

Video Data Modelling To Support Hybrid Query

Lilly Suriani Affendey[†], Ali Mamat[†], Hamidah Ibrahim[†] and Fatimah Ahmad^{††},

University Putra Malaysia, Serdang, Selangor, Malaysia

Summary

Information contained in unstructured video data needs to be extracted and must be appropriately modeled in order to support storage and content retrieval. A video data model should be expressive enough to capture several characteristics inherent to video. Previous video data models lack connection among the video structure, the semantic and the image contents. Hence the types of queries supported are limited to the data models used. This paper proposes a video data model that would allow users to formulate hybrid queries on different attributes of video. The video data model captures the hierarchical structure of video (sequence, scene, shot and key frame), as well as high-level concepts (object, activity, and event) and low-level visual features (colour, texture, shape and location). With this representation, queries for the content and/or the specific hierarchical structure using similarity-based matching of low-level visual features as well as exact matching of textual attributes are supported. Experiments to compare query formulation using single types against hybrid query showed that hybrid query gives more accurate results. Thus, strengthening the need for a video data model that supports queries on more than one type of video attributes.

Key words:

Video data modelling, video representation, video query, video database, video features.

1. Introduction

Content-based video retrieval requires many changes in a multimedia database management system, mainly in the modelling and querying techniques [1][2]. A video data model should include elements that represent inherent structural properties of video as well as elements that represent the video content. Thus in this multimedia environment, queries can be based not only on exact matching of textual data, but also on degree of similarity of visual features. Therefore, an appropriate query mechanism is required to formulate queries on these properties. The query process basically consists of three interactive steps: query formulation, query processing and query results presentation. This involves finding

expressive methods for conveying what is desired, the capability to match what is expressed with what is there, and ways to evaluate the outcome of the search. There are several ways to formulate queries on video data namely, textually such as SQL-like languages [3][4][5], graphically [6][7], or query algebra [8]. Text-based query methods that rely on string pattern-matching or keyword look-up are not adequate for all types of data, particularly visual and auditory data [9][10]. Therefore, it is not reasonable to assume that all types of multimedia data can be described sufficiently with words alone.

While not abandoning the conventional text-based methods, we propose to integrate them with visual means of expressing and constructing queries based on attributes that are otherwise impossible to express textually. The rest of the paper is organized as follows: The next section gives the background on video data management. Section 3 reviews the video data model. Section 4 describes the video database system architecture. Section 5 discusses the experiments and results. Finally the paper is concluded in Section 6.

2. Background

Along with the growth of the information era and technology, abundant collections of multimedia data such as image, graphic, audio and video became available. Among these media, video contains the most features making it the most expressive and thus very complex to manage [11][12]. Video provides a rich and lively resource for multimedia applications. Video resources can be divided into various categories such as instruction, entertainment, scientific recording, news, etc. However the availability of video resources does not necessarily imply accessibility and manipulability of video data.

Large amount of video data and its audio-visual nature has made their manipulation very challenging. Problems are encountered with respect to the retrieval of the video content. Ordinary retrieval techniques are not suitable for a practical usage within digital video libraries. The high volume of video data makes free browsing almost impossible. It is tedious and time consuming for a user to browse through a huge collection of video clips and find

the desired part. However, the retrieval process can rely on textual annotation which is added as metadata through the semi-automatic cataloguing process. Nevertheless, this process requires a considerable amount of time to annotate. Furthermore, the text associated can sometime be vague and incomplete due to subjective human perception.

Data that characterize the information contained in video data can be called metadata. Metadata is defined to be any data description that "tell us something" about video content that can be used as index terms for video retrieval. Although any suitable representation can be used to represent metadata, text is commonly used. Note that metadata can include other representation such as key frames, colour histograms, feature vector, etc. Like image, video contains visual features such as colour, shape, texture as well as spatial information. In addition, video contains temporal and motion features, audio, as well as semantic content. Semantic content of the video is the idea or knowledge it conveys to the users. Identifying those features will help to develop video data models to represent, index and techniques to access it. The choice and importance of the features depend on the purpose and use of the video data.

Video data management is important since there is a growing interest in querying the vast amount of digital video resources. Management of sequential video clips consists of modelling, indexing, and querying [2][5][13][14].

