DC-Image for Real Time Compressed Video Matching

Saddam Bekhet, Amr Ahmed and Andrew Hunter

Abstract This chapter presents a suggested framework for video matching based on local features extracted directly from the DC-image of MPEG compressed videos, without full decompression. In addition, the relevant arguments and supporting evidences are discussed. Several local feature detectors will be examined to select the best for matching using the DC-image. Two experiments are carried to support the above. The first is comparing between the DC-image and I-frame, in terms of matching performance and computation complexity. The second experiment compares between using local features and global features regarding compressed video matching with respect to the DC-image. The results confirmed that the use of DC-image, despite its highly reduced size, is promising as it produces higher matching precision, compared to the full I-frame. Also, SIFT, as a local feature, outperforms most of the standard global features. On the other hand, its computation complexity is relatively higher, but it is still within the real-time margin which leaves a space for further optimizations that could be done to improve this computation complexity.

Keywords Compressed domain · DC-image · Global features · Local features · MPEG · SIFT · Video matching

S. Bekhet (✉) · A. Ahmed · A. Hunter
School of Computer Science, University of Lincoln,
Brayford Pool, Lincoln LN6 7TS, UK
e-mail: sbekhet@lincoln.ac.uk

A. Ahmed
e-mail: aahmed@lincoln.ac.uk

A. Hunter
e-mail: ahunter@lincoln.ac.uk

G.-C. Yang et al. (eds.), *Transactions on Engineering Technologies*,
DOI: 10.1007/978-94-017-8832-8_37,

1 Introduction

The volume of video data is rapidly increasing, more than 72 hours of video are uploaded to YouTube every minute [1], and counters are still running fast. This is attributed to recent advance in multimedia technology. The majority of available video data exists in compressed format (e.g. MPEG), and the first step towards efficient video content retrieval is extraction of low level features, directly from compressed domain without full decompression to avoid the expensive computations and large memory requirement involved in decoding such compressed videos. Working on compressed videos is beneficial because of its richness of additional, pre-computed, features such as DCT coefficients, motion vectors and macro blocks types. DC coefficients specifically could be used to reconstruct a video frame with minimal cost [2]. However, most of the current techniques are still inefficient in directly handling compressed videos, without decompressing them first, which is a waste of valuable processing time and memory resources. All those advantages of detecting similarity from compressed videos are also expected to contribute to other higher-level layers of semantic analysis and annotation of videos, among other fields. An MPEG video consists of "**I**", "**P**" and "**B**" frames encoded using Discrete Cosine Transform (DCT) [3]. The DCT algorithm works by dividing an input image into 8×8 blocks (default block size). For each block, the DCT is computed and the result consists of one DC coefficient and 63 AC coefficients per block. A DC-image of an I-frame is the collection of all its DC coefficients, in their corresponding spatial arrangements. The DC image is 1/64 of its original I-frame size. Figure 1a shows an illustration of the DCT block structure. Figure 1b depicts samples of DC-images reconstructed from different I-frames.

The DC-image is usually an image of size around 40×30 pixels. However, the DC-image was found to retain most of the visual features of its corresponding full I-frame. It has also been found that human performance on scene recognition drops by only 7 % when using small images relative to full resolution images [4], as depicted in Fig. 2. This is very useful for computer vision algorithms, especially in relation to computation complexity of achieving the same complex tasks on the DC-image. Taking advantage of the this tiny size, fast reconstruction and richness of visual content, the DC-image could be employed effectively alone or in conjunction with other compressed domain features (AC coefficients, macro-block types and motion vectors) to detect similarity between videos for various purposes; as automated annotation [5] or copy detection or any other higher layer built upon similarity between videos.

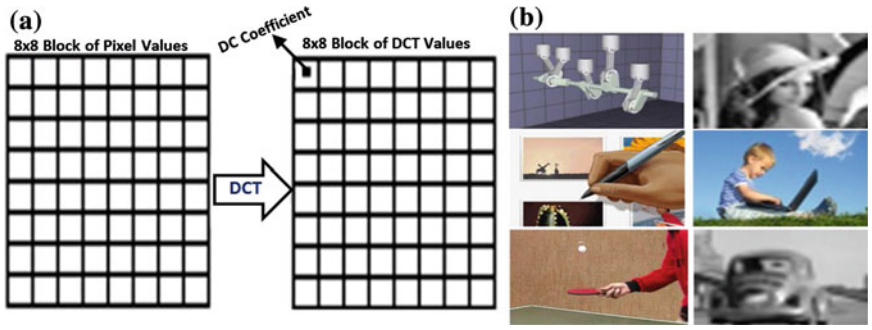

Fig. 1 **a** DCT block structure, showing the location of the DC coefficient, and **b** sample reconstructed DC-images, of size 30 × 40 (images are stretched for illustration)

