Content-Based Indexing and Retrieval Using MPEG-7 and X-Query in Video Data Management Systems

Dian Tjondronegoro and Yi-Ping Phoebe Chen

Published:
World Wide Web: Internet and Web Information Systems, 5, 207–227 (2002)

Current advances in multimedia technology enable ease of capturing and encoding digital video. As a result, video data is rapidly growing and becoming very important in our life. It is because video can transfer a large amount of knowledge by providing combination of text, graphics, or even images. Despite the vast growth of video, the effectiveness of its usage is very limited due to the lack of a complete technology for the organization and retrieval of video data. To date, there is no "perfect" solution for a complete video data-management technology, which can fully capture the content of video and index the video parts according to the contents, so that users can intuitively retrieve specific video segments. We have found that successful content-based video datamanagement systems depend on three most important components: key-segments extraction, content descriptions and video retrieval. While it is almost impossible for current computer technology to perceive the content of the video to identify correctly its key-segments, the system can understand more accurately the content of a specific video type by identifying the typical events that happens just before or after the key-segments (specific-domainapproach). Thus, we have proposed a concept of customisable video segmentation module, which integrates the suitable segmentation techniques for the current type of video. The identified key-segments are then described using standard video content descriptions to enable content-based retrievals. For retrieval, we have implemented XQuery, which currently is the most recent XML query language and the most powerful compared to older languages, such as XQL and XML-QL.

Keywords: content-based video data management, key-segments extraction, MPEG-7 content descriptions, XQuery retrieval

1. Introduction

Current rapid advances of the Internet and Multimedia technology enable huge amount of digital videos to be produced or downloaded, stored and organized, before they are retrieved, or shared on a daily basis. This phenomenon triggers a fast growth of video data, and thus, we need to have a content-based Video Data Management System, which can index the video according to its content, to allow users browse the video effectively. Content-based VDMS requires video indexing process, where video is broken into smaller and more manageable chunks before they are summarised using a description scheme. The success of current video indexing technology is limited because complete solution for automatic video indexing is not available. Automatic video indexing technique is essential to reduce the required time and human work in indexing large video data. For example, it would be very cumbersome and error-prone to manually scan through a long video to manually identify the key-segments and describing them. However, the biggest problem lies in the fact that computer is simply not a human, like us. It does not actually understand the content. Instead all it does is "computerizing" the semantic meaning of the video, by examining the features of a video, such as the characteristics of the audio and its moving pictures. Thus, computer relies on the computerized understanding of the video to extract the key-segments, as well as labelling them. However, it is important to note that each different video type has different features and characteristics. For example, only sport videos have crowd cheers and commentator's voice.

The unavailability of a complete video indexing solution is also a consequence of the fact that although previous video indexing researches have offered various possible videoindexing solutions, their efforts have not been coordinated and standardised. For instance, some researches focus on analysing the sound feature of a video to automatically detect important video segments, while others may use video's moving image or artificial texts characteristics. However, this is reasonable since automatic video indexing is still a very large and complex issue, involving multi-disciplinary fields. For example, automatic audio analysis for video segmentation may require the formulation of audio characteristics, and speech recognition.

Hence, we acknowledge the importance of specific-domain approach, which aims to focus on a particular video type to improve the accuracy and effectiveness of the contentbased VDMS for that particular type of video. Later, the specific-domain VDMS can be adjusted or modified to suit other domains, particularly other domains that have a close relationship in terms of the video characteristics. In this paper, we will show how we can benefit from specific-domain approach, by showing how we implemented the approach in designing our content-based VDMS for soccer videos. We focus our discussions on its most important three parts: key-segment extraction, video content descriptions, and the retrieval. Please note that our designed VDMS uses the currently active-evolving technologies of MPEG-7 for our video content descriptions, and XQuery for the retrieval, thus, this paper only reflects their current status.

2. Related works

Some recent studies have already used specific-domain approach in developing video segmentation techniques [2,5,7–9,11,12,14] as well as developing solutions for content video description schemes [1,4,6,10,13] and XML document retrieval [3].
In order to automate content-based video segmentation, Nepal et al. [9] investigated the typical events for detecting goals during basketballmatch, to build temporal models, which drives the decision to choose audio-visual features that can be used to automatically detect the typical events. Zhou et al. [14] use a similar approach to use a rule-based indexing system for basketball video. Rui et al. [12] used an experiment with baseball games to test the accuracy of detecting the combination of two highly correlated audio aspects, which are highly correlatedwith exciting events: commentator's excited speech and baseball pitch and hit.