Modelling is concerned with representing the structural properties of video as well as its contents. Structural elements of video are derived by segmenting the video into small manageable units. Video segmentation or video parsing, also known as temporal segmentation, is the process of detecting boundaries between consecutive camera shots and segmenting video streams into elemental units.

Video indexing has previously been text-based, but recent research has made use of visual feature extraction as a basis for content-based indexing [10][15]. Visual features are low-level image features such as colour and texture. In low-level indexing, visual features are automatically extracted from the video data, organized based on some distance metric and later retrieved using similarity-based matching techniques.

However, the main limitation is the lack of semantics attached to the visual features. Semantics describe video concepts on a higher level, such as the description of

objects, activities and events. And this is usually what an average user has in mind when formulating the query.

As video involves a huge amount of data, the need to efficiently query this data becomes significant. Depending on the underlying data model, video data can be queried based on the structure (shot, scene, etc.) or by their content (descriptive textual data, low-level visual features, etc). Currently content-based video retrieval can be classified into two types, namely, textual annotation-based and visual content-based [16][17][18][19][20]. The following sections describe in detail the on video data modeling.

3. Video Data Model

Modelling video is the process of translating raw video data into an internal representation to capture video semantics. Information contained in unstructured video data needs to be extracted and must be appropriately modeled in order to support storage and content retrieval. A video data model should be able to capture and represent various types of information about multimedia objects and provide a sophisticated representation of video in terms of low-level features, but also high-level concepts. In video data modelling, we can make distinction between the structure and the content of a video. Segmenting the video into small manageable units derives the structural elements of a video. This is known as the structured modelling approach [21]. This approach divides the video sequences into shots as a basic unit of manipulation. Information stored at the shot level typically includes the start and end boundaries, descriptions, dates, locations, and so on. The problem with basic structured modelling approach is the difficulty for the user to retrieve part of a video shot and also to access a particular video shot for different purposes. These problems are mainly due to the lack of contextual information between video shots. To model contextual information, higher-level constructs, called scenes are introduced to group together shots that share some common attributes. Others have supplemented the basic structured model with presentation and logic structures [22]. What contents to present and how to present them is defined in the presentation structure. The logic structure defines the logical relationships among the content objects.

The structured modelling approach has evolved into segmentation-based approach [23]. In this approach, the video data is recursively broken down into scenes, shots and frames. Key frames are extracted from shots and their visual features will then be used for indexing. Each

key frame can be treated as an image, allowing the use of existing techniques, with some extensions, that have been developed to model and query image data.

On the other hand, video can be modeled based on their content. Video content can be categorized in different levels, namely, raw data, low-level visual features and semantic. Raw video data is made up of elementary video units with some general video attributes such as format, frame rate, etc. The low-level visual features include colour, texture, and shape. The semantic are high-level concepts such as objects, events, and activities. Due to the different levels of video content, the requirements for the extraction are diversified. Extraction of visual features is domain independent and can be done automatically. However, the extraction of semantic is rather complex and tedious since it requires domain knowledge and user interaction.

Under the content-based modeling approach we can see two important things. Firstly, feature-based models use extracted features such as colour, shape and texture to represent video content, but they do not provide semantics that describe high-level concepts of a video such as objects, activities and events. Secondly, in the semantic models, textual annotations are used to represent high-level concepts. This approach has drawbacks as well, since the process of video annotation is mostly done manually. This process is not only tedious, subjective to the annotations perception, but also time consuming. Despite the drawbacks, the only way to assign meanings to high-level concepts is by textual annotations. Obviously, an integrated approach that supports both visual and semantic content is a complementary solution. On top of that the structured model provides a good basis for the integration of these two content-based approaches.

Video content can be queried at different levels namely, raw video data, visual features or semantic level. The simplest way to query a video is to query the raw video data directly. For example, a user can select all news videos produced by TV3. Querying on the visual feature level is a common technique used in image retrieval systems. This technique can also be used in video retrieval system where the key frames images are used as indexes. The common approach for querying visual features is through query by example. Querying at semantic level is limited by the annotation process. The descriptions are likely to be incomplete and subjective. Queries may be formulated using the standard query language (SQL) or its extension. Among the query languages reviewed, none support homogeneous querying at different levels. Users are unable to specify query from

different levels in a single query formulation. To support homogeneous querying at different levels the underlying video data model must be able to deal with all different variations of video content as well as to provide appropriate mechanisms for query formulation. Another important characteristic of a video query language is visualization. Formulation of queries through visual query interface is more appropriate especially in multimedia environment where many types of information are inherently visual.

