Video Data Mining Using Configurations of Viewpoint Invariant Regions

Josef Sivic and Andrew Zisserman
Robotics Research Group, Department of Engineering Science
University of Oxford
http://www.robots.ox.ac.uk/∼vgg

Abstract

We describe a method for obtaining the principal objects, characters and scenes in a video by measuring the reoccurrence of spatial configurations of viewpoint invariant features. We investigate two aspects of the problem: the scale of the configurations, and the similarity requirements for clustering configurations.

The problem is challenging firstly because an object can undergo substantial changes in imaged appearance throughout a video (due to viewpoint and illumination change, and partial occlusion), and secondly because configurations are detected imperfectly, so that inexact patterns must be matched.

The novelty of the method is that viewpoint invariant features are used to form the configurations, and that efficient methods from the text analysis literature are employed to reduce the matching complexity.

Examples of 'mined' objects are shown for a feature length film and a sitcom.

1. Introduction

The objective of this work is to extract significant objects, characters and scenes in a video by determining the frequency of occurrence of spatial configurations. The intuition is that spatial configurations that have a high rank will correspond to these significant objects. For example, the principal actors will be mined because the spatial configuration corresponding to their face or clothes will appear often throughout a film. Similarly, a particular set or scene that reoccurs (e.g. the flashbacks to Rick's cafe in Paris in 'Casablanca') will be ranked higher than those that only occur infrequently (e.g. a particular tree by the highway in a road movie).

There are a number of reasons why it is useful to have commonly occurring objects/characters/scenes for various applications. First, they provide entry points for visual search in videos or image databases (this is the "page zero" problem in image retrieval systems). Second, they can be used in forming video summaries – the basic elements of a summary will often involve the commonly occurring objects [2, 7, 26] and these are then displayed as a storyboard. A third application area is in detecting product placements in a film – where frequently occurring logos or labels will be prominent.

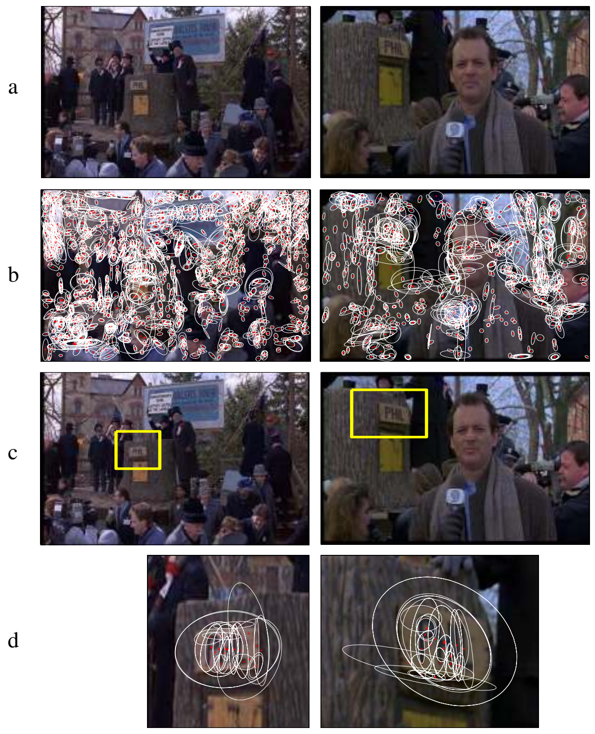


Figure 1: (a) Two frames from the movie 'Groundhog Day'. (b) The two frames with detected affine co-variant regions superimposed. (c) An example of a scene region that has been automatically 'mined' because it occurs frequently throughout the movie. This particular region is detected in 22 shots. (d) Close-up of the region with affine co-variant regions superimposed. A subset of these ellipses correspond, and it is this correspondence that supports this particular cluster.

Data mining, or knowledge discovery, in large databases is a well established research pursuit, particularly for text databases. The aim is to identify previously unknown, valid, novel, potentially useful, and understandable patterns in the database [8]. Even in the case of text this is seen as non-trivial. However, text has the advantages of having a grammar and sentences. This gives a natural granularity to the task – documents can be clustered for example on co-occurring words within sentences.

The visual task is substantially more challenging. First, there is no natural segmentation into sentences, indeed there is not even a natural ordering of an image. A solution to this problem in natural language analysis is to use a sliding window [14] to measure word co-occurrence. In this paper we borrow the idea of a sliding window, which here becomes a sliding region. A second reason the visual task is challenging is because the visual descriptors may not match (they may be occluded, or not detected) or even mismatched.

Our aim is to identify frequently co-occurring parts of the visual *scene* rather than the image – if an object is imaged at twice the size in one frame as another we would wish to identify these as two instances of the same object, even though the image region covered is very different. For this reason our visual descriptors must be invariant to at least scale, and we will employ descriptors that have affine invariance. An example of a typical cluster that is obtained using the methods of this paper is shown in figure 1.