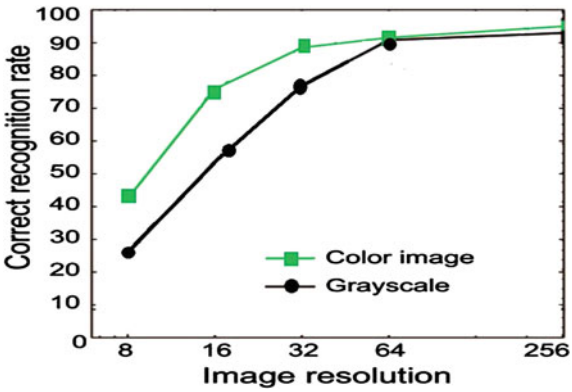

Fig. 2 Human eye performances on scene recognition as a function of image resolution [4]

2 Related Work

In this section, previous key work related to video matching in compressed domain is reviewed, focusing on the DC-image since it is a powerful feature compared to other MPEG features as depicted in Table 1. However, as the DC-image, is a small or lower-resolution image, the relevant work on low-resolution small images will also be reviewed. Initially the term "tiny image" was introduced in [4] during an attempt to construct a database of 80 million tiny images of size 32 × 32, labeled using non abstract English nouns, as listed in WordNet [6]. The aim of this work was to perform object and scene recognition by fusing semantic information extracted from WordNet with visual features extracted from the images, using nearest neighbor methods. Image similarity was computed using two measures; the first is the sum of squared differences (SSD), over the first 19 principal components of each image pixel values. The second similarity measure accounts for the potential small scaling (up to 10 pixels) and small translations (within a 5 × 5 window), by performing exhaustive evaluation of all possible image shifts. The concept of tiny

Table 1 MPEG compressed stream features

Feature	Type	Pros.	Cons.
DC coefficients	Spatial	• Partial decompression needed to extract from I frames [37]	• Cannot generate interest points easily due to its small size [37]
		• Used as a replacement of I frames [37]	• Full decompression needed to be extracted from P & B frames
		• Fast in applying complex operations	
		• Could be extracted either in grayscale or full color	
		• DC image of I frame could be used as a key frame of the entire GOP	
AC coefficients	Spatial	• Partial decompression needed to extract it	• Do not reveal any visual information unless reconstructed [3]
Motion vectors	Temporal		• Describe movement of a block
		• Partial decompression needed to extract	• Do not encode motion information across GOP's [38]
		• A pre-computed motion feature	• Only available for P & B frames
			• Do not encode visual information
Macroblock types	Spatial	• Partial decompression needed to extract	• Encodes only metadata about block compression information (eg. intra coded, skipped) [39]
		• Suitable for copy detection and fingerprinting [39]	• Do not encode visual information

image was then adopted and extended in [7–9] in an attempt to build a database of tiny videos. Approximately 50,000 tiny videos were used, in conjunction with the 80 million tiny images database, to enhance object retrieval and scene classification. Videos were collected with all their available metadata (e.g. title, description and tags), also all video frames were resized to 40 $\times$ 30 pixels stored as one dimensional vector that aggregates all the three color channels. Same similarity measures from tiny images [4] were adopted. Later the work was extended for the purpose of video retrieval using keyframes [7]. However, available video metadata were utilized, which is not always available neither accurate, in addition videos were treated as a set of unrelated images during the matching. Thus, our work is more focused on videos before they could have any tags or meta-data available which can be seen as a phase that can help in building such datasets for later use.

In the compressed domain, the DC-image has been used widely in shot-boundary detection and video segmentation due to its small size [10–14]. It was also utilized for keyframe extraction, instead of parsing the full frame [15–17], or even for video summarization purpose [18, 19]. For video retrieval, in [20] the DC-image was used to detect keyframes, then attention analysis is carried out on