As large amount of video data becomes available, the need to model and query them efficiently is extremely important. We investigate techniques that enable modelling and querying the video structure together with the semantic and low-level visual content. Although video contain other features such as audio and spatio-temporal, we limit our work to the semantic and low-level visual features of video. This is because, since video shot can be represented by one or more images (key frames), a video query can be reduced to the problem of image query. In content-based image retrieval, query is on semantic and low-level features.

The followings present our video data model.

Definition 1:

A video data model is a 5-tuple,

$$(V_Id, V_Clip, V_Attr, \mathcal{R}, I)$$

where:

- V_Id is a unique identifier of an instance of Video,
- V_Clip is a reference to the video clip itself, which can be referenced as an external binary file,
- V_Attr is an attribute component that may be used to describe bibliographic data (eg. title, type, source),
- \mathcal{R} is a set of relations, $\{R_1, \dots, R_k\}$, each of which has an attribute called V_Id in its scheme. These relations contain the description of the hierarchical structure of the video, and
- I is an image table. Image table is a special relation, which contains complex data type representing low-level features of the key frame image.

An example of a video table comprising of V_Id , V_Clip and V_Attr is as follows:

Video(v_id, video_clip, title, creator, category, description, keyword, publisher, contributor, dates, video_type, video_format, source, language, relation, coverage, rights)

where $V_Attr = \{title, creator, category, description, keyword, publisher, contributor, dates, video_type, video_format, source, language, relation, coverage, rights\}$

Since \mathfrak{R} and I are separate relations, their definitions are given below.

Definition 2:

\mathfrak{R} is maintained in a standard relational database as an ordinary relation and contains information about video as a whole.

$$\mathfrak{R} = \{Shot, Scene, Sequence\}.$$

Definition 3:

A shot is a set of contiguous video frames denoted with the *start* and *end* boundaries, representing a continuous action in time and space by a single camera. A shot-instance is a 4-tuple,

$$(V_Id, Sh_Id, Kf_Id, Sh_Attr)$$

where:

- V_Id is a unique identifier of an instance of Video,
- Sh_Id is a unique identifier of an instance of Shot,
- Kf_Id is a unique identifier of an instance of I , and
- Sh_Attr is an attribute component that is used to describe the shot, as well as semantics of the video (eg. object).

An example of a Shot relation is as follows:

Shot (v_id , sh_id , kf_id , $kfbegin$, $kfend$, $kfobject$, $description$, $duration$)

where $Sh_Attr = \{kfbegin, kfend, kfobject, description, duration\}$

Definition 4:

Shots that are taken in the same space, focusing on an object or objects of interest form a *scene*. A scene may contain one or more video shots and/or scenes. A scene is a 3-tuple,

$$(V_Id, Sc_Id, Sc_Attr)$$

where:

- V_Id is a unique identifier of an instance of Video,
- Sc_Id is a unique identifier of an instance of Scene, and
- Sc_Attr is an attribute component that is used to describe the scene, as well as semantics of the video (eg. activity).

An example of a Scene relation is as follows:

Scene (v_id , sc_id , $kfbegin$, $kfend$, $activity$, $description$, $transcript$, $duration$)

where $Sc_Attr = \{kfbegin, kfend, activity, description, transcript, duration\}$

Definition 5:

A series of related scenes form a *sequence*. A sequence is a 3-tuple,

$$(V_Id, Seq_Id, Seq_Attr)$$

where:

- V_Id is a unique identifier of an instance of Video,
- Seq_Id is a unique identifier of an instance of Sequence, and
- Seq_Attr is an attribute component that is used to describe the sequence, as well as semantics of the video (eg event).

