveness in supporting the user's search process. It should therefore combine CBR and advanced visualization. We call such an integrated system a visual search system.

To realize visual search systems, recent systems apply similarity based visualization techniques giving a more informative and effective interface [Moghaddam et al. 2004; Nguyen and Worring 2005; Heesch and Ruger 2004; Rubner 1999; Santini et al. 2001]. In Nguyen and Worring [2005], we present comparisons of existing techniques in visualizing image collections and conclude that for an optimal-similarity-based visualization system, three requirements have to be obeyed: *overview, structure preservation*, and *visibility*.

(i) *Overview Requirement*. Ensure that the set displayed represents the whole collection since not all images from the collection can be shown on the screen at once.

(ii) *Structure Preservation Requirement*. Preserve the relations between images in the original feature space on the screen.

(iii) *Visibility Requirement*. Keep the content of displayed images visible to make interaction feasible.

The similarity based visualization we proposed in Nguyen and Worring [2005] satisfies these requirements. It forms the basis for our interactive system, which in this article is extended with an active learning component.[1]

[1] This system is part of the MediaMill system, which received the Best Techinical Demo Award at ACM Multimedia [Snoek et al. 2005].

The ultimate goal of a search system is to have real users working with it. Thus, when optimizing retrieval, one should not only evaluate the individual steps, but the integral process. However, an experiment with real users introduces a large number of variables influencing the result such as age, sex, level of expertise, and the type of questioning. Therefore, obtaining repeatable results from experiments such as the one in Rodden et al. [2001] is difficult and time consuming. When the scenarios and criteria for success have been made explicit, the user and its actions can be simulated. It effectively allows for decomposition of the evaluation of the visualization methodology and the design of the interface and system performance [Ivory and Hearst 2001]. Therefore, we develop an evaluation method integrating both user interaction as well as system aspects in a simulated scenario where many of the free parameters can be optimized before the user study.

In this article, we aim for a similarity based search system combining visualization and interaction. The system builds upon the visualization and active learning components described above. The key issue here is that we integrate these elements into a unique framework. Within the system, there are many free parameters like the similarity function and the method for processing relevance feedback. To optimize the performance of the system, we consider an interactive category search scenario in various large image collections. The search tasks range from finding images sharing simple properties such as images of the same object, to complex properties, for instance, images of a person entering a vehicle.

This article is organized as follows: To optimize the search task, each step is analyzed to find the best solution in Section 2. In Section 3, we present experiments of the proposed system for three different classes of data: ALOI (Amsterdam Library of Object Images), the Corel dataset, and the TRECVID dataset.

2. METHODS

In this section, we first briefly describe the system of Nguyen and Worring [2005], and from there its proposed active learning extension. Finally, we describe the built-in optimization method.

2.1 Similarity-Based Visualization

The system in Nguyen and Worring [2005] is based on the requirements (i), (ii), (iii) from the introduction. Cost functions are introduced for each of them.

Let us consider an image collection $\mathcal{I} = \{I_1, I_2, \ldots, I_N\}$ described using features. For similarity based visualization, a projection has to be made from the high dimensional feature space to the *visualization space*. The ISOSNE in Nguyen and Worring [2005] turns out the best one among others in obeying the *structure preservation requirement*. In this projection, nearest neighbors of each image in the feature space are computed to create a nearest neighbor graph. The graph-based distance is then used to redefine the distances between non-neighboring images. Finally, the nonlinear mapping SNE obtains the positions of images in visualization space. To evaluate this requirement, a structure preservation cost function $\mathcal{C}_S$ is defined. Assume $\mathcal{P} = \{P_{ij}\}$ and $\mathcal{Q} = \{Q_{ij}\}$ are the discrete probability density functions on relations between N images in the high dimensional space and the visualization space, respectively, then we have:

$$\mathcal{C}_S = \sum_{i}^{N} \sum_{j}^{N} P_{ij} \log \frac{P_{ij}}{Q_{ij}}. \tag{1}$$

Thus, when this function is minimized, the relations between images are optimally preserved in the visualization space. As a consequence, similar images tend to be grouped together. They are likely to overlap on the screen. It influences the interaction process, because the user can miss the relevant ones as they are hidden. Therefore, the need for the *visibility requirement*. The cost function of this requirement $\mathcal{C}_V$ is based on the inscribed circles of the two images, for simplicity, assuming all images are square and have equal size [Nguyen and Worring 2005]. Let R denote the radius of the inscribed circle of an image, and $\mathcal{O}_{ij}$ the overlap area of the inscribed circles of image I_i and I_j. The visibility area of these two images will be $1 - \frac{\mathcal{O}_{ij}}{\pi R^2}$. Then the cost function for the visibility requirement is given by:

$$\mathcal{C}_V = \frac{1}{N(N-1)} \sum_{i=1}^{N} \sum_{j=1, j \neq i}^{N} \left(1 - \frac{\mathcal{O}_{ij}}{\pi R^2}\right). \tag{2}$$

In reality, images can come in different sizes, in this case, the value R will be varied for different images.

To satisfy the *overview requirement*, $\mathcal{I}$ is divided into a number of clusters in visualization space. A set containing selected images from each cluster, called the representative set, is the basis for presentation to the user. To measure how well this set represents the whole collection, the modified Hubert statistic [Dubes 1987] is used as the overview cost function $\mathcal{C}_O$. Assume we have n clusters. Let $d(I_i, I_j)$ denote the distance between image I_i and I_j and $d_c(I_i, I_j)$ the distance between the two cluster centers containing the two images. We have:

$$\begin{aligned} r &= (1/M) \sum\sum d(I_i, I_j) d_c(I_i, I_j), & M &= n(n-1)/2, \\ M_c &= (1/M) \sum\sum d_c(I_i, I_j), & M_p &= (1/M) \sum\sum d(I_i, I_j), \\ \sigma_c^2 &= (1/M) \sum\sum d_c^2(I_i, I_j) - M_c^2, & \sigma_p^2 &= (1/M) \sum\sum d^2(I_i, I_j) - M_p^2. \end{aligned}$$

The overview cost function $\mathcal{C}_O$ is defined as:

$$\mathcal{C}_O = \frac{r - M_p M_c}{\sigma_p \sigma_c} \tag{3}$$

The above requirements are conflicting. For example, to satisfy the overview requirement, the number of representative images should be large. Because of the fixed size of the visualization space, the more images the higher chance of overlapping images, hence, the visibility requirement will be violated. On the other hand, while preserving visibility, images are spread out, original relations between them are changed that is, structure is not preserved. Therefore, two balancing functions between the above cost functions, namely $\mathcal{C}_1$ and $\mathcal{C}_2$, are needed. The first one is the relation between overview and structure preservation. The second one is between structure preservation and visibility. The two parameters influencing the cost functions are the size n of the representative set and the positions $\mathbf{y}$ of the images in visualization space. From Nguyen and Worring [2005], we have:

$$\mathcal{C}_1(n) = \lambda_1 \mathcal{C}_O(n) + (1 - \lambda_1)\mathcal{C}_V(n) \tag{4}$$

$$\mathcal{C}_2(\mathbf{y}) = \lambda_2 \mathcal{C}_S(\mathbf{y}) + (1 - \lambda_2)(1 - \mathcal{C}_V(\mathbf{y})) \tag{5}$$

where $0 \leq \lambda_1, \lambda_2 \leq 1$, n is the size of the representative set and $\mathbf{y}$ denotes the set of positions of images in the visualization space.

In brief, after the collection is projected to the visualization space, images are clustered into n groups. The k-means algorithm is selected to do the clustering because of its simplicity and good performance. Each time a set of n images representing the clusters is displayed. The value of n is the result of optimizing the balancing function $\mathcal{C}_1$ using exhaustive search:

$$n_{\text{opt}} = \underset{n \in [2..n_{\max}]}{\operatorname{argmax}} \ \mathcal{C}_1(n). \tag{6}$$

Because of the limitation of the display, there is a restricted number of images which can be shown to the user simultaneously. The maximum number of displayed images depends on the size of the image, the size of the display, and the area of the images visible to the user. Let us assume all images have equal sizes denoted w and h for width and height, respectively. Let the size of the display be W and H. Finally, assume that for interaction, displayed images should be visible for at least $v\%$. We have:

$$n_{\max} = \frac{H \times W}{h \times w \times \frac{v}{100}}.$$

In our system, v is set to 75% for which we consider it feasible for a user to view the image content. With $W = 900$, $H = 1024$, $w = h = 64$, from the above equation, $n_{\max}$ is set to 300.

Positions $\mathbf{y}$ are found as

$$\mathbf{y}_{\text{opt}} = \min_{\mathbf{y}} \mathcal{C}_2(\mathbf{y}) \tag{7}$$

by gradient descent.

In Figure 1, examples of displaying different sets of images are shown. It can be seen that similar images are indeed placed close to each other in the visualization space and almost every image is visible.

2.2 Features and Similarity Functions

In Nguyen and Worring [2005], one feature and one similarity function is used. As these two parameters also affect the overall performance of the system, we extend the consideration with different features and similarity functions.

2.2.1 *Features.* As mentioned in Section 1, features are generally task-dependent. Using shape features to detect circles with constraints in their spatial relations to find cars may give satisfying results, but will completely fail when applied in searching for flowerbeds. Here we aim at a generic method, therefore, we select from literature two generic features, one for color and one for texture.