full I-frames, to detect salient objects. SIFT [21] is applied to detect interest points and track them in successive spatial salient regions to build their corresponding trajectories. Color and texture were used to describe each salient region for matching purpose. But this method fails when either the visual features of the foreground object is not distinct or when video background contain rich details as it will produce meaningless salient regions which is not distinctive for a given video. An approach to match video shots and cluster them into scenes proposed in [22], the idea was taking into account variable number of frames to represent a shot (instead of only one keyframe). They utilized frame-to-frame matching based on color histograms computed for every DC or DC + 2AC depending on frame size, for frame of size 320×240 DC-image is selected and for frame size of 160×120DC + 2AC is selected, this makes representative frame images are always of size 40×30 and contains sufficient information for extraction, but with more full decompression for smaller size frames which affects the real-time processing. Regarding generating video signatures using the DC-image, in [23] matching between video clips was done using signatures built by extracting color values (Y-U-V) from DC-images sequence to form three different quantized histograms per each frame. The similarity between two videos is computed using sliding window technique, trying to find the best set of matching frames using histogram intersection. The approach of ordinal measures were applied on DC-images of each color component (Y-U-V) separately to generate fingerprint features for each frame which are accumulated to form video signature for later video matching [24]. Dimitrova et al. [25] demonstrated using DC coefficients of (Y-U-V) components separately and motion vectors to build signature for video retrieval. Signature was extracted from every frame and concatenated to form full video signature where hamming distance used to rank videos based on sliding window technique to determine the set of frames to compute signature from. A noticeable remark is that such approach used the DC-image as a set of numeric values leaving all the visual information behind, in addition to the slow operation of the sliding window technique as it applies an exhaustive search process to align two signatures together.

From a high level perspective, techniques that utilize the DC-image could be classified based on feature extraction level, feature types and applications as depicted in Fig. 3. For feature extraction level, there are two levels. The first; where every frame in video is being processed to extract low level features for later retrieval or signature building. The Second type is more compact and tries to reduce the amount of features being extracted by using keyframes only. Both approaches have disadvantages as they ignore the temporal dimension of a video and handles video as a bag of still images. Moreover, window alignment techniques will be needed in this case, which is based on exhaustive search among frames to find the best matching frames sequence. Regarding video signature built on those approaches it will be large and includes redundant information due to the concatenation of individual frames/keyframes signatures which violates the compactness of the signature. Furthermore for keyframe based schemes, there is

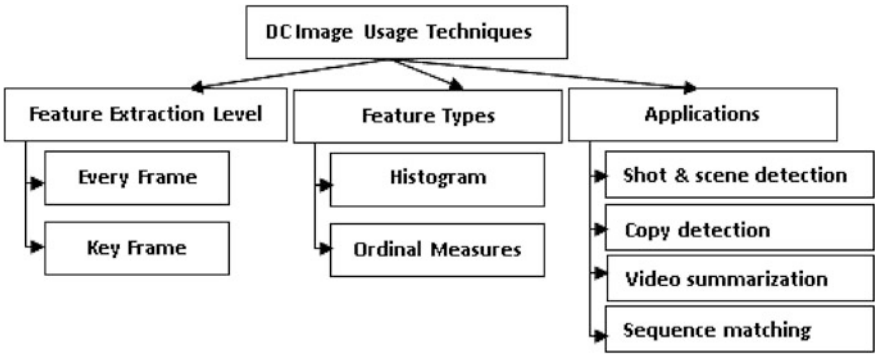

Fig. 3 DC images usage techniques

no fixed selection criteria for those keyframes which could be applied to all videos; some techniques uses the first and last frames within a shot as keyframes, while others uses the middle frame so, the resultant video signature may differ for same video with different keyframe selection criteria's.

For feature types that could be extracted from DC-images, exists: histogram [26] which is a global feature computed on frame level or video level (less common) where similarity between videos depends on the similarity of underlying histograms. Disadvantages of histograms are: (1) relatively high computational cost (pixel level processing) (2) high dependency on underlying color space as each one exhibit significant variations in color representation (3) histogram as a global feature don't capture neither spatial nor temporal information. The second common feature is ordinal measures [27], which also a global feature originally used for stereo matching and later adopted for video retrieval. The idea works by partitioning an image into equal-sized sub-images (blocks), then those sub-images are ranked based on their respective average color. Then, the finial ordering represents the ordinal matrix of the image. Ordinal measures are invariant to luminance change and histogram equalization effects within frame level only, but it is not invariant to geometric transformations, also it is based on color information only which is not robust against color format change. In addition, as a type of global feature it does not capture neither spatial nor temporal information. Recently it has been extended to capture the temporal dimension of videos, as the blocking process could be extended across video frames [28].