An example of a Sequence relation is as follows:

Sequence (v_id , seq_id , $kfbegin$, $kfend$, $event$, $description$, $duration$)

where $Seq_Attr = \{kfbegin, kfend, event, description, duration\}$

Definition 6:

I is an image table, which contains the key frame images and the signature of the visual features namely color, texture, shape and location.

$$I(Kf_Id, Kf_Image, Kf_Signature, Kf_Attr)$$

where:

- Kf_Id is a unique identifier of an instance of I ,
- Kf_Image is a reference to the Key Frame image object itself, which is stored as an external binary file. In our implementations its data type is ORDSYS.ORDImage,
- $Kf_Signature$ is a feature vector representation of the object Kf_Image . In our implementation, its data type is ORDSYS.ORDSignature, and
- Kf_Attr is an attribute component that may be used to describe the object present in the image (if any).

An example of an Image relation is as follows:

Image(kf_id , kf_image , $kf_signature$)

The proposed video data model combines the two important characteristics of video, namely the structure and the content. We only focus on two specific video contents, which are semantic (object, activity, event) and low level visual features (colour, texture, shape and location), since a video query can be reduced to the problem of image query. Other characteristics of video such as motion and audio are beyond the scope of our research work.

Using this video data model, we are able to represent the hierarchical structure of video which are shot, scene and sequence. With this representation, we are able to query for the specific hierarchical structure. In addition we can include searches on video segments that have similar low-level visual features. Since visual attributes are used, the query system should be able to perform similarity-based matching. Furthermore, we can still formulate queries for exact matching of textual attributes. The following section discusses the theoretical foundation used for the combination of exact and similarity-based queries. When formulating hybrid queries in a single mode, the dissimilarities between traditional database queries and multimedia queries require a different solution for evaluating the atomic queries.

3.1 Combining Exact and Similarity-Based Queries

An unweighted query Q consists of n atomic queries q_1, \dots, q_n which are regarded as predicates. A compound query ($n > 1$) combines atomic queries using the binary sentential connectives **and** (\wedge) and **or** (\vee). Beside these connectives the monadic connective, the negation operator **not** (\neg), is likewise important. A query can be defined as follows: $Q := q \mid (Q [\wedge \vee] Q) \mid \neg Q \mid (Q)$.

There are essential differences between querying multimedia data and traditional databases. The most important difference is that a response to a multimedia query typically provides not only a set of objects, but also the grades with which these objects qualify in terms of similarities. This is in contrast to a traditional database where the answer to a query is simply a set of values.

Atomic query for multimedia data is much harder to evaluate than an atomic query in a relational database. For example, when querying images it is reasonable and natural to ask for images that are somehow similar to some example image. In response to a query, a multimedia system might typically return a sorted list of items in the database that match the query best. Hence, there must be a notion of similarity, so that the most similar images are retrieved.

For every object o_j in the database ($o_j \in \text{DB}$) the similarity to the query is expressed by the object's overall score μ . Therefore, every atomic query q_i is evaluated and its score μ_i is determined by a similarity distance function $Dq_i(o_j) = \mu_i$. The overall score for an object is calculated using a scoring rule S_Q . The scoring rule S_Q assigns a numerical value (score) $\mu = S_Q(\mu_1, \dots, \mu_n)$ to each object, based on

the evaluation of the object's scores for each atomic query. A score $0 \leq \mu \leq 1$ describes the degree of similarity between the object in the database and the query. The closer a score to the value 1 is, the higher is the similarity. $\mu = 1$ means the retrieved object is a perfect match and $\mu = 0$ means the object does not fulfill the query at all. The overall score allows a ranking of qualified objects. The higher the score, the better the ranks of an object in the result list.

Another important issue that arises for fuzzy queries in a multimedia database, but not for traditional queries in a standard database is weighting the importance of subqueries. The concept of weighting is used to ensure that users can specify and formulate their preferences regarding their search criteria in an adequate way. The user can assign a weight θ_i with $\theta_i \in I$ and $I = [0, 1]$ to each atomic query q_i . When calculating the overall score μ for an object the weights also have to be taken into account. Therefore, the weights θ_i are incorporated into the underlying scoring rule S_Q . Thus the weighted scoring rule S_Q^\ominus is a function of q and θ :

$$S_Q^\ominus(\mu_1, \dots, \mu_n, \theta_1, \dots, \theta_n) = \sum \mu^* \theta_i.$$

Compound queries are Boolean combinations of atomic queries. In compound query there could be a mismatch where the result of some queries is a sorted list, and for other queries, it is a set. These differences cause us to consider new mechanism to calculate the overall score. How do we combine such queries in Boolean combinations?

