Efficient visual search of videos cast as text retrieval

Josef Sivic and Andrew Zisserman

***Abstract*— We describe an approach to object retrieval which searches for and localizes all the occurrences of an object in a video, given a query image of the object. The object is represented by a set of viewpoint invariant region descriptors so that recognition can proceed successfully despite changes in viewpoint, illumination and partial occlusion. The temporal continuity of the video within a shot is used to track the regions in order to reject those that are unstable.**

Efficient retrieval is achieved by employing methods from statistical text retrieval, including inverted file systems, and text and document frequency weightings. This requires a visual analogy of a word which is provided here by vector quantizing the region descriptors. The final ranking also depends on the spatial layout of the regions. The result is that retrieval is immediate, returning a ranked list of shots in the manner of Google [6].

We report results for object retrieval on the full length feature films 'Groundhog Day', 'Casablanca' and 'Run Lola Run', including searches from within the movie and specified by external images downloaded from the Internet. We investigate retrieval performance with respect to different quantizations of region descriptors and compare the performance of several ranking measures. Performance is also compared to a baseline method implementing standard frame to frame matching.

***Index Terms*— Object recognition, viewpoint and scale invariance, text retrieval**

I. Introduction

The aim of this work is to retrieve those key frames and shots of a video containing a particular object with the ease, speed and accuracy with which Google [6] retrieves text documents (web pages) containing particular words. This paper investigates whether a text retrieval approach can be successfully employed for this task.

Identifying an (identical) object in a database of images is now reaching some maturity. It is still a challenging problem because an object's visual appearance may be very different due to viewpoint and lighting, and it may be partially occluded, but successful methods now exist [17], [18], [20], [25], [33]–[35], [41], [42]. Typically an object is represented by a set of overlapping regions each represented by a vector computed from the region's appearance. The region extraction and descriptors are built with a controlled degree of invariance to viewpoint and illumination conditions. Similar descriptors are computed for all images in the database. Recognition of a particular object proceeds by nearest neighbour matching of the descriptor vectors, followed by disambiguating using local spatial coherence (such as common neighbours, or angular ordering), or global relationships (such as epipolar geometry or a planar homography).

J. Sivic is with INRIA, WILLOW Project-Team, Laboratoire d'Informatique de l'Ecole Normale Superieure (CNRS/ENS/INRIA UMR 8548), 45 rue d'Ulm, 75230 Paris Cedex 05, France. E-mail: josef@di.ens.fr. This work was carried out whilst J.S. was with the Department of Engineering Science, University of Oxford.

A. Zisserman is with the Department of Engineering Science, University of Oxford, Parks Road, Oxford, OX1 3PJ, UK. E-mail: az@robots.ox.ac.uk

We explore whether this type of approach to recognition can be recast as text retrieval. In essence this requires a visual analogy of a word, and here we provide this by vector quantizing the descriptor vectors. However, it will be seen that pursuing the analogy with text retrieval is more than a simple optimization over different vector quantizations. There are many lessons and rules of thumb that have been learnt and developed in the text retrieval literature and it is worth ascertaining if these also can be employed in visual retrieval.

The benefits of the text retrieval approach is that matches are effectively pre-computed so that at run-time frames and shots containing any particular object can be retrieved with no-delay. This means that any object occurring in the video (and conjunctions of objects) can be retrieved even though there was no explicit interest in these objects when descriptors were built for the video. However, we must also determine whether this vector quantized retrieval misses any matches that would have been obtained if the former method of nearest neighbour matching had been used.

A. Review of text retrieval

Text retrieval systems generally employ a number of standard steps [3]: the documents are first parsed into words, and the words are represented by their stems, for example 'walk', 'walking' and 'walks' would be represented by the stem 'walk'. A stop list is then used to reject very common words, such as 'the' and 'an', which occur in most documents and are therefore not discriminating for a particular document. The remaining words are then assigned a unique identifier, and each document is represented by a vector with components given by the frequency of occurrence of the words the document contains. In addition the components are weighted in various ways (such as inverse document frequency weighting, described in more detail in section IV). All of the above steps are carried out in advance of actual retrieval, and the set of vectors representing all the documents in a corpus are organized as an *inverted file* [45] to facilitate efficient retrieval. An inverted file is structured like an ideal book index. It has an entry for each word in the corpus followed by a list of all the documents (and position in that document) in which the word occurs.

A query text is treated in a similar manner: its vector of weighted word frequencies is computed. Matching documents are obtained by measuring the similarity between the query and document vectors using the angle between the vectors. In addition the returned documents may be ranked by the correspondence of the word ordering and separation with the query.

B. Paper outline

In this paper we explore visual analogies of each of these steps. Section II describes the visual descriptors used. Section III then describes their vector quantization into visual 'words', and sections IV and V show how the text retrieval techniques are

Digital Object Indentifier 10.1109/TPAMI.2008.111

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.

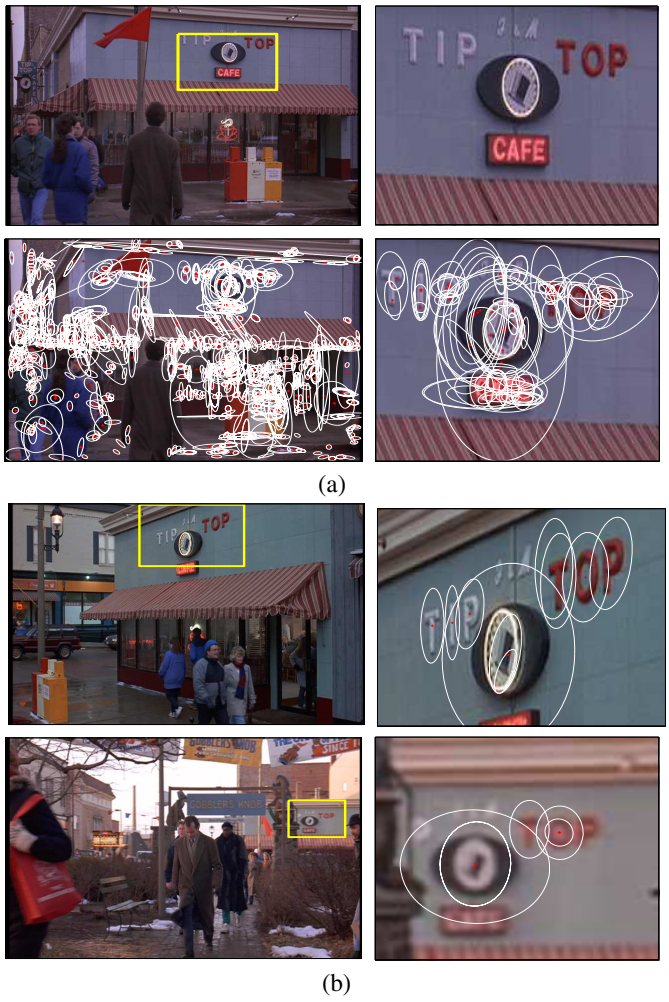


Fig. 1. **Object query example I.** (a) Top row: (left) a frame from the movie 'Groundhog Day' with an outlined query region and (right) a close-up of the query region delineating the object of interest. Bottom row: (left) all 1039 detected affine covariant regions superimposed and (right) close-up of the query region. (b) (left) two retrieved frames with detected regions of interest and (right) a close-up of the images with affine covariant regions superimposed. These regions match to a subset of the regions shown in (a). Note the significant change in foreshortening and scale between the query image of the object, and the object in the retrieved frames. For this query there are four correctly retrieved shots ranked 1, 2, 3 and 12. Querying all the 5,640 keyframes of the entire movie took 0.36 seconds on a 2GHz Pentium.

applied in the visual domain. Finally, in section VI, we evaluate the proposed approach on a ground truth set of six object queries. We investigate retrieval performance with respect to various visual vocabularies and compare the performance of several ranking measures. Performance is also compared to a baseline method implementing standard frame to frame matching without vector quantization. Object retrieval results, including searches from within the movie and specified by external images, are shown on feature films: 'Groundhog Day' [Ramis, 1993], 'Casablanca' [Curtiz, 1942] and 'Run Lola Run' [Tykwer, 1998].

Although previous work has borrowed ideas from the text retrieval literature for image retrieval from databases (e.g. [40] used the weighting and inverted file schemes) to the best of our knowledge this is the first systematic application of these ideas to object retrieval in videos. This paper is an extended version of [39].

II. Viewpoint Invariant Description

The goal is to extract a description of an object from an image which will be largely unaffected by a change in camera viewpoint, the object's scale and scene illumination, and also will be robust to some amount of partial occlusion. To achieve this we employ the technology of viewpoint invariant segmentation developed for wide baseline matching [20], [25], [33], [41], [42], object recognition [18], [25], and image/video retrieval [35], [39]. The idea is that regions are detected in a viewpoint invariant manner – so that for images of the same scene, the pre-image of the region covers the same scene portion. It's important to note that the regions are detected *independently* in each frame. A comprehensive review of viewpoint invariant (also called affine covariant) region detectors, and a comparison of their performance can be found in [22].

In this work, two types of affine covariant regions are computed for each frame. The first is constructed by elliptical shape adaptation about a Harris [12] interest point. The method involves iteratively determining the ellipse centre, scale and shape. The scale is determined by the local extremum (across scale) of a Laplacian, and the shape by maximizing intensity gradient isotropy over the elliptical region [4], [16]. The implementation details are given in [20], [33]. This region type is referred to as Shape Adapted (SA).

The second type of region is constructed by selecting areas from an intensity watershed image segmentation. The regions are those for which the area is approximately stationary as the intensity threshold is varied. The implementation details are given in [19]. This region type is referred to as Maximally Stable (MS).