3 Proposed Approach

In this section, our proposed DC-image based system for video matching is introduced. The proposed idea is to utilize local features, such as SIFT [21] and SURF [29] on the small DC-images and track them across consecutive frames to compare

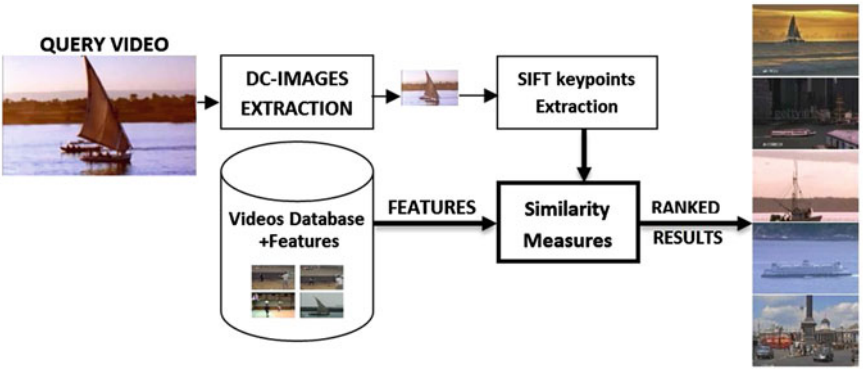

Fig. 4 Proposed system structure to measure videos similarity based on DC image

the similarity between videos. This idea introduces some challenges regarding local features extracting in such small size images, as discussed later. Figure 4 shows block diagram of the proposed system. The main stages of the system are:

1. Decoding video and extracting grayscale DC-image sequence.
2. Extracting SIFT keypoints and their descriptors, in each DC-image.
3. Video matching, using the extracted features.

The following sub-sections describe those stages, including challenges and our contribution to facilitate the video matching using small DC-images, without performing full decompression.

3.1 Extracting the DC Image Sequences

The process starts by decoding a video and extracting luminance DC-images sequence from I-frames only. Following Table 1, there are extra reasons for focusing on the DC-image, includes:

- I-frame's DC-image is the quickest part that could be extracted from a compressed video without performing full decompression of video stream.
- I-frames in GOPs (Group Of Pictures) are inserted by the encoder when there is large residual change (residual is the amount of motion estimation error accumulated at the end of GOP), this could be analogous to keyframes within a scene, in other words as a keyframe is reprehensive to a scene, DC-image of an I-frame could be used as representative of a GOP. In addition, GOPs could be merged to specific length to limit number of DC-images and map them to be key frames like.

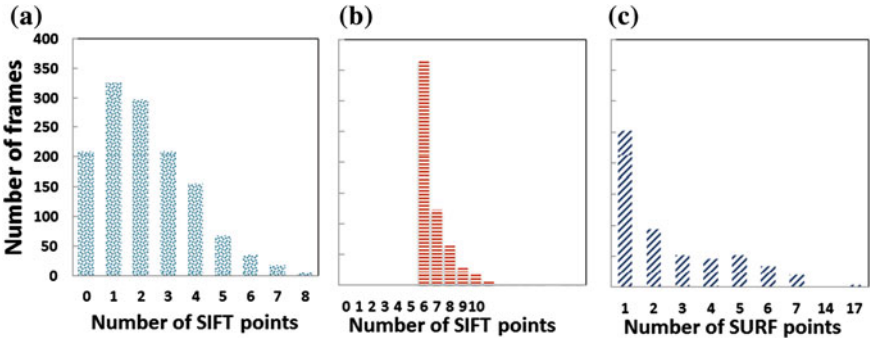

Fig. 5 **a** Number SIFT points per frame before adjusting sigma value. **b** Number SIFT points per frame after adjusting sigma value. **c** Number SURF points per frame

- I-frames will give about 10:1 initial compaction ratio assuming 10 frames per GOP [30] on average which means lower computations and faster results.
- Human eye is sensitive to small changes in luminance rather than chrominance [4]. Thus we can rely on luminance DC-image only.

3.2 Extracting Keypoints and Descriptors

The second stage in the proposed framework is extraction of keypoints and their respective descriptors. During our experiments we used SIFT and SURF for extracting keypoints, as they are the mostly reported effective feature detectors algorithms. While a typical full image of size 500 $\times$ 500 could generate more than 2,000 interest points [21]. However, most of the DC-images would generate less than three SIFT key points, which is not enough for matching [21]. We did an experiment using TRECVID BBC RUSHES [31] dataset to investigate the amount of local features a DC-image could generate. Figure 5a, c shows that $\sim$63 % of frames generate less than three keypoints for SIFT and SURF respectively. Since SIFT was reported for better keypoints localizing than SURF [32–34] we adapted SIFT by iteratively adjusting sigma value (the amount of gaussian blurring applied to an image) to generate a minimum of six keypoints in each DC-image. Figure 5b shows number of SIFT points per frame after our adjustment and enforcing the minimum of six SIFT keypoints per each DC-image. With this enforcement, we facilitated for the DC-image to be used for video matching. Regarding the precision of matching videos, using DC-images compared with the full I-frame, it will be presented later in Sect. 4.