Previous work has applied clustering methods to detected faces in videos [3, 6] in order to automatically extract the principal cast of a movie. A similar approach could be used to cluster other objects classes that can now be fairly reliably detected, for example cars [1, 4, 23]. However, in the method investigated here spatial configurations are clustered directly, rather than first detecting object classes and then clustering within these classes. Previously co-occurrence clusters have been used to support texture classification and segmentation. For example Schmid [21] and Lazebnik *et al.* [11] clustered co-occurrences of textons and viewpoint invariant descriptors respectively.

In the following sections we first provide a review of the visual descriptors used (section 2) for image representation. We then describe the spatial configuration of these descriptors (section 3), and the method of computing the frequency of occurrence across all frames of the video (section 4). Examples of the resulting clusters are given (in section 5) and we also discuss the issue of assessing ground truth on tasks with this quantity of data.

The method will be illustrated for the feature length film “Groundhog Day” [Ramis, 1993] and an episode from the BBC situation comedy “Fawlty Towers” [‘A Touch of Class’, 1975]. The video is first partitioned into shots using standard methods (colour histograms and motion compensated cross-correlation [12]), and the significance of a cluster will be assessed by the number of shots and keyframes that it covers.

2. Quantized viewpoint invariant descriptors

We build on the work on viewpoint invariant descriptors which has been developed for wide baseline matching [15, 17, 20, 25, 27], object recognition [13, 18, 19], and image/video retrieval [22, 24].

The approach taken in all these cases is to represent an image by a set of overlapping regions, each represented by a vector computed from the region’s appearance. The region segmentation and descriptors are built with a controlled degree of invariance to viewpoint and illumination conditions. Similar descriptors are computed for all images, and matches between image regions across images are then obtained by, for example, nearest neighbour matching of the descriptor vectors, followed by disambiguating using local spatial coherence, or global relationships (such as epipolar geometry). This approach has proven very successful for lightly textured scenes, with robustness up to a five fold change in scale reported in [16].

Affine co-variant regions In this work, two types of affine co-variant regions are computed for each frame. The first is constructed by elliptical shape adaptation about an interest point. The implementation details are given in [17, 20]. The second type of region is constructed using the maximally stable procedure of Matas *et al.* [15] where areas are selected from an intensity watershed image segmentation. 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 1600. An example is shown in figure 1.

Each elliptical affine invariant region is represented by a 128-dimensional vector using the SIFT descriptor developed by Lowe [13]. Combining the SIFT descriptor with affine covariant regions gives region description vectors which are invariant to affine transformations of the image.

Vector quantized descriptors The SIFT descriptors are vector quantized using K-means clustering. The clusters are computed from 474 frames of the video, with about 6K clusters for Shape Adapted regions, and about 10K clusters for Maximally Stable regions. All the descriptors for each frame of the video are assigned to the nearest cluster centre to their SIFT descriptor.

Vector quantizing brings a huge computational advantage because descriptors in the same clusters are considered matched, and no further matching on individual descriptors is then required. Following our previous work [24] we will refer to these vector quantized descriptors as *visual words*.

As in [24] very common and uncommon words are suppressed. The frequency of occurrence of single words

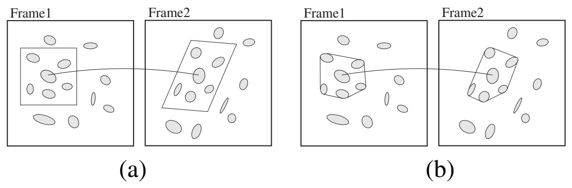


Figure 2: Two definitions of a spatial configuration. (a) An area (square) centred at an affine covariant region. (b) The convex hull of the region's N nearest spatial neighbours. The figures show, for each type of configuration of affine covariant regions, an affine geometric transformation between two frames. Note that in (a) the mapped region is not square, but in (b) the convex hull is mapped to the convex hull. Provided no regions are missing or mismatched (b) is a similarity invariant definition (it is not affine invariant because anisotropic scaling does not preserve relative distances). However, in practice regions are missing and are mismatched. In this work (b) is used.

across the whole video (database) is measured, and the top and bottom 5% are stopped. This step is inspired by a stop-list in text retrieval applications where very common words (such as 'the') and very rare words are discarded. A stop list is very important in our case, since otherwise features (such as specularities) that occur very frequently (in almost every frame) dominate the results.

Final representation The video is represented as a set of key frames, and each key frame is represented by the visual words it contains and their position. This is the representation we use from here on for data mining. The original raw images are not used other than for displaying the mined results. Thus the film is represented by a n_w by n_k matrix `M` where n_w is the number of visual words (the vocabulary) and n_k the number of key frames. Each entry of `M` specifies the number of times the word appears in that frame.