Two types of regions are employed because they detect different image areas and thus provide complementary representations of a frame. The SA regions tend to be centred on corner like features, and the MS regions correspond to blobs of high contrast with respect to their surroundings such as a dark window on a grey wall. Both types of regions are represented by ellipses. These are computed at twice the originally detected region size in order for the image appearance to be more discriminating. For a 720×576 pixel video frame the number of regions computed is typically 1,200. An example is shown in Figure 1.

Each elliptical affine covariant region is represented by a 128-dimensional vector using the SIFT descriptor developed by Lowe [18]. In [21] this descriptor was shown to be superior to others used in the literature, such as the response of a set of steerable filters [20] or orthogonal filters [33], and we have also found SIFT to be superior (by comparing scene retrieval results against a ground truth [39]). One reason for this superior performance is that SIFT, unlike the other descriptors, is designed to be invariant to a shift of a few pixels in the region position, and this localization error is one that often occurs. Combining the SIFT descriptor with affine covariant regions gives region description vectors which are invariant to affine transformations of the image. Note, both region detection and the description is computed on monochrome versions of the frames, colour information is not currently used in this work.

To reduce noise and reject unstable regions, information is aggregated over a sequence of frames. The regions detected in each frame of the video are tracked using a simple constant velocity dynamical model and correlation [34], [38]. Any region which does not survive for more than three frames is rejected. This 'stability check' significantly reduces the number of regions to about 600 per frame.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.

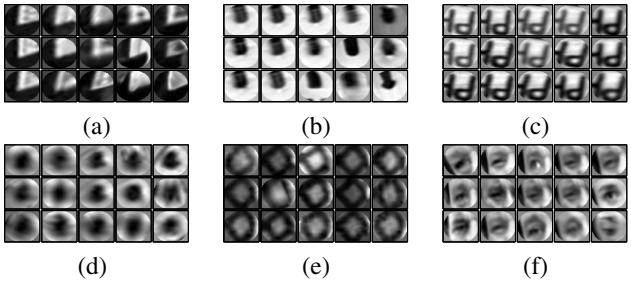


Fig. 2. Samples of normalized affine covariant regions from clusters corresponding to a single visual word: (a–c) Shape Adapted regions; (d–f) Maximally Stable regions. Note that some visual words represent generic image structures, e.g. corners (a) or blobs (d), and some visual words are rather specific, e.g. a letter (c) or an eye (f). Samples in each plate were generated uniformly from all occurrences of the particular visual word in the movie and are shown sorted (in a scan-line order) according to the Mahalanobis distance (1) from the cluster center.

III. Building a visual vocabulary

The objective here is to vector quantize the descriptors into clusters which will be the visual 'words' for text retrieval. The vocabulary is constructed from a subpart of the movie, and its matching accuracy and expressive power are evaluated on the entire movie, as described in the following sections. The running example is for the movie 'Groundhog Day'.

The vector quantization is carried out by K-means clustering, though other methods (K-medoids, histogram binning, mean shift, etc) are certainly possible. Recent works have demonstrated the advantages of using a vocabulary tree [24] or a randomized forest of k-d trees [28] to reduce search cost in the quantization stage.

A. Implementation

Each descriptor is a 128-vector, and to simultaneously cluster all the descriptors of the movie would be a gargantuan task. Instead, a random subset of 474 frames is selected. Even with this reduction there still remains around 300K descriptors that must be clustered.

Mahalanobis distance is used as the distance function for K-means clustering. The distance between two descriptors $\mathbf{x}_1$, $\mathbf{x}_2$, is then given by

$$d(\mathbf{x}_1, \mathbf{x}_2) = \sqrt{(\mathbf{x}_1 - \mathbf{x}_2)^\top \Sigma^{-1} (\mathbf{x}_1 - \mathbf{x}_2)}. \quad (1)$$

The covariance matrix Σ is determined by (i) computing covariances for descriptors throughout tracks within several shots, and (ii) assuming Σ is the same for all tracks (i.e. independent of the region) so that covariances for tracks can be aggregated. In this manner sufficient measurements are available to estimate all elements of Σ. The Mahalanobis distance enables the more noisy components of the 128–vector to be weighted down, and also decorrelates the components. Empirically there is a small degree of correlation. As is standard, the descriptor space is affine transformed by the square root of Σ so that Euclidean distance may be used.

6,000 clusters are used for Shape Adapted regions, and 10,000 clusters for Maximally Stable regions. The ratio of the number of clusters for each type is chosen to be approximately the same as the ratio of detected descriptors of each type. The K-means algorithm is run several times with random initial assignments of points as cluster centres, and the lowest cost result used. The number of clusters was chosen empirically to maximize matching performance on a ground truth set for scene retrieval [39]. The object retrieval performance with respect to the number of clusters is tested on a new ground truth set for object retrieval in section VI.

Figure 2 shows examples of the regions which belong to particular clusters, i.e. which will be treated as the same visual word. The clustered regions reflect the properties of the SIFT descriptors which penalize intensity variations amongst regions less than cross-correlation. This is because SIFT emphasizes orientation of gradients, rather than the position of a particular intensity within the region.

The reason that SA and MS regions are clustered separately is that they cover different and largely independent regions of the scene. Consequently, they may be thought of as different vocabularies for describing the same scene, and thus should have their own word sets. In the same way as one vocabulary might describe architectural features and another the material quality (e.g. defects, weathering) of a building.

IV. Visual indexing using text retrieval methods

In text retrieval each document is represented by a vector of word frequencies. However, it is usual to apply a weighting to the components of this vector [3], rather than use the frequency vector directly for indexing. In the next sub-section we describe the standard weighting that is employed, and the visual analogy of document retrieval to frame retrieval. The following sub-sections then describe the visual analogue of a stop list, and the method used to rank images based on the spatial layout of their visual words.

A. Term frequency–inverse document frequency weighting

The standard weighting [3] is known as 'term frequency–inverse document frequency' (*tf–idf*) and is computed as follows. Suppose there is a vocabulary of V words, then each document is represented by a vector

$$\mathbf{v}_d = (t_1, ..., t_i, ..., t_V)^\top \quad (2)$$

of weighted word frequencies with components

$$t_i = \frac{n_{id}}{n_d} \log \frac{N}{N_i}, \quad (3)$$

where n_{id} is the number of occurrences of word i in document d, n_d is the total number of words in the document d, N_i is the number of documents containing term i, and N is the number of documents in the whole database. The weighting is a product of two terms: the *word frequency*, n_{id}/n_d, and the *inverse document frequency*, $\log N/N_i$. The intuition is that the word frequency weights words occurring more often in a particular document higher (compared to word present/absent), and thus describes it well, whilst the inverse document frequency downweights words that appear often in the database, and therefore do not help to discriminate between different documents.

At the retrieval stage documents are ranked by the normalized scalar product (cosine of angle)

$$\mathrm{sim}(\mathbf{v}_q, \mathbf{v}_d) = \frac{\mathbf{v}_q{}^\top \mathbf{v}_d}{\|\mathbf{v}_q\|_2 \ \|\mathbf{v}_d\|_2} \quad (4)$$

between the query vector $\mathbf{v}_q$ and all document vectors $\mathbf{v}_d$ in the database, where $\|\mathbf{v}\|_2 = \sqrt{\mathbf{v}^\top \mathbf{v}}$ is the L_2 norm of $\mathbf{v}$.

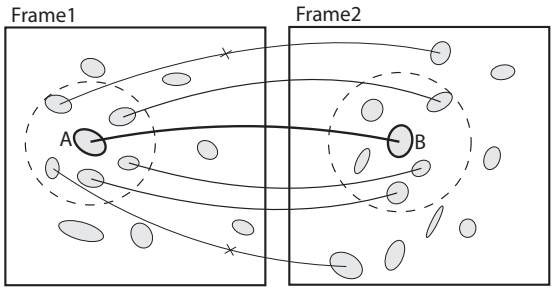


Fig. 3. Illustration of spatial consistency voting. To verify a pair of matching regions (A,B) a circular search area is defined by the k (=5 in this example) spatial nearest neighbours in both frames. Each match which lies within the search areas in both frames casts a vote in support of match (A,B). In this example three supporting matches are found. Matches with no support are rejected.

Note that if document and query vectors are pre-normalized to have unit L_2 norm, then (4) can be rewritten as

$$\mathrm{sim}(\mathbf{v}_q, \mathbf{v}_d) = {\mathbf{v}_q}^\top \mathbf{v}_d = 1 - \frac{1}{2}\|\mathbf{v}_q - \mathbf{v}_d\|_2^2. \qquad (5)$$

As a consequence of (5), sorting documents according to their ascending (squared) L_2 distance to the query vector produces the same ranking as sorting using the (descending) angle score (4).

In our case the query vector is given by the frequencies of visual words contained in a user specified sub-part of an image, weighted by the inverse document frequencies computed on the entire database of frames. Retrieved frames are ranked according to the similarity of their weighted vectors to this query vector.

B. Stop list

Using a stop list analogy the most frequent visual words that occur in almost all images are suppressed. In our case the very common words are large clusters of over 2K points. These might correspond to small specularities (highlights), for example, which occur in many frames. The effect of applying a stop list is evaluated on a set of ground truth queries in section VI.

Figure 4 shows the benefit of imposing a stop list – very common visual words occur in many places in an image and can be responsible for mis-matches. Most of these are removed once the stop list is applied. The removal of the remaining mis-matches is described next.

C. Spatial consistency

Google [6] increases the ranking for documents where the searched for words appear close together in the retrieved texts (measured by word order). This analogy is especially relevant for querying objects by an image, where matched covariant regions in the retrieved frames should have a similar spatial arrangement [34], [35] to those of the outlined region in the query image. The idea is implemented here by first retrieving frames using the weighted frequency vector alone, and then re-ranking them based on a measure of spatial consistency.

