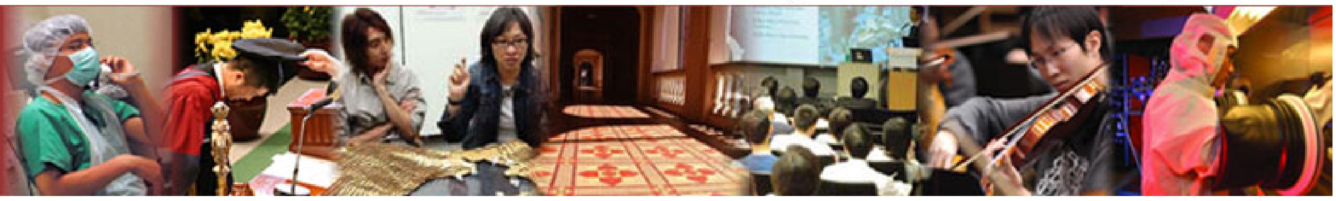


Object-based Surveillance Video Retrieval System With Real-Time Indexing Methodology

Jacky S-C. Yuk, Kwan-Yee K. Wong, Ronald H-Y. Chung, K. P. Chow, Francis Y-L. Chin, and Kenneth S-H. Tsang

Department of Computer Science
The University of Hong Kong
Pokfulam, Hong Kong

Abstract. This paper presents a novel surveillance video indexing and retrieval system based on object features similarity measurement. The system firstly extracts moving objects from the videos by an efficient motion segmentation method. The fundamental features of each moving object are then extracted and indexed into the database. During retrieval, the system matches the query with the features indexed in the database without re-processing the videos. Video clips which contain the objects with sufficiently high relevance scores are then returned. The novelty of the system includes: 1. A real-time automatic indexing methodology achieved by a fast motion segmentation, such that the system is able to perform on-the-fly indexing on video sources; and 2. an object-based retrieval system with fundamental features matching approach, which allows user to specify the query by providing an example image or even a sketch of the desired objects. Such an approach can search the desired video clips in a more convenient and unambiguous way comparing with traditional text-based matching.

1 Introduction

As the number of video clips grow rapidly all over the world, a fast video retrieval system, which allows users to search their desired video clips efficiently, becomes more and more demanded. Most of the existing popular video searching engines [1, 2] rely on text-based annotations and descriptions of the video clips input by the clip owners. These engines search for the desired clips by matching the keywords input by the user against a large set of annotations. However, this approach always needs extensive time and intensive man-power to annotate and describe those video clips. Furthermore, ambiguous search results usually occur due to the fact that different people usually have different interpretations on the video contents.

Automated video indexing and/or annotation has become one of the key issues in recent years. This approach highly reduces the time and man-power required during the indexing stage. The systems inroduced in [10, 7] perform video retrieval according to some similarity measurements between video shots. Although, it has been demonstrated that this approach can give excellent results, user is usually required to provide a sample video clip, which may not be always possible. Besides, the processing of the sample video clip for querying is also time consuming, resulting in an inefficient retrieval system.

The systems presented in [5, 9] employ a classification approach, and they classify video shots by some pre-defined text class labels. This approach demonstrates powerful abilities in searching the desired video clips, and is able to be incorporated with the traditional text-based video searching engines. However, this kind of systems is applicable only when the contents of the video clips have already been seen and properly associated with one of the pre-defined class labels. Adding new class label always requires re-processing all the videos again which is highly time consuming.

Smith and Khotanzad [8] suggest an object-based approach for video retrieval by first using JSEG segmentation method to segment each I/P frame into regions. At most ten of the largest segmented regions are selected as interested objects to be tracked. The MPEG-7 texture and color features of the tracked objects are then used to describe the video shots. The system demonstrates an interesting content-based approach for video retrieval. However, it requires a computationally intensive segmentation step. The system, again, also requires a sample video clip as a query in the retrieval process, which is not desirable.

In this paper, we propose an efficient object-based video indexing and retrieval system Instead of using computationally intensive segmentation for extracting the objects as in [8], we employ an efficient motion segmentation method as described in [4]. Such a motion segmentation method allows the system to segment and locate the motion objects in an efficient manner according to the consistency of the motion vectors encoded in MPEG-4 videos, with no limitation on the number of objects in each video shot. The system tracks each moving object until it disappears. Features of each tracked object are also extracted, and are then indexed as metadata into the database. By doing so, the indexing is thus no longer based on shot but each moving object itself. This approach allows users to retrieve video clips by simply providing the information of the objects of interest. In the proposed system, user is able to search for the video clips containing the desired objects by providing a snapshot, photo, or even hand-drawn images of the objects. Therefore, user can specify the query in a more convenient and intuitive way, and also allows the retrieval process to be more efficient. Basically, the system indexes the dominant color and edge direction histograms as the features for future retrieval. Some other features like color histograms, shape descriptors are also implemented and tested. This paper, however, only presents the results by using dominant color and edge direction histograms since experimental results show that these two features are efficient, and more reliable than the other features especially when performing retrieval based on hand-drawn images.