video 2 \ Video1	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12	F13	F14	F15	F16	F17
F1	66	83	50	33	83	66	50	50	50	0	33	83	66	66	66	83	66
F2	50	66	50	33	50	50	33	66	0	50	50	66	66	50	90	66	50
F3	50	66	50	33	50	50	33	50	16	33	50	50	50	50	66	50	33
F4	42	42	42	28	42	28	28	57	28	42	28	42	42	42	42	42	28
F5	66	66	50	33	50	50	50	50	50	50	50	50	66	33	66	50	33
F6	66	66	33	16	50	50	50	50	33	50	50	66	50	33	66	50	33
F7	28	57	71	57	28	57	42	57	57	57	71	71	57	4	57	75	75
F8	42	42	57	71	28	42	28	57	57	57	42	42	28	28	42	42	42
F9	33	50	66	50	33	50	50	50	50	50	50	33	33	50	50	50	50
F10	66	66	83	83	66	66	66	66	50	66	66	50	50	66	66	50	50
F11	66	50	66	83	50	50	66	66	33	50	50	33	33	50	50	33	50

Fig. 6 Finding matching similarity score between two videos

3.3 Video Matching Using DC-Images and SIFT

The third and final stage in our proposed framework is the actual matching between videos. For simplicity, we adopted the frame-to-frame matching, as for each video pair; we compute a similarity measure between SIFT keypoints taking into account the temporal order of video frames. This is done by searching for the longest optimal matching frames sequence between two videos using dynamic programming. Optimality in this case, means finding the best matching video frames that maximizes the overall similarity score with respect to the temporal order of frames. Figure 6 shows a sample confusion matrix of given two videos and the optimal matching values for their respective frames are highlighted in grey. Following is the pseudo-code of the dynamic programming algorithm used to compute the optimal matching cost:

```
SET M to video.1 frames number +1;
SET N to video.2 frames number +1;
CREATE_MATRIX OPT_MATCH[M][N];
INITALIZE DISTANCE to all frame-to-frame similarity based SIFT;
SET OPT_MATCH to 0;
FOR I=1 to M DO
    FOR J=1 to N DO
      SET OPT_MATCH[I][J] to MAX   of
      {
                    OPT_MATCH [I-1][J],
                    OPT_MATCH[I-1][J-1]+DISTANCE[I-1][J-1],
                    OPT_MATCH[I][J-1]
      }
RETURN OPT_MATCH[M-1][N-1] ;
```

Where **DISTANCE** is the confusion matrix between both video frames, computed based on the number of underlying matched SIFT keypoints, and **OPT_MATCH** is the matrix which will contain the finial matching score, this value will be located in location (**M − 1**, **N − 1**) and **MAX** is function returns the maximum value of a given group of numbers. The algorithm works by scanning the confusion matrix from left to right and from up to bottom trying to find the highest match for each frame taking into account the previous and next frames matching scores, in addition to the sequence of frames.

We can see that our proposed dynamic programming algorithm performs one to one mapping as each video frame will be matched to only one frame in the other video. Since the matching is one-to-one some frames may not be matched at all, for two reasons; the first that it might reduce the overall matching value between videos (e.g. frames 1, 4 in video 1). The second case happens if the currently matching videos are of different number of frames (e.g. frames 6, 7, 13 and 16 in video 1).

4 Experiments and Results

In this section we explain the experiments and present the results that support our work explained earlier. This section presents two experiments; the first is regarding comparing the DC-image to I-frame, in terms of matching performance and computation complexity. The second experiment compares between using local features and global features in compressed video matching, with respect to the small size images. We used the TRECVID BBC RUSHES [31] standard data set for video retrieval which contains diverse set of challenging videos; mainly man-made moving objects (cars, tanks, planes and boats). But, since the videos were only available in uncompressed format; all the videos were re-encoded to MPEG-2 format with frame size (352 × 240), so that all the DC-images are of equal size (44 × 30 pixels). The experiments ran on Intel Core i3-3.30 GHZ computer with 4 Gb of RAM.

