Retrieving Objects, People and Places from a Video Collection: TRECVID'12 Instance Search Task

Christian Schulze
German Research Center for Artificial Intelligence
MADM Group
Kaiserslautern - Germany
Email: christian.schulze@dfki.de

Sebastian Palacio
German Research Center for Artificial Intelligence
MADM Group
Kaiserslautern - Germany
Email: sebastian.palacio@dfki.de

***Abstract*—We participated in 2012's TRECVID instance search task (INS) and wanted to measure how much we can positively impact the performance of a state-of-the-art video retrieval system based on local features and relying solely on content-based retrieval methods. Our agenda consisted in the implementation of some incremental additions to the system that included filtering of local features, custom codebook generation and tailored ranking metrics for the indexed videos. We got three versions of our system tested which iteratively included algorithms that consistently pushed further the performance of the system. Given the terms under which the system had to be implemented – ground-truth was not available–, we used artificially generated datasets to get an idea of how much progress were we making with each additional component. We showed that improvements requiring small computational and human effort can already have positive impacts on the system's performance.**

1) **Runs:**

Nr.	Run ID.	Description
i	F_X_NO_madmDfki_1	Custom codebook with refinement, tie-breaking and relaxed masking.
ii	F_X_NO_madmDfki_2	General codebook with refinement, tie-breaking and relaxed masking.
iii	F_X_NO_madmDfki_3	General codebook without refinement, default scoring and tight masking.

2) **Differences: Run (iii) constitutes the baseline with some fine-tuned parameters. Run (ii) applies some improvements such as match filtering of local descriptors, a re-ranking scheme for videos and relaxation of the queries' masks. Run (i) only differs form run (ii) in that it uses a codebook that was built on the same videos that have been indexed; the other systems were using a generic codebook trained on some independent data.**
3) **Contribution to performance: Just by fine-tuning the parameters of the baseline system, an increase in performance was already noticeable. One important step towards improvement was achieved via the Hough refinement. Some slight improvements were possible by using the custom codebook and the relaxed masking.**
4) **Major findings: Simple additions to a basic video retrieval system are already an important step towards the correct ranking of videos. All achieved with relatively small effort in terms of computational and human effort.**

I. Introduction

Since the advent and posterior spread of custom video consumption through platforms like YouTube™ or Flickr™, the field of video retrieval has been drawing more and more attention given the potential uses and great advantages that such systems could offer in areas like advertising, user-customized video catalogs, filtering, etc. Concretely, the ability to recognize, track and retrieve different visual entities within a corpus of videos has been the subject of numerous investigations [1]–[6]. To study today's performance of such systems, a scenario [7] can be considered where, given a collection of videos and given a query image delimiting a sample of an entity (i.e., a place, person or object), a sorting of the videos can be retrieved such that they are more likely to contain the entity represented in the query image the higher they are ranked.

The first step to tackle this task deals with how to extract information from the videos in a way that it gives a thorough but concise representation. As we will see, several approaches have been devised in order to index and represent videos considering the aforementioned properties. By focusing on the fact that a video is nothing but a sequence of images, a reduction on the amount of data to process can be achieved via frame sub-sampling. The most basic approach is to sub-sample frames at regular intervals. This ensures an even distribution of the contents in a video while keeping processing overhead to a minimum. On the other hand, this approach completely disregards the nature of the contents in the video and yields the issue of setting an ad-hoc sampling step. Another approach would be to rely on shot-boundary detection algorithms [8]–[10] in order to segment the video in a more content-oriented fashion and hence, gain more understanding about its internal structure. This extra step allows a more compressed representations of videos now that their contents can be analyzed and sampling can be made on a per-shot basis. Some other information could also be retrieved from the videos in order to make more informed decisions on how to represent them in a database, namely information about motion [11], background segmentation or 3D geometry [11]–[13]. Although this information can be extracted from the videos, it remains unclear how can it be of any advantage in a scenario such as the one described before.

As we shall see, and based on a framework where videos are being represented as a collection of images, extracting information that allows an indexing system to identify and, later on, match different entities from those videos, can be addressed with the fusion of various algorithms and techniques. Previous work [14], [15] has determined that the use of local descriptors is better suited for object recognition rather than their global counterparts. This raises the question about the criterion under which a region is to be considered of interest or not. Plenty of effort [16], [17] has been put to settle this matter but almost as much effort as with the question regarding the best descriptor to use [18]–[20]. Lots of other additional information have been successfully extracted from the same regions of interest as shown in [21].