The paper is organized as follows. Section 2 presents the details of the segmentation and tracking of the moving objects. Section 3 describes the feature extraction of each tracked object and the indexing process of the database. Section 4 describes the object-based video retrieval process, and Section 5 presents the retrieval results. The conclusion and discussion of the future works in Section 6.

2 Moving Object Segmentation and Tracking

The moving objects appeared in the video scenes are segmented by an efficient block-based segmentation method as proposed in [4]. Since the segmentation method can utilize the motion vectors of video stream like MPEG-4/H.264, it allows the system to segment moving objects very efficiently, and enables the system to perform video indexing and recording in real-time. The segmented objects are then tracked by the commonly used Kalman filter tracker.

2.1 Efficient Motion Segmentation

As suggested in [4], the motion block segmentation is performed based on a region growing approach according to the motion vector consistency between blocks. A motion block $B_{i,j}$ located at (i, j) with motion vector $\boldsymbol{MV}_{i,j}$ is grouped into a region $\boldsymbol{R}$ when the following condition is satisfied:

$$\frac{\sum_{B_{i_r,j_r}\in\boldsymbol{R}} d(B_{i,j}, B_{i_r,j_r})}{\|\boldsymbol{R}\|} < \epsilon_{mv} \tag{1}$$

where ϵ_{mv} is a pre-defined threshold. $\|\boldsymbol{R}\|$ denotes the number of blocks in the region $\boldsymbol{R}$, and $d(B_{i,j}, B_{i_r,j_r})$ denotes the consistency index between $B_{i,j}$ and B_{i_r,j_r}, such that:

$$d(B_{i,j}, B_{i_r,j_r}) = |[i_r - i, j_r - j][\boldsymbol{MV}_{i_r,j_r} - \boldsymbol{MV}_{i,j}]| \tag{2}$$

Figure 1 shows some moving object segmentation results of different video sequences with 16x16 block size, and the segmentation gives fairly good object boundaries.

2.2 Object Tracking

After motion segmentation, the location of each segmented object is tracked by commonly used Kalman filter tracker. At each time instance t when a particular object n is being tracked, a set of feature vectors $\boldsymbol{V}(n,t) = \{\boldsymbol{v}_i(n,t)\}$ is extracted. The final feature vector set $\boldsymbol{V}(n)$ for the object n is then defined as $\{\boldsymbol{v}_i(n)\}$, such that each component vector $\boldsymbol{v}_i(n)$ is the mean of the corresponding $\boldsymbol{v}_i(n,t) \in \boldsymbol{V}(n,t)$, i.e.,

$$\boldsymbol{v}_i(n) = \frac{\sum_{t_s(n)\le t<t_l(n)} \boldsymbol{v}_i(n,t)}{T} \tag{3}$$

where $t_s(n)$ is the time instance when the object n appears and starts being tracked in the scene, and $t_l(n)$ is the time instance when the object n leaves the scene, and $T = t_l(n) - t_s(n)$.

3 Feature Extraction and Indexing

The system mainly utilizes the dominant colors $\boldsymbol{v}_{dc}$ and the edge direction histograms $\boldsymbol{v}_{ed}$, respectively, as the feature vectors in order to form the feature vector set $\boldsymbol{V}$ for indexing and retrieval. The implementation of these two feature vectors is an modified version of that suggested in MPEG-7 [3, 6], which will be discussed in more details in