Gong et al. [5] presented a method on segmenting soccer match according to the court position (for example, in midfield, on left corner area, and so on). Their method is firstly to detect the court white line, then eliminate the noise (e.g., from the players), and trim the line to fix broken lines due to player occlusions, and finally match the line captured with a predefined pattern to justify which part of the court is contained by a particular scene. Lienhart [8] has proposed some methods to analyse the information presented by the text displayed on the video frames.

Pan et al. [11] used a HiddenMarkovModel (HMM) to model slow motion replays, and uses an inference algorithm to compute the probability of a slow motion replay segment, as well as localising the boundaries of the segment. Babaguchi et al. [2] proposed to

detect slow motion replay by detecting pairs of DVEs (Digital Video Effects), perceived as scene transition, which usually displayed before and after a slow motion replay segment. After detecting the replays, they search the actual live scene (before the replay), which is typically a scene with typical framing of shot from a fixed angle of a main camera (e.g., in soccer the main camera is the top-view camera). Kobla et al. [7] proposed a technique to identify slowmotion replays applied directly to an MPEG compressed video. Theirmethod uses the macro block, motion and bit-rate information, which are directly accessible from MPEG video with very minimal decoding, and thus increasing the processing speeds. Avaro et al. [1] has summarised the overview of MPEG-7 and Pfeiffer et al. [10] used MPEG-7 for their TV anytime application, which showed how to describe a program summary, using a soccer match description as an example, while Fatemi et al. [4] used MPEG-7 to index and retrieve TV news program. Van Beek et al. [13] used soccer match descriptions to give an example of the MPEG-7 Multimedia Description Schemes (MDS) semantic content descriptions and summary DS. Moreover, Hunter et al. [6] showed how to use content descriptions in describing news video. Since MPEG-7 has decided to use XML, we found XQuery in its working draft proposed by Boag et al. [3].
Based on these previous works on video segmentation techniques, we will propose "customisable segmentation module" in the next section.

3. Customisable video segmentation module

Since the first requirement of our designed content-based VDMS is key-segments extraction process, we investigated the typical events that happen before and after a goal is scored (in a soccer match), and we have found the following typical events to develop some keyevent models for a soccer video. These models can be used to improve the accuracy of detecting the goal segments within a soccer video.

Excitement in the audience and the match commentator's voice.

• In a soccer match, when exciting events happen, the level of crowd noise becomes much louder, and commentator's voice raises its tone (indicates the excitement). Moreover, when interesting events happen, commentator indicates that to the viewer, e.g., ". . .quite a superb effort. . ." or when goal is scored, commentator will say, "That's a goal! What a marvellous goal! He scores!"

• If a goal is not scored (i.e. only attempt for goal happened), the crowd cheer loudness is not sustained and commentator's voice is back to normal. On the other hand, when a goal is scored, the crowd noise is sustained for more than 3 seconds and the commentator's voice sometimes also very loud or the tone is emphasizing. However, there are some occasions when crowd noise is very loud during a normal play; because the supporters are cheering their team to score or to maintain a good result.

The camera is currently focused on showing the goal.

• Soccer has a very big court and involves many players. Thus, the camera mainly shows top-views (tower view), and follows on the ball's movement. Thus, when the video frames show the court are near the goal for longer than 2 seconds, usually interesting event happens and a goal is more likely to be scored.

Slow motion replays and the scoreboard display.

• When interesting events just happened, a slow motion replay usually follows the event. When a goal is scored, definitely slow motion replays will be played for 2 until 4 times, depending on the editing decision (basically try to use all the best camera angles to view the goal). Following the slow motion, a scoreboard is displayed within the next 2–7 seconds. It is also possible that the replay comes together with the scoreboard display. However, scoreboard display and slow motion replays vary significantly depending on the editing works done by the TV broadcasting. For example, some soccer videos always display the scoreboard throughout the game. Thus, the only way to detect the goal is when the scoreboard is just updated. Nevertheless, usually in this case, when a goal is just scored, a bigger text display is shown to tell viewers who just scored the goal and the timestamp when the goal is scored.