We denote a feature set as $\mathcal{F} = \{F^1, F^2, \ldots, F^l\}$, where l is the number of features. For each image $I_i \in \mathcal{I}$, $\vec{F}_i = (F_i^1, F_i^2, \ldots, F_i^l)$ is the corresponding feature vector. The global color histogram is the most commonly used feature because of its simple computation, and its invariance to rotation and small changes in the images. The L*a*b color space, compared to color spaces such as RGB and HSI, has the advantage that it is designed to be perceptually uniform. A distance in color space leads to equal human color difference perception. In our experiment, the L*a*b histogram is computed using 32 bins for each color channel, so in total, each image is represented by a 96 dimensional feature vector, that is, $l = 96$.

The texture feature is the Wiccest (Weibull Invariant Color Contrast ESTimator). This is a new class of color invariant features introduced by Geusebroek in [Geusebroek and Smeulders 2005]. Wiccest features combine color invariance with natural image statistics. Color invariance aims to remove accidental lighting conditions such that the result is constant under varying illumination color, shadow effects and shading, while natural image statistics efficiently represent image data. We implement the feature extraction following Snoek et al. [2005]. An image is first divided into 3×3 regions. In each region the Wiccest features with 2 Wiccest parameters (β, γ) in 6 color channels are computed. This means that each region is represented by a vector of 12 values. Therefore, an image is represented by a vector of $l = 108$ dimensions.

2.2.2 *Similarity or Distance Functions.* Similarity is interpreted as a distance function. The more similar two images, the smaller the distance between them. In literature, a number of distance functions have been proposed [Basseville 1989; Rubner et al. 2001]. The simplest ones are in the family of

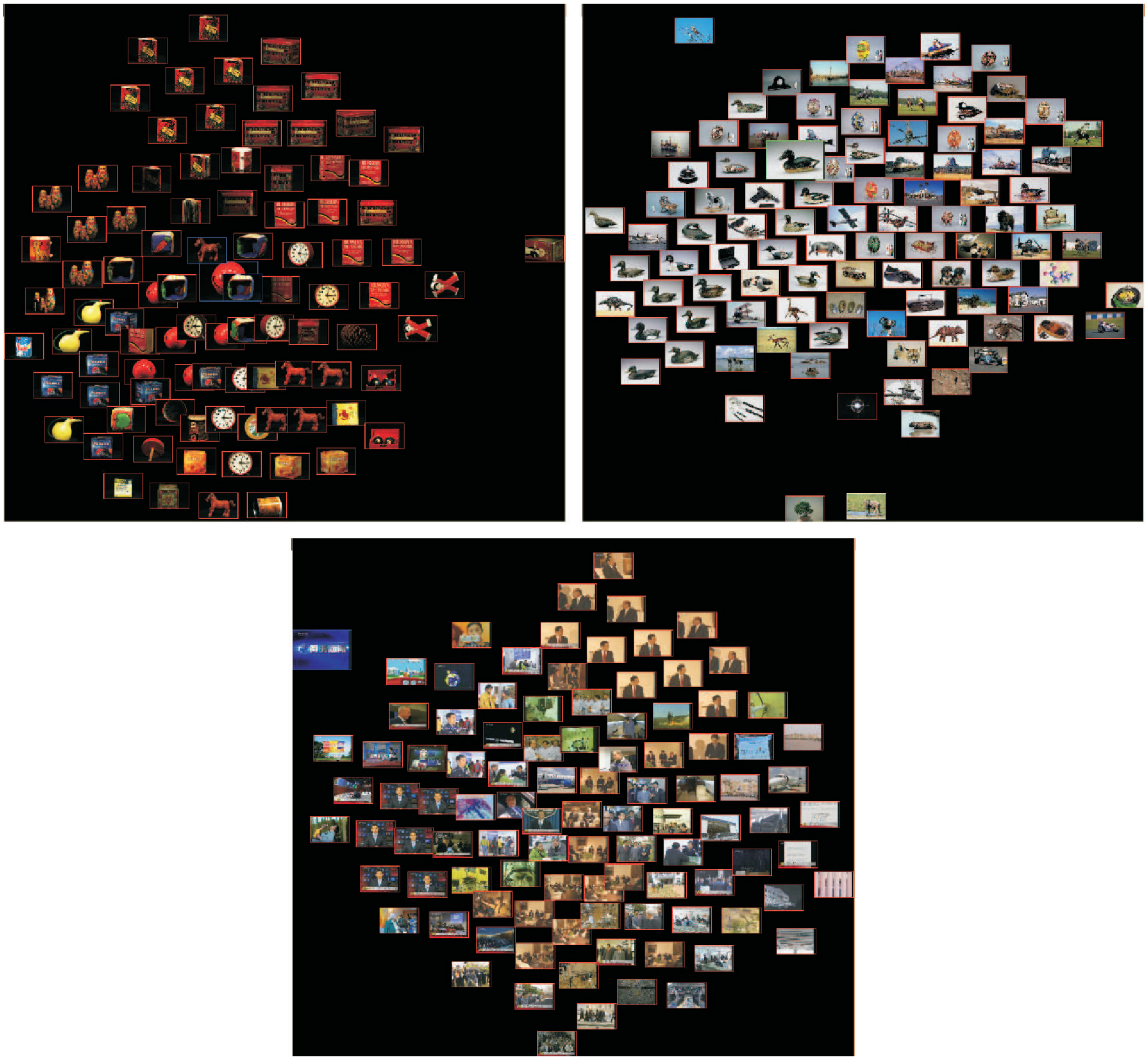

Fig. 1. Examples of visualizing images with similar ones are close together.

Mikowski based functions:

$$d(I_i, I_j) = \left(\sum_{t=1}^{l} (F_i^t - F_j^t)^m \right)^{\frac{1}{m}} \tag{8}$$

with $m = 1$ we have the city-block distance L_1, $m = 2$ is the Euclidean distance L_2, and a special case is the maximum value distance (Chebyshev) L_∞.

Some distance functions measure the difference of two probability distributions. Examples are Kullback Leibler distance and its symmetric version Jeffrey divergence.

Kullback divergence: $d(I_i, I_j) = \sum_{t=1}^{l} \left(F_i^t - F_j^t\right) \left(\log\left(\frac{F_i^t}{F_j^t}\right)\right)$

Jeffrey divergence: $d(I_i, I_j) = \sum_{t=1}^{l} F_i^t \log\left(\frac{F_i^t}{\frac{F_i^t + F_j^t}{2}}\right) + F_j^t \log\left(\frac{F_i^t}{\frac{F_i^t + F_j^t}{2}}\right)$

Some other distance functions are also used often, for instance the Bhattacharyya, and Matusita distance:

Bhattacharyya distance: $d(I_i, I_j) = -\log\left(\sum_{t=1}^{l} \sqrt{F_i^t F_j^t}\right)$

Matusita distance: $d(I_i, I_j) = \sqrt{\sum_{t=1}^{l} \left(\sqrt{F_i^t} - \sqrt{F_j^t}\right)^2}$

Though it is not an explicit rule that one should use a specific distance function for a specific feature, it is a tacit consent. For example, the common way of comparing images represented by L*a*b color histograms is Euclidean distance:

$$d_{\mathrm{L^*a^*b}}(I_i, I_j) = \sqrt{\sum_{t=1}^{l} \left(F_i^t - F_j^t\right)^2}. \tag{9}$$

The Wiccest features are Weibull-based. A distance measure for two Weibull distributions with parameters (β_1, γ_1) and (β_2, γ_2), is defined in [Geusebroek and Smeulders 2005] as

$$d_{\mathrm{Wiccest}} = 1 - \frac{\min(\beta_1, \beta_2)}{\max(\beta_1, \beta_2)} \frac{\min(\gamma_1, \gamma_2)}{\max(\gamma_1, \gamma_2)}. \tag{10}$$

As the Wiccest features are computed for all the regions in the partitioning of the image, we have:

$$d_{\mathrm{Wiccest}}(I_i, I_j) = 1 - \frac{1}{N_r} \sum_{t=1}^{N_r} \frac{\min\left(\beta_i^t, \beta_j^t\right)}{\max\left(\beta_i^t, \beta_j^t\right)} \frac{\min\left(\gamma_i^t, \gamma_j^t\right)}{\max\left(\gamma_i^t, \gamma_j^t\right)}, \tag{11}$$

where N_r is the number of regions in each image. In our case $3x3$ regions are used hence $N_r = 9$.

2.3 Relevance Feedback and Active Learning Algorithm

We now extend the visualization method described with an active learning component. A common way of giving feedback is to have the user label a set of images as positive (relevant images) and/or negative (non-relevant images). For an overview, see Rui et al. [1998] and Zhou and Huang [2003]. To provide a set of images, the system actively selects images which are most informative. Labelled images are used as the training set [Gosselin and Cord 2004; Nguyen and Smeulders 2004; Zhang et al. 2001]. After learning, a new set of images is then selected, and the process is iterated. This process is known as active learning.