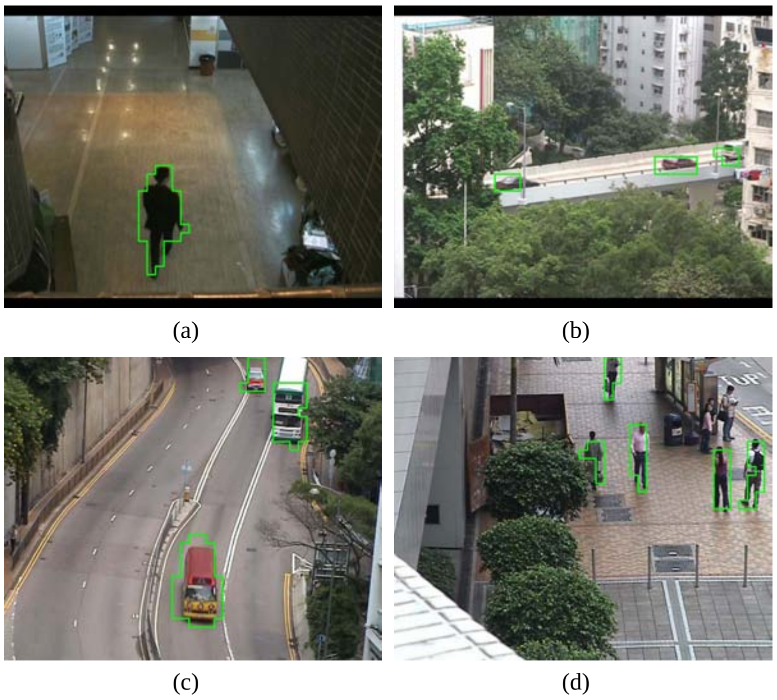


Fig. 1. Examples of moving object segmentation results of different video sequences at different view points

the following section. Furthermore, the system also performs Gaussian mixture modeling (GMM) of the background on the down-sampled video frames. The down-sampling ensures the efficiency of indexing process, but retains an informative background model such that most of the irrelevant background pixels can be ignored to achieve a more accurate retrieval results. The extracted $\boldsymbol{V}$ together with the tracking information (including the object trajectory, aspect ratio, size, etc) are then indexed as metadata into database for later retrieval.

3.1 Features

Dominant Color The dominant color feature vector $\boldsymbol{v}_{dc}$ is constructed from the color histograms instead of the computationally intensive segmentation approach as described in [6]. In the current implementation, only the hue component of the HSV color space is used, i.e., the dominant color feature vector only considers the color component, such that the feature is relatively insensitive to lighting conditions and saturation level of different cameras. The i-th color histogram bin value $chist_i$ is defined as:

$$chist_i = \sum_{(x,y)\in \boldsymbol{B}(n)} \delta_i(h_{x,y}) \tag{4}$$

where $h_{x,y}$ is the hue pixel value at location (x, y), and $\boldsymbol{B}(n)$ is the motion blocks of the segmented object n. $\delta_i(h_{x,y})$ is a delta function such that

$$\delta_i(h_{x,y}) = \begin{cases} 1 \; h_{x,y} \text{ is at foreground AND} \\ \quad h_{x,y} \text{ falls into the } i\text{-th bin range,} \\ 0 \text{ otherwise} \end{cases} \tag{5}$$

For each bin i, a vector $\boldsymbol{v}_{chist_i} =< c_i, p_i >$ is constructed where c_i the mean color value of bin i. $p_i = \frac{chist_i}{\sum_j chist_j}$ is the percentage of bin i. The dominant color feature vector $\boldsymbol{v}_{dc}$ is then constructed from the vectors $\boldsymbol{v}_{chist_i}$ of the three bins with largest p_i.

Edge Direction Histograms The edge direction histograms are constructed by firstly dividing the bounding box of the segmented object into 4x4 blocks. The sobel edge direction of each pixel within each block $b_{i,j}$ is determined, and the corresponding edge direction histograms are constructed. As suggested in [6, 3], 5 direction histogram bins are used including vertical edges, horizontal edges, 45^o edges, 135 ^{o} edges, and non-directional edges. However, experiments showed that the histograms are always dominated by the non-directional edges, especially for those objects with high portion of homogeneous color region. Therefore, the current implementation only uses the first 4 edges for histograms construction, and ignores the non-directional edges. The edge direction histograms of $b_{i,j}$ at direction dir is constructed as:

$$ehist_{i,j,dir} = \nu_{i,j,dir} \sum_{(x,y)\in b_{i,j}} \delta_{dir}(e_{x,y}) \tag{6}$$

where $e_{x,y}$ is the sobel edge direction at location (x, y), and $\nu_{i,j,dir}$ is the normalize factor. δ_{dir} is the delta function such that

$$\delta_{dir}(e_{x,y}) = \begin{cases} 1 \;\; e_{x,y} \text{ is at foreground AND} \\ \quad\; |e_{x,y}| > \epsilon_e \text{ AND} \\ \quad\; e_{x,y} \text{ falls into the bin direction,} \\ 0 \;\; \text{otherwise} \end{cases} \tag{7}$$

Where the threshold ϵ_e is used to determine if $e_{x,y}$ is a non-directional edge or not.