3.1. Key event models

Based on these findings, some key-event models of goal segments in a soccer match can be developed. To understand our key-event models, it is important to note that a key-event model is generally defined as follows:

Pre-con → [n seconds] → Event → [n seconds] → Post-con.

• Pre-con indicates the event that happens before the key-segment.
• [n seconds] indicates the time gap between two different events.
• Event indicates the key event, which is contained as the key-segment.
• Post-con indicates the event that happens after the key-segment.

Thus, since our event is goal, please note that in our models any events included before the event is pre-con, consequently, the events included after goal is post-con.

Model 1.

Crowd cheer → [0.5 seconds] → Excited speech from commentator
→ [2 seconds] → Goal
→Crowd cheer and commentator's loud voice are sustained for 3 seconds.

The limitation of this model is the events where crowd cheer is loud and commentator's voice is raised due to an attempt for goal or a foul just being committed or when there is a fight going on. Thus, there are a lot of reasons other than a goal scored that may affect crowd cheer and commentator's voice.

Model 2.

Position of play near the goal → [2 seconds] → Goal.

It is very obvious that this model is not always true, as there are more cases when only attempts on goal happen, but no goal is scored yet. Also, when capturing events that happen near the goal, such as corner kicks, free kicks and penalty kicks, the camera view is focused to the court parts near the goal. However, this model is most useful when we try to locate "interesting plays" in a soccer match since they are mostly likely to occur when the position of play is near goal.

Model 3.

Goal → [5 seconds] → Slow motion replay → [5 seconds] → Scoreboard display.

The limitation of this model is slow motion replay also displayed after other interesting events, such as fouls and attempts for goal. Scoreboard display may be displayed just to keep the viewers up-to-date with the current score state. Thus, it is important to ensure that the scoreboard display is updated to reveal that a goal has been scored.

Model 4 is the combination of models 1 and 2.

Model 5 is the combination of models 2 and 3.

Model 6 is the combination of models 1 and 3.

Model 7 is the combination of models 1, 2 and 3.

For example, model 4 would look like the following:

Crowd cheer → [0.5 seconds] → Excited speech from commentator

→ [2 seconds] → Position of play near the goal → [2 seconds] → Goal
→Crowd cheer and commentator's loud voice are sustained for 3 seconds

As we have shown, each model is not always accurate, thus, models 4–7 try to combine each model in order to overcome the limitations of each model. However, the more model to be processed, then inevitably the more the workload would be. Hence, while we can experiment on combining the models, we should consider the resources consumed (processing time, memory space taken, etc.) to process each of the models. The goal should be to minimize the work load, but still can achieve an acceptable accuracy. Hence, a future work needs to be done to analyse the processing time of each model, so that we can decide on the best way to combine these models.

After we have recognised the typical events, which are revealed in key-event models, we use these models to integrate the concept of some segmentation techniques in detecting each of the typical events. First, we designed some sound analysis techniques to detect the crowd cheer and excited commentator's voice, as well as the technique to detect the keywords within the speech that mentions something about a goal just happened. Second, we designed the method to detect the editing works that separate the slow motion replay scenes before and after it were played, to locate the slow motion scenes. Then since we knowthat a goal segment is before its replay scenes, we can combine the live scene with the replay scenes to get the whole goal segment. Third, we designed the techniques to identify those video shots that reveal the position of the play, to then recognise the line marks on the soccer field, to determine the position of play. Finally, we also took the benefit of the techniques to analyse the text in the score board, so that the system can determine whether the score status has been updated (different from previous score board displayer) since when a goal is just scored, the scoreboard display will reveal the updated score board.

Our integrated segmentation techniques are then encapsulated in a segmentation tools library, which is used for our customisable video segmentation module concept. We will discuss our concepts using the diagram in Figure 1 and a process illustration, which will describe the purpose of each component in terms of how they can work cooperatively within our segmentation module. Please note that in this diagram, squares represent the data components, while ellipses represent the process components.

3.2. Process illustration

The following illustration is an example on how our customisable video segmentationmodule works. The components of the video segmentation module will be highlighted as italic texts. In addition, the content examples for each component are described following the process illustration.

Preprocessing. The live video input (e.g., steaming video from TV) is digitized and recorded in video buffer. The classification tool checks the user profile, which tells to filter the contents of the buffer, by selecting only soccer videos.