Spatial consistency can be measured quite loosely by requiring that neighbouring matches in the query region lie in a surrounding area in the retrieved frame. It can also be measured very strictly by requiring that neighbouring matches have the same spatial layout in the query region and retrieved frame. In our case the matched regions provide the affine transformation between the query and retrieved image so a point to point map is available for this strict measure.

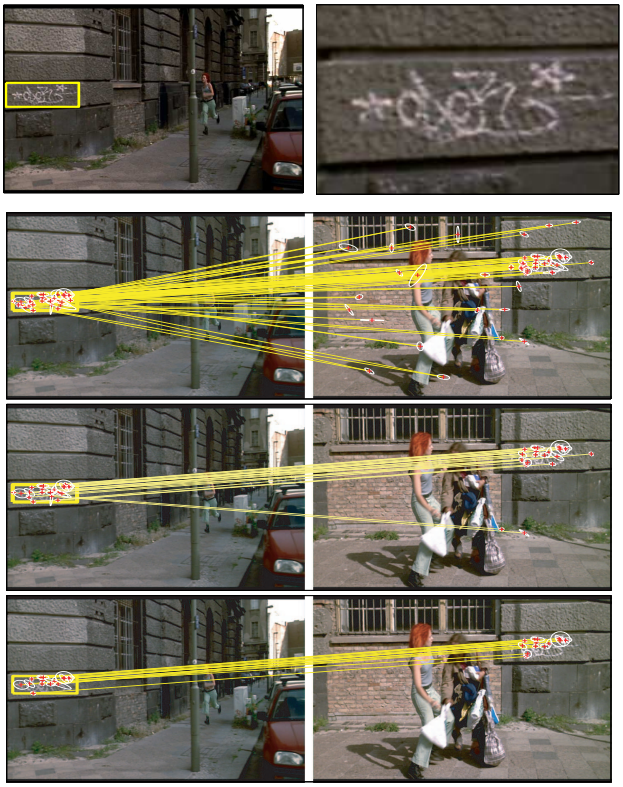


Fig. 4. **Matching stages.** Top row: (left) Query region and (right) its close-up. Second row: Original matches based on visual words. Third row: matches after using the stop-list. Last row: Final set of matches after filtering on spatial consistency.

We have found that a good performance is obtained at the less constrained end of this possible range of measures. A search area is defined by the 15 nearest spatial neighbours of each match in the query and target frames. Each region which also matches within the search areas casts a vote for that frame. Matches with no support are rejected. The spatial consistency voting is illustrated in figure 3. To discount repeated structures, which we found are responsible for many highly ranked false positives, matches with the same visual word label are not allowed to vote for each other, and each match can accumulate at most one vote from one distinct visual word. The final score of the frame is determined by summing the spatial consistency votes, and adding the frequency score $\mathrm{sim}(\mathbf{v}_q, \mathbf{v}_d)$ given by (4). Including the frequency score (which ranges between 0 and 1) disambiguates ranking amongst frames which receive the same number of spatial consistency votes. The object bounding box in the retrieved frame is determined as the rectangular bounding box of the matched regions after the spatial consistency test. This test works well as is demonstrated in the last row of figure 4, which shows the spatial consistency rejection of incorrect matches. The object retrieval examples presented in this paper employ this ranking measure and amply demonstrate its usefulness.

Other measures which take account of the affine mapping between images may be required in some situations, but this involves a greater computational expense. We return to this point in section VII.

1) **Pre-processing (off-line)**

- Detect affine covariant regions in each keyframe of the video. Represent each region by a SIFT descriptor (section II).
- Track the regions through the video and reject unstable regions (section II).
- Build a visual vocabulary by clustering stable regions from a subset of the video. Assign each region descriptor in each keyframe to the nearest cluster centre (section III).
- Remove stop-listed visual words (section IV-B).
- Compute tf–idf weighted document frequency vectors (section IV-A).
- Build the inverted file indexing structure (section V).

2) **At run-time (given a user selected query region)**

- Determine the set of visual words within the query region.
- Retrieve keyframes based on visual word frequencies (section IV-A).
- Re-rank the top $N_s(= 500)$ retrieved keyframes using spatial consistency (section IV-C).

Fig. 5. The Video Google object retrieval algorithm. Example retrieval results are shown in figures 6 and 7.

Fig. 6. **Object query example II: Groundhog Day.** A screenshot of the running object retrieval system showing results of object query 3 from the query set of figure 8. The top part of the screenshot shows an interactive timeline, which allows the user to browse through the retrieved results on that page in a chronological order. The bottom part of the screenshot shows the first seven ranked shots from the first page of retrieved shots. Each shot is displayed by three thumbnails showing (from left to right) the first frame, the matched keyframe with the identified region of interest shown in white, and the last frame of the shot. The precision-recall curve for this query is shown in figure 9.

V. Object retrieval using visual words

We first describe the off-line processing. A feature length film typically has 100K-150K frames. To reduce complexity, roughly one keyframe is used per second of video, which results in 4K-6K keyframes. Descriptors are computed for stable regions in each keyframe (stability is determined by tracking as described in section II). The descriptors are vector quantized using the centres clustered from the training set, i.e. each descriptor is assigned to a visual word. The visual words over all frames are assembled into an inverted file structure where for each word, all occurrences and the position of the word in all frames are stored.

At run-time a user selects a query region, which specifies a set of visual words and their spatial layout. Retrieval then proceeds in two steps: Firstly, a short list of $N_s = 500$ frames are retrieved based on their tf–idf weighted frequency vectors (the bag of words model), and those are then re-ranked using spatial consistency voting. The frequency based ranking is implemented using Matlab's sparse matrix engine and the spatial consistency re-ranking is implemented using the inverted file structure. The entire process is summarized in figure 5 and examples are shown in figures 6 and 7.

It is worth examining the time complexity of this retrieval architecture and comparing it to that of a method that does not vector quantize the descriptors. The huge advantage of the quantization is that all descriptors assigned to the same visual word are considered matched. This means that the burden on the run-time matching is substantially reduced as descriptors have effectively been pre-matched off-line.

In detail, suppose there are N frames, a vocabulary of V visual words, and each frame contains R regions and M distinct visual words. $M < R$ if some regions are represented by the same visual word. Each frame is equivalent to a vector in $\mathbb{R}^V$ with M non-zero entries. Typical values are $N = 10,000$, $V = 20,000$ and $M = 500$. At run-time, the task is to compute the score of (4) between the query frame vector $\mathbf{v}_q$ and each frame vector $\mathbf{v}_d$ in the database (another situation might be to only return the n closest frame vectors). The current implementation exploits sparse coding for efficient search as follows. The vectors are pre-normalized (so that the denominator of (4) is unity), and the computation reduces to one scalar product for each of the N frames. Moreover, only the $m \leq M$ entries which are non-zero in both $\mathbf{v}_q$ and $\mathbf{v}_d$ need to be examined during each scalar product computation (and typically there are far less than R regions in $\mathbf{v}_q$ as only a subpart of a frame specifies the object search). In the worst case if $m = M$ for all documents the time complexity is $O(MN)$.

If vector quantization is *not* used, then two alternative architectures are possible. In the first, the query frame is matched to each frame in turn. In the second, descriptors over all frames are combined into a single search space. As SIFT is used, the dimension, D, of the search space will be 128. In the first case the object search requires finding matches for *each* of the R descriptors of the query frame, and there are R regions in each frame, so there are R searches through R points of dimension D for N frames, a worst case cost of $O(NR^2D)$. In the second case, over all frames there are NR descriptors. Again, to search for the object requires finding matches for *each* of the R descriptors

(a)

(b)

Fig. 7. **Object query example III: Casablanca.** (a) Keyframe with user specified query region (lamp). (b) Screenshot showing the first eight ranked shots. Each shot is displayed by three thumbnails showing (from left to right) the first frame, the matched keyframe with the identified region of interest shown in white, and the last frame of the shot.

in the query image, i.e. R searches through NR points, again resulting in time complexity $O(NR^2D)$.

Consequently, even in the worst case, the vector quantizing architecture is a factor of RD times faster than not quantizing. These worst case complexity results can be improved by using efficient nearest neighbour or approximate nearest neighbour search [18], [28], [36].

Processing requirements: The region detection, description and visual word assignment takes about 20 seconds per frame (720×576 pixels) in this implementation but this is done off-line. Optimized implementations currently run at 5Hz [24]. In terms of memory requirements, the inverted file for the movie 'Groundhog Day' takes about 66MB and stores about 2 million visual word occurrences (this is with the 10% most frequent words removed). For each visual word occurrence we store: (i) the frame number, (ii) the x and y position in the frame and (iii) the distance to the 15th nearest neighbour in the image to define the radius of the search region for spatial consistency re-ranking. For comparison, storing 128-dimensional descriptors in double precision (8 bytes) for two million regions would take about 2GB.

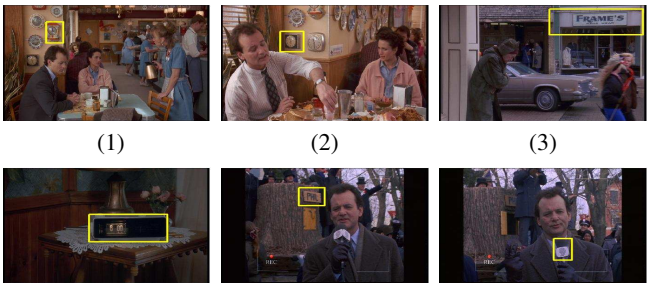
VI. Experiments

Here we evaluate the object retrieval performance over the entire movie on a ground truth test set of six object queries. First, in sections VI-A and VI-B, we introduce the ground truth queries and compare performance and retrieval times with a baseline method implementing standard frame to frame matching (without quantization). In part this retrieval performance assesses the expressiveness of the visual vocabulary, since only about 12% of ground truth keyframes (and the invariant descriptors they contain) were included when clustering to form the vocabulary. In section VI-C we discuss typical failure modes and give a qualitative assessment of retrieval performance. Finally, we study the object retrieval performance with respect to different visual vocabularies (section VI-D), and investigate in depth various frequency ranking and weighting methods (section VI-E).