A solution for combining traditional database query and multimedia query was proposed in terms of "graded" or "fuzzy" sets [24]. A graded set is a set of pairs (x, g) , where x is an object (such as a tuple), and g (the grade or score) is a real number in the interval $[0, 1]$. We can think of a graded set as corresponding to a sorted list, where the objects are sorted by their grades. Therefore, a graded set is a generalization of both a set and a sorted list.

A number of different rules for evaluating Boolean combinations of atomic formulas in fuzzy logic are given as follows [25]. Consider the standard rules of fuzzy logic, where if x is an object and Q is a query, then let $\mu_Q(x)$ denote the score or grade of x under the query Q .

If we assume that $\mu_Q(x)$ is defined for each atomic query Q and each object x , then it is possible to extend to queries that are Boolean combination of atomic queries via the following rules.

Conjunction rule:

$$\mu_{A \wedge B}(x) = \min \{ \mu_A(x), \mu_B(x) \}$$

Disjunction rule:

$$\mu_{A \vee B}(x) = \max \{ \mu_A(x), \mu_B(x) \}$$

Negation rule:

$$\mu_{\neg A}(x) = 1 - \mu_A(x)$$

Details of formula for incorporating weights in scoring rules can be found in [26].

The next section describes the architecture for the prototype of the proposed video data model.

4. Video Database System (VBDS) Architecture

The Video Database System (VBDS) has three main modules namely the video shot detection, annotation and query interface, which are connected to an object-relational database management system. Since the main focus of this paper is on modeling, it will not discuss in detail the modules of VBDS. Figure 1 shows the VBDS architecture.

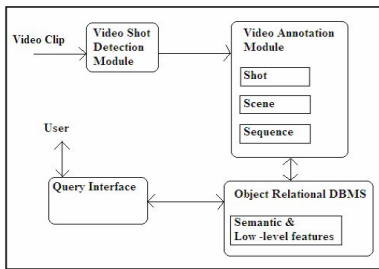


Figure 1. Video database system (VBDS) architecture

The rest of this section will describe on the Query Interface. The query interface has three tabs that support query by text, image and the combination of both text and image attributes as shown in Figure 2 , Figure 3 and Figure 4 respectively.

The Query by Text interface as shown in Figure 2 supports query for bibliographic attributes of video (eg. video identifier) using exact matching of the descriptive textual data (eg. object).

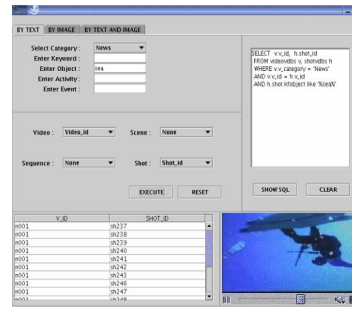


Figure 2: The Query by Text Tab

A user selects a video category and enters one of the semantic attributes for matching. He/she then needs to select the bibliographic attributes (eg. video id, shot id as shown in Figure 2). By clicking the *SHOW SQL* button, the corresponding SQL statement will be generated in the text area on the upper right hand side of the interface. If the user clicks the *EXECUTE* button, the SQL statement will be executed and the query result will be displayed in the lower left hand section. If the user selects a row, the particular video segment can be played in the media player in the lower right hand section.

The Query by Image interface as shown in Figure 3 supports query by image similarity. Query formulation starts by the selection of a video category. Depending on the category selected, a set of example images will be displayed in the form of thumbnails. The user then chooses an image to be used as an example for the query. The colour, shape, texture and location sliders can be adjusted to indicate matching similarity. The accuracy slider is used to adjust the threshold value, which will be used to compute the degree of similarity. Next the user can click on the *SHOW SQL* button to see the corresponding SQL statement. If the *EXECUTE* button is clicked, the SQL statement will be executed and the query result will be displayed in the lower left section. A user can choose to play the corresponding shot by clicking on the row in the result list.