In addition to the local histograms of 4x4 blocks, [6, 3] also suggest to construct the semi-global histograms and the global histograms as shown in figure 2. The semi-global and global histograms can be efficiently computed by accumulating the local histograms, i.e., the edge direction histograms feature vector $\boldsymbol{v}_{ed}$ consist of 64 local histogram bins $ehist_i^l$, 52 semi-global histogram bins $ehist_i^s$ and 4 global histogram bins $ehist_i^g$, where i denotes the i-th histogram of local, semi-global or global block.

4 Video Retrieval

The system performs object-based video retrieval by samples. Users can search for the objects appeared in the video clips by providing snapshots or hand-drawn images as queries. Some of the sample query images are shown in figure 3. The system extracts the set of feature vectors $\boldsymbol{V}_q = \{\boldsymbol{v}_{dc}^q, \boldsymbol{v}_{ed}^q\}$ of the input image as described in section

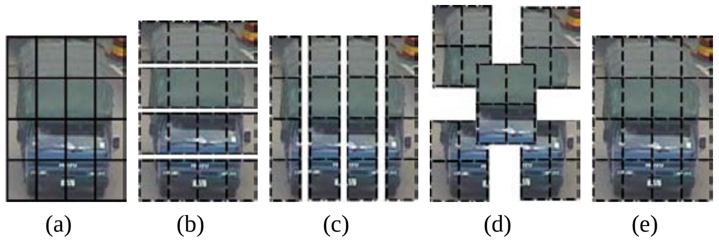


Fig. 2. Bounding box of the segmented object is divided into 4x4 blocks for construction of edge direction histograms. (a) 16 local blocks, (b)(c)(d) 4 horizontal, 4 vertical and 5 neighboring sub-images of semi-global blocks respectively, (e) the whole bounding box image is used for global edge direction histograms.

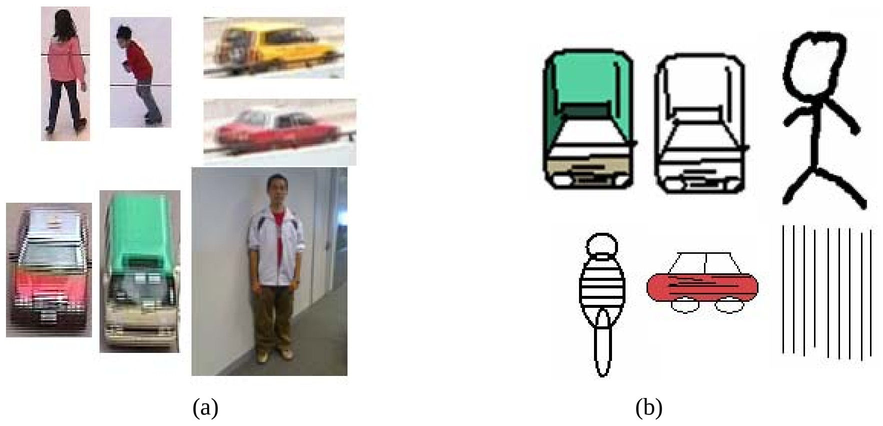


Fig. 3. Sample query images (a) snapshots/photos (b) hand-drawn images

3.1. $\boldsymbol{V}_q$ is then used to match with the feature vector sets $\boldsymbol{V}(n) = \{\boldsymbol{v}_{dc}(n), \boldsymbol{v}_{ed}(n)\}$ of each segmented object n already stored in the database.

4.1 Similarity Measurements

To measure whether an object matches the query, the dominant color distance D_{dc} and edge direction histograms distance D_{ed} are defined:

$$D_{dc}^2(\boldsymbol{v}_{dc}, \boldsymbol{v}_{dc}^{'}) = \sum_{i=1}^{3} p_i^2 + \sum_{j=1}^{3} p_j^{'2} + \sum_{i=1}^{3}\sum_{j=1}^{3} 2a_{i,j}p_i p_j^{'} \tag{8}$$

$$D_{ed}(\boldsymbol{v}_{ed}, \boldsymbol{v}_{ed}^{'}) = \sum_{i=1}^{64} |ehist_i^l - ehist_i^{'l}| + 5 \times \sum_{i=1}^{4} |ehist_i^g - ehist_i^{'g}| + \sum_{i=1}^{52} |ehist_i^s - ehist_i^{'s}| \tag{9}$$

In (8) $a_{i,j}$ is the similarity coefficient between color c_i and c_j, such that $a_{i,j} = 1 - d_{i,j}/d_{max}$ where $d_{i,j} = |c_i - c_j|$ and d_{max} is the maximum allowable distance. Here, c_i and p_i are the color and percentage components of the i-th dominated color.