In literature, the active learning methods mostly use SVM as a feedback learning base [Chen et al. 2001; Manevitz and Yousef 2004; Guo et al. 2001]. SVM was first introduced in data mining research as a classification method. Given a training set $\mathcal{I}_T$ of $r \ll N$ images, each image $I_i \in \mathcal{I}_T$ is represented as a point $\mathbf{x}_i$ with label $l_i \in \{-1, 1\}$. The aim is to produce a model that predicts the class for unlabelled

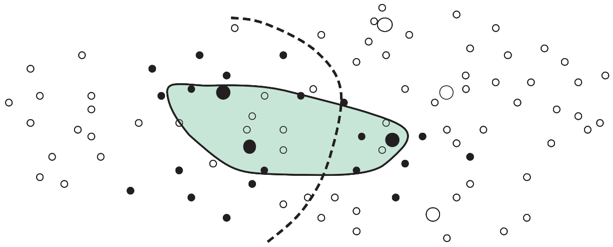

Fig. 2. An example of a decision boundary. The small circles are unlabelled data, the bigger ones are the labelled ones used as training data. The empty circles are irrelevant images, and the filled circles are relevant ones. In case of one-class SVM, only the labelled images are used as positive examples, SVM finds the optimal border to cover those training examples (solid boundary). The dashed line is the result in case of two-class SVM.

images by creating a hyperplane that separates the collection into a positive and a negative class based on the user feedback. This requires to optimize the following function:

$$\begin{array}{ll} \min_{\omega,b,\xi} & \frac{1}{2}\omega^T\omega + C\sum_{i=1}^{r}\xi_t \\ \text{subject to} & l_i(\omega^T\phi(\mathbf{x}_i)+b) \geq 1-\xi_i, \\ & \xi_i \geq 0. \end{array} \tag{12}$$

with C a penalty parameter for the error term. Training image $\mathbf{x}_i$ is mapped to a higher dimensional space by the function ϕ. The function $K(\mathbf{x}_i, \mathbf{x}_j) \equiv \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$ is called the kernel function. The SVM originally used linear classifiers. Several non-linear kernel functions have been introduced later. In Chang and Lin [2001], the authors point out that the radial basis function is a good choice, which we will use in this article, namely

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma\|\mathbf{x}_i - \mathbf{x}_j\|^2),$$
$$\gamma > 0 \text{ is a kernel parameter.}$$

The performance of SVM depends on the selection of its parameters. However, there is no optimal set of parameters that works well in all cases. The choice of default parameters are a good option for building a general search system using SVM [Chang and Lin 2001]. Therefore, we also use the default parameters as suggested in Chang and Lin [2001].

With different ways of giving relevance feedback, $\mathcal{I}_T$ can contain either both positive and negative examples or only one of those. Hence, two main approaches for learning are available. The first approach is based on both positive and negative examples, which is called two-class SVM. At each iteration, both sets are used to find the best classifier separating positive examples from negative ones [Zhang et al. 2001]. The second one is based on only positive examples known as one-class SVM. The main argument for using this approach over the other one is that in a large collection, the number of relevant images is normally much smaller than the total size of the collection. Moreover, the distribution of non-relevant ones is unpredictable and therefore it is difficult to find the borders for those. Hence, in this approach when interacting with the images displayed, the user labels positive examples only. By assuming similar images are clustered in feature space, the algorithm learns the border of the area covering as much as possible of those examples, that is, the set of images one is searching for Chen et al. [2001] and Manevitz and Yousef [2004].

After training is finished, there are two common ways of selecting a new set of images to display for another round of feedback. The first approach is selecting images in the positive class with maximum distance to the border, which have highest chance of being relevant to the search task. Alternatively, in the second approach, images closest to the border are returned, and this approach is said to provide the most informative set to the user [Chen et al. 2001; Manevitz and Yousef 2004].

Those images returned are then also labelled as positive or negative by the user. The system repeats the process to obtain new feedback information. The iteration is stopped when the performance satisfies given constraints such as the number of iterations, time limitation, or simply that the user does not want to give any more feedback.

Finally, when the interaction process is finished, images inside the decision boundary are ranked by their distances to the border. Images having maximum distance are supposedly most relevant to the search task.

2.4 Scheme

Up to this point, we have analyzed the two main techniques used in the system namely similarity based visualization (SimVis) and active learning ($\text{SimVis}_{\text{Active}}$). We will now define the proposed scheme combining these techniques. The user starts the search without any examples at hand, and the general scenario contains the query initialization step and the interactive step.

The SimVis system requires the preparation of the projections, and representative set. We can add these steps in the interactive stage, but usually this is computationally impractical. Therefore, they are computed beforehand. First, features of images in the collection are selected and extracted. Distances between images are then obtained with a selected similarity function. After that, ISOSNE is applied to project images to the visualization space. Next, we employ k-means to cluster images into n clusters. A set of images selected from different clusters form the representative set of the collection. Information of each image belonging to a certain group, and its position in the visualization space are stored as offline data.

Figure 3 shows the interactive stage. Each time a set of n images is displayed. In the first screen, the representative set is shown to the user. He then uses the system to explore the collection and find relevant images. Particularly, if the currently displayed set contains any positive examples, the user selects that image and asks for a set of nearest neighbors expecting to find more images that are similar. The number of nearest neighbors displayed is set to k. Those neighbors are based on the selected similarity function.

The system satisfying the visibility requirement, assures feasible interaction with all images displayed. Moreover, employing the advantage of similarity based visualization, instead of clicking on an individual image for labelling, the system allows the user to select several images by dragging a rectangle around images in the same category. As a consequence, our system can reduce the number of actions needed from the user. In case there is no positive image in the current set, the user asks the system to display another representative set. This set contains images, which have not been displayed before and are closest to the previous representative set.

In the feedback step, training examples are selected by the user. When a certain number of examples are provided, the SVM trains the support vectors. We use the well-known SVM library developed by Chang and Lin [2001], which gives one-class as well as two-class SVM implementations. As indicated, the parameters of the SVM implementations are set to their default values.

2.5 Scenario Optimization

First of all, we restate the search scenario that we will work on:

CATEGORY SEARCH: *Finding as many images as possible belonging to a certain category.*

In the general scheme above, there are several degrees of freedom in each step either from the user or the system point of view. This leads to the difficulty of finding the factors that are user specific and those that are objectively improving the system. In the introduction, we have pointed out that the use

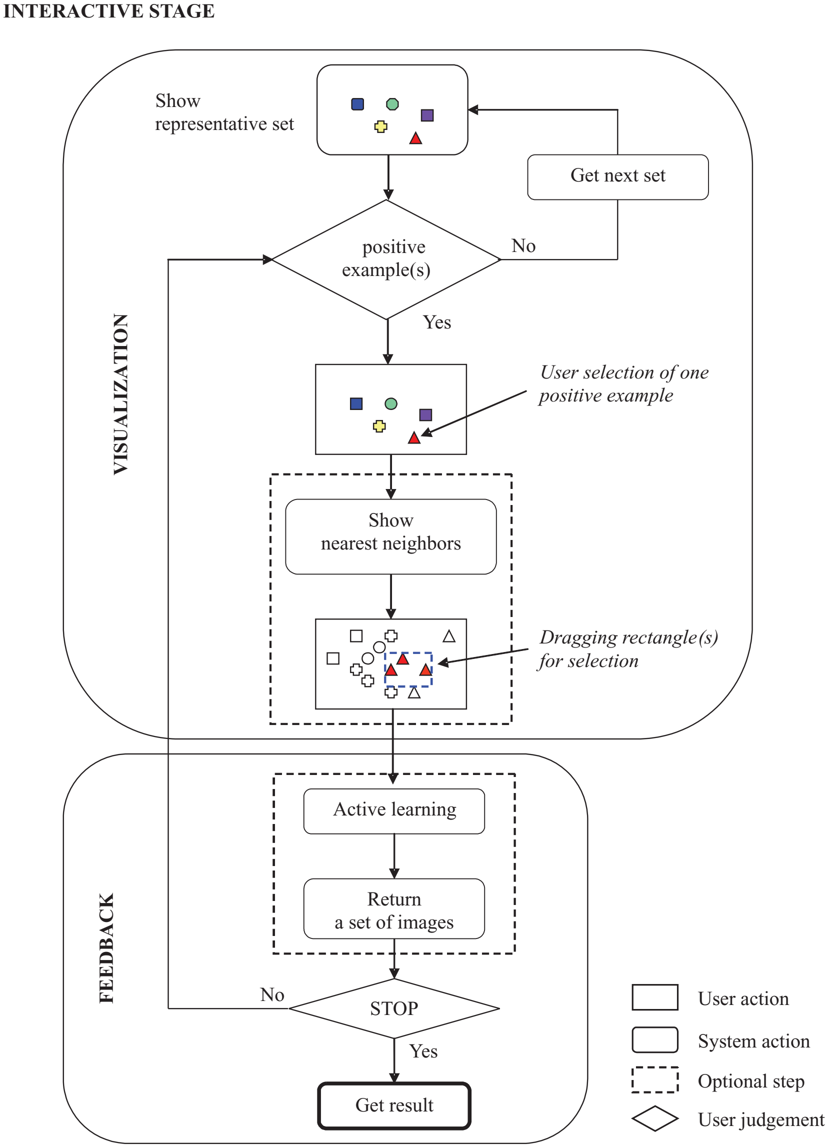

Fig. 3. The interactive stage of the SimVis$_{\text{Active}}$ system.

of a simulated users and system actions helps in optimizing the overall performance. Hence, in this section, all possible system and user actions in the proposed scheme are simulated.

The main concentration is the interactive stage, where the *system actions* are:

- Selection of images to display following the balance between the overview and visibility requirement.
- Adjustment of those images on the display such that the balancing cost function between the visibility and the structure preservation requirements are optimized.
- Remembering examples selected or removed by the user.
- Reaction to user requests such as: get feedback, show neighbors, and show results.

Secondly, we simulate *user actions* including user judgment. There are four typical types of actions, which count as one action:

- Selecting one image
- Clicking to display another set of representative images to find more examples.
- Clicking to go to the cluster corresponding to the current example.
- Dragging a rectangle around a set of positive images. In the worst case, a rectangle contains only one image.