A. Retrieval performance against a baseline

The performance is compared to a baseline method implementing standard frame to frame matching. The goal is to evaluate the potential loss of performance due to the descriptor quantization. In the baseline method, the same detected regions and descriptors (after the stability check) in each keyframe are used. The detected affine covariant regions within the query area in the query keyframe are sequentially matched to all 5,640 keyframes in the movie. For each keyframe, matches are obtained based on the descriptor values using nearest neighbour matching with a threshold ϵ on the distance. This results in a single or no match between each query descriptor and each keyframe. Euclidean distance is used here. Keyframes are ranked by the number of matches, and shots are ranked by their best scoring keyframes. Note that the baseline method is essentially equivalent to pooling all descriptors from all 5,640 keyframes into a single database and performing an 'ϵ-nearest neighbour search' for each query descriptor. In more detail, the ϵ-nearest neighbour search amounts to finding all points in the database within (Euclidean) distance ϵ of the query descriptor with an additional uniqueness constraint that only the best matching descriptor from each keyframe is retained. This is a type of descriptor matching method used by Schmid and Mohr [35] and later by Lowe [18].

The performance of the proposed method is evaluated on six object queries in the movie Groundhog Day. Figure 8 shows the query frames and corresponding query regions. Ground truth occurrences were manually labelled in all the 5,640 keyframes (752 shots). Retrieval is performed on keyframes as outlined in section IV and each shot of the video is scored by its best scoring keyframe. Performance is measured using a precision-recall plot for each query. Precision is the number of retrieved ground truth shots relative to the total number of shots retrieved. Recall is the number of retrieved ground truth shots relative to the total number of ground truth shots in the movie. Precision-recall plots are shown in figure 9. The results are summarized using the Average Precision (AP) in figure 9. Average Precision is a scalar



Object	# of kfrms	# of shots	# of regions
1 Red clock	138	15	31
2 Black clock	120	13	29
3 Frames sign	92	14	123
4 Digital clock	208	23	97
5 Phil sign	153	29	26
6 Microphone	118	15	19

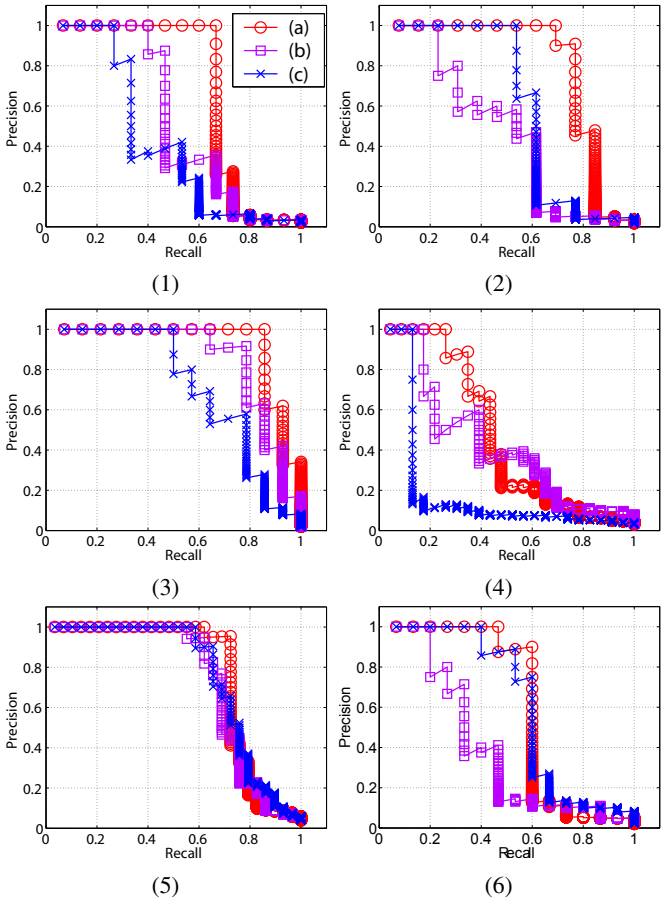
Fig. 8. Query frames with outlined query regions for the six test queries with manually obtained ground truth occurrences in the movie Groundhog Day. The table shows the number of ground truth occurrences (keyframes and shots) and the number of affine covariant regions lying within the query rectangle for each query.

valued measure computed as the area under the precision-recall graph and reflects performance over all recall levels. An ideal precision-recall curve has precision 1 over all recall levels, which corresponds to Average Precision of 1. Note that a precision-recall curve does not have to be monotonically decreasing. To illustrate this, say there are 3 correct shots out of the first 4 retrieved, which corresponds to precision $3/4 = 0.75$. Then, if the next retrieved shot is correct the precision increases to $4/5 = 0.8$.

It is evident that for all queries the average precision of the proposed method exceeds that of using frequency vectors alone – showing the benefits of using the spatial consistency to improve the ranking. On average (across all queries), the tf–idf frequency ranking method performs comparably to the baseline method. This demonstrates that using visual word matching does not result in a significant loss in performance against the standard frame to frame matching. Further examining the precision-recall curves in figure 9 we note that the performance is biased towards high precision at lower recall levels. In practice, this might be acceptable for some applications: for example a visual search of videos/images on the Internet, where the first few correctly retrieved videos/images (and their corresponding web-pages) might contain the relevant information. We note, however, that for some other applications, where finding all instances of an object is important (e.g. surveillance), higher precision at higher recall levels might be preferable.

Figures 1, 6, 10 and 11 show example retrieval results for four object queries for the movie 'Groundhog Day', figure 7 shows a retrieval example for the black and white film 'Casablanca' and figure 12 shows a retrieval example for the movie 'Run Lola Run'. Movies 'Casablanca' and 'Run Lola Run' are represented by 5,749 and 3,768 keyframes, respectively, and a new visual vocabulary was built for each of the two movies, as described in section III.

Figure 13 shows an example of a search by an image from outside the 'closed world' of the film. The image was preprocessed as outlined in section II. Searching for images from other sources opens up the possibility for product placement queries, or searching movies for company logos, or particular buildings



	obj 1	obj 2	obj 3	obj 4	obj 5	obj 6	avg
(a)	0.70	0.81	0.93	0.48	0.77	0.62	**0.72**
(b)	0.55	0.49	0.86	0.43	0.73	0.41	**0.58**
(c)	0.44	0.62	0.72	0.20	0.76	0.62	**0.56**

Average precision (AP) for the six object queries: (a) frequency and spatial ranking, (b) frequency ranking only, (c) baseline

Fig. 9. Precision-recall graphs (at the shot level) for the six ground truth queries on the movie Groundhog Day. Each graph shows three curves corresponding to (a) frequency ranking (tf–idf) followed by spatial consistency re-ranking (circles), (b) frequency ranking (tf–idf) only (squares), and (c) the baseline method implementing standard frame to frame matching (stars). Note the significantly improved precision at lower recalls after spatial consistency re-ranking (a) is applied to the frequency based ranking (b). The table shows average precision (AP) for each ground truth object query for the three different methods. The last column shows mean average precision over all six queries.

or types of vehicles.

B. *Retrieval time*

The average query time for the six ground truth queries on the database of 5,640 keyframes is 0.82 seconds with a Matlab implementation on a 2GHz Pentium. This includes the frequency ranking and spatial consistency re-ranking. The spatial consistency re-ranking is applied only to the top $N_s = 500$ keyframes ranked by the frequency based score. This restriction results in no loss of performance (measured on the set of ground truth queries).

The query time of the baseline matching method on the same database of 5,640 keyframes is about 500 seconds. This timing includes only the nearest neighbour matching performed using linear search. The region detection and description is also done off-line. Note that on this set of queries our proposed method has

Fig. 10. **Object query example IV: Groundhog Day.** (a) Keyframe with user specified query region (Phil sign), (b) close-up of the query region and (c) close-up with affine covariant regions superimposed. (d-g) (first row) keyframes from the 1st, 4th, 10th, and 19th retrieved shots with the identified region of interest, (second row) a close-up of the image, and (third row) a close-up of the image with matched elliptical regions superimposed. The first false positive is ranked 21st. The precision-recall graph for this query is shown in figure 9 (object 5). Querying 5,640 keyframes took 0.64 seconds on a 2GHz Pentium.

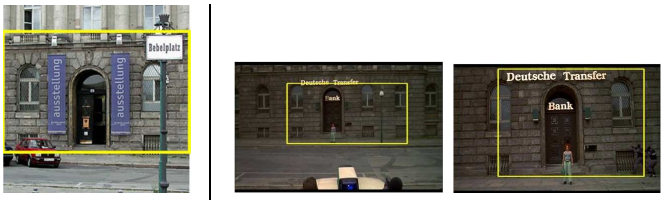


Fig. 13. **Searching for a location in the movie 'Run Lola Run' using an external image downloaded from the Internet.** (a) A query frame. (b),(c) Frames from two shots, ranked 3 and 7, correctly retrieved from the movie. All three images show the same building – a museum in Berlin, which was redesigned to look like a bank in the movie. In the first 20 retrieved shots there are 3 correct matches (ranked 3, 7 and 11).