3. Spatial configuration definition

We wish to determine the frequency of occurrence of spatial configurations in scene space throughout the video. This immediately raises two questions: (1) what constitutes a spatial configuration? i.e. the neighbourhood structure and extent; and (2) what constitutes a viewpoint invariant match of a spatial configuration across frames?

For example, one natural definition would be to start from a particular detected elliptical region $\mathbf{p}$ in one frame and define the neighbourhood as all detected regions within an area (a square say) centred on $\mathbf{p}$. The size of the square determines the scale of the configuration, and the neighbours of $\mathbf{p}$. In other frames detected elliptical regions matching $\mathbf{p}$ are determined, and a match between $\mathbf{p}$ and $\mathbf{p}'$ in a second frame also determines the 2D affine transformation between the regions. This affine transformation can then be used to map the square surrounding $\mathbf{p}$ to its corresponding parallelogram in the second frame, and thereby determines the neighbours of $\mathbf{p}'$ in the second frame as those elliptical regions lying inside the parallelogram. The two neighbourhoods could be deemed matched if the affine transformation maps all the elliptical neighbours of $\mathbf{p}$ onto corresponding elliptical neighbours of $\mathbf{p}'$. These definitions are illustrated in figure 2(a).

There are a number of problems with such a strict requirement for matching. Foremost is that many of the neighbours may not match for a variety of reasons including: (i) they are not detected in the second frame due to feature detection problems or occlusion, or (ii) they are not mapped by an affine transformation because they lie on a non-planar surface or another surface entirely, or (iii) the affine transformation is not sufficiently accurate since it is only estimated from a 'small' local region.

The approach we adopt here is to use the data itself to define the neighbourhood. To be definite the neighbourhood of an elliptical region $\mathbf{p}$ is the convex hull of its N spatial nearest neighbours in the frame (see figure 2). Similarly the neighbourhood of the matching region $\mathbf{p}'$ is the convex hull of its N nearest neighbours. The two configurations are deemed matched if M of the neighbours also match, where usually M is a small fraction of N (e.g. 2 out of 10). The *scale* (extent) of the neighbourhood is governed by N.

These definitions have the advantage of being robust to the errors mentioned above (unstable affine transformation, some neighbours not matching, etc). The apparent disadvantage in the neighbourhood definition is that it is not invariant to changes of scale. For example if the frame of $\mathbf{p}'$ is imaged at higher zoom than that of $\mathbf{p}$, then one might expect that there will be additional elliptical regions detected about $\mathbf{p}'$ because extra textured detail can be resolved. In turn this would mean that the N neighbourhood of $\mathbf{p}'$ will only be a subset of the N neighbourhood of $\mathbf{p}$. However, provided M neighbours of $\mathbf{p}$ are included in this subset then the configurations are still matched.

It might be thought that such a loose definition would give rise to many false positive matches of neighbourhoods, and although these occur, they can be removed with further geometric filtering. An example is that the corresponding regions are required to be in a star graph configuration [9]. Using the relative scale between the matched regions $\mathbf{p}$ and $\mathbf{p}'$ to map the neighbourhood (experiments not included here through lack of space), generates more false positives than the neighbourhood definition above. This is because an overestimation of the scale change maps a small set of neighbours onto a large set, and the chances of some of these matching is then increased. Other examples of geometric filtering are mentioned in the following section. What is most important is not to miss any matches (i.e. no false negatives).

Since the elliptical region descriptors are vector quantized into 'visual words' we are essentially describing each neighbourhood simply as a 'bag of words', where the actual spatial configuration of the words is not significant within the neighbourhood.

In the following section we investigate the frequency of configuration occurrence over a range of scales with $N = 20, 50$, and 100.

4. Implementation

In this section we describe the data structures and algorithms that are used to efficiently compute the frequency of occurrence of the neighbourhoods defined in the previous section.

The algorithm consists of three stages. First, only neighbourhoods occurring in more than a minimum number of keyframes are considered for clustering. This filtering greatly reduces the data and allows us to focus on only significant (frequently occurring) neighbourhoods. Second, significant neighbourhoods are matched by a progressive clustering algorithm. Third, the resulting clusters are merged based both on spatial and temporal overlap.

To avoid prohibitive computational expense, in the first stage neighbourhoods are conditioned on a detected region, and a neighbourhood match is only considered further if this central region is matched. However, the second stage allows neighbourhood matches missed due to non-matched central regions to be recovered.

These stages are now explained in more detail. We will use the particular example of a neighbourhood defined by $N = 20$ descriptors, of which $M = 3$ are required to match. The film is represented by a set of 2,820 keyframes (a keyframe every two seconds).