Some other important aspects regarding the indexing of images for object retrieval, deals with the structures and metrics to store the extracted features. Such items have been usually addressed by taking what is known to work in the domain of text-retrieval. A brief but comprehensive review of such methods and their adaptation to the image domain can be found in [1]. Finally, it only remains to investigate how to compare the information obtained from a query to determine whether a match is found in the indexed data or not. In [22], [23] some of the commonly used distance measures and their implications when matching similar features are reviewed. Furthermore, additional criteria has been devised in order to filter the amount of matches obtained after such a procedure as shown in [18], [21].

We investigated the performance of some of this methods to retrieve videos that contain specific entities in a consistent and yet scalable way, adapting some of the standard procedures to fit our requirements.

II. Datasets

For TRECVID12's instance search task, a set of FLICKR™ videos were made available in collaboration with the AXES project [1]. These videos have been split into clips of arbitrary length totaling over 70000 files. The videos are, despite of the splitting, assumed to be independent from each other. Within the dataset, a wide range of topics, qualities, lengths, sources and formats can be found (Fig. 1). Although all files have been encoded using the *webm* format, artifacts such as interlacing or very coarse quantization (e.g. in the case of videos originally encoded using MPEG) are still noticeable and pose further challenges for the classification engine.

To evaluate the system, a collection of images was also provided. Each image showed an object, person or place that represented the entity to be retrieved from the videos. An additional binary image mask indicates the shape and location of each entity within the image as shown in Fig. 2. Finally, all query images came with information about what kind of entity were they containing (i.e., object, person or place) as well as an ID of the particular entity they represent (in order to identify which set of query images contained the same entity).

Given the nature of the task, the dataset was lacking any kind of labeling or metadata that allow us to infer any additional information other than the one obtained via the image content itself. We then created a small set of 30 images taken

[1]http://www.axes-project.eu/

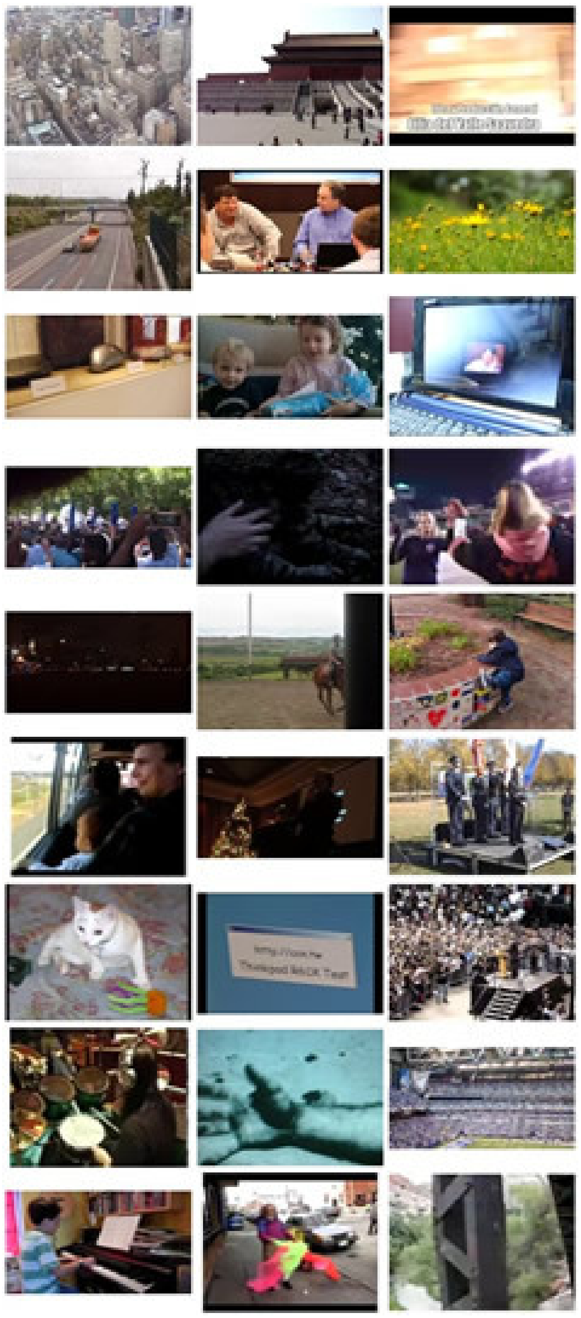

Fig. 1. Random sample of videos from the dataset used for TRECVID12's instance search task. Various sources (amateur, TV), topics and qualities comprise the video collection.