The first three actions are straightforward to implement. For the last one, we proposed an algorithm for finding the minimum number of rectangles needed to cover all relevant images in the currently displayed set [Nguyen and Worring 2005]. Briefly, for each relevant image, the system finds a rectangle containing the maximum number of neighboring relevant images such that none of the irrelevant ones is inside. The pseudo-code for simulating the user action of dragging a rectangle is:

```
Rectangle-search(an image set M)
   For each element m in M
       If (m has not been annotated as positive)
           X = sort(neighbors(m)) on distance to m;
           R = draw_rectangle(m);
           For each element x in X, where category(x)=category(m)
               store(R);
               R = draw_rectangle({R, m}); //increase size of the rectangle.
               If R contains z, where category(z)!=category(m)
                   R = store(R);
                   break;
               else
                   annotate(x);
```

Figure 4 shows the simulated scenario with simulated user actions and simulated system actions associated with the scheme. The *counting user actions* will calculate the number of actions the simulated user needs during the interactive process. The *optimization* part involves the adjustments of parameters in the simulated scenario.

3. EXPERIMENTS

3.1 Data Preparation

For studying the proposed methods, we use three different image collections with widely varying characteristics. First, the ALOI (Amsterdam Library Object Images) dataset which contains high quality images of 1000 objects [Geusebroek et al. 2005]. Each image contains one object, and this object is captured in different orientations and lighting conditions. In total, the dataset has 110,250 images. For each object, we select 12 different recording conditions, which includes 4 different in-plane viewing angles (0°, 30°, 90°, 270°), 6 different illumination angles, and 2 illumination colors. The 12 images of the object form one category. Thus, the dataset has 12,000 images in total, with 1000 categories. As images of the same class capture the same object, we expect a lot of structures in the information space spanned by these images. Figure 5 shows the 12 different images of one object.

The second one is the well-known Corel dataset, which is used often in existing search systems. Images in this dataset are also of high quality. Different from the first collection, they are classified into semantic categories, where images in one category might vary widely in visual appearance. We select a collection of 33326 Corel images with different scenes such as winter and surfing, and objects such as flowers, buses and birds. As ground truth, we use the pre-defined categorization of Corel. Thus, we have 460 categories where the size of each category is different. An example from this collection is shown in Figure 6.

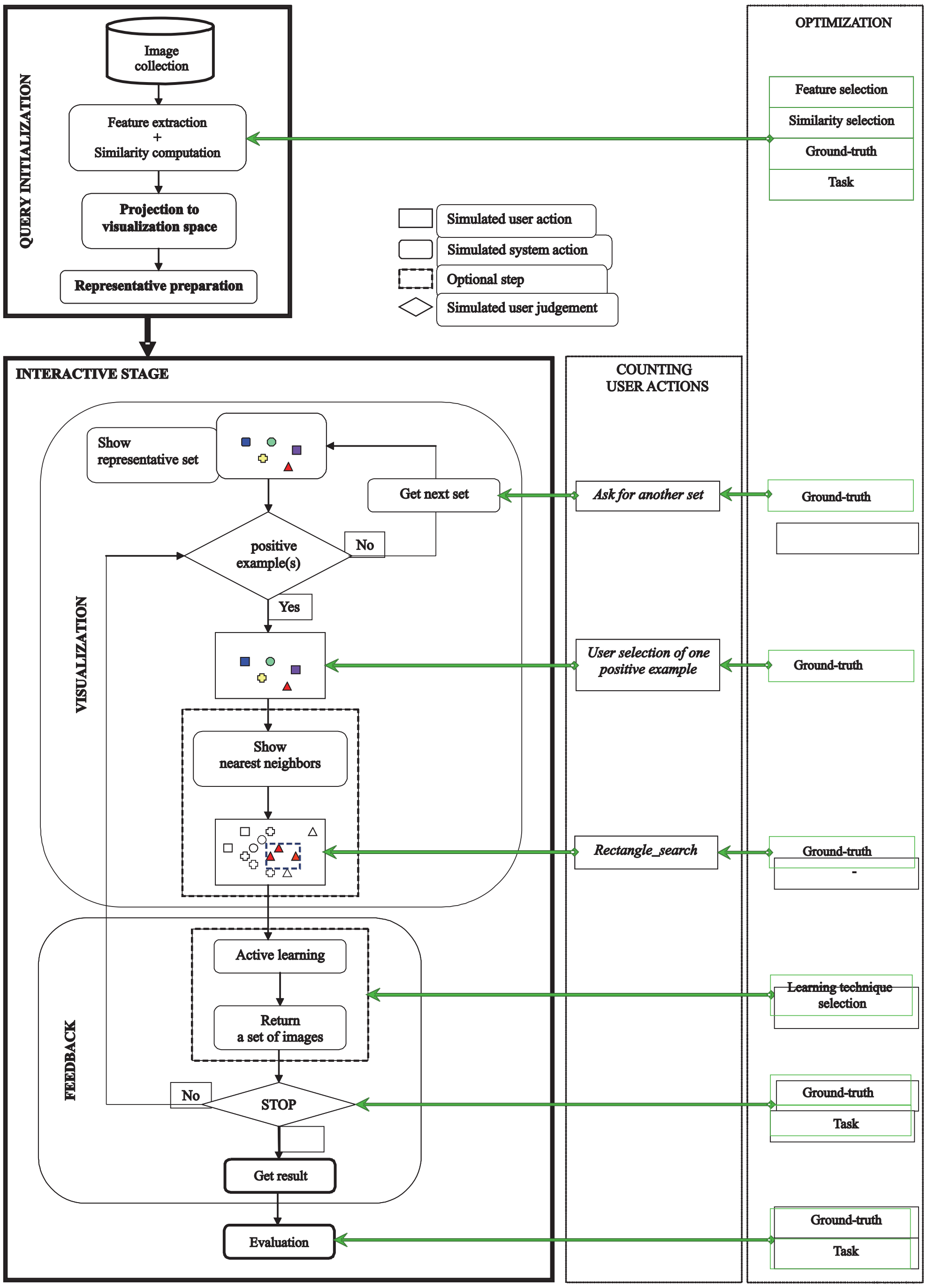

Fig. 4. Scheme with simulated user and system actions.

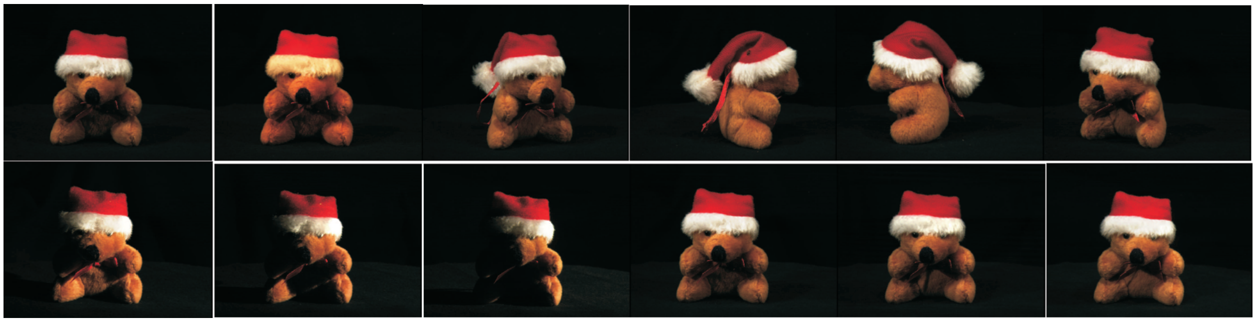

Fig. 5. Left-right, top-down order: two images with changing illumination colors, images under 4 different viewing angles, images with 6 different illumination angles. Each object is captured in total of 12 different conditions.

Fig. 6. An example of images in the category *Bus* from the Corel dataset.

Fig. 7. An example of images in the category *People shaking hands* from the TRECVID dataset.

The final collection contains images taken from the TRECVID 2005 news video archive [Snoek et al. 2005]. As we deal with images, each video is first segmented into shots; each shot is then represented by a keyframe. Thus, TRECVID05 contains 45765 images. TRECVID defines a set of 24 topics such as *images of Tony Blair*, or *images of people shaking hands*. For these 24 search topics pooled ground-truth information is available and these topics are thus used as categorization. The selection of this dataset into the evaluation is challenging in many aspects. As images are extracted from videos, which are encoded, the quality of this dataset is low compared to the other two. Moreover, they have a semantic categorization, and within these categories elements vary largely in content (see Figure 7).

3.2 Selection of Similarity/Distance Functions

With the L*a*b and Wiccest selected as features, this section concentrates on selecting suitable similarity or distance functions. The primary role of the distance function is to add structure to the data. In the ideal structure, the distance yields semantic clusters, where neighboring images belong to the same category. Therefore, the aim of this section is to choose the distance function able to return the maximum number of relevant nearest neighbors. This is important as in the projection step, nearest neighbors are first computed after which the mapping tries to preserve those neighbors in the projected space. So the k-nearest neighbors of an image ideally belong to the same category.

Now let $\mathcal{I}_C$ be the set of images in category C. For evaluation of the distance functions in 2.2.2, we compute for each image $I_i \in \mathcal{I}_C$ the set $\mathcal{I}_i^k$ of k nearest neighbors. Results are evaluated, using the ground truth categorization. Recall and precision values for image I_i are:

$$\text{recall}(I_i) = \frac{\|\mathcal{I}_i^k \bigcap \mathcal{I}_C\|}{||\mathcal{I}_i^k||}, \quad \text{precision}(I_i) = \frac{\|\mathcal{I}_i^k \bigcap \mathcal{I}_C\|}{||\mathcal{I}_C||}. \tag{13}$$

From there, we compute mean recall and mean precision as:

$$\text{mean recall} = \frac{\sum_{I_i \in \mathcal{I}_C} recall(I_i)}{||\mathcal{I}_C||}, \quad \text{mean precision} = \frac{\sum_{I_i \in \mathcal{I}_C} precision(I_i)}{||\mathcal{I}_C||}, \tag{14}$$

where ||.|| denotes the size of a set.