Fig. 14. Examples of missed (low ranked) detections for objects 1, 2 and 4 from figure 8. In the left image the two clocks (objects 1 and 2) are imaged from an extreme viewing angle and are barely visible – the red clock (object 2) is partially out of view. In the right image the digital clock (object 4) is imaged at a small scale and significantly motion blurred. Examples shown here were also low ranked by the baseline method.

achieved about 600-fold speed-up compared to the baseline linear search.

C. Qualitative assessment of performance

Examples of frames from low ranked shots are shown in figure 14. Appearance changes due to extreme viewing angles, large scale changes and significant motion blur affect the process of extracting and matching affine covariant regions. The examples shown represent a significant challenge to the current object matching method.

Currently, the search is biased towards (lightly) textured regions which are repeatably detected by the applied affine covariant region detectors [22]. Examples of challenging object searches are shown in figure 15.

Typical failure modes include the following cases: (i) no regions are detected on the query object (e.g. object 'A' in figure 15). Such queries return no results. (ii) Extracted affine regions (and descriptors) generalize only over a very limited range of viewpoint and lighting variations. This typically happens when affine regions are extracted on object boundaries (with depth discontinuities) and therefore include other objects or background, or when affine regions are detected on image structures arising from lighting effects such as specular reflections or shadows. An example is shown in figure 15, object 'B' (coffee pot). Other examples include textureless (bottles, mugs) or thin and wiry objects (bicycles, chairs). (iii) Extracted affine regions are highly unstable (e.g. change/disappear over time). Examples include

Fig. 11. **Object query example V: Groundhog Day.** (a) Keyframe with user specified query region (tie), (b) close-up of the query region and (c) close-up with affine covariant regions superimposed. (d-g) (first row) keyframes from the 1st, 2nd, 4th, and 19th retrieved shots with the identified region of interest, (second row) a close-up of the image, and (third row) a close-up of the image with matched elliptical regions superimposed. The first false positive is ranked 25th. Querying 5,640 keyframes took 0.38 seconds on a 2GHz Pentium.

highly deformable objects such as people's clothing (see object 'C' in figure 15) or unstructured non-rigid objects such as running water or leaves moving in the wind.

The range of searchable objects can be extended by adding other covariant regions (they will define an extended visual vocabulary), for example those of [42]. Including shape and contour based descriptors [5], [23] might enable matching textureless or wiry [7] objects. Finally, an interesting direction is developing specialized visual vocabularies for retrieving instances of object classes, such as a face-specific visual vocabulary for retrieving faces of a particular person in a video [37].

D. Vocabulary investigation

In the following experiments, we vary the parameters of the object retrieval system such as the number of words in the visual vocabulary, the size of the stop-list and the size of the retrieval database.

1) Varying the number of words of the visual vocabulary: The goal here is to evaluate the performance of the proposed object retrieval system for different cardinalities of the visual vocabulary. The visual vocabulary is built as described in section III, and retrieval is performed as outlined in section IV, using both the frequency ranking and spatial consistency re-ranking steps. The top 10% most frequent visual words are stopped. The proportion of SA to MS regions is kept constant (=3/5) throughout the experiments. The results are summarized in figure 16. The best performance is obtained for a visual vocabulary size of 16,000.

The size of the visual vocabulary is clearly an important parameter which affects the retrieval performance. When the number of clusters is too small, the resulting visual words are non-discriminative generating many false positive matches. On the other hand, when the number of clusters is too large, descriptors from the same object/scene region in different images can be assigned (due to e.g. noise) to different clusters generating false negative (missed) matches.

Recently, Nister and Stewenius [24] proposed a visual vocabulary organized in a tree together with a hierarchical scoring scheme, which seems to overcome the difficulty of choosing a particular number of cluster centres.

2) Effect of the stop list: Table I evaluates the effect of varying the size of the stop list on the performance of the proposed object retrieval system (after the spatial consistency re-ranking). The best performance (mean Average Precision 0.72) is obtained when 10% of the most frequent visual words are stopped. This amounts to stopping 1,600 most frequent visual words out of the vocabulary of 16,000. Note that stopping the 1,600 most frequent visual words removes about 1.25 million visual word occurrences (out of the total of about 3.2 million) appearing in the 5,640 keyframes of the movie 'Groundhog Day'.

3) Evaluating generalization performance of the visual vocabulary: To test the generalization performance of the visual vocabulary we evaluate the object retrieval performance on the 5,640 keyframes of 'Groundhog Day' with different visual vocabularies. The results are shown in table II. Visual vocabularies (a) from 'Groundhog Day', and (b) from 'Casablanca', were built

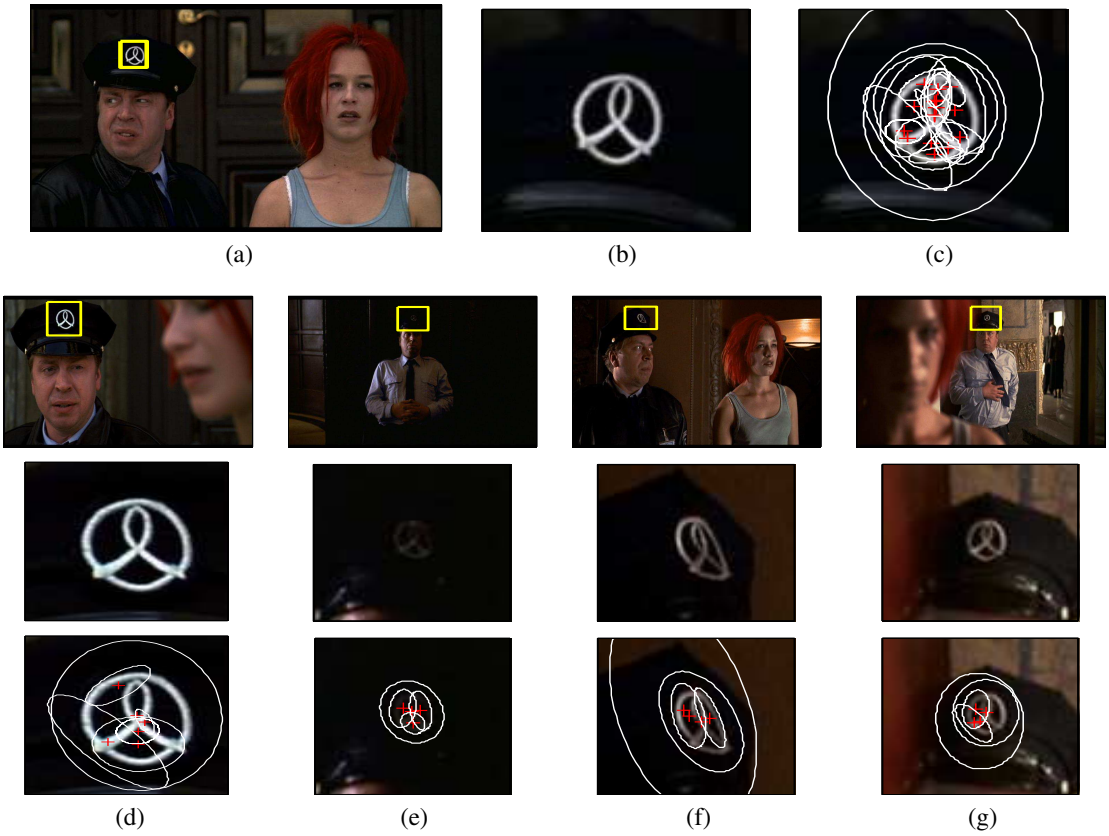


Fig. 12. **Object query example VI: Run Lola Run.** (a) Keyframe with user specified query region (a logo on a hat), (b) close-up of the query region and (c) close-up with affine covariant regions superimposed. (d-g) (first row) keyframes from the 3rd, 9th, 10th, and 11th retrieved shots with the identified region of interest, (second row) a close-up of the image, and (third row) a close-up of the image with matched elliptical regions superimposed. The first false positive is ranked 12th. Querying 3,768 keyframes took 0.36 seconds on a 2GHz Pentium.

Visual Vocab.	kfrms	K	Obj. 1	Obj. 2	Obj. 3	Obj. 4	Obj. 5	Obj. 6	Average
(a) Ghd.	474	16,000	0.70	0.78	0.94	0.46	0.78	0.62	**0.71**
(b) Casa.	483	16,000	0.25	0.31	0.47	0.20	0.48	0.15	**0.31**
(c) Casa.+Ghd.	474, 483	32,000	0.58	0.92	0.93	0.51	0.69	0.53	**0.69**

TABLE II

Generalization performance. PERFORMANCE FOR OBJECT RETRIEVAL ON **5,640** KEYFRAMES OF 'GROUNDHOG DAY' WITH RESPECT TO DIFFERENT VISUAL VOCABULARIES. (A) VISUAL VOCABULARY OF 16,000 VISUAL WORDS BUILT FROM 474 KEYFRAMES OF 'GROUNDHOG DAY'. (B) VISUAL VOCABULARY OF 16,000 VISUAL WORDS BUILT FROM 483 KEYFRAMES OF 'CASABLANCA'. (C) VISUAL VOCABULARY OF 32,000 VISUAL WORDS OBTAINED BY CONCATENATING VISUAL VOCABULARIES (A) AND (B). PERFORMANCE IS MEASURED BY AVERAGE PRECISION ON THE SIX GROUND TRUTH QUERIES FROM 'GROUNDHOG DAY' SHOWN IN FIGURE 8.