An object n in the database is considered to be matched if $D_{dc}(\boldsymbol{v}_{dc}(n), \boldsymbol{v}^q_{dc}) < \epsilon_{dc}$ and $D_{dc}(\boldsymbol{v}_{ed}(n), \boldsymbol{v}^q_{ed}) < \epsilon_{ed}$, where ϵ_{dc} and ϵ_{ed} are pre-defined thresholds for dominant color and edge direction histograms respectively.

The overall distance $D_{overall}$ is calculated as $D_{overall} = \alpha_{dc} D_{dc} + \alpha_{ed} D_{ed}$ where α_{dc} and α_{ed} are the pre-defined weighting. The relevance R between the retrieved objects and the query is then defined:

$$R = \begin{cases} (1 - \frac{D_{overall}}{D_{max}}) \times 100\% & \text{if } D_{overall} \leq D_{max}, \\ 0\% & \text{otherwise} \end{cases} \tag{10}$$

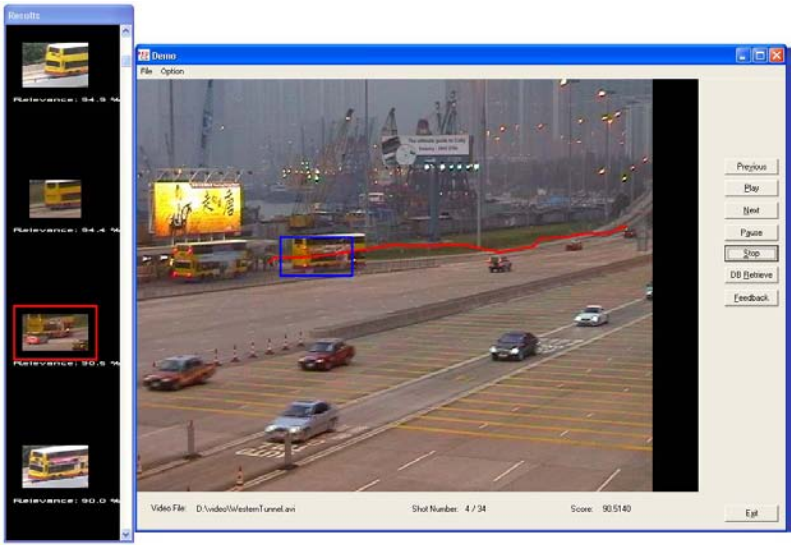
where D_{max} is the maximum allowable distance.

5 Experiments and Results

The proposed object-based indexing and retrieval system is evaluated by 19 surveillance videos with D1 resolution (720x576 pixels) which are captured by static cameras. Without any optimization, the indexing process was already running mostly in real-time at about 24-25 frames per second on a PC with Intel Core(TM)2 CPU 6300 @ 1.86 GHz and 1 Gb RAM. The process can be even faster when the videos are down-sampled into CIF resolution (352x288 pixels) without significant degradation of the retrieval results. There are totally 1421 objects segmented from the video, of which all their associated features are indexed into the database. The objects are mainly human beings, vehicles, and very few background objects like windy trees due to background movement and camera vibrations.

Figure 4 shows a demonstration of the user interface of the retrieval system. The left column shows snapshots of the retrieved objects. Users can select any retrieved object and playback the video clip which contains the object. The corresponding object trajectory and location are also overlaid in the videoas red line and blue bounding box. Figures 5 and 6 show the examples of some retrieval results. The query images are shown at the bottom-left of each sub-figures, and the corresponding snapshots of 12 retrieved objects with the highest relevance R as defined in (10) are shown in the next 3 columns, which are displayed in the order of top-down and then left-right according to R. In our experiments, foreground of a query image usually occupies more than 80% of the whole image area. The retrieval results are even better when the background are masked out. On the other hand, if the background becomes clutter or its area increases, the results will degrade gradually.