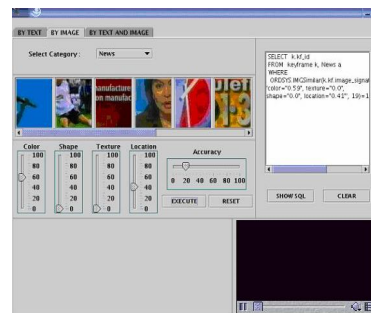


Figure 3 : The Query by Image Tab

The Query by Text and Image interface as shown in Figure 4 supports hybrid query formulation using a combination of text and image features in a single query mode.



Figure 4. The query by text and image tab

Through this interface a user can combine querying using semantic and image features. Query formulation starts by the selection of a video category. A user can then enter one of the semantic attributes for matching. He/she can choose the bibliographic attributes to retrieve, an example image to use and then adjust the values of the sliders for similarity matching. Next the user can click on the SHOW SQL button to see the corresponding SQL statement. If the EXECUTE button is clicked, the SQL statement will be executed and the query result will be displayed in the lower left section. A user can choose to play the corresponding video segment by clicking on the row in the result list.

5. Experiments and Results

To illustrate the query formulation supported by VDBS, a news segment broadcasted by a local television channel was used as an example. Each segment can be annotated based on the visual as well as audio features. The news video is a 30 minutes clip and to generate the key frame images, the video was preprocessed using the Video Shot Detection module. A total of 352 shots represented by 352 key frames were obtained. Visual features are extracted from each key frame and they are stored in the database. Using the Video Annotation Module each of the 352 shots was annotated. Objects, activities, events as well as description of shots are stored in the database.

The rest of this section shows several examples of query formulation and the results. For each category of query, namely Query by Text, Query by Image and Hybrid Query, three examples will be shown. In these examples, the shot id, key frame id, object and key frame images are presented as the query results.

Based on our experiments on query formulation using single types and hybrid query mechanism in the previous section, we will discuss the results that we have obtained. For each query using different query types we plotted graphs to compare the number of images retrieved for various threshold values.

Figure 5 shows a comparison between query by image and hybrid query for 'sea'. The graph plots the threshold values against the number of images retrieved. The result shows the Hybrid Query type gives better results compared to Query by Images since the numbers of images retrieved were reduced. The result of the Query by Text was not shown in the graph as the threshold values are not relevant to this query type. Nevertheless, in the Query by Text there were 16 images retrieved. When compared to Query by Text, the Hybrid Query results was the same for threshold values 14 and above. However, by lowering the threshold value, the number of images retrieved was reduced.

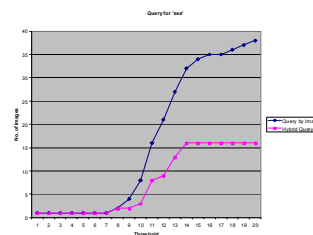


Figure 5: Comparison between Query by Image and Hybrid Query for 'sea'

Figure 6 shows a comparison between Query by Image and Hybrid Query for 'football'. The graph plots the threshold values against the number of images retrieved. The result shows the Hybrid Query type gives better results compared to Query by Image. In the Query by Text there were 33 images retrieved. When compared to query by Text, the Hybrid Query also shows better results where the number of images retrieved was less than 33. In fact by reducing the threshold values, the number of images retrieved was also reduced.

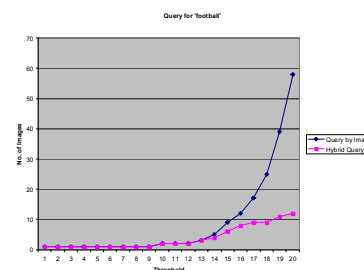


Figure 6: Comparison between Query by Image and Hybrid Query for 'football'

Figure 7 shows a comparison between Query by Image and Hybrid Query for ‘woman’. The graph plots the threshold values against the number of images retrieved.

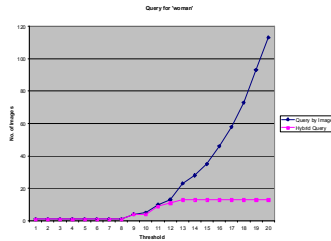





Figure 7: Comparison between Query by Image and Hybrid Query for ‘woman’

Again, the result shows the Hybrid Query type gives better results compared to Query by Image. In the Query by Text there were 18 images retrieved. When compared to query by Text, the Hybrid Query also shows better results where the number of images retrieved was less than 33. In fact by reducing the threshold values, the number of images retrieved was also reduced.