4.1 DC-Image Versus I-Frame

The purpose of this experiment is to evaluate the performance of the DC-image, in terms of matching and computational complexity, compared to the corresponding I-frame. The experiment used the framework explained in Fig. 4. Regarding matching time based on the DC-image with SIFT [21] features; it took a total of 58.4 min for all videos in the dataset, while it took a total of 166.6 h for the same dataset using the full I-frame. The average time (per frame) is 0.017 s for the DC-image, compared to 1.05 s for the I-frame (time includes reconstruction, SIFT keypoints extraction and matching). This shows that the computation complexity using the DC-image is only 1.6 % of the corresponding I-frame, which means a

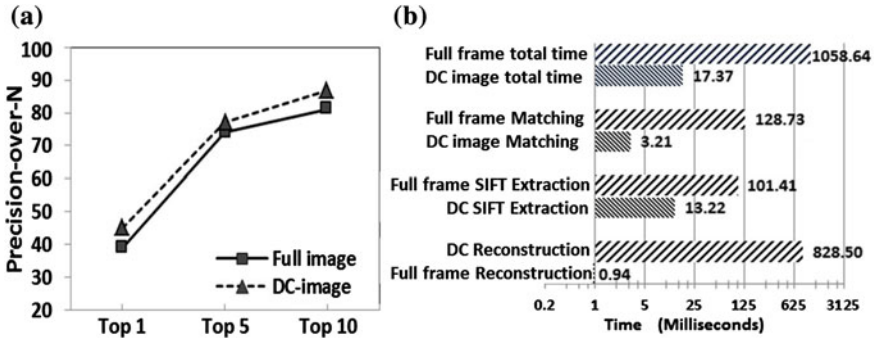

Fig. 7 **a** DC-image versus I-frame retrieval precision, and **b** DC-image versus I-frame timing performance

total reduction of 98.4 % in processing time. Figure 7b shows the timing details for the DC-image and the I-frame respectively. To compare the matching precision, we adopted the precision-over-N [35] standard measure over ranks 1, 5 and 10. The DC-image, despite its highly reduced size, was found to have a slightly higher precision than the I-frame at all ranks, as depicted in Fig. 7a.

4.2 Local Versus Global Features

The purpose of this experiment is to evaluate the performance of using local and global features, on the DC-image, in terms of matching precision and computational complexity. The experiment also used the framework described earlier in Fig. 5. For local features, we utilized SIFT [21] as a local feature descriptor, in addition to dense SIFT [36] to verify the results for a larger number of keypoints. For global features, we applied matching based on the luminance histogram, ordinal measures [27] and the pixel difference [8].

The results, presented in Fig. 8a, shows that SIFT as a local feature descriptor outperforms dense SIFT, in addition SIFT outperforms global feature descriptors by 15.4 % (compared to ordinal matching as the highest precision global feature method). However, SIFT's computation complexity was the highest, as depicted in Fig. 8b. SIFT took 16.43 ms to match two DC-image frames, compared to only 2 ms in pixel difference matching (maximum time in case of global features). But SIFT still works within real-time margin, while producing better matching performance. Knowing that all measures are being used in their generic form, using dynamic programming to incorporate the temporal dimension based on the finial fame-to-frame confusion matrix. We also developed results visualization software, a snapshot is depicted in Fig. 9 based on real example of SIFT matching using the BBC RUSHES dataset.

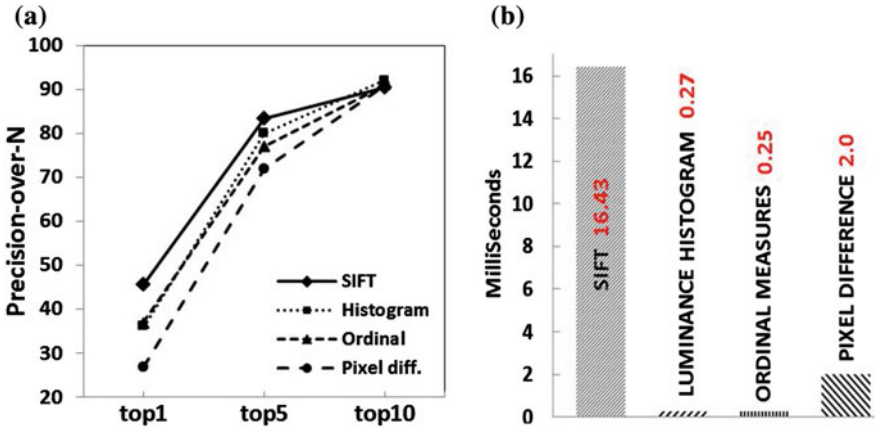

Fig. 8 **a** DC image retrieval precision-over-N curves using SIFT-luminance histogram-ordinal measures-pixel difference, and **b** DC timing analysis using SIFT-luminance histogram-ordinal measures-pixel difference