Size of stop list (%)	mean Average Precision
0	0.66
1	0.66
5	0.71
10	**0.72**
20	0.65

TABLE I

Effect of the stop list. MEAN AVERAGE PRECISION OF THE PROPOSED OBJECT RETRIEVAL METHOD WITH THE VARYING SIZE OF STOP LIST. THE MEAN AVERAGE PRECISION IS COMPUTED OVER THE SIX GROUND TRUTH OBJECT QUERIES FROM FIGURE 8. THE VOCABULARY SIZE IS 16,000 VISUAL WORDS.

as outlined in section III, i.e. vector quantization was performed only within frames of one movie. Visual vocabulary (c) was obtained by concatenating visual vocabularies (a) and (b). Using the visual vocabulary built from 'Casablanca' (b) for retrieval in 'Groundhog Day' results in a performance drop in comparison to the performance of the 'Groundhog Day' vocabulary (a). On the other hand, case (c), simple concatenation of vocabularies (a) and (b), brings the performance almost to the original level (a). Note that in all three cases, (a)-(c), the top 5% most frequent visual words are stopped. Using the 10% stop-list lowers the performance (measured by the mean average precision) of vocabulary (b) and (c). This might be attributed to higher importance of more general (and more common) visual words in this case.

4) Increasing the database size: Here we test the retrieval performance on a larger database composed of 11,389 keyframes from the two movies 'Groundhog Day' (5,640 keyframes) and 'Casablanca' (5,749 keyframes). The same ground truth set of queries from the movie 'Groundhog Day' (figure 8) is used here but the additional keyframes from 'Casablanca' act as distractors potentially lowering the precision/recall of the retrieved results.

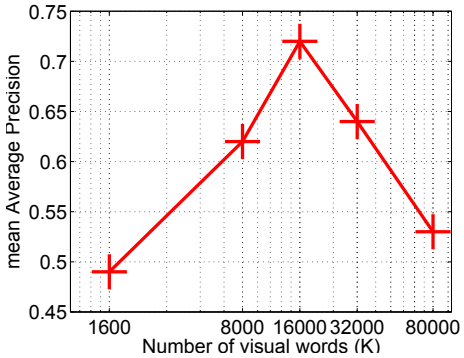
Fig. 15. **Examples of challenging object searches.** (a) Keyframe from the movie 'Groundhog Day' with three query regions (denoted A,B,C). (b) The same keyframe with affine covariant regions superimposed. (c,e,g) Query region close-ups. (d,f,h) Query close-ups with affine covariant regions superimposed. (i-l) Example retrievals for object B (i-j) and object C (k-l). Each column shows (top) a keyframe from the retrieved shot with the identified region of interest, (middle) a close-up of the image, and (bottom) a close-up of the image with matched affine regions superimposed. **Query analysis:** Query A (plain wall) does not contain any visual words and hence returns no results. Query B (coffee pot) retrieves two shots taken from very similar viewpoints (the 2nd ranked is shown in (i)). Other shots are not retrieved as affine regions extracted on the object either include background or cover specular reflections, which change with viewpoint and lighting conditions. The first false positive (ranked 3rd) is shown in (j). Query C (the white shirt) retrieves three correct shots (the 2nd ranked is shown in (k)) but most of the matches are on the background object. This is because affine regions on the shirt are detected on creases, which change as the person moves. The first false positive (ranked 4th) is shown in (l).

Visual Vocab.	kfrms	K	Obj. 1	Obj. 2	Obj. 3	Obj. 4	Obj. 5	Obj. 6	Average
(a) Ghd.	474+0	16,000	0.62	0.61	0.92	0.35	0.71	0.54	0.63
(b) Ghd.+Casa.	474+483	16,000	0.49	0.56	0.85	0.37	0.68	0.48	0.56
(c) Ghd.+Casa.	474+483	24,000	0.63	0.50	0.76	0.39	0.73	0.51	0.59
(d) Ghd.+Casa.	474+483	32,000	0.61	0.65	0.84	0.52	0.80	0.54	**0.66**
(e) Ghd.+Casa.	474+483	40,000	0.68	0.59	0.78	0.51	0.79	0.49	0.64

TABLE III

Increasing the database size. PERFORMANCE FOR OBJECT RETRIEVAL ON A DATABASE OF **11,389** KEYFRAMES FROM TWO MOVIES ('GROUNDHOG DAY' AND 'CASABLANCA') WITH RESPECT TO DIFFERENT VISUAL VOCABULARIES. SEE TEXT. PERFORMANCE IS MEASURED BY AVERAGE PRECISION ON THE SIX GROUND TRUTH QUERIES FROM 'GROUNDHOG DAY' SHOWN IN FIGURE 8.

The test was performed with five different visual vocabularies: (a) the original vocabulary of 16,000 visual words computed from 'Groundhog Day'; (b)–(e) vocabularies clustered from 474 'Groundhog Day' keyframes and 483 'Casablanca' keyframes into different vocabulary sizes, varying between 16,000 – 40,000 visual words. Results are summarized in table III. In all cases the top 10% most frequent visual words were stopped. Examining results for the vocabulary (a), we observe that increasing the database



# of visual words (K)	mean Average Precision
1,600	0.49
8,000	0.62
16,000	**0.72**
32,000	0.64
80,000	0.53

Fig. 16. **Changing the vocabulary size.** Performance for object retrieval on **5,640** keyframes of 'Groundhog Day' with respect to vocabularies of different sizes. The graph and table show the mean Average Precision computed over the six ground truth object queries from figure 8.

size by adding the extra distractor keyframes from 'Casablanca' lowers the mean Average Precision from 0.72 to 0.63 (cf figure 9, method (a)). The best performance on the extended database (mean Average Precision 0.66) is achieved for vocabulary (d), where descriptors from 'Groundhog Day' and 'Casablanca' are pooled together and jointly clustered into 32,000 visual words. This suggests that including descriptors from 'Casablanca' in the vocabulary building step is beneficial and reduces confusion between 'Groundhog Day' and 'Casablanca' objects. Again, note that the number of visual words is an important parameter, which significantly influences the final performance. Similar 'quantization' effects were observed on the database composed of only 'Groundhog Day' keyframes (figure 16) but with the best performance at 16,000 visual words.

E. Comparison of term frequency weighting and ranking methods

In this section, we describe alternative term frequency weighting and ranking schemes and compare their performance with the standard tf–idf weighting (described in section IV-A). Performance is evaluated on the ground truth set of queries of figure 8. Spatial consistency is not applied.

1) Freq-L_2: In this method document vectors are formed using only absolute term frequencies,

$$t_i = n_{id}, \tag{6}$$

and query and document vectors are normalized to have unit L_2 norm. Note that starting from relative term frequencies,

$$t_i = \frac{n_{id}}{n_d}, \tag{7}$$

gives the same document vector as starting from absolute term frequencies (6), as the L_2 normalization cancels the n_d term in the denominator of (7). Similarity is computed using the normalized scalar product (4). The reason for including this method is to compare the term frequency weighting with the tf–idf weighting and assess the contribution of the inverse document frequency term.

2) Freq-L_1: In this method document vectors are formed using term frequencies (6) but are normalized to have unit L_1 norm (instead of L_2), $\|\mathbf{v}_q\|_1 = 1$, $\|\mathbf{v}_d\|_1 = 1$, where $\|\mathbf{v}\|_1 = \sum_{i=1}^{V} |t_i|$. Using L_1 normalization is equivalent to using relative term frequencies (7). The similarity score is computed using L_1 distance as

$$1 - \frac{1}{2}\|\mathbf{v}_q - \mathbf{v}_d\|_1. \tag{8}$$

The goal here is to compare the L_1 and L_2 based normalization and similarity score.

3) Freq-χ^2: Here document vectors are treated as normalized histograms (probability distributions) over terms [13], [15], [27], [43], [44], i.e. relative word frequencies (7) are used (vectors are normalized to sum to one). Similarity between two vectors (normalized histograms) is computed using the χ^2 distance [15], [30], [43] as

$$1 - \frac{1}{2}\chi^2(\mathbf{v}_q, \mathbf{v}_d), \tag{9}$$

where

$$\chi^2(\mathbf{v}_q, \mathbf{v}_d) = \sum_{i=1}^{V} \frac{(t_{qi} - t_{di})^2}{(t_{qi} + t_{di})}. \tag{10}$$

4) Freq-KL: As in the 'Freq-χ^2' method above, document vectors are treated as probability distributions over terms, but the dissimilarity score between the query vector and document vectors is computed using the *Kullback–Leibler (KL) divergence* [13], [27], [44]

$$D_{KL}(\mathbf{v}_q \| \mathbf{v}_d) = \sum_{i=1}^{V} t_{qi} \log \frac{t_{qi}}{t_{di}}. \tag{11}$$

Note that the Kullback–Leibler divergence is not symmetric, $D_{KL}(\mathbf{v}_q \| \mathbf{v}_d) \neq D_{KL}(\mathbf{v}_d \| \mathbf{v}_q)$. In particular, note that document terms which are not present in the query have limited effect on the $D_{KL}(\mathbf{v}_q \| \mathbf{v}_d)$ as the corresponding t_{qi} are zero. This is an important difference from the χ^2 distance based ranking (9) as the χ^2 distance is symmetric and penalizes terms which are present in the document vector $\mathbf{v}_d$ and missing in the query vector $\mathbf{v}_q$.

5) tf-idf-KL: In this method document vectors are formed using the tf–idf weighted visual word frequencies (3). Document vectors are then normalized to sum to one and the dissimilarity score between the query vector and document vectors is computed using the KL divergence (11). The goal is to compare performance of this method with the 'Freq-KL' method above and evaluate the contribution of the idf weights.