Based on these three comparisons, we summarize the results for the three query types. Table 1 summarizes the query results for the three types of query formulation based on colour weight 0.6 and location weight 0.4.

Table 1: Query results for the three types of query formulation

Query Types Query Object	Text (semantic)		Image (low-level visual features) Colour:0.6, Location:0.4		Hybrid Colour:0.6, Location: 0.4	
	#	%	#	%	#	%
 37701 sea	16	4.55	Threshold:20		Threshold:20	
			38	10.79	16	4.55
			Threshold:7		Threshold:7	
			1	0.28	1	0.28
 43621 football	33	9.38	Threshold:20		Threshold:20	
			58	16.48	12	3.41
			Threshold:7		Threshold:7	
			1	0.28	1	0.28
 1292 woman	18	5.11	Threshold:20		Threshold:20	
			113	32.10	13	3.69
			Threshold:9		Threshold:9	
			1	0.28	1	0.28

Note: The figures show the numbers(#) and percentages(%) of images retrieved

6. Conclusions

Modelling video is an important step in video data management. Previous video data models lack connection among the video structure, the semantic and the image contents. Hence the types of queries supported are limited to the data models used.

We present a video data model that captures the hierarchical structure and contents of video. Based upon the segmentation-based approach, our model considers video structure to be made of sequences, scenes and shots. Meanwhile, based upon the content-based approach key frame images are used to represent each shot. The low-level visual features of these images are extracted to represent image content. In addition, we annotate sequence, scene and shot for semantic contents. A combination of these two approaches has enabled the followings:

- a) Capture the hierarchical structure of video, namely, sequence, scene, shot and key frame.
- b) Capture both semantic and low-level visual features of video.
- c) Allow queries on video segments, semantic as well as low-level visual features.

Through the prototype VDBS, we were able to compare the query results of single type query formulation against those obtained from hybrid query formulation.

References

- [1] T. Lee, L. Sheng, T. Bozkaya, N.H. Balkir, , Z.M. Ozsoyoglu, G. Ozsoyoglu, “Querying Multimedia Presentations Based on Content,” *IEEE Transactions on Knowledge and Data Engineering*, Vol 11, No.3, pp. 361-385, 1999.
- [2] C. Declair and M.S. Hacid, “Modeling and Querying Video Data: A Hybrid Approach.” Proceedings of the IEEE Workshop on Content-based Access of Image and Video Libraries CBAIVL’98, Santa Barbara, USA, 1998.
- [3] S. Adali, K.S. Candan, S. Chen, K. Erol, and V.S. Subrahmanian, “The Advanced Video Information System: Data Structures and Query Processing”, *ACM Multimedia Systems Journal*, Vol 4, pp.172-186,1996.
- [4] E. Hwang, and V.S. Subrahmanian, “Querying Video Libraries,” *Journal of Visual Communication and Image Representation*, Vol 7, No. 1, pp. 44-60, 1996.
- [5] M.E. Donderler, “Data Modeling and Querying for Video Databases”, Ph. D. Thesis, Bilkent University, 2002.
- [6] J. H. Lim, “Explicit Query Formulation with Visual Keywords.” *ACM Multimedia 2000*, pp. 407-412. Los Angeles, CA USA, 2000.
- [7] J. Assfalg, A. D. Bimbo, and M. Hirakawa, “A Mosaic-based Query Language for Video Databases.” Proceedings of The IEEE International Symposium on Visual Languages (VL’00) pp. 23, 2000.