Figure 8 shows results for each collection. The results are surprising. The default distance L_2 used for the L*a*b feature is not the best one. Its performance is even worse than L_1. For all of the three collections, the Matusita function is the best out of all the distance functions considered for L*a*b. For the Wiccest feature, the standard distance function, as expected, performs significantly better than the rest.

From the experiments, we decide to use the Matusita function in the case of L*a*b, and the original Wiccest distance for comparison of images described with the Wiccest feature.

3.3 System Setup

In the query initialization or the offline stage, because of the need for a balance between the different requirements, parameters need to be set such as the number of clusters and the values for λ_1 and λ_2. We use the result from Nguyen and Worring [2005], and λ_1 and λ_2 (in the equations Eq. 4 and Eq. 5) are set to 0.5 and 0.9, respectively. In this reference, we experimented with different values for n in Eq. 4, $n = 100$ turned out being optimal.

K-means algorithm with $n = 100$ is applied to the whole collection. With large images sets, the original k-means algorithm has two major disadvantages: high memory requirements and large computation time it requires the calculation of the $N \times N$ distance matrix, where N is number of images. As this process is done only one time in the offline stage, timing is not an important issue. To overcome the memory problem, we use the competitive learning technique for the k-means algorithm [Rumelhart and Zipser 1985].

The data preparation for the experiments is shown in Figure 9. Each of the selected features is computed for each image collection and stored. For both the projection from high dimensional feature space and the visualization space, neighbors are found using the corresponding distance functions.

3.4 Evaluation Criteria

The benefit of using SVM is clearly proved in previous work, our aim in this paper is to prove that the combination of SVM and similarity based visualization improves the results even further.

To evaluate and compare different methods, both recall and precision are important. It is however easier to have one single measurement. Therefore, another popular measure is the average precision [Voorhees and Harman 2001; Zhu 2004] which calculates the area under the precision and recall curve. The values of those measurements are in the range [0, 1].

Average precision *is the sum of the precision at each relevant hit in the result set divided by the total number of relevant images in the collection.*

Besides, high average precision, our aim is to reduce the number of user actions needed to get there. Hence, the average precision is computed as function of the interactive effort.

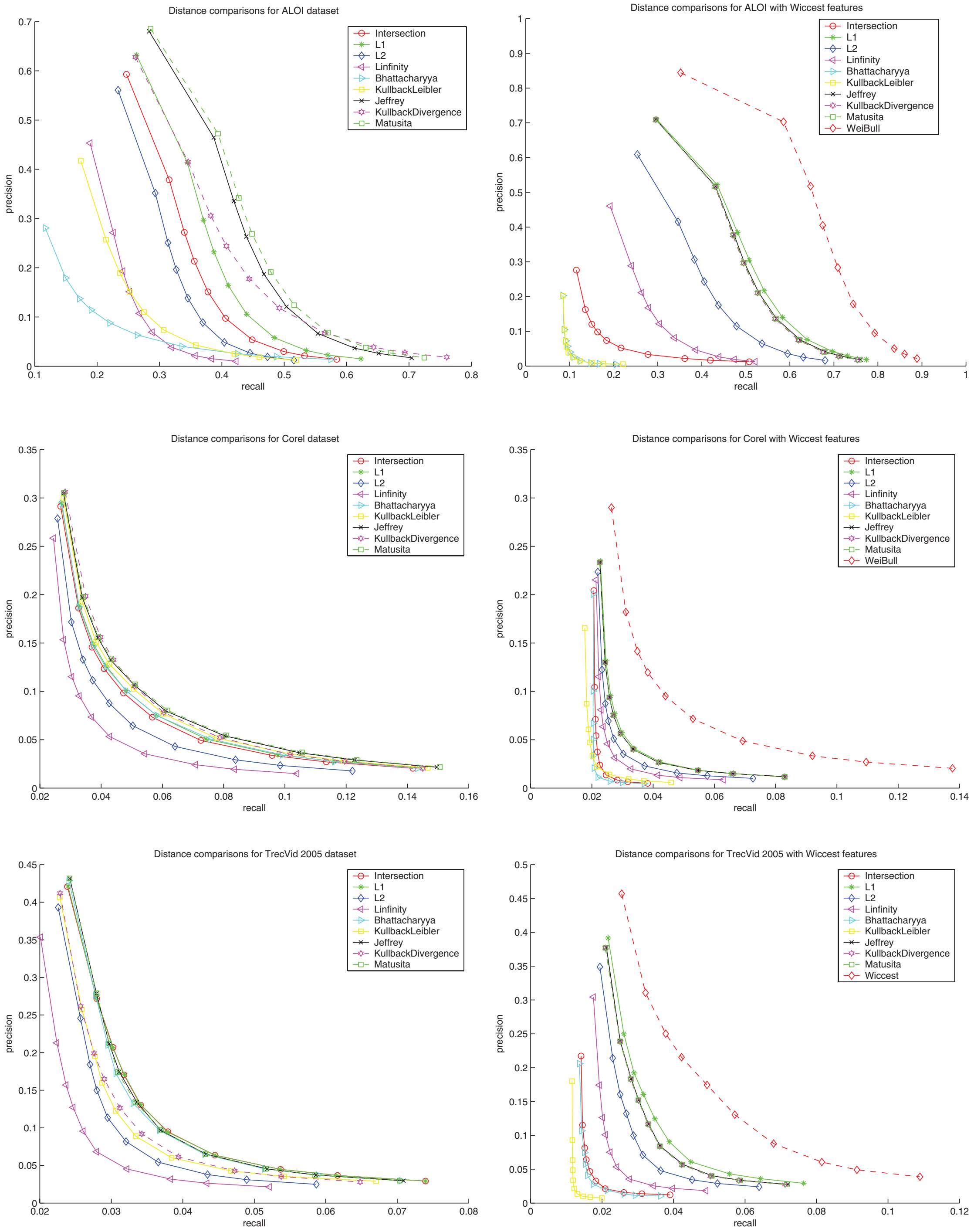

Fig. 8. Comparison of distance functions: The first column gives results for L*a*b; in top-down order the ALOI, Corel, and TRECVID collection. In a similar way, the second column gives results for the Wiccest features.

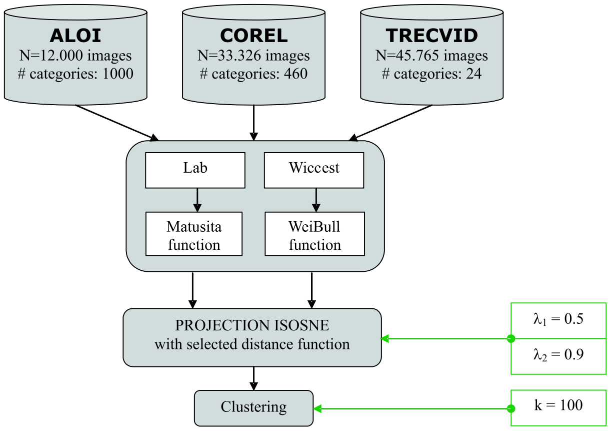

Fig. 9. Query initialization in experiments.

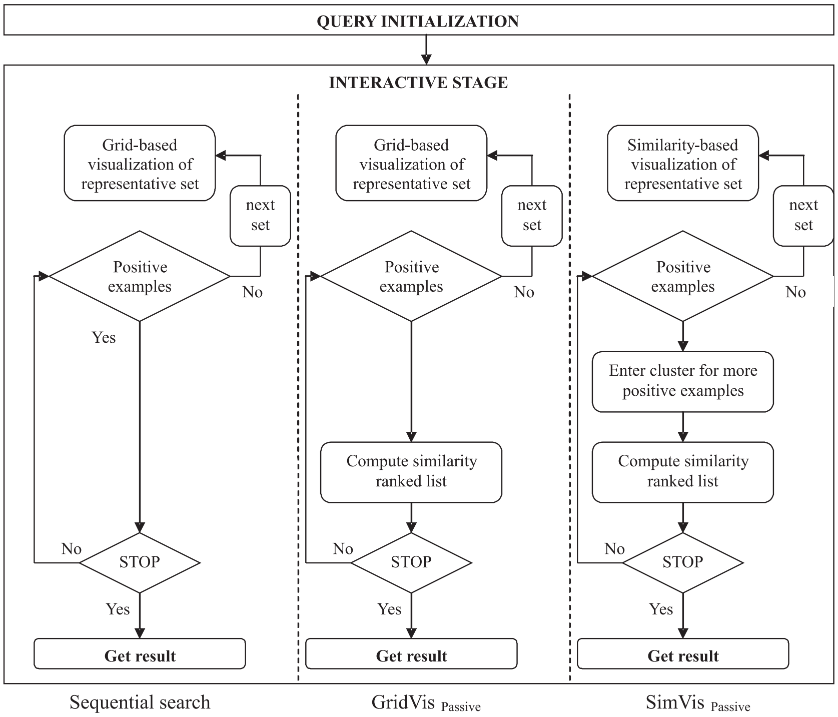
QUERY INITIALIZATION
INTERACTIVE STAGE
Grid-based visualization of representative set
next set
Positive examples
No
Yes
STOP
No
Yes
Get result
Compute similarity ranked list
Similarity-based visualization of representative set
Enter cluster for more positive examples
Sequential search
GridVis Passive
SimVis Passive

Fig. 10. Three different search approaches. From left to right: sequential search, $\mathbf{GridVis_{Passive}}$, $\mathbf{SimVis_{Passive}}$.

Interactive effort *is the total number of actions one needs to interact with the system to get a certain result.*

For comparison, we consider the search system with or without SimVis and with or without active learning. First, to provide the baseline, without both of the optional steps, we get exhaustive search, without any system support. In the baseline, the user gradually goes through the collection to find all relevant images. The system displays 100 random images in sequence (we name this visualization GridVis, i.e., grid-based visualization), the user selects positive images one-by-one from the images in the visualization. This search is called **sequential search**. When finished, another random set is shown. The process is repeated until all images of the category are found.