6) Freq-Bhattacharyya: As above, document vectors are treated as probability distributions over terms, i.e. visual word frequencies (6) are used and query and document vectors are normalized to have unit L_1 norm, $\|\mathbf{v}_q\|_1 = 1$, $\|\mathbf{v}_d\|_1 = 1$. The similarity score between the query vector and document vectors is measured using the Bhattacharyya coefficient [2], [9],

$$B(\mathbf{v}_q, \mathbf{v}_d) = \sum_{i=1}^{V} \sqrt{t_{qi}\, t_{di}}. \tag{12}$$

The Bhattacharyya coefficient can be geometrically interpreted [2], [9] as a cosine of the angle between vectors $\mathbf{u}_q = (\sqrt{t_{q1}}, \ldots, \sqrt{t_{qV}})^\top$ and $\mathbf{u}_d = (\sqrt{t_{d1}}, \ldots, \sqrt{t_{dV}})^\top$. Note that both $\mathbf{u}_q$ and $\mathbf{u}_d$ have unit L_2 norm since $\mathbf{v}_q$ and $\mathbf{v}_d$ have unit L_1 norm.

7) tf-idf-Bhattacharyya: Here document vectors are formed using the tf–idf weighted visual word frequencies (3). Document vectors are then normalized to sum to one and the similarity score between the query vector and document vectors is computed using the Bhattacharyya coefficient (12). The goal is to compare performance of this method with the 'Freq-Bhattacharyya' method above and evaluate the contribution of the idf weights.

8) Binary: Here document vectors are binary, i.e. $t_i = 1$ if the word i is present in the document and zero otherwise. Similarity is measured using the (unnormalized) scalar product $\mathbf{v}_q{}^\top \mathbf{v}_d$. This similarity score simply counts the number of distinct terms in common between the query and the retrieved document. Note that this method can be also viewed as an intersection of binary (un-normalized) histograms, $\mathbf{v}_q$ and $\mathbf{v}_d$.

In addition to the binary vector method described above we introduce four other binary vector based methods: *Binary-*L_2, *Binary-*L_1, *Binary-*χ^2 and *Binary-*KL. These methods are analogous to methods described above, i.e. the same normalization and similarity score is used. The only difference is that the initial document vectors (before normalization) are binary rather than based on term frequencies (6). The reason for including the 'binary' methods is to assess the importance of using term frequencies. Note that the *Binary-Bhattacharyya* method is not included as it produces the same document ranking as the *Binary-*L_2 method.

9) Performance comparison: Precision-recall plots for the different term frequency ranking methods are shown in figure 17. Results are summarized using Average Precision (AP) in the table in figure 17.

The best average performance over all queries (mean AP 0.61) is achieved by the 'tf-idf-Bhattacharyya' frequency ranking method (a), which combines the 'tf–idf' term weighting with the Bhattacharyya ranking score. Relatively high performance (mean AP 0.58–0.60) is also achieved by Kullback-Leibler divergence methods (b,c,d) and the standard 'tf–idf' method (e), described in section IV-A. Considerably worse results (mean AP 0.26–0.40) are obtained using χ^2 (j,k) and L_1 (l,m) distance based methods.

The L_1 (l,m) and χ^2 (j,k) methods perform poorly on queries 1–2 and 5–6. By close inspection of the results we found that this is due to highly ranked false positive images with small total number (10–50) of visual words and only 1–2 visual words common with the query.

Note also the superior performance (measured by the mean Average Precision) of the KL divergence method (c) to the χ^2 method (j). This can be attributed to the asymmetry of the KL divergence as discussed above.

By comparing each frequency method with its corresponding binary method we also note that using term frequencies seems to produce slightly better ((j,k) and (l,m)) or equal ((g,h) and (c,d)) results, measured by the mean Average Precision. The superior performance (measured by the mean AP) of the tf–idf methods (a,b,e) compared with their frequency based counterparts (f,c,g) may be attributed to the positive contribution of the inverse document frequency weighting.

In all the above experiments the top 5% most frequent visual words were stopped. If the 10% stop-list is used, the performance of method (b) goes down slightly to mean average precision 0.59. The performance of methods (a,e) remains the same. Note that methods (a,b,e) use the tf–idf weighting. More interestingly, performance of the other methods (c,d,f–m), which do not use the tf–idf weighting, slightly increases (by on average 0.035). For example, the mean average precision of methods (f) and (g) increases from 0.56 and 0.54 to 0.59 and 0.57, respectively, which makes them comparable to their tf–idf counterparts (a) and (e). This suggests that applying a stop-list has a similar effect to using tf–idf weights. In other words, the inverse document frequency (idf) weighting component might be viewed as a 'soft' stop-list, down-weighting very common visual words. Applying the stop-list, however, has the additional benefit of discarding mismatches (as was illustrated in figure 4), which helps in the spatial consistency re-ranking stage (cf table I), and is also useful for localizing objects in images.

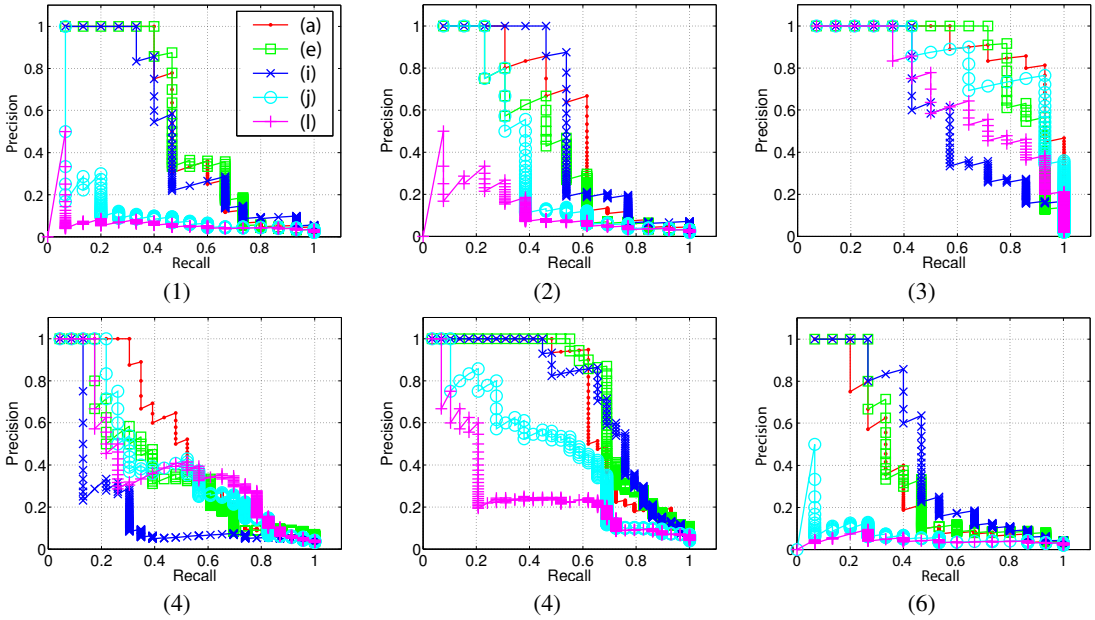
10) Discussion: The proposed object retrieval system uses the normalized scalar product (method (e)) for initial visual word frequency based ranking of video frames, but methods based on Kullback-Leibler divergence and Bhattacharyya coefficient seem to produce similar (or slightly better) results on our ground truth set of test queries. As observed in the text retrieval literature [3], inverse document frequency (idf) weighting consistently improves retrieval performance. Interestingly, the L_1 distance based ranking (method (l)) performs very poorly on our data, which is in contrast with experiments performed by Nister and Stewenius [24] on their dataset. We think this might be attributed to: (i) different statistics of extracted (quantized) visual descriptors and/or (ii) different statistics of the dataset used for experiments. Our dataset contains queries for small objects in highly cluttered and possibly changing background, whereas Nister and Stewenius query mostly by entire images (with some change of camera viewpoint).

VII. Discussion and Conclusions

We have demonstrated a scalable object retrieval architecture, which utilizes a visual vocabulary based on vector-quantized viewpoint invariant descriptors. The vector quantization does not appear to introduce a significant loss in retrieval performance (precision or recall) compared to nearest neighbour matching.

Currently, descriptors are assigned to the nearest cluster centre using linear search. Recently however, efficient search methods using hierarchical tree structured vocabulary [24], vocabulary indexed by randomized trees [28], or descriptor indexing by decision trees [14], [26] have been used. Hierarchical vocabularies [11], [24] can also reduce descriptor quantization effects and can, to some extent, overcome the difficulty with choosing the number of cluster centres.

The spatial consistency re-ranking was shown to be very effective in improving the precision and removing false positive matches. However, the precision could be further improved by a more thorough (and more expensive) verification, based on a stricter measure of spatial similarity (e.g. angular ordering of regions [35], region overlap [10], deformable mesh matching [29], or common affine geometric transformation [18], [28]). Unless the system is being designed solely to retrieve rigid objects, care must be taken not to remove true positive matches on deformable objects, such as people's clothing, by using measures that apply only to rigid geometry. To reduce the computational cost, verification can be implemented as a sequence of progressively more thorough (and more expensive) filtering stages. Spatially verified returns can be used to automatically expand the initial user-given query with additional visual words leading to a significantly improved retrieval performance [8].