The query images in figures 5(a), 5(b) and 5(c) are randomly sampled from the 19 indexed videos. The system is able to retrieve the corresponding objects and also objects which are similar to them. The query images in figures 5(e) and 5(d) are unseen objects taken by a digital camera. The system is also able to retrieve relevant objects as shown. Results show that the system performs the retrieval by snapshot/photo images quite well especially when the objects have sharp dominant colors. However, if the dominant colors tend to be white/black/grey, the system may not be able to distinguish them due



Relevance: 94.9 %
Relevance: 94.4 %
Relevance: 90.6 %
Relevance: 90.0 %

Video File: D:\video\WesternTunnel.avi Shot Number: 4 / 34 Score: 90.5140

Fig. 4. User interface of the proposed video retrieval system.

to the limitation of the hue component in differentiating gray levels. As shown in figure 5(e) and 5(d), objects with dominant black/grey colors are also retrieved. Simply using other color space like RGB/CIE LUV or adding the saturation/intensity components may help to handle this kind of retrievals, but this also makes the system to be very sensitive to lighting changes or camera settings. Slight changes in lighting or camera saturation may cause the system fail in the retrieval. Further research is needed for more robust but efficient color searching under different lighting and camera settings.

Figure 6 shows the retrieval results using hand-drawn query images which are drawn with MS Paint. For figures 6(d) and 6(e), only the edge direction features are used in retrieval since there is no color information in the query images, while the other three use both dominant color and edge direction histograms. The results show the system can retrieve objects using hand-drawn images very well. This demonstrates an interesting ability of the system. Users can perform video retrieval even by drawing the desired objects when the snapshots/photos of the objects are not available.

6 Conclusion and Future Works

The proposed system demonstrates an efficient and interesting object-based video indexing and retrieval approach to retrieve moving objects which are similar to the query image. Unlike class-based video retrieval, the proposed system allows more freedom for users to specify the searching query. Knowledge-based information like “what is the object” can also be retrieved by further post-processing and analysis of the indexed metadata.

Furthermore, the proposed system also demonstrates its potential ability in digital surveillance industry. For examples, operators may take hours or even days to analyze recorded videos to search for a suspect car or person that passed through certain monitoring areas. With the help of the proposed system, operators may search for the suspect by simply providing a snapshot/photo/drawing of the suspect, and then all the video clips, which captured such a suspect, can be retrieved. The time for performing the labor intensive searching operations can therefore be significantly reduced.

The current proposed system is targeted to extract and index moving objects in the scene. For static background objects, we may need another object extraction method which is able to extract those static objects in an efficient manner. One of the possible solution is to perform general image segmentation on the background image given by the GMM background modeling. In this way, the computational intensive segmentation process is required only once at the end of each scene, and therefore, the processing time for video indexing can be highly reduced.

Besides, there are two major directions for future works. Firstly, the system is currently using MySQL version 5.0.2 to provide the database and indexing operations. This can causes the retrieval process being very slow when the database size grows, due to the curse of dimensionality of image feature vector as stated in [5]. Traditional R-Tree, SR-Tree, and SS-Tree will not work either for the high dimensional image feature vectors. As a result, further research effort is required on video database structure to speed-up the retrieval mechanism. Secondly, the current system performs video retrieval based on matching of elementary features. These features, as stated earlier, can further be analyzed to give knowledge-based information. This motivates us to continue research on performing knowledge-based video retrieval which allows higher level query like "Find a girl in red clothes who is skating". In addition, we will also continue to search for a more robust and reliable elementary features, especially a more reliable color feature, to further enhance retrieval accuracy.