Second, the first optional step is again removed, that is, the visualization method is GridVis, and the second one is replaced by the standard interactive search algorithm, leading to $\mathbf{GridVis_{Passive}}$. In this search, after a number of interactions, the selected images are used as query set. Similarity values between all images in the collection to the query set are computed based on a predefined similarity function. A ranked list based on these similarity values is obtained. The top 100 most similar ones are returned. From there, average precision values are also computed. We report the mean average precision at interactive effort intervals of 5 over all the search categories for each collection.

As a third method, we apply the first optional step to the sequential search, to find what performance in search can be achieved without an advanced learning algorithm, $\mathbf{SimVis_{Passive}}$.

Finally, the proposed scheme, $\mathbf{SimVis_{Active}}$ (in Figure 4) is compared to all the others.

In summary, four different search strategies are considered

- **Sequential**: exhaustive search with gradual annotation of relevant images.
- $\mathbf{GridVis_{Passive}}$: interactive search using grid based visualization, no support of similarity based visualization.
- $\mathbf{SimVis_{Passive}}$: interactive search with similarity based visualization.
- $\mathbf{SimVis_{Active}}$: interactive search in the proposed scheme combining similarity based visualization and active learning.

3.5 Selection of Active Learning Methods

In this experiment, the two SVM approaches in Section 2.3 are considered. Parameters for the two methods are taken from Chang and Lin [2001].

Figure 11 shows the comparison results with the three image collections. As discussed above, the two-class SVM does not work well if there is unbalance between the size of the relevant and irrelevant set. In the ALOI collection, the number of images in each category is rather small (12 images), the two-class SVM will classify many irrelevant ones as relevant. Moreover, images of the same category in this collection have a visual similar appearance. Therefore, using one-class SVM out-performs two-class SVM.

In the Corel collection, as the number of relevant ones is reasonable with 100 or more images, the performance of two-class SVM is improved. Because of the visual appearance of images in the same category of this collection are varied, relevant images can be spread out instead of grouping into one unique area. This makes the difficulty for both the one-class and the two-class SVM. Overall, the performances of the two techniques are comparable, but again the speed factor is a concern as the size of training data increases rapidly during interaction with the two-class SVM. The same arguments are applied to the result of the third collection, the TRECVID, which is shown in the last two figures. Again, the performance of the one-class SVM is higher.

Results prove our expectation on the performance of the two techniques. Despite the fact that more information is given in the training examples for two-class SVM than in one-class SVM, it turns out the one-class SVM works better in general. One of the reasons is that the number of non-relevant images is very large while the relevant set is much smaller. A way to gain a better performance in this particular problem of the two-class SVM is to adjust the parameters such as setting the penalty for the negative class higher. However, doing this iteratively according to the user relevant feedback is very difficult. Moreover, after one feedback iteration the size of the training data is increased especially the number

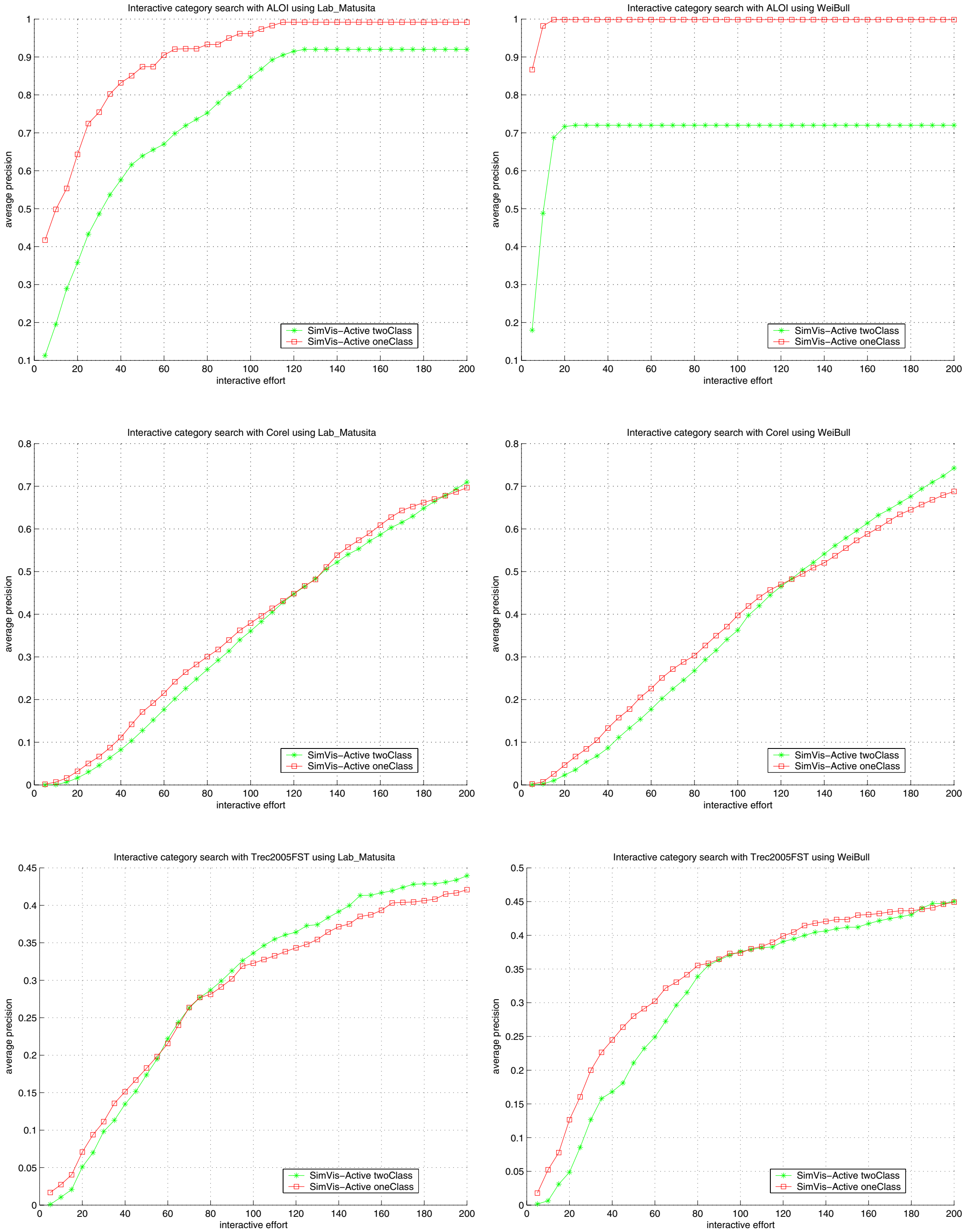

Fig. 11. Comparison of SVM performance. The first column gives results with L*a*b, top-down the ALOI, Corel, and TRECVID collection. In a similar way, the second column gives results for the Wiccest features.

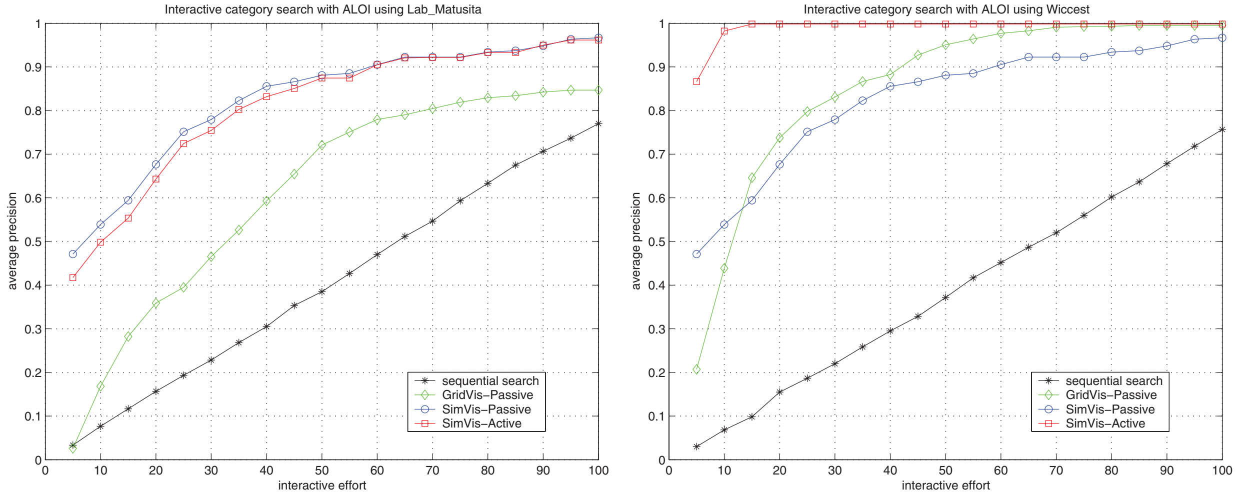

Fig. 12. Results of *interactive effort* vs. average precision with the ALOI data. Left to right: with the L*a*b, with the Wiccest features.

of negative examples, hence the training time takes longer. In interactive search, processing speed is an important element. All the computation and display should be in acceptable time as the user does not have the patience to wait long for the next iteration. One-class SVM, requiring fewer examples, will be a better option.

One-class SVM does have the disadvantage that when relevant images fall into different clusters or they are scattered, it may concentrate on one part of the feature space containing relevant images, ignoring other potentially relevant areas. Nevertheless, in the given search scenario, within a certain search time, the number of correct ones returned are more important than how diverse the distribution of correct ones is. Therefore, we select the one-class SVM.