Highlighting process (5 steps!). First, segmentation agent looks at the user profile, which tells to only select goal segments as highlights. If there is no user preference that has been input, agent asks user to enter interactively their options. Users can let agent to use the latest usage history to determine what they want.

Second, segmentation agent checks the video storage if there is a non-highlighted soccer video in the video storage. If all soccer videos have been highlighted, segmentation agent will not continue processing, or ask if user wants to re-do the highlighting process.

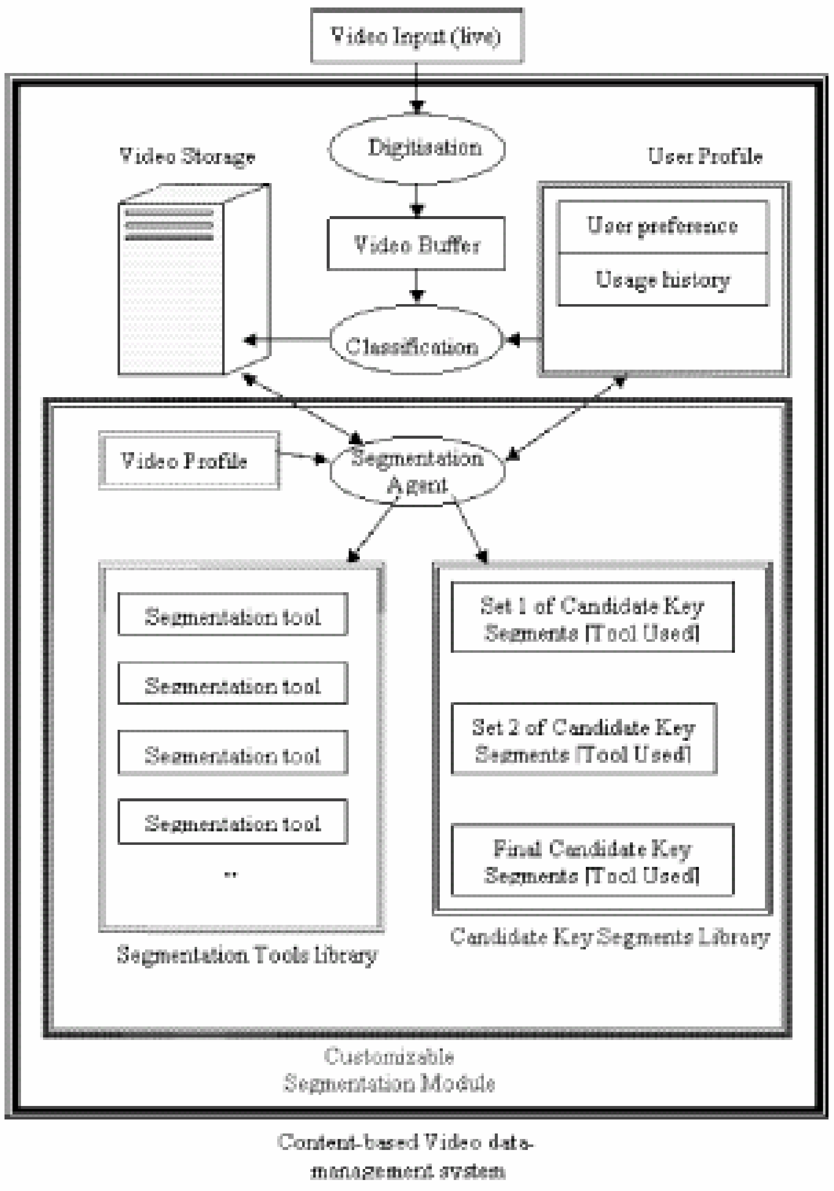

Video Input (live)
Digitisation
Video Storage
User Profile
Video Buffer
User preference
Usage history
Classification
Video Profile
Segmentation Agent
Segmentation tool
Segmentation tool
Segmentation tool
Segmentation tool
..
Set 1 of Candidate Key Segments [Tool Used]
Set 2 of Candidate Key Segments [Tool Used]
Final Candidate Key Segments [Tool Used]
Segmentation Tools library
Candidate Key Segments Library
Customizable Segmentation Module
Content-based Video data-management system

Figure 1. **Customisable video segmentation module.**

Third, segmentation agent selects the video profile for soccer video. Video profile contains the typical events for detecting goal segment; in order to justify what segmentation techniques (from the segmentation tools library) can be used.