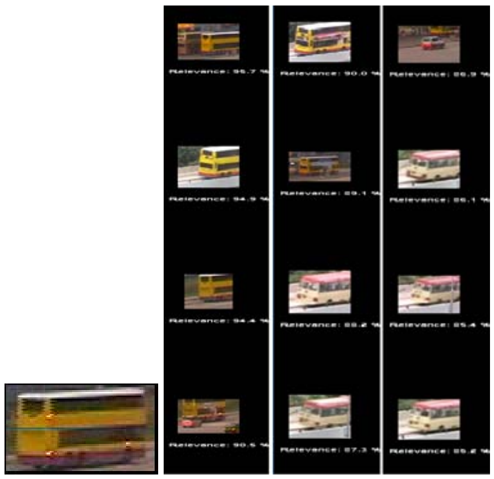
References

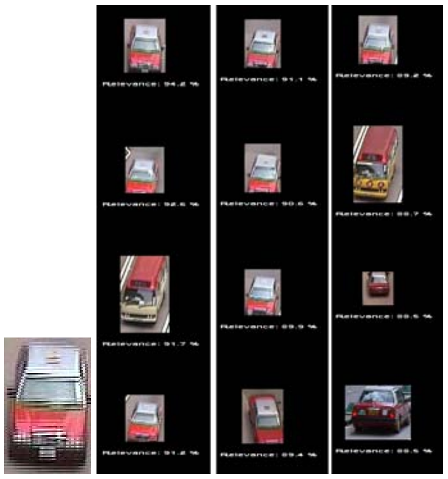
1. http://video.google.com.
2. http://www.youtube.com.
3. Iso/iec 15938-3 information technology – multimedia content description interface – part 3, visual. May 2002.
4. R. H. Y. Chung, F. Y. L. Chin, K.-Y. K. Wong, K. P. Chow, T. Luo, and H. S. K. Fung. Efficient block-based motion segmentation method using motion vector consistency. In *Proc. IAPR Conference on Machine Vision Applications*, pages 550–553, Tsukuba Science City, Japan, May 2005.
5. S. Fan, X.Q. Zhu, A.K Elmagarmid, W.G. Aref, and L. Wu. Classview: Hierarchical video shot classification, indexing, and accessing. *IEEE Transactions on Multimedia*, 6(1):70–86, Febrary 2004.
6. B.S. Manjunath, P. Salembier, and T. Sikora. *Introduction to MPEG-7, Multimedia Content Description Interface*. Wiley, 2002.
7. J.H. Oh and N. Baskaran. An efficient technique for video shot indexing using low level features. In *Proc. of WORKSHOP ON MULTIMEDIA SEMANTICS 2002 in conjunction with 29th Annual Conference on Current Trends in Theory and Practice of Informatics (SOFSEM 2002)*, Milovy, Czech Republic, November 2002.
8. M. Smith and A. Khotanzad. An object-based approach for digital video retrieval. In *Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC 2004)*, volume 1, pages 456–459, Los Alamitos, CA, USA, April 2004.

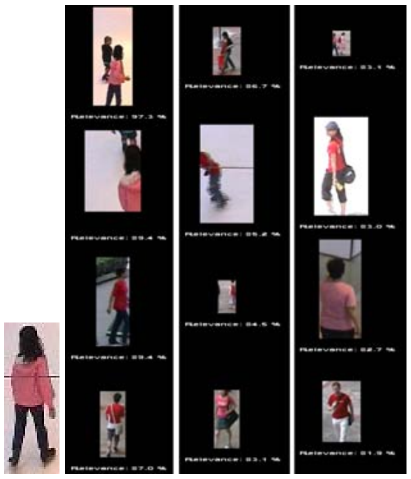
9. C. Taskiran, J-Y. Chen, A. Albiol, L. Torres, C.A. Bouman, and E.J. Delp. Vibe: a compressed video database structured for active browsing and search. *IEEE Transactions on Multimedia*, 6(1):103–118, Febrary 2004.
10. H. Yi, D. Rajan, and L.T. Chia. A motion based scene tree for browsing and retrieval of compressed videos. In *2nd ACM International Workshop on Multimedia Databases*, pages 10–18, Washington DC, USA, November 2004.

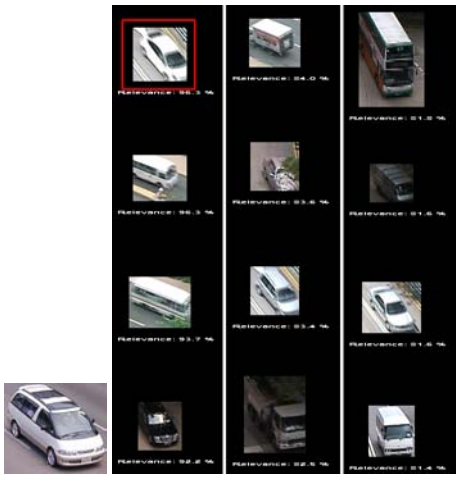
Acknowledgment

This research was jointly sponsored by Verint Systems (Asia Pacific) Limited and the Innovation and Technology Commission of the Government of the Hong Kong Special Administrative Region, under the Grant ITS/072.


(a)


(b)


(c)


(d)

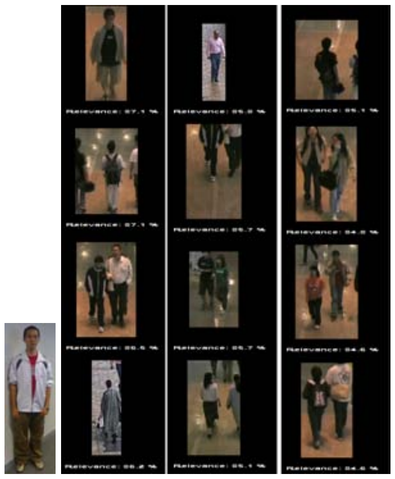

(e)

Fig. 5. Retrieval results by sample snapshots/photo images. The query images are shown at the bottom-left of each sub-figures, and the corresponding snapshots of 12 retrieved objects with the highest relevance R as defined in (10) are shown in the next 3 columns, which are displayed in the order of top-down and then left-right according to R.