Neighbourhood representation matrix The N-neighbourhood about each detected region is represented as a (very sparse) m-dimensional vector $\mathbf{x} = (x_1, \ldots, x_i, \ldots, x_m)^\top$, where m is the number of visual words. The vector is binary, i.e. entry x_i is set to 0/1 depending whether visual word i is absent or present within the neighbourhood. Comparing two neighbourhoods i, j can be naturally expressed as a dot product between their corresponding vectors $\mathbf{x}_i{}^\top \mathbf{x}_j$. The value of the dot product is the number of distinct visual words the two neighbourhoods have in common. Note that the binary counting discounts multiple occurrences of a visual word within the neighbourhood. This naturally suppresses (1) repeated structures (such as windows on a building facade), and (2) multiple firings of the feature detector at a point (a known problem [17]). The whole video is than represented by a $m \times n$ matrix $\mathtt{X} = [\mathbf{x}_1 \ldots \mathbf{x}_j \ldots \mathbf{x}_n]$, where m is number of visual words and n is the number of neighbourhoods extracted from the video. Note that both m and n can be quite large, e.g. m is typically 16K-22K and n could be several million. Note that matrix $\mathtt{X}$ is very sparse, roughly 0.002% entries are non-zero in the case of the 20 neighbourhoods.

Stage I: neighbourhood stability filter The goal here is to efficiently extract neighbourhoods occurring in more than a minimum number of keyframes. Similar 'minimum support pruning' techniques are a common practice in the data mining literature [8].

Two neighbourhoods are deemed matched if they have at least M visual words in common, i.e. if the dot product of their corresponding neighbourhood vectors is greater than M. The difficulty is that comparing all neighbourhoods against each other is a $O(n^2)$ problem in the number of neighbourhoods. To reduce the complexity of the matching we use that fact that neighbourhoods are constructed around a central visual word, and therefore only neighbourhoods with the same central visual word need to be considered for matching. This reduces the complexity to $O(\sum_{i=1}^{m} n_i^2)$, where n_i is the number of times the visual word i appears throughout the video.

In the case of the movie 'Groundhog Day' with about 10^6 neighbourhoods the method requires only about 10^8 dot products in comparison to about 10^{12} for the full $O(n^2)$ method. This translates to about 5 minutes running time (implemented using matlab sparse matrix engine on a 2GHz Pentium) in comparison to a month (estimated) for the $O(n^2)$ method.

The potential drawback is that the proposed method relies on the central feature being detected and matched correctly by the appearance quantization. However, this does not pose a significant problem since the neighbourhoods are largely overlapping (a neighbourhood is formed around each elliptical region). Consequently each object is likely to be represented by several overlapping neighbourhoods, which decreases the chance of an object being lost (scored poorly).

The result of the filtering algorithm is a score (vote) for every neighbourhood in all the keyframes of the video. In total there are about 1.2 million neighbourhoods in all the keyframes. Neighbourhoods which have score greater than 10 (are matched in at least ten distinct frames) and occur in more than one shot are kept. This reduces the data to about 55,000 neighbourhoods.

Stage II: clustering the neighbourhoods from the filtering The result of the filtering is that a particular neighbourhood will in general be multiply represented. For example, if a word $\mathbf{p}$ occurs in one frame, and corresponds to a word $\mathbf{p}'$ in another frame, then there will be a neighbourhood based around both $\mathbf{p}$ and $\mathbf{p}'$ because the filtering considered every word in each frame.

To merge these repeated neighbourhoods we carry out a greedy progressive clustering algorithm guided by the scores computed in the filtering stage. The algorithm starts at the neighbourhood with the highest score and finds all the neighbourhoods which have at least M words in common. The matching neighbourhoods are combined into a cluster and removed from the data set. This is repeated until no neighbourhoods remain. If several neighbourhoods match within one frame only the best matching one is extracted. At this stage the match on the central region of the neighbourhood is not required (as long as at least M other regions within the neighbourhood match).

The similarity threshold M controls how 'tight' the resulting clusters are. It M is too low clusters contain mismatches. If M is too high the data is partitioned into a large number of small clusters where neighbourhoods are typically found only within one shot. Here $M = 3$.

The advantage of the greedy algorithm over K-clustering, e.g. K-medoids [10], algorithms is that we do not have to specify the number of clusters K, which would be difficult to guess in advance. In contrast to the standard progressive clustering which is initialized at random starting points the current algorithm is guided by the similarity score computed in the filtering stage.

This clustering stage results typically in several thousand clusters.

Stage III: spatial-temporal cluster growing In the previous clustering stage each neighbourhood is allowed to have at most one match in each frame, which typically gives several 'parallel' clusters which have matches in the same keyframes, e.g. neighbourhoods centred on the left eye and right eye of the same person. Here the task is to identify and merge such clusters. Starting from the largest cluster, clusters which have temporal overlap for a certain proportion of keyframes and spatially share at least one region are considered for merging.