3.6 Experimental Results on the Full Scenario

For each query, we simulate the user search actions following the proposed scheme. The search stops when the average precision reaches 1.

The ALOI results are presented in Figure 12. Since images of the same category are close in information space, even GridVis$_{\text{Passive}}$ performs well in the beginning. However, at some point, it cannot find more relevant images, while the average precision of the proposed approach still increases. Secondly, results show that using Wiccest features gives better performance in this dataset than L*a*b. The average precision reaches 1 very fast especially with our approach. The search time is improved significantly. Where in the baseline the user needs on average 80 actions to get all the relevant images, only 20 actions are needed in SimVis$_{\text{Active}}$.

Results in Figure 13 are for the Corel collection. The GridVis$_{\text{Passive}}$ performs worst, even worse than the sequential baseline. This is expected, as the color histogram is too poor to distinguish high-level concepts. With similarity-based visualization, the result of SimVis$_{\text{Passive}}$ is improved significantly. At the beginning, when only limited interactive effort has been made, that is, a small number of training samples are available, the GridVis$_{\text{Passive}}$ even yields better results than the SimVis$_{\text{Passive}}$ and SimVis$_{\text{Active}}$. When sufficient examples are provided, performance of the proposed system becomes the best. Overall, Wiccest features do not improve the performance upon simple L*a*b histograms for this dataset.

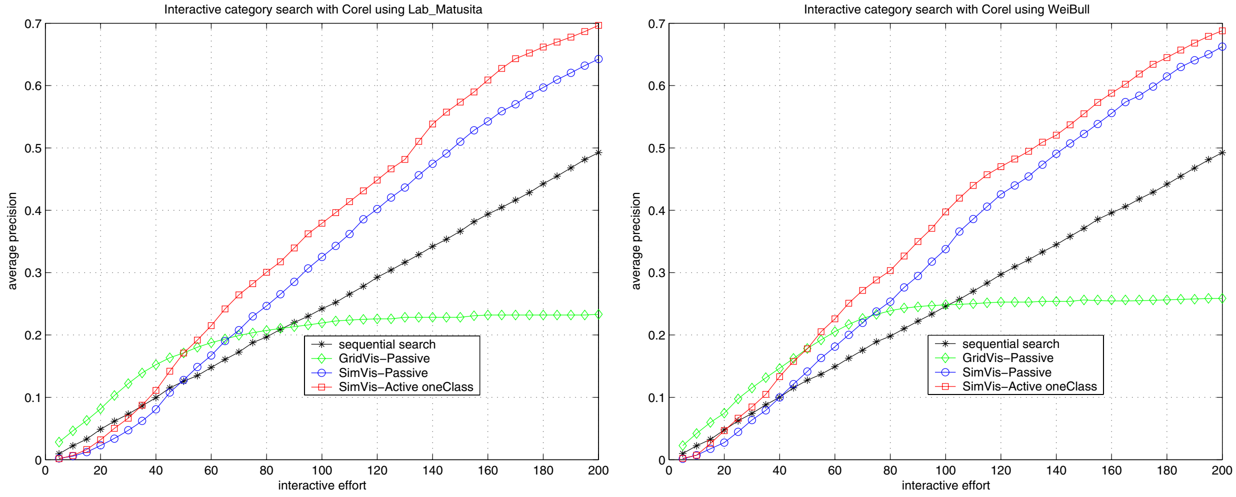

Fig. 13. Results of *interactive effort* vs. average precision with the Corel data. Left to right: with the L*a*b, with the Wiccest features.

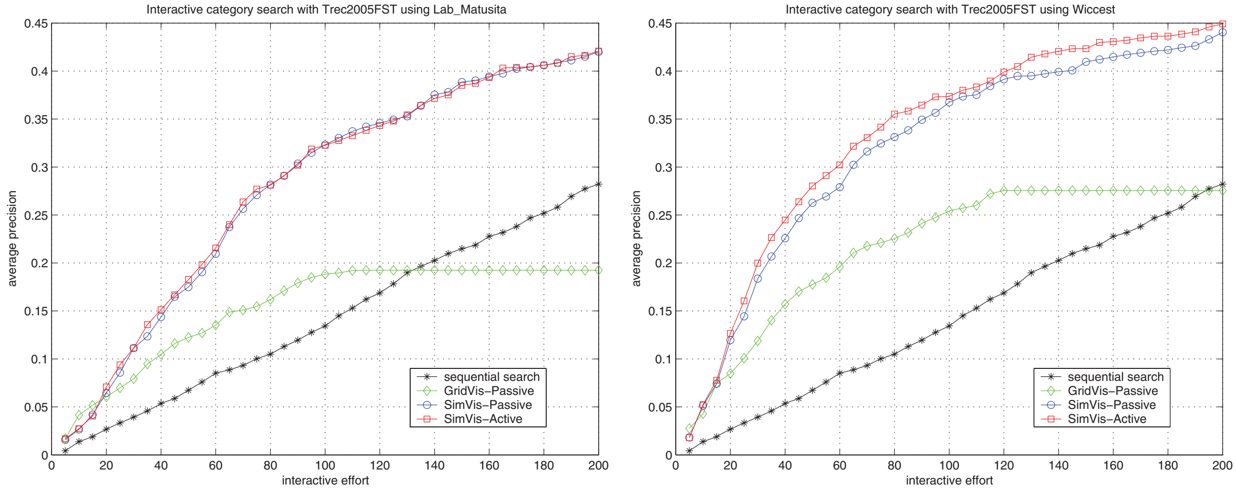

Fig. 14. Results of *interactive effort* vs. average precision with the TRECVID data. Left to right: with the L*a*b, with the Wiccest features.

Figure 14 gives the results for search topics in TRECVID data. It is obvious that the baselines are very low as the search tasks are semantic. The global histogram L*a*b is too simple to obtain the relevant images, meaning that the relevant images are easily scattered into different clusters. Therefore, though the performance of the proposed system is improved, on average the different between $\text{SimVis}_{\text{Passive}}$ and $\text{SimVis}_{\text{Active}}$ is not as significant as in case of the ALOI and Corel collection. To search through this collection, filtering based on a textual query or a semantic concept query, as for example in Snoek et al. [2005], should be applied first to reduce the whole collection to a limited set of more homogeneous images. From there, our approach can be employed to do the search.

In summary, the proposed approach works well in all three different image collections with different image features. It shows that the combination of advanced visualization and active learning does lead

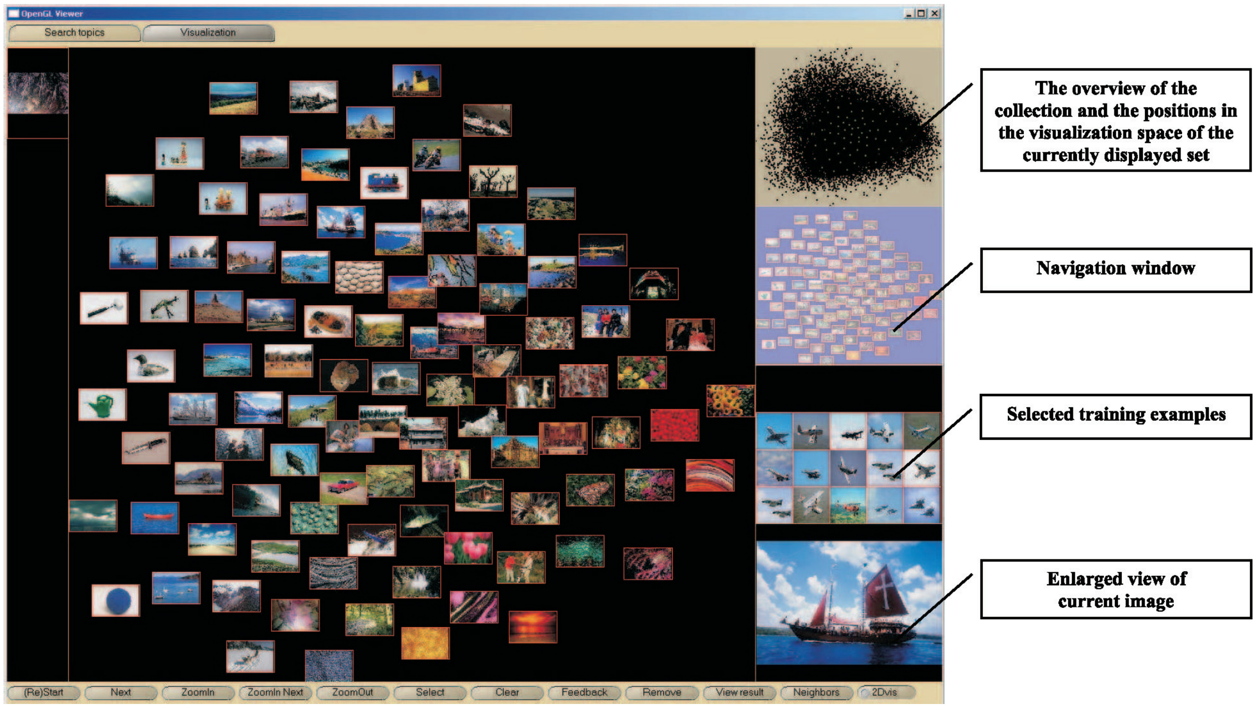
OpenGL Viewer
Search topics
Visualization
The overview of the collection and the positions in the visualization space of the currently displayed set
Navigation window
Selected training examples
Enlarged view of current image
(Re)Start
Next
ZoomIn
ZoomIn Next
ZoomOut
Select
Clear
Feedback
Remove
View result
Neighbors
2Dvis

Fig. 15. Screen-dump of the interactive search system with the Corel dataset.

to a better search performance. Results with the Wiccest features in general are better than with the simple L*a*b color histogram, especially with the ALOI collection where images of the same search category are very similar. Figure 15 show an example of the system interface, where interactions between the user and the system take place.