from the already available videos and manually generated the corresponding masks, metadata and links to the videos from which they were taken. This small dataset was intended to serve as a very rough basis to quickly estimate the performance of the system. Additionally, we randomly extracted complete frames from all the videos in the original dataset (along with the ID of the videos they were taken from) to further grasp a sense of performance and to take more informed decisions about the fine-tuning of some of the system's parameters. For this last dataset, random squared crops were taken out of the

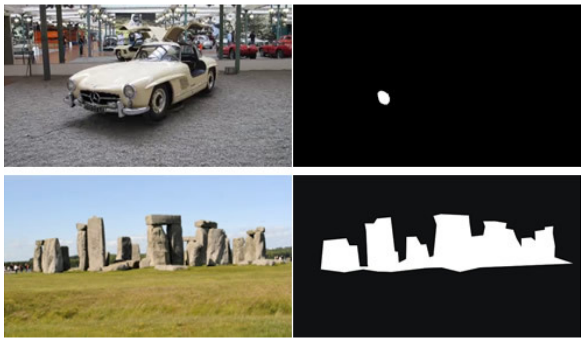

Fig. 2. Sample of the query images. Each image depicted an instance of either a person an object or a place (left). Furthermore each query came with a mask (right) to indicate the position and shape of the entity.

frames to serve as "artificial entities" at querying time.

III. Approach

A. Baseline

The initial setup for our system follows the general pipeline described before and illustrated on Fig. 3. After the basic system was set up, we made several incremental additions in order to improve performance and overcome known issues with some of the methods and algorithms at use. The specifics of each stage of the pipeline is outlined below:

1) Video Sampling: Each video clip gets sampled at regular intervals of 1 second which in average resulted in 10 frames per video. This keeps processing overhead to a minimum and ensures that the video is sampled evenly. For videos being shorter than 1 second the center frame was taken. Since the video clips were mostly about 10s long, cuts between shots are rather unlikely to occur and hence including a shot-boundary detection step will mainly increase the processing cost.

2) Feature Extraction: All resulting frames are passed through an interest point detector based on a mixture of corners and blobs. On one hand, a multi-scale Harris-Laplace corner detection [24] scanned each frame which was then passed on to a Difference of Gaussians blob detector (DoG) [25]. Between the two, an average of 700 interest points were collected per frame. Blobs and corners complement each other and improve the description of the characteristics of entities much better than just by using one method alone. The detected interest points are then described using SIFT [18]. In order to simplify the indexing of the SIFT features, interest points get quantized using a hierarchically generated codebook (using hierarchical K-Means) of one million clusters so that the descriptors end up grouped together depending on the cluster they fall into.

3) Representation and Indexing: The quantized features are passed on to an inverted file structure implemented through *Lucene engine* [26]. Within Lucene, a video v is represented as a document with as many *fields* as interest points have been detected. Each field contains the information about the location of every interest point as well as its SIFT descriptor association to the codebook bin. At the end, a video can be seen as one list of occurring cluster numbers of *all* SIFT descriptors that were extracted from the sampled frames of v.