A cluster can also have some keyframes missing due to for example mismatched regions which caused the neighbourhood to have low occurrence score. Therefore we also attempt a temporal extension of clusters into the missing frames. The situation can be imagined as two parallel tubes weaving through the keyframes – the tubes must spatially overlap or at least touch each other to be considered for merging, but some parts of one of the tubes are missing. In such cases we examine the vicinity of the neighbourhood which is present in the frame for evidence of the missing neighbourhood.

The examples presented in section 5 are clusters after a single pass of the merging algorithm. After the merging stage we end up with 50-500 clusters depending on the scale. Table 1 summarizes the basic statistics for neighbourhoods and the resulting clusters at the various stages of the algorithm.

Note that expressing neighbourhoods using (sparse) vectors $\mathbf{x}_i$ allows for very efficient computation of certain neighbourhood comparisons, e.g. counting the number of distinct visual words in common, or 'histogram' like comparisons (where proper normalization of $\mathbf{x}_i$ might be needed). On the other hand, such a representation does not allow for efficient computation of operations where position of the regions (or ordering with respect to the central region [25]) needs to be taken into account.

Neighbourhood size N	20	50	100
initial # neighbourhoods	1,190,162	1,190,162	1,190,162
after filtering	55,414	144,631	195,546
initial # of clusters	2,575	5,061	1,769
# of merged clusters	489	350	44

Table 1: Basic statistics for neighbourhoods of different sizes. For the 20-neighbourhood scale the minimum number of key-frames support required in filtering is 10, for the 50- and 100- neighbourhoods it is 5. This stronger constraint results in a smaller number of filtered neighbourhoods for the 20-neighbourhood scale. Examples of final merged clusters are shown in figures 3 and 4.

Object	frm pr	frm rc	sht pr	sht rc
5 'Phil'	0.98	0.87	0.95	0.95
6 'microphone'	0.94	0.64	1.00	0.78
7 'red clock'	1.00	0.98	1.00	1.00
8 'black clock'	0.96	0.77	1.00	0.77
9 'frames'	0.89	0.89	0.83	0.91

Table 2: Precision/recall (pr / rc) measured on keyframes and shots for five mined clusters (obj 5 – obj 9) from figure 3. The ground truth for these five objects was obtained by manually labeling 2,820 keyframes of the movie Groundhog Day. Since the objects are rather small an object was considered present in a frame if it was at least 50% visible, i.e. greater than 50% unoccluded.

5. Examples

Figures 3 and 4 show samples from different clusters found for the scales of 20, 50 and 100 neighbourhood in the movie 'Groundhog Day'. Figure 5 shows samples from clusters found at the 30-neighbourhood scale on the 'Fawlty Towers' episode.

Appraisal. Generally, smaller consistent objects, e.g. faces and logos or objects which change background frequently or get partially occluded, tend to appear at the smaller scale. An example would be the two clocks on the wall in the cafe (objects 7 and 8 of figure 3). Even though they are on the same wall, in some keyframes or shots one of them is out of view or is occluded so that they are mined as two separate clusters at the smaller scale.

An interesting example is the 'frames' shop sign (object 9 of figure 3) which is extracted as a separate cluster at the 20-neighbourhood scale, and can be seen again as a subset of the a 100-neighbourhood scale cluster which covers the