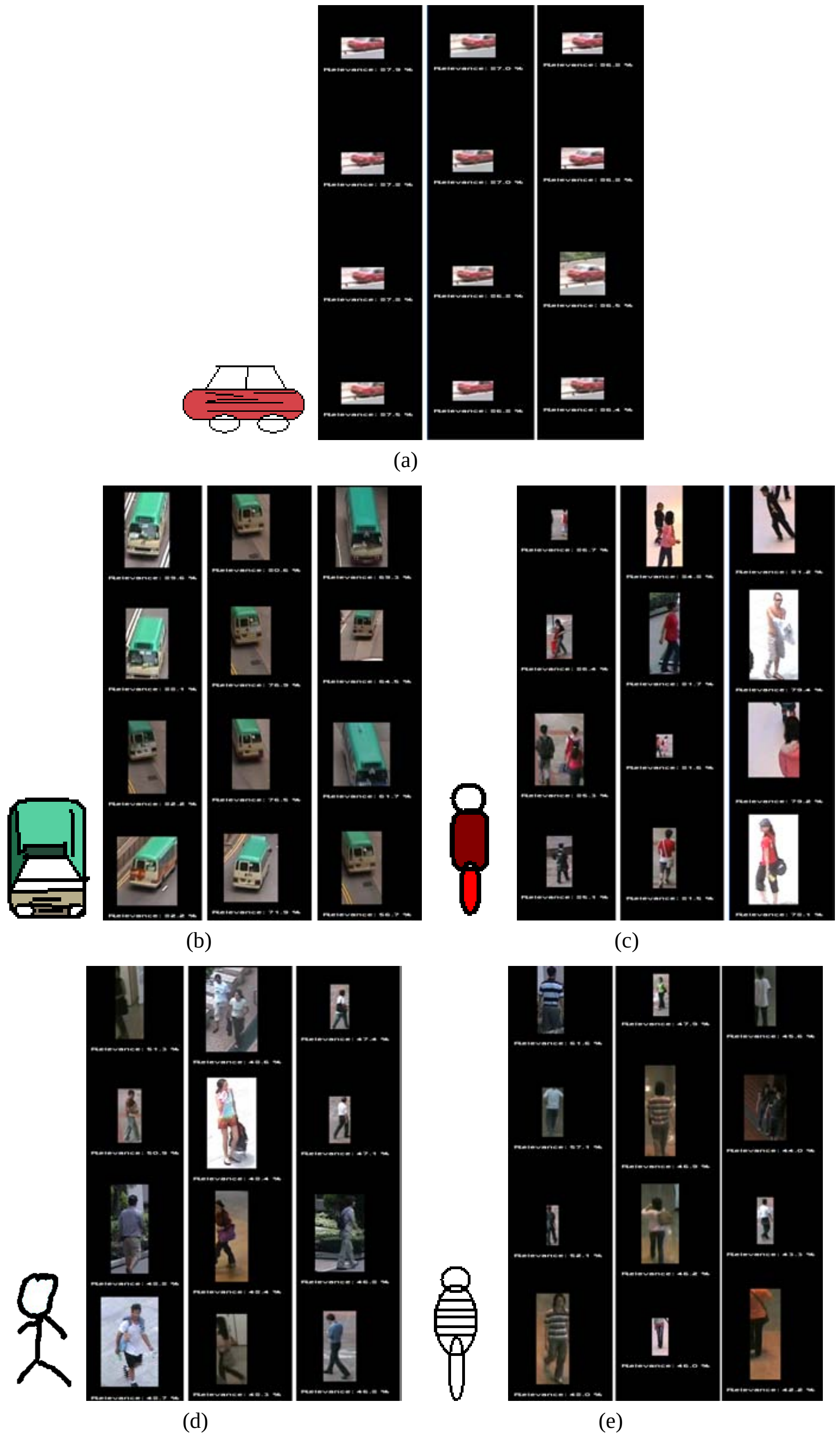


Fig. 6. Retrieval results by sample hand-drawn images. The query images are shown at the bottom-left of each sub-figures, and the corresponding snapshots of 12 retrieved objects with the highest relevance R as defined in (10) are shown in the next 3 columns, which are displayed in the order of top-down and then left-right according to R.