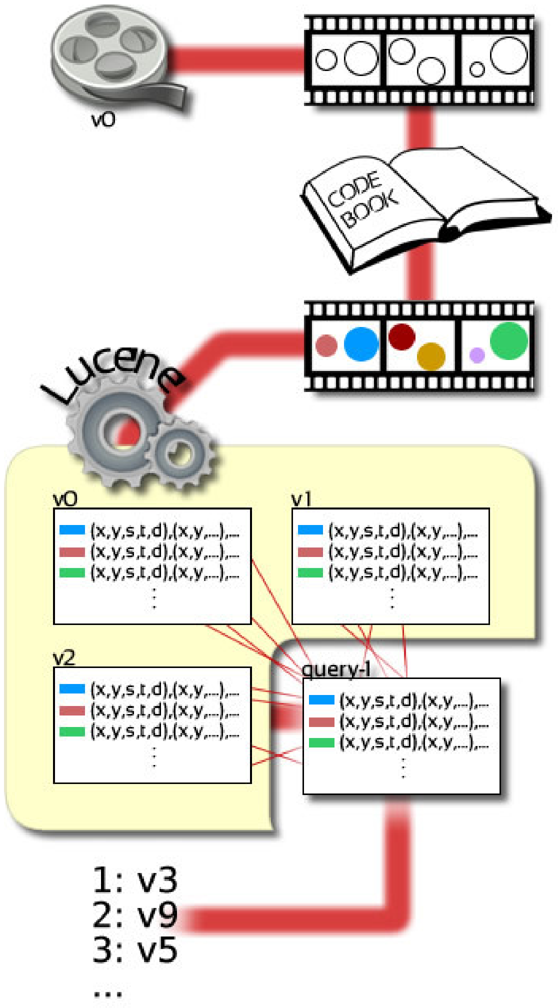

Fig. 3. Basic pipeline of the proposed system. Frames get sampled from each video and all sampled frames are scanned using a corner and blob detectors. Interest points get quantized using a codebook and then get indexed by Lucene's engine in an inverted index. When a query comes, it gets processed the same way a frame would so that similar interest points can be matched against the indexed representations of the videos within Lucene and finally, a ranking of the videos is constructed based on the amount of matches found between the query and the videos.

4) Querying: When a query image comes in, it gets treated as any other video frame, namely when referring to the detection, description and codebook-quantization of local features. Having arrived at this point the mask is taken into account to discard any interest point whose origin lays in the masked-out area. We repeat this process for all queries belonging to the same entity and accumulate them all to create just one query per entity. These union of interest points are passed to Lucene to create a document in a similar fashion as for the videos. A query can now be compared against the videos by finding matches between interest points and then, a ranking of the videos can be generated based on how many corresponding matches were found between the query and the indexed clips.

B. Improvements

Several incremental additions where made to the baseline in order to improve performance and overcome known issues with some of the aforementioned methods and algorithms. All new phases requiring evaluations made use of the two self-generated datasets described in Section II.

1) Parameter Optimization: At first we focused on the fine-tuning of all the parameters related to the interest point detectors. The variables causing most of the impact were, on the side of the Harris corner detection algorithm, the threshold to determine whether a corner was present or not as well as the number of scales to run the detection on. For the DoG blob detector, the threshold that defines a blob and the size of the scale space were carefully followed and modified.

2) Refinement and Tie-Breaking: In order to get rid of undesired and noisy matches between interest points, a *Hough refinement* [27] was applied on the Lucene result such that initial matches with inconsistent scale and orientation (up to some range) were removed prior to ranking the final result. Recall that the main criterion to sort the videos in the general pipeline is the amount of matches between queries and indexed documents. Refinement works by quantizing the shifts in position $\Delta x, \Delta y$ and scale $\Delta\sigma$ as well as the shifts in orientation $\Delta\theta$ between two matching interest points. After classifying all Δs, only the matches with the most frequently occurring position and orientation shifts are kept and the other matches are discarded. This filtering sometimes forces the system to discard too many matches and yield list of videos with the same ranking value. In this case, the original unfiltered ranking based on the amount of raw matches is used as a second instance, tie-breaking criterion.

3) Custom Codebook: Since it is not the aim of the experiment to create a completely general video retrieval system [7], we crafted an additional codebook with one million entries for the quantization of descriptors trained on data extracted from the same videos that were being indexed. This favors the segmentation of the feature space in a way that corresponds better with the distribution of the features taken from the videos.

4) Relaxed Masking: We relaxed the way of filtering out the interest points of a query given its mask as follows: we chose to keep not only the interest points occurring within the unmasked area exclusively but also kept the ones whose area intersected the mask-free region of the query. This way, interest points falling in the edge of the desired entity are safe from being erroneously discarded because of small offsets with respect to the mask.

IV. Results

We conducted three main experiments combining some of the elements mentioned before and compared them with the run of the baseline system.

a) F_X_NO_madmDfki_3: This run established the baseline for the system. After fine-tuning the system's parameters, we ran this experiment using a generic codebook, relying only on Lucene's native scores for matching interest points of queries and videos and omit any kind of refinement, namely relaxed masking and hough refinement.

b) F_X_NO_madmDfki_2: Here we included the hough refinement in the scoring phase of the system along with the tie-breaking criterion based on the original Lucene scores. We also made use of relaxed masking to include interest points detected on the edges of the unmasked entity. The same generic codebook was also used here.

c) F_X_NO_madmDfki_1: The only difference with the previous run is the use of a codebook based on the data extracted from the videos.