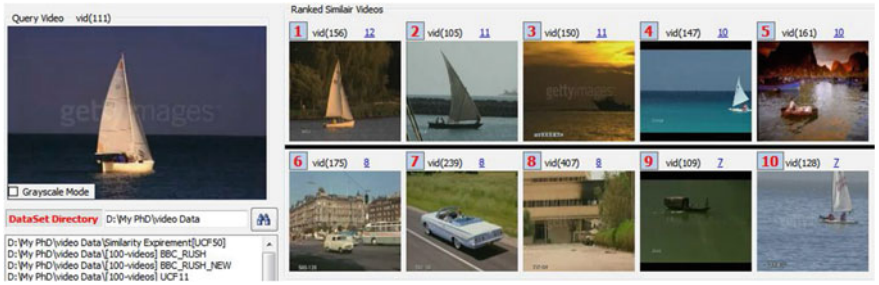

Fig. 9 Snapshot of visualization software based on SIFT matching

5 Conclusion and Future Work

In this paper, we presented a framework for video matching based on local features extracted only from the DC-images of MPEG compressed videos, without full decompression. Also, supporting experiments regarding DC-image precision and complexity versus the full I-frame were presented. But we had to address the issue of using SIFT on such small-size images, before it could be used. The results show that the DC-image, despite its small size, produces similar (if not better) similarity precision scores, compared to its corresponding I-frame. But using the DC-image has dramatically improved the computational performance ($\sim$62 times faster), which makes it a high candidate for more sophisticated use. Also, local features, such as SIFT, were compared to standard global features for the purpose of video similarity. The results shows that using SIFT, on DC-image only, slightly

outperformed the accuracy of the global features. On the other hand, the computational complexity of using SIFT is relatively higher than those for the global features. But SIFT extraction and matching is still within the real-time margins, and still we have a number of optimizations to be introduced to reduce this computation complexity. We also plan to introduce more complex matching, instead of the frame-to-frame approach, and better incorporate the temporal information actively.

Acknowledgments This work is funded by SouthValley University, Egypt.

References

1. YouTube Statistics [Online]. Available http://www.youtube.com/yt/press/statistics.html (2013)
2. S. Bekhet, A. Ahmed, A. Hunter, Video matching using DC-image and local features, in *Proceedings of the World Congress on Engineering*, UK (2013), pp. 2209–2214
3. A.B. Watson, Image compression using the discrete cosine transform. Math. J. **4**, 81 (1994)
4. A. Torralba, R. Fergus, W.T. Freeman, 80 million tiny images: A large data set for nonparametric object and scene recognition. IEEE Trans. Pattern Anal. Mach. Intell. **30**, 1958–1970 (2008)
5. A. Altadmri, A. Ahmed, A framework for automatic semantic video annotation. Multimedia Appl. Tools **64**(2), 1–25 (2013)
6. G. Miller, C. Fellbaum, *Wordnet: An Electronic Lexical Database* (1998)
7. A. Karpenko, P. Aarabi, Tiny videos: A large data set for nonparametric video retrieval and frame classification. IEEE Trans. Pattern Anal. Mach. Intell. **33**, 618–630 (2011)
8. A. Karpenko, P. Aarabi, Tiny videos: A large dataset for image and video frame categorization, in *11th IEEE International Symposium on Multimedia (ISM '09)* (2009), pp. 281–289
9. A. Karpenko, P. Aarabi, Tiny videos: non-parametric content-based video retrieval and recognition, in *Tenth IEEE International Symposium on Multimedia (ISM 2008)* (2008), pp. 619–624
10. B.-L. Yeo, B. Liu, A unified approach to temporal segmentation of motion JPEG and MPEG compressed video, in *Proceedings of the International Conference on Multimedia Computing and Systems* (1995), pp. 81–88
11. J. Meng, Y. Juan, S.F. Chang, Scene change detection in a MPEG compressed video sequence, in *IS&T/SPIE Symposium Proceedings* (1995)
12. J.-L. Zheng, M.-J. Li, M.-X. Zhang, J. Zhou, Z.-D. Liu, An effective framework of shot segmentation based on I-frame in compressed-domain videos, in *International Conference on Wavelet Analysis and Pattern Recognition* (2012), pp. 84–90
13. P. Xu, L. Xie, S.F. Chang, A. Divakaran, A. Vetro, H. Sun, Algorithms and system for segmentation and structure analysis in soccer video, in *Proceedings of ICME* (2001), pp. 928–931
14. A. Divakaran, H. Ito, H. Sun, T. Poon, Scene change detection and feature extraction for MPEG-4 sequences, in *Electronic Imaging'99* (1998), pp. 545–551
15. G. Liu, J. Zhao, Key frame extraction from MPEG video stream, in *Third International Symposium on Information Processing* (2010), pp. 423–427
16. E.K. Kang, S.J. Kim, J.S. Choi, Video retrieval based on scene change detection in compressed streams. IEEE Trans. Consum. Electron. **45**, 932–936 (1999)