Fourth, segmentation agent checks the user profile to determine what type of processing user wants, e.g., quick processing, moderate or slowest. Similarly, if no user preference determines this, users are asked to enter their options, otherwise they can let agent to use the latest usage history.
Fifth, segmentation agent uses quick processing that utilizes sound and text analysis from the segmentation tools library. Sound analysis will produce set 1 of candidate key segments, which will be used as the input for text analysis to confirm if the candidates

were key events. Set 2 (final set in this case) of candidate key segment is produced after eliminating false detections in set 1. Final set of candidate key segments is the final output that users perceive as the soccer highlights.

To further clarify our illustration above, we give some presumable examples for the values of the inputs and outputs for the segmentation module at the time of processing as follows.

Input
User profile:
User preference:
◦ Only soccer match to be recorded.
◦ Highlights goal segments in a soccer video.
◦ Do quick processing.
Usage history: not needed since there is a user preference.
• Video storage:
Some full-length soccer videos.
• Video profile:
Soccer video profile.
• Events model for detecting goals in a soccer video:
    ◦ Model 1: Position of play near the goal→[5 seconds]→Goal.
    ◦ Model 2: Crowd cheer→[1 second]→Commentator's voice raised →[2 seconds] → Goal → Crowd cheer is sustained and commentator's voice is loud for >3 seconds.
    ◦ Model 3: Goal → [5 seconds] → slow motion replay → [7 seconds] → scoreboard display.
• Segmentation tools that can be used:
    ◦ Fastest detection: sound analysis, text analysis.
    ◦ Moderate: fastest + slow-motion replay detection.
    ◦ Slowest: moderate + position of play analysis.
• Segmentation tools library.
• Sound analysis package.
• Position of play identifier.
• Text analysis.
• Slow motion replay detection.
Input–output
Set of candidate key segments (please note that key segments are the output of the segmentation techniques, and also used as the input for the next video segmentation techniques).

4. Using MPEG-7 descriptors to summarise the content of a video

To enable users to retrieve the video segments according to their contents, key-segments are labelled to summarise the content of a video. For this purpose, we have investigated MPEG-7, which is proposed by the World Wide Web Consortium (W3C) to become the standard multimedia description scheme and has become a Final Committee Draft on September 2001. MPEG-7 tries to standardise the Description Definition Language (DDL), which defines the set of descriptors that we can use or modify for our purposes. The descriptors can be used for describing the content of video and audio clips, and the description scheme provides the structure to combine those descriptors to make the video descriptions. So for example, we can describe video by its audio and video clips, and each of video clips may use both the audio and visual descriptors. After we investigated the generic video description schemes provided by MPEG-7, we will propose our video summary scheme in this section. It is important to note that we have added some domain-specific annotations that are specific for soccer videos, and can be used generally for other team-based sports, to enable more precise retrieval (for example, we can query "what is the final score of the game?"). However, while we try to benefit specific annotations, our schema is still robust since we use the "standard" MPEG-7 structured annotations wherever possible (e.g., who, what action). Moreover, highlights components in this diagram are what we have been referring as key-segments. The following are the components of our video summary (as illustrated in Figure 2):

• A video summary contains an overall summary and a hierarchical summary.
• Overall summary describes the whole content of the video, and provides the link to the full-length video.
• Hierarchical summary structures multiple highlights, for example, we can group goal 1, goal 2, into a highlight summary named "goals."
• Similarly to overall summary, highlight segment descriptions include the link to the media, while describing the contents.
• Each highlight can link to key video clip and key audio clip. Within those two we may also have the link to the key frame and key sound. For example, for a soccer goal segment, the key-frame may show the face of the goal scorer, and the key sound is when the commentator says that the player has just scored.
• We have shown here that sequential summary can be combined to the descriptions, to describe the properties of the key frame, key sound or the texts displayed.

To show how our video summary works, the following MPEG-7 based description schema is proposed for a Video Sport Library.