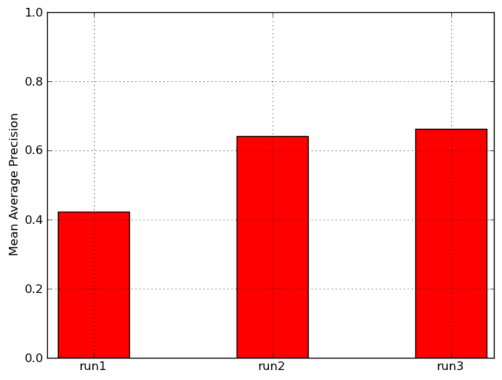

Fig. 4. Performance of the system using the self-generated dataset. `run1` is the basic system, `run2` uses Hough refinement, tie-breaking and relaxed masking; `run3` quantizes interest points with a codebook trained on data from the videos.

All three experiments were also conducted on the artificial randomly cropped dataset described in Section II and which was based on frames extracted from TRECVID's original dataset. Figure 4 summarizes the contributions to performance from the different improvements that were previously proposed. The performance of the systems using TRECVID's queries for evaluation is shown in Fig. 5.

V. Discussion

When comparing the two improved systems against the baseline, we see how they outperform the simpler system (i.e., the one without any of the proposed additions). Only on topic 9056 (Pantheon) the baseline system achieved better performance than their improved counterparts. We argue that this phenomenon occurs because the entity under consideration occupies the whole frame, i.e., had an empty mask, and its geometry can be considered as self-similar (in concrete, the tails in the roof are nothing but concentric rectangles). In this case, the filtering step has a negative impact since it removes matches that do not have a consistent orientation and position displacement with respect to the predominant places and orientations which, in addition to the way the videos pan over the pantheon's dome, make this naive filtering unsuitable for this kind of topic. Despite of that single example, the results in general confirm that the methods included in the improved systems effectively addresses some of the issues of

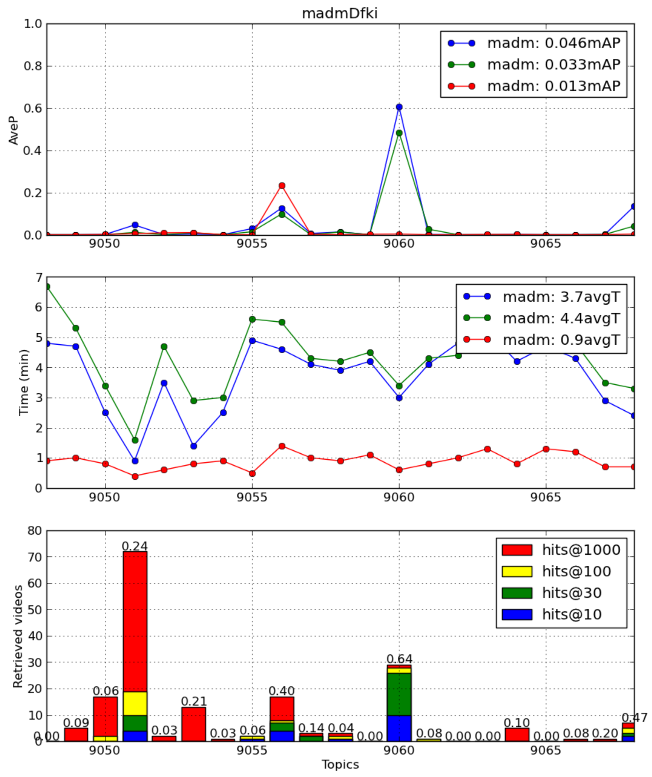

Fig. 5. Results of the system using the queries provided for TRECVID. Above: average precision of the different runs for all topics. The number in the labels correspond to the mean average precision of that run. Middle: time spent processing a topic, i.e., the time it took to match all queries belonging to a single category. Below: number of retrieved true-positives. Each bar shows how many videos were retrieved at different ranks (10, 30, 100, 1000) for the best run based on its mAP. The number above each bar corresponds to the ratio of totally retrieved videos w.r.t. the total amount of videos for that topic that could have been retrieved.