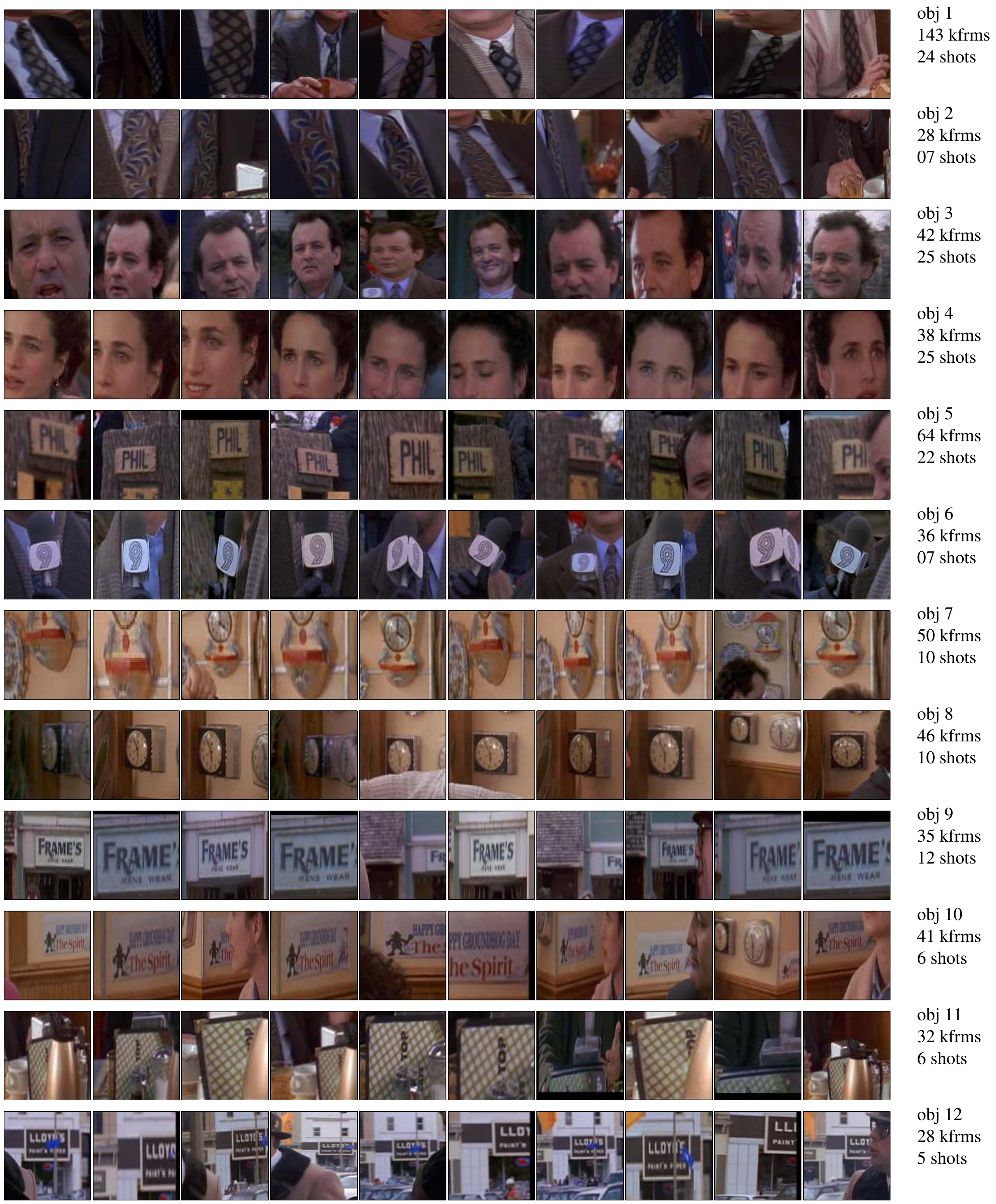
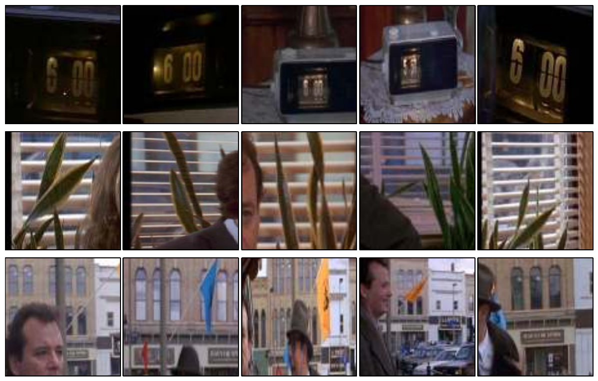


Figure 3: Groundhog Day. Examples of mined clusters at the 20 neighbourhood scale. Each row shows ten samples from one cluster. The first two rows show two different ties of the main character. The next two rows correspond to faces of the two main characters. The remaining rows show various objects that occur often in the movie. The images shown cover a rectangular convex hull of the matched neighbourhoods within the frame plus a margin of 10 pixels. The rectangles are resized to squares for this display.


(a)

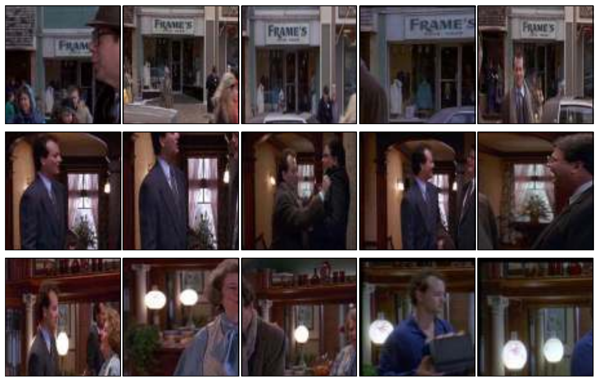

(b)

Figure 4: Groundhog Day. Objects and scenes mined on the scale of (a) 50-neighbourhood and (b) 100-neighbourhood. The clusters extend over (a) 7,21,3 shots, (b) 7,3,5 shots (top-down).

whole shop entrance (row 1 of figure 4b).

Even though the clustering procedure is done carefully so that minimal number of mismatched neighbourhoods get clustered we inevitably have clusters containing 'outliers'. More severe tests to prune out such mismatching neighbourhoods might be necessary. A possibility is to use the alignment procedure [5] to proof check the matches or even propagate existing affine invariant regions to repair misdetections. The expense of such method would not be an issue since they would be applied only within one cluster. It is at this point that other geometric consistency tests can be reintroduced. For example, that all corresponding regions have a similar change in scale between frames.