Preamble (wrapper)
```
<schema xmlns="http://www.w3.org/2000/10/XMLSchema"
      xmlns:mpeg7="http://www.mpeg7.org/2001/MPEG-7 schema"
      targetNameSpace="[the URI by which the current schema is to be
           identified]/VIDSchema"
      elementFormDefault="unqualified"
      attributeFormDefault="unqualified">

      <import namespace="http://www.mpeg7.org/2001/MPEG-7 Schema">
```

Sport Video Library

```
<element name="SportVidLib">
      <complexType name="SportVideoLibraryType">
           <sequence>
                <element name="video" type="VideoSummaryType"
                     minOccurs="1"
                maxOccurs=unbounded"/>
           </sequence>
      </complexType>
</element>

<complexType name="VideoSummaryType">
      <sequence>
           <element name="OverallSummary"
                type="vid:OverallSummary Type"/>
           <element name="HierarchicalSummary"
           type="vid:HierarchicalSummaryType"/>
      </sequence>
      <attribute name="genre" use="required" type="string">
      <attribute name="type" use="required" type="string">
</complexType>
```

We illustrated the following definitions, which are used for our schema.

Name	Definition
`SportVidLib`	An element which instantiates `VideoStorageType.`
`SportVideoLibraryType`	A complex type that contains 1 or more video summaries, whose genres are "sport."
`VideoSummary`	Element of the sport video library which instantiates the `VideoSummaryType.`
`VideoSummaryType`	A complex type that summarises a video, whose genre is "sport."
`OverallSummary`	Element of `VideoSummary` that describes the overall content of the video, and contains the link to of the full-length video.
`HierarchicalSummary`	Element of `VideoSummary` that describes the key events in a soccer match (e.g., score), and includes the links to the highlights in the full length video.
`Genre`	Defines the genre of the video (e.g., sport, news, movies, etc.).
`Type`	Defines in more detail the type of the video. For example, if the genre is "sport," the type can be soccer, basketball, rugby, etc.

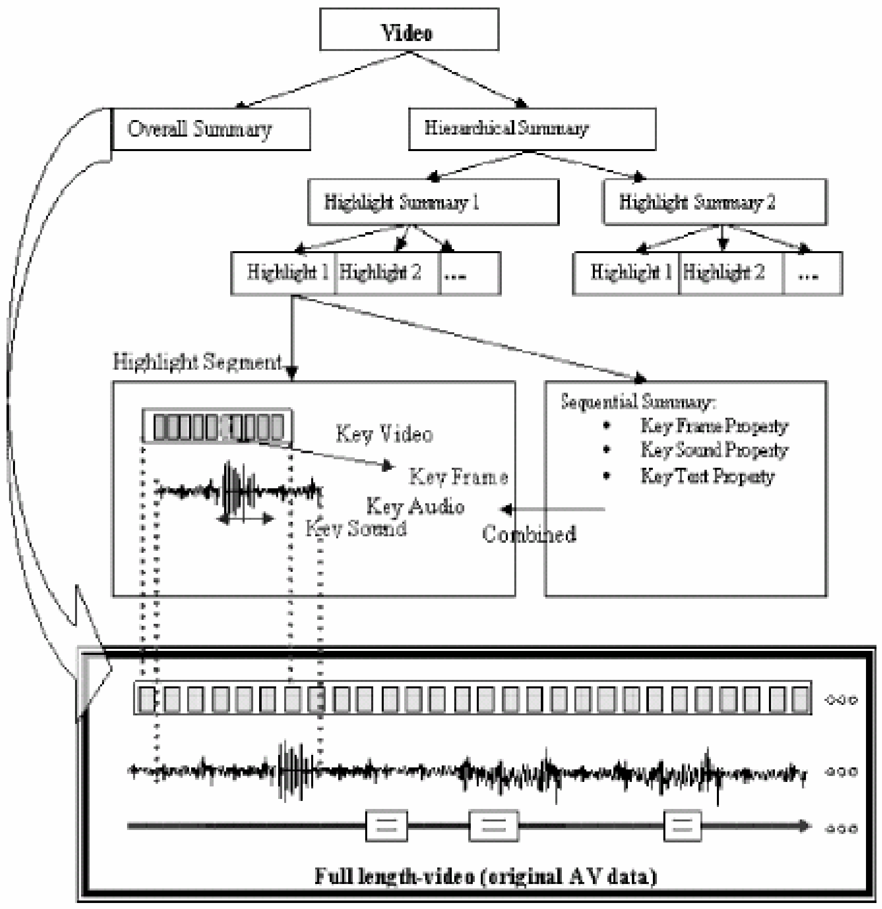