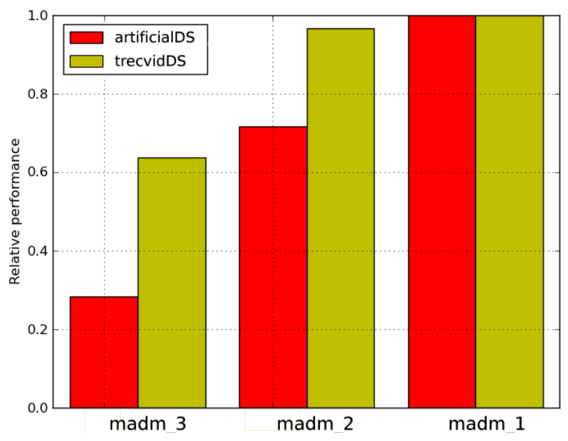

Fig. 6. Comparison of relative performance using the artificial dataset and the official queries provided by the NIST

local feature matching in almost all scenarios. Besides, by looking at the relative improvements (in terms of mean average precision *mAP*) obtained by using the artificially generated datasets and comparing it to the ones obtained with the "real" data (Fig. 6), we see a consistent correlation between the two and therefore, confirming that the use of this self-crafted data served as a reliable indicator for progress. It becomes also clear how such a retrieval system can be significantly enhanced with relatively small effort.

On the other hand, we argue that allowing interest points that only describe a small portion of the visual information, i.e, interest points at high scale, to be part of a video description and given the poor quality of some videos, the ranking gets severely affected since the information extracted from such small regions is insufficient and misleading when compared against some bigger and less noisy ones. This is why we consider that some videos not belonging to any of the listed topics, got highly ranked even for completely different topics. Low resolution videos of nature and crowds often reoccurred within the rankings. Note that in these two cases, the issue of self-similarity may also be another source for misclassification.

It is also important to highlight the fact that, to our impression, there were videos containing regions that were similar to the searched objects without the regions actually being an instance of the actual objects which brings up the rather challenging issue of the semantic gap. Although the queries already came, as pointed out earlier, with some meta-data indicating whether it contained a person, a place or an object, we opt for not using this information at all, since it would require another component to be added to the system which, if not correctly set, would produce more noise than the information that could effectively be taken from it. Besides, one of our initial aims was to test how well a simple system could perform in such a task and embedding semantic interpretation of images is, though quite fascinating, out of the scope of this research.

VI. Conclusion

We investigated the impact of several methods and algorithms on the performance of an instance search video retrieval system applied to a challenging, big and diverse dataset. Our major finding centers in that even by implementing a few improvements over a state-of-the-art baseline system such as custom interest point clustering, filtering of local feature matches and tailored ranking metrics, a significant improvement can be achieved in the way the videos are ranked when searching for particular entities. By fine tuning the parameters of the interest point detectors, codebook quantization, query masking, match refinement and ranking scores, a relative improvement of over 30% can be achieved.

We found that a great source of noise in the fused signals for ranking, was caused by small interest points that were taken as part of a video's description. We argue that these points provide very little discrimination about the region they describe. This holds specially true as the quality of the videos decreases or when describing non-rigid and self-similar objects.

These issues could be avoided in principle by filtering out small sized interest points but at the cost of a reduced ability to search for small or texture-poor objects. This last idea along with some other algorithms including semantic indexing, geometry consistency or the use of other descriptors to further improve the system, remain open for future research.

Acknowledgment

We would like to thank Damian Borth and Adrian Ulges at the DFKI for their extra support while putting together the system as well as their advise in organizational matters regarding TRECVID.

References

[1] 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*, vol. 2, Oct. 2003, pp. 1470–1477. [Online]. Available: http://www.robots.ox.ac.uk/~vgg

[2] J. Sivic, F. Schaffalitzky, and A. Zisserman, "Efficient object retrieval from videos," in *European Signal Processing Conference*, 2004.

[3] S. fu Chang, W. Chen, and H. Sundaram, "Videoq: A fully automated video retrieval system using motion sketches," in *Proc. 4 th IEEE Workshop on Applications of Computer Vision*, 1998, pp. 27–0.

[4] D. Borth, C. Kofler, and A. Ulges, "Smart video buddy - content-based live recommendation," in *Proceedings of the International Conference on Multimedia and Expo*, IEEE. IEEE, 7 2011.