17. O.N. Gerek, Y. Altunbasak, Key frame selection from MPEG video data, in *Electronic Imaging* (1997), pp. 920–925
18. J.C.S. Yu, M.S. Kankanhalli, P. Mulhen, Semantic video summarization in compressed domain MPEG video, in *Proceedings International Conference on Multimedia and Expo*, vol. 3 (2003), pp. 329–332
19. J. Almeida, N.J. Leite, R.S. Torres, Online video summarization on compressed domain. J. Vis. Commun. Image Represent. (2011)
20. H.-P. Gao, Z.-Q. Yang, Content based video retrieval using spatiotemporal salient objects, *in 2010 International Symposium on Intelligence Information Processing and Trusted Computing (IPTC)* (2010), pp. 689–692
21. D.G. Lowe, Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. **60**, 91–110 (2004)
22. M.M. Yeung, B. Liu, Efficient matching and clustering of video shots, in *Proceedings of the International Conference on Image Processing.* vol. 1 (1995), pp. 338–341
23. M.R. Naphade, M.M. Yeung, B.L. Yeo, A novel scheme for fast and efficient video sequence matching using compact signatures, in *Proceedings of SPIE, Storage and Retrieval for Media Databases* (2000), pp. 564–572
24. R. Mohan, Video sequence matching, in *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 6 (1998), pp. 3697–3700
25. N. Dimitrova, M.S. Abdel-Mottaleb, Video retrieval of MPEG compressed sequences using dc and motion signatures, in *Video Retrieval of MPEG Compressed Sequences using DC and Motion Signatures*, 1999
26. R.C. Gonzalez, E.W. Richard, *Digital Image Processing*, 3rd edn edn. (Prentice Hall, Englewood Cliffs, 2008), pp. 142–143
27. D.N. Bhat, S.K. Nayar, Ordinal measures for visual correspondence, in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, (CVPR '96)* (1996), pp. 351–357
28. J. Almeida, N.J. Leite, R. S. Torres, Comparison of video sequences with histograms of motion patterns, in *18th IEEE International Conference on Image Processing (ICIP)* (2011), pp. 3673–3676
29. H. Bay, A. Ess, T. Tuytelaars, L. Van Gool, Speeded-up robust features (SURF). Comput. Vis. Image Underst. **6**(110), 346–359 (2008)
30. D. Fuentes, R. Bardeli, J.A. Ortega, L. Gonzalez-Abril, A similarity measure between videos using alignment, graphical and speech features. Expert Syst. Appl. **39**, 10278–10282 (2012). 9/1
31. A. Basharat, Y. Zhai, M. Shah, Content based video matching using spatiotemporal volumes. Comput. Vis. Image Underst. **110**, 360–377 (2008)
32. K. Mikolajczyk, C. Schmid, A performance evaluation of local descriptors, in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition,* vol. 2 (2003), pp. II-257–II-263
33. L. Juan, O. Gwun, A comparison of sift, pca-sift and surf. Int. J. Image Process. (IJIP) **3**, 143–152 (2009)
34. K.E.A. van de Sande, T. Gevers, C.G.M. Snoek, Evaluating color descriptors for object and scene recognition. IEEE Trans. Pattern Anal. Mach. Intell. **32**, 1582–1596 (2010)
35. D.M. Powers, Evaluation: From precision, recall and f-factor to roc, informedness, markedness and correlation, in *School of Informatics and Engineering*, Flinders University, Adelaide, Australia, Tech.Rep.SIE-07-001, 2007
36. T. Tuytelaars, Dense interest points, in *2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2010), pp. 2281–2288
37. S. Bekhet, A. Ahmed, A. Hunter, Video matching using DC-image and local features, in *Lecture Notes in Engineering and Computer Science: Proceedings of the World Congress on Engineering, WCE 2013*, London, 3–5 July 2013, pp. 2209–2214

38. N. Dimitrova, F. Golshani, Motion recovery for video content classification. ACM Trans. Inf. Syst. (TOIS) **13**, 408–439 (1995)
39. A.S. Abbass, A.A.A. Youssif, A.Z. Ghalwash, *Compressed Domain Video Fingerprinting Technique Using the Singular Value Decomposition* (2012)
40. P. Panchal, S. Merchant, Performance evaluation of fade and dissolve transition shot boundary detection in presence of motion in video, in *1st International Conference on Emerging Technology Trends in Electronics, Communication and Networking (ET2ECN)* (2012), pp. 1–6