	Object 1	Object 2	Object 3	Object 4	Object 5	Object 6	Average
(a) tf-idf-Bhatta	0.54	0.58	0.91	0.53	0.71	0.37	**0.61**
(b) tf-idf-KL	0.67	0.59	0.72	0.24	0.78	0.57	**0.60**
(c) frq-KL	0.66	0.61	0.67	0.24	0.77	0.56	**0.59**
(d) bin-KL	0.66	0.61	0.67	0.24	0.76	0.57	**0.59**
(e) frq-tf-idf	0.55	0.49	0.86	0.43	0.73	0.41	**0.58**
(f) frq-Bhatta	0.50	0.57	0.86	0.44	0.68	0.33	**0.56**
(g) frq-L_2	0.57	0.49	0.77	0.39	0.70	0.31	**0.54**
(h) bin-L_2	0.45	0.60	0.87	0.38	0.63	0.30	**0.54**
(i) bin	0.51	0.59	0.63	0.23	0.74	0.49	**0.53**
(j) frq-χ^2	0.16	0.38	0.86	0.44	0.49	0.09	**0.40**
(k) bin-χ^2	0.17	0.27	0.82	0.43	0.47	0.09	**0.37**
(l) frq-L_1	0.09	0.16	0.71	0.43	0.30	0.05	**0.29**
(m) bin-L_1	0.11	0.10	0.61	0.39	0.29	0.05	**0.26**

Average precision (AP) for the six object queries.

Fig. 17. **Comparison of frequency ranking methods.** Precision-recall graphs (at the shot level) for the six ground truth queries on the movie Groundhog Day comparing performance of different term frequency ranking methods. The table shows average precision (AP) for each ground truth object query. The last column shows mean average precision over all six queries. Note that precision–recall graphs are shown only for methods (a), (e), (i), (j) and (l) from the table, so that the curves are visible.

The method in this paper allows retrieval for a particular visual aspect of an object. However, temporal information within a shot may be used to group visual aspects, and enable object level retrieval [32], [38].

It's worth noting some differences between document retrieval using a bag-of-words, and frame retrieval using a bag-of-visual-words: (i) because visual features overlap in the image, some spatial information is implicitly preserved (i.e. randomly shuffling bits of the image around will almost certainly change the bag-of-visual-words description). This is in contrast to the bag-of-words representation of text, where all spatial information between words (e.g. the word order or proximity) is discarded. (ii) An image query typically contains many more visual words than a text query – as can be seen in figure 8 a query region of a reasonable size may contain 30-100 visual words. However, since the visual words are a result of (imperfect) detection and also might be occluded in other views, only a proportion of the visual words may be expected to match between the query region and target image. This differs from the web-search case where a query is treated as a conjunction, and all words should match in order to retrieve a document/web-page. (iii) Internet search engines exploit cues such as the link structure of the Web [6] and web-page popularity (the number of visitors over some period of time) to compute a static rank [31] of web-pages. This query independent rank provides a general indicator of a quality of a web-page and enables more efficient and in some cases more accurate retrieval. For example, the inverted file index can be ordered by the static rank allowing the retrieval algorithm to access the high quality documents first. An interesting research question would be to develop an analogue to static ranking for video collections.

A live demonstration of the 'Video Google' system on two publicly available movies ('Charade' [Donen, 1963] and 'Dressed to Kill' [Neill, 1946]) is available on-line at [1].

ACKNOWLEDGMENT

We are very grateful for suggestions from and discussions with Mike Brady, Alyosha Efros, Michael Isard, Joe Levy, and David Lowe. We would like to thank James Philbin for developing the user interface. This work was funded by the Mathematical and Physical Sciences Division of the University of Oxford and EC Project Vibes.

REFERENCES

[1] http://www.robots.ox.ac.uk/∼vgg/research/vgoogle/.

[2] F. Aherne, N. Thacker, and P. Rockett. The Bhattacharyya metric as an absolute similarity measure for frequency coded data. *Kybernetika*, 34(4):363–368, 1998.

[3] R. Baeza-Yates and B. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press, ISBN: 020139829, 1999.

[4] A. Baumberg. Reliable feature matching across widely separated views. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 774–781, 2000.

[5] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(24):509–522, 2002.

[6] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In *7th Int. WWW Conference*, 1998.

[7] O. Carmichael and M. Hebert. Shape-based recognition of wiry objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(12):1537–1552, 2004.

[8] O. Chum, J. Philbin, J. Sivic, M. Isard, and A. Zisserman. Total recall: Automatic query expansion with a generative feature model for object retrieval. In *Proceedings of the International Conference on Computer Vision*, 2007.

[9] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):564–575, 2003.

[10] V. Ferrari, T. Tuytelaars, and L. Van Gool. Simultaneous object recognition and segmentation by image exploration. In *Proceedings of the European Conference on Computer Vision*, volume 1, pages 40–54, 2004.

[11] K. Grauman and T. Darrell. The pyramid match kernel: Discriminative classification with sets of image features. In *Proceedings of the International Conference on Computer Vision*, pages I:357–364, October 2005.

[12] C. G. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of the 4th Alvey Vision Conference, Manchester*, pages 147–151, 1988.

[13] J. Lafferty and C. Zhai. Document language models, query models, and risk minimization for information retrieval. In *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 111–119, New York, NY, USA, 2001. ACM Press.

[14] V. Lepetit, P. Lagger, and P. Fua. Randomized trees for real-time keypoint recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages II:775–781, 2005.

[15] T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *International Journal of Computer Vision*, 43(1):29–44, June 2001.

[16] T. Lindeberg and J. Gårding. Shape-adapted smoothing in estimation of 3-D depth cues from affine distortions of local 2-D brightness structure. In *Proceedings of the 3rd European Conference on Computer Vision, Stockholm, Sweden*, LNCS 800, pages 389–400, May 1994.

[17] D. Lowe. Local feature view clustering for 3D object recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Kauai, Hawaii*, pages 682–688. Springer, December 2001.

[18] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[19] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *Proceedings of the British Machine Vision Conference*, pages 384–393, 2002.

[20] K. Mikolajczyk and C. Schmid. An affine invariant interest point detector. In *Proceedings of the 7th European Conference on Computer Vision, Copenhagen, Denmark*, pages I:128–142. Springer-Verlag, 2002.

[21] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages II:257–263, 2003.

[22] K. Mikolajczyk, T. Tuytelaars, C. Schmid, A. Zisserman, J. Matas, F. Schaffalitzky, T. Kadir, and L. Van Gool. A comparison of affine region detectors. *International Journal of Computer Vision*, 65(1–2):43–72, 2005.

[23] K. Mikolajczyk, A. Zisserman, and C. Schmid. Shape recognition with edge-based features. In *Proceedings of the British Machine Vision Conference*, 2003.

[24] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages II:2161–2168, 2006.

[25] S. Obdrzalek and J. Matas. Object recognition using local affine frames on distinguished regions. In *Proceedings of the British Machine Vision Conference*, pages 113–122, 2002.

[26] S. Obdrzalek and J. Matas. Sub-linear indexing for large scale object recognition. In *Proceedings of the British Machine Vision Conference*, 2005.

[27] P. Ogilvie and J. Callan. Language models and structured document retrieval. In *Proceedings of the Initiative for the Evaluation of XML Retrieval Workshop (INEX 2002)*, 2002.

[28] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2007.

[29] J. Pilet, V. Lepetit, and P. Fua. Real-time non-rigid surface detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages I:822–828, June 2005.

[30] W. Press, S. Teukolsky, W. Vetterling, and B. Flannery. *Numerical Recipes in C (2^{nd} Ed.)*. Cambridge University Press, 1992.

[31] M. Richardson, A. Prakash, and E. Brill. Beyond PageRank: Machine learning for static ranking. In *15th International Conference on World Wide Web*, pages 707–715, 2006.

[32] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce. Segmenting, modeling, and matching video clips containing multiple moving objects. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages II:914–921, 2004.

[33] F. Schaffalitzky and A. Zisserman. Multi-view matching for unordered image sets, or "How do I organize my holiday snaps?". In *Proceedings of the 7th European Conference on Computer Vision, Copenhagen, Denmark*, volume 1, pages 414–431. Springer-Verlag, 2002.

[34] F. Schaffalitzky and A. Zisserman. Automated location matching in movies. *Computer Vision and Image Understanding*, 92:236–264, 2003.

[35] C. Schmid and R. Mohr. Local greyvalue invariants for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5):530–534, May 1997.

[36] C. Silpa-Anan and R. Hartley. Localization using an imagemap. In *Australasian Conference on Robotics and Automation*, 2004.

[37] J. Sivic, M. Everingham, and A. Zisserman. Person spotting: video shot retrieval for face sets. In *International Conference on Image and Video Retrieval (CIVR 2005), Singapore*, pages 226–236, 2005.

[38] J. Sivic, F. Schaffalitzky, and A. Zisserman. Object level grouping for video shots. *International Journal of Computer Vision*, 67(2):189–210, 2006.

[39] J. Sivic and A. Zisserman. Video Google: A text retrieval approach to object matching in videos. In *Proceedings of the International Conference on Computer Vision*, pages II:1470–1477, October 2003.

[40] D. M. Squire, W. Müller, H. Müller, and T. Pun. Content-based query of image databases: inspirations from text retrieval. *Pattern Recognition Letters*, 21:1193–1198, 2000.

[41] D. Tell and S. Carlsson. Combining appearance and topology for wide baseline matching. In *Proceedings of the 7th European Conference on Computer Vision, Copenhagen, Denmark*, LNCS 2350, pages 68–81. Springer-Verlag, May 2002.

[42] T. Tuytelaars and L. Van Gool. Wide baseline stereo matching based on local, affinely invariant regions. In *Proceedings of the 11th British Machine Vision Conference, Bristol*, pages 412–425, 2000.

[43] M. Varma and A. Zisserman. A statistical approach to texture classification from single images. *International Journal of Computer Vision*, 62(1–2):61–81, April 2005.

[44] M. Varma and A. Zisserman. Unifying statistical texture classification frameworks. *Image and Vision Computing*, 22(14):1175–1183, 2005.

[45] I. H. Witten, A. Moffat, and T. Bell. *Managing Gigabytes: Compressing and Indexing Documents and Images*. Morgan Kaufmann Publishers, ISBN:1558605703, 1999.