[5] H. K. Ekenel, T. Semela, and R. Stiefelhagen, "Content-based video genre classification using multiple cues," in *Proceedings of the 3rd international workshop on Automated information extraction in media production*, ser. AIEMPro '10. New York, NY, USA: ACM, 2010, pp. 21–26. [Online]. Available: http://doi.acm.org/10.1145/1877850.1877858

[6] D. Borth, J. Hees, M. Koch, A. Ulges, C. Schulze, T. M. Breuel, and R. Paredes, "Tubefiler: an automatic web video categorizer," in *Proceedings of the 17th International Conference on Multimedia 2009, Vancouver, British Columbia, Canada, October 19-24, 2009*, W. Gao, Y. Rui, A. Hanjalic, C. Xu, E. G. Steinbach, A. El-Saddik, and M. X. Zhou, Eds. ACM, 2009, pp. 1111–1112.

[7] A. F. Smeaton, P. Over, and W. Kraaij, "Evaluation campaigns and trecvid," in *MIR '06: Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval*. New York, NY, USA: ACM Press, 2006, pp. 321–330.

[8] A. Dailianas, R. B. Allen, and P. England, "Comparison of automatic video segmentation algorithms," in *In SPIE Photonics West*, 1995, pp. 2–16.

[9] J. S. Boreczky and L. A. Rowe, "Comparison of video shot boundary detection techniques," 1996, pp. 170–179.

[10] R. Lienhart, "Comparison of automatic shot boundary detection algorithms," 1999, pp. 290–301.

[11] B. K. P. Horn and B. G. Schunck, "Determining optical flow," *ARTIFICAL INTELLIGENCE*, vol. 17, pp. 185–203, 1981.

[12] A. G. Bors and I. Pitas, "Optical flow estimation and moving object segmentation based on median radial basis function network," *IEEE Trans. on Image Processing*, vol. 7, pp. 693–702, 1998.

[13] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," 1981, pp. 674–679.

[14] T. Deselaers, D. Keysers, and H. Ney, "Features for image retrieval: an experimental comparison," *Inf. Retr.*, vol. 11, no. 2, pp. 77–107, Apr. 2008. [Online]. Available: http://dx.doi.org/10.1007/s10791-007-9039-3

[15] ——, "Features for image retrieval - a quantitative comparison," in *In DAGM 2004, Pattern Recognition, 26th DAGM Symposium*, 2004, pp. 228–236.

[16] D. Lisin, M. Mattar, M. Blaschko, E. Learned-Miller, and M. Benfield, "Combining local and global image features for object class recognition," in *Computer Vision and Pattern Recognition - Workshops, 2005. CVPR Workshops. IEEE Computer Society Conference on*, june 2005, p. 47.

[17] K. Zagoris, S. A. Chatzichristofis, and A. Arampatzis, "Bag-of-visual-words vs global image descriptors on two-stage multimodal retrieval," in *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*, ser. SIGIR '11. New York, NY, USA: ACM, 2011, pp. 1251–1252. [Online]. Available: http://doi.acm.org/10.1145/2009916.2010144

[18] D. G. Lowe, "Object recognition from local scale-invariant features," 1999.

[19] H. Bay, T. Tuytelaars, and L. V. Gool, "Surf: Speeded up robust features," in *In ECCV*, 2006, pp. 404–417.

[20] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," 2005.

[21] H. Jegou, M. Douze, and C. Schmid, "Hamming embedding and weak geometric consistency for large scale image search," in *In ECCV*, 2008.

[22] X. Chen and T.-J. Cham, "Discriminative distance measures for image matching," in *Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04) Volume 3 - Volume 03*, ser. ICPR '04. Washington, DC, USA: IEEE Computer Society, 2004, pp. 691–695. [Online]. Available: http://dx.doi.org/10.1109/ICPR.2004.310

[23] Y. Rubner, "Perceptual metrics for image database navigation," 1999.

[24] K. Mikolajczyk and C. Schmid, "An affine invariant interest point detector," in *In Proceedings of the 7th European Conference on Computer Vision*, 2002, pp. 0–7.

[25] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," 2003.

[26] E. Hatcher and O. Gospodnetic, *Lucene in Action (In Action series)*. Greenwich, CT, USA: Manning Publications Co., 2004.

[27] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, Nov. 2004. [Online]. Available: http://dx.doi.org/10.1023/B:VISI.0000029664.99615.94