Comparison with ground truth There are two criteria which could be used to evaluate the results: (1) Were all potential objects mined, (2) If an object has been mined, were all occurrences of this object found. Whereas the second criteria is relatively easy to verify by checking for all occurrences of a mined object in a particular video. The ground truth for the first criteria is much more difficult to establish

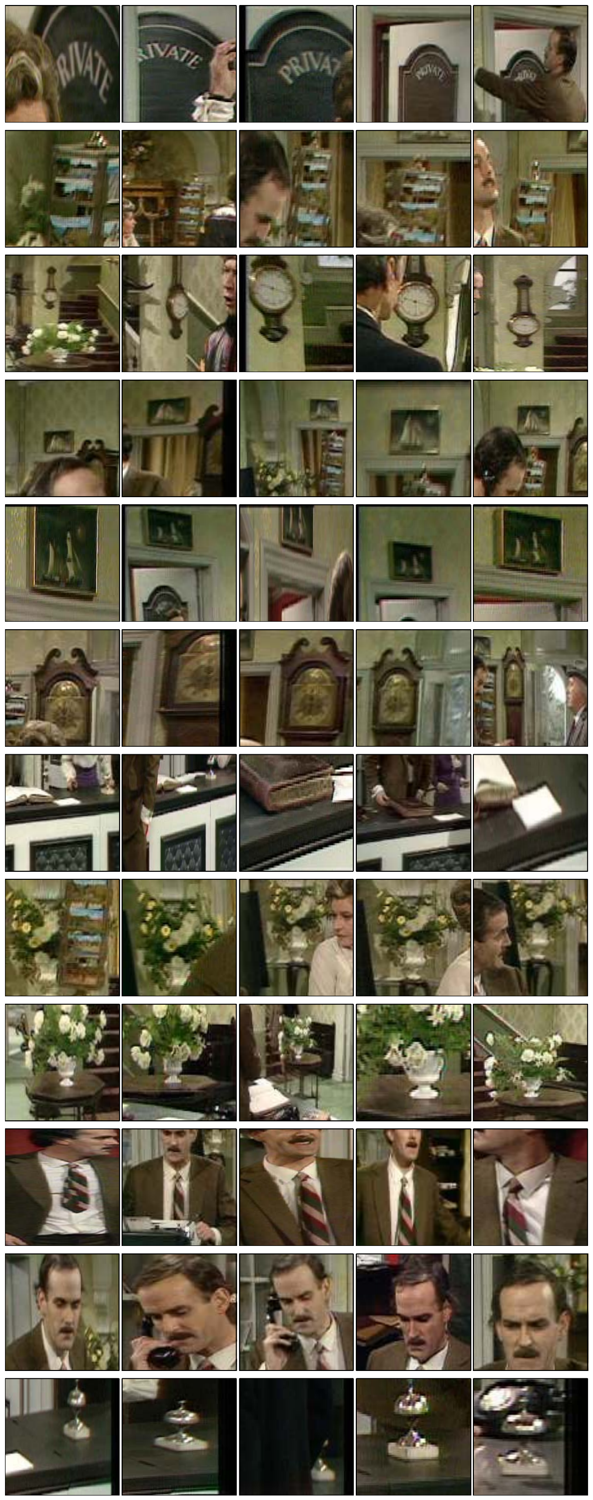


Figure 5: Fawlty Towers. Examples of objects and scenes mined on the scale of 30-neighbourhood in the first episode of the TV series Fawlty Towers. The clusters extend over 48, 43, 19, 28, 13, 26, 24, 19, 14, 27, 19 and 9 shots (top-down).

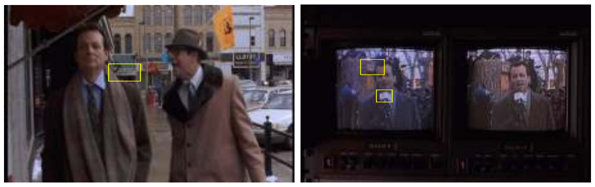


Figure 6: Examples of missed occurrences of objects 9, 5 and 6.

on the whole feature length movie.

To assess the algorithm performance, occurrences of five objects (objects 5-9 from figure 3) were manually marked in the 2,820 keyframe of the movie Groundhog Day. Precision and recall of the corresponding clusters is shown in table 2. Missed occurrences are mostly because of non-detected or mismatched features due to extreme pose/scale changes, or severe defocus. Examples of missed occurrences are shown in figure 6.

Were all potential objects mined? The search is biased towards lightly textured regions that are detectable by the feature detectors used (corner like features, blob like features). We can not tell if a particularly coloured wall-paper occurs often unless it is somewhat textured.