4. CONCLUSION

In interactive search, most systems only concentrate on improving the performance of active learning based on user feedback. Therefore, the way images are shown to the user has received little attention. On the other hand, in evaluating the interaction interface between the user and the system-related work is mostly interested in the visualization. A visual search system combining similarity based visualization and active learning using support vector machines has been proposed. In this scheme, we analyze possible relations between visualization and relevance feedback. From there, an optimal solution is presented.

On the way of building an optimal system, we have implemented different distance functions for comparing two feature vectors. Our experiments show that the selection of distance function is strongly dependent on the chosen feature. The default function may not be the best option, for instance for L*a*b histograms, the Matusita function is certainly better than the commonly used Euclidean function. A comparison between one-class and two-class SVM is also presented. Our conclusion is that for interactive search, the two methods are generally having equal performance, but one-class SVM is much faster, therefore, better suited for interactive search.

To prove the efficiency of our method, we have selected three different classes of image collections. The first collection, ALOI, has high quality and exhibits structure in the high dimensional feature space as images in a category are of the same object. The Corel collection also has high quality but has a semantic categorization; hence, difficult for visual search. The last one, a collection of broadcast news shots used in TRECVID 2005 from CNN, Chinese, and Arabic broadcasters, is the most challenging with varying visual appearance and semantics in each category.

Evaluating the system with objective evaluation, where all possible actions are simulated, is also one of our contributions in this paper. This choice overcomes the limitation of having real users or subjective evaluation. The proposed simulated actions can be easily applied to other systems.

Experiments show that for the ALOI collection, our system reduces the user interactive effort by a factor of 4 compared to the search baseline. For example, after 10 actions it reaches an average precision of almost 1.0, whereas it needs up to 70 actions in the baseline to get similar results. In case of Corel, on average, the performance of the proposed approach is 3 times better with the same number of user actions, and almost 2 times for the TRECVID collection. Our system, therefore, eases user search tasks. With fewer actions it is possible for the user to get more images that are relevant.

The performance of the new class of color invariant features, the Wiccest features, is still domain dependent. It works well, specifically for the ALOI collection, when images of the same category search are very similar. For collections where similarity of images is more abstract such as in the TRECVID collection, its performance does not outperform the simple color histogram feature. That means for a general search system, using the color histogram is still a good option.

In this article, we concentrate on visual exploration of the collection as a whole while ignoring the query specification step. However, the proposed scheme easily applies to systems which use a query as filtering step. Especially in cases where the search is based on semantic categorization, for instance in the TRECVID collection, our visual based search scheme works well [Snoek et al. 2005].

With the tremendous growth of image collections, a very time consuming user interaction process will result. Our approach can be considered as the beginning of a new generation of advanced search mechanisms. The scenario-based evaluation method is a general scheme for evaluating interactive retrieval systems. Many other search scenarios can be objectively evaluated in order to find effective integrated methods.

ACKNOWLEDGMENTS

The work is within the ImIk project (Interactive disclosure of Multimedia Information and Knowledge) sponsored by IOP MMI (Man-Machine Interaction) and the BSIK MultimediaN project. Dr. J. M. Geusebroek is thanked for providing the ALOI collection as well as the Wiccest features.

REFERENCES

Basseville, M. 1989. Distance measures for signal processing and pattern recognition. *Sig. Process 18*, 349–369.

Bederson, B. 2001. PhotoMesa: A zoomable image browser using quantum treemaps and bubblemaps. In *Proceedings of the ACM Symposium on User Interface Software and Technology, CHI Letters*. Vol. 3. 71–80.

Chang, C. and Lin, C. 2001. LIBSVM: A library for support vector machines. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Chen, M., Christel, M., Hauptmann, A., and Wactlar, H. 2005. Putting active learning into multimedia applications: Dynamic definition and refinement of concept classifiers. In *MULTIMEDIA '05: Proceedings of the 13th Annual ACM International Conference on Multimedia*. ACM, New York, 902–911.

Chen, Y., Zhou, X., and Huang, T. 2001. One-class SVM for learning in image retrieval. In *Proceedings of the International Conference on Image Processing*. Vol. 1. 34–37.

Dubes, R. 1987. How many clusters are best?—An experiment. *Pattern Recog. 20*, 6, 645–663.

Geusebroek, J., Burghouts, G., and Smeulders, A. 2005. The Amsterdam library of object images. *Int. J. Comput. Vision 61*, 1, 103–112.

Geusebroek, J. M. and Smeulders, A. W. M. 2005. A six-stimulus theory for stochastic texture. *Int. J. Comput. Vision 62,* 1/2, 7–16.

GOSSELIN, P. AND CORD, M. 2004. RETIN AL: An active learning strategy for image category retrieval. In *Proceedings of the International Conference on Image Processing*.

GUO, G., JAIN, A., MA, W., AND ZHANG, H. 2001. Learning similarity measure for natural image retrieval with relevance feedback. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*.

HEESCH, D. AND RUGER, S. 2004. Three interfaces for content-based access to image collections. In *Proceedings of the International Conference on Image and Video Retrieval*. 491–499.

IVORY, Y. AND HEARST, M. 2001. The state of the art in automating usability evaluation of user interfaces. *ACM Comput. Surv. 33*, 4, 470–516.

KEIM, D. 2002. Information visualization and visual data mining. *IEEE Trans. Visual. Comput. Graph. 7*, 1, 100–107.

MANEVITZ, L. AND YOUSEF, M. 2004. One-class SVMs for document classification. *J. Mach. Learn. Res. 2*, 139–154.

MOGHADDAM, B., TIAN, Q., LESH, N., SHEN, C., AND HUANG, T. 2004. Visualization and user-modeling for browsing personal photo libraries. *Int. J. Comput. Vision 56*, 1-2, 109–130.

NAPHADE, M., BASU, S., SMITH, J., LIN, C., AND TSENG, B. 2002. Modeling semantic concepts to support query by keywords in video. In *Proceedings of the IEEE International Conference on Image Processing*. IEEE Computer Society Press, Los Alamitos, CA.

NGUYEN, G. AND WORRING, M. 2005. Similarity based visualization of image collections. In *Proceedings of the 7th International Workshop on Audio-Visual Content and Information Visualization in Digital Libraries*.

NGUYEN, H. AND SMEULDERS, A. 2004. Everything gets better all the time, apart from the amount of data. In *Proceedings of the International Conference on Image and Video Retrieval*.

RODDEN, K., BASALAJ, W., SINCLAIR, D., AND WOOD, K. 2001. Does organisation by similarity assist image browsing? In *Proceedings of the ACM Conference on Human Factors in Computing Systems*. ACM, New York, 190–197.

RUBNER, Y. 1999. Perceptual metrics for database navigation. Ph.D. dissertation, Stanford University, Stanford, CA.

RUBNER, Y., PUZICHA, J., TOMASI, C., AND BUHMANN, J. 2001. Empirical evaluation of dissimilarity measures for color and texture. *Comput. Vision Image Understand. J. 84*, 1, 25–43.

RUI, Y., HUANG, T., AND MEHROTRA, S. 1998. Relevance feedback: A power tool for interactive content-based image retrieval. *IEEE Trans. Curcuits Video Tech. 8*, 5, 644–655.

RUMELHART, D. AND ZIPSER, D. 1985. Feature discovery by competitive learning. *Cogn. Sci. 9*, 75–112.

SANTINI, S., GUPTA, A., AND JAIN, R. 2001. Emergent semantics through interaction in image databases. *IEEE Trans. Knowl. Data Eng. 13*, 3, 337–351.

SHNEIDERMAN, B. 1996. The eyes have it: A task by data type taxonomy for information visualizations. In *Proceedings of the IEEE Conference on Visual Languages*. IEEE Computer Society Press, Los Alamitos, CA, 336–343.

SMEULDERS, A., WORRING, M., SANTINI, S., GUPTA, A., AND JAIN, R. 2000. Content-based image retrieval at the end of early years. *IEEE Trans. Pattern Anal. Mach. Intel. 22,* 12, 1349–1380.

SNOEK, C., WORRING, M., VAN GEMERT, J., GEUSEBROEK, J., KOELMA, D., NGUYEN, G., DE ROOIJ, O., AND SEINSTRA, F. 2005. Mediamill: Exploring news video archives based on learned semantics. In *Proceedings of ACM Multimedia* (Best Technical Demonstration Award). ACM, New York.

SWAIN, M. AND BALLARD, D. 1991. Color indexing. *Int. J. Comput. Vision 7*, 1, 11–32.

TONG, S. AND CHANG, E. 2001. Support vector machine active learning for image retrieval. In *Proceedings of the ACM Interneltional Conference on Multimedia*. Vol. 9. ACM, New York, 107–118.

VOORHEES, E. AND HARMAN, D. 2001. Trec-10 proceedings appendix on common evaluation measures. http://trec.nist.gov/pubs/trec10/appendices/measures.pdf.

ZHANG, L., LIN, F., AND ZHANG, B. 2001. Support vector machine learning for image retrieval. In *Proceedings of the International Conference on Image Processing*. 721–724.

ZHOU, X. AND HUANG, T. 2003. Relevance feedback in image retrieval: A comprehensive overview. *Multimed. Syst. 8*, 6, 536–544.

ZHU, M. 2004. Recall, precision and average precision. http://www.stats.uwaterloo.ca/stats_navigation/techreports/04WorkingPapers/2004-09.pdf.

Received December 2005; revised August 2006; accepted November 2006