- [8] R. Hjelsvold, R. Midstram, and O. Sandsta, "Searching and Browsing a Shared Video Database." *Multimedia Database Systems: Design and Implementation Strategies*, pp. 89-122. Kluwer Academic Publishers. 1996.
- [9] G. Ahanger, D. Benson, and T.D.C. Little, "Video Query Formulation," *Storage and Retrieval for Image and Video Database III, SPIE 2420*, pp. 280-291, 1995.
- [10] Y. Rui, T.S. Huang, and S. Chang, "Image Retrieval: Past, Present, and Future," *Journal of Visual Communication and Image Representation*, Vol 10, pp.1-23. 1999.
- [11] J. C. Lee, Q. Li and W. Xiong, VIMS: A Video Information Management System. *Multimedia Tools and Applications* 1997; 4:7-28.
- [12] D. Poncelion, S. Srinivasan, A. Amir, D. Petkovic, and D. Diklic, Key to Effective Video Retrieval: Effective Cataloging and Browsing. In Proceedings of the ACM Multimedia 1998; 99-107.
- [13] Zhang, H. J., Low, C. Y., Smoliar, S. W., and Wu, J.H. Video Parsing, Retrieval and Browsing: An Integrated and Content-Based Solution. In Proceedings of the third ACM International Conference on Multimedia 1995, San Francisco, California: 15-24.
- [14] W. Al-Khatib, Y. F. Day, A. Ghafoor, and P. B. Berra, Semantic Modeling and Knowledge Representation in Multimedia Databases. *IEEE Transactions on Knowledge and Data Engineering* 1999;11(1):64-80.
- [15] K. Tanaka, Y. Ariki, and K. Uehara, Organization and Retrieval of Video Data. In Proceedings of the IEICE Transactions on Information and System 1999; E82-D (1):34-44.
- [16] H. Jiang, D. Montesi and A.K. Elmagarmid, VideoText Database Systems. In Proceedings of the IEEE Multimedia Systems, ICMCS 1997; 344-351. Ottawa, Canada.
- [17] S. Dagtas, W. Al-Khatib, A. Ghafoor, and A. Khokhar, Trail-Based Approach for Video Data Indexing and Retrieval. In Proceedings of IEEE Multimedia Computing and Systems, 1999; Vol. 1:235-239.
- [18] O. Marques, and B. Furht. Issues in Designing Contemporary Video Database Systems. In Proceedings of the International Conference on Internet and Multimedia System and Applications (IASTED) 1999, Nassau, Bahamas.
- [19] R. Tusch, H. Kosch, and L. Boszormenyi, VIDEX: An Integrated Generic Video Indexing Approach. In Proceedings of the Eighth ACM International Multimedia Conference, California, USA 2000; 448-451.
- [20] Fan, J., Luo, H., Elmagarmid, A., Concept-Oriented Indexing of Video Databases: Towards Semantic Sensitive Retrieval and Browsing. In Proceedings of IEEE Transactions on Image Processing 13, 2004; 7:974-992.
- [21] T.S. Chua, and L.Q. Ruan, "A Video Retrieval and Sequencing System," *ACM Transactions on Information Systems*. Vol 13, No. 4, pp. 373-407, 1995.
- [22] Y. Tonomura, A. Akutsu, Y. Taniguchi, and G. Suzuk, "Structured Video Computing." *IEEE Multimedia*, Vol 1, No. 3, pp. 34-43, 1994.
- [23] L. Chen, T. Ozsu, and V. Oria, "Modeling Video Data For Content Based Queries: Extending the DISIMA Image Data Model," Proceedings of 9th International Conference on Multimedia Modeling, Taiwan, pp. 169-189, January 2003.
- [24] R. Fagin, "Combining Fuzzy Information from Multiple Systems," Proceedings of ACM Symposium on Principles of Database Systems (PODS'96), pp. 216-226, Montreal, Canada. 1996.
- [25] R. Fagin, "Fuzzy Queries in Multimedia database Systems," Proceedings of 17th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS'98), pp. 1-10, Seattle, USA, 1998.
- [26] R. Fagin and E.L. Wimmers, "A Formula for Incorporating Weights into Scoring Rules", *Journal of Theoretical Computer Science* Vol 239, pp. 309-338, Elsevier Science, 2000.



Lilly Suriani Affendey received her Bachelor of Computer Science from University of Agriculture, Malaysia in 1991 and M. Sc in Computing from the University of Bradford, UK in 1994. 1999, respectively. In 2006 she received her PhD from University Putra Malaysia. Her research interest is in Multimedia Databases. She is currently a lecturer in University Putra Malaysia.