Discovering the faces clusters is surprising since the feature detection methods are not specifically designed to work on faces (or deformable objects). We can not claim to find all occurrences of Bill Murray's face in the whole movie. He appears in a much larger range of poses with a variety of expressions. Also both the clusters contain some mismatches (for other faces, not other objects).

6. Discussion and Future work

We have demonstrated that interesting and salient objects, faces and background scenes can be extracted by clustering on viewpoint invariant configurations. Of course there is room for improvement — currently the search is biased towards textured regions and other regions are missed.

However, we are now at a point where the clustered configurations are of sufficient quality that they may be used as a basis for more extensive co-occurrence (spatial and temporal) exploration.

Acknowledgements We are grateful for discussions with Alyosha Efros, Mark Everingham, Andrew Fitzgibbon and Frederik Schaffalitzky. Funding was provided by EC project CogViSys.

References

[1] S. Agarwal and D. Roth. Learning a sparse representation for object detection. In *Proc. ECCV*, pages 113–130, 2002.

[2] A. Aner and J. R. Kender. Video summaries through mosaic-based shot and scene clustering. In *Proc. ECCV*. Springer-Verlag, 2002.

[3] S. Eickeler, F. Wallhoff, U. Iurgel, and G. Rigoll. Content-Based Indexing of Images and Video Using Face Detection and Recognition Methods. In *ICASSP*, 2001.

[4] R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale-invariant learning. In *Proc. CVPR*, 2003.

[5] V. Ferrari, T. Tutelaars, and L. Van Gool. Wide-baseline multiple-view correspondences. In *Proc. CVPR*, pages 718–725, 2003.

[6] A. W. Fitzgibbon and A. Zisserman. On affine invariant clustering and automatic cast listing in movies. In *Proc. ECCV*, volume 3, pages 304–320. Springer-Verlag, 2002.

[7] Y. Gong and X. Liu. Generating optimal video summaries. In *IEEE Intl. Conf. on Multimedia and Expo (III)*, pages 1559–1562, 2000.

[8] R.J. Hilderman and H.J. Hamilton. *Knowledge discovery and measures of interest*. Kluwer, 2001.

[9] P. Hong and T. Huang. Mining inexact spatial patterns. In *Workshop and Discrete Mathematics and Data Mining*, 2002.

[10] L. Kaufman and P. J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons, NY, USA, 1990.

[11] S. Lazebnik, C. Schmid, and J. Ponce. Affine-invariant local descriptors and neighborhood statistics for texture recognition. In *Proc. ICCV*, 2003.

[12] R. Lienhart. Reliable transition detection in videos: A survey and practitioner's guide. *International Journal of Image and Graphics*, 2001.

[13] D. Lowe. Object recognition from local scale-invariant features. In *Proc. ICCV*, pages 1150–1157, 1999.

[14] C. Manning and H. Schutze. *Foundations of Statistical Natural Language Processing*. MIT Press, 1999.

[15] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *Proc. BMVC.*, pages 384–393, 2002.

[16] K. Mikolajczyk and C. Schmid. Indexing based on scale invariant interest points. In *Proc. ICCV*, 2001.

[17] K. Mikolajczyk and C. Schmid. An affine invariant interest point detector. In *Proc. ECCV*. Springer-Verlag, 2002.

[18] S. Obdrzalek and J. Matas. Object recognition using local affine frames on distinguished regions. In *Proc. BMVC.*, pages 113–122, 2002.

[19] F. Rothganger, S. Lazebnik, C. Schmid, and J. Ponce. 3d object modeling and recognition using affine-invariant patches and multi-view spatial constraints. In *Proc. CVPR*, 2003.

[20] F. Schaffalitzky and A. Zisserman. Multi-view matching for unordered image sets, or "How do I organize my holiday snaps?". In *Proc. ECCV*, volume 1, pages 414–431. Springer-Verlag, 2002.

[21] C. Schmid. Constructing models for content-based image retrieval. In *Proc. CVPR*, volume 2, pages 39–45, 2001.

[22] C. Schmid and R. Mohr. Local greyvalue invariants for image retrieval. *IEEE PAMI*, 19(5):530–534, 1997.

[23] H. Schneiderman and T. Kanade. A statistical method for 3d object detection applied to faces and cars. In *Proc. CVPR*, 2000.

[24] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *Proc. ICCV*, 2003.

[25] D. Tell and S. Carlsson. Combining appearance and topology for wide baseline matching. In *Proc. ECCV*, LNCS 2350, pages 68–81. Springer-Verlag, 2002.

[26] B. Tseng, C.-Y. Lin, and J. R. Smith. Video personalization and summarization system. In *MMSP*, 2002.

[27] T. Tuytelaars and L. Van Gool. Wide baseline stereo matching based on local, affinely invariant regions. In *Proc. BMVC.*, pages 412–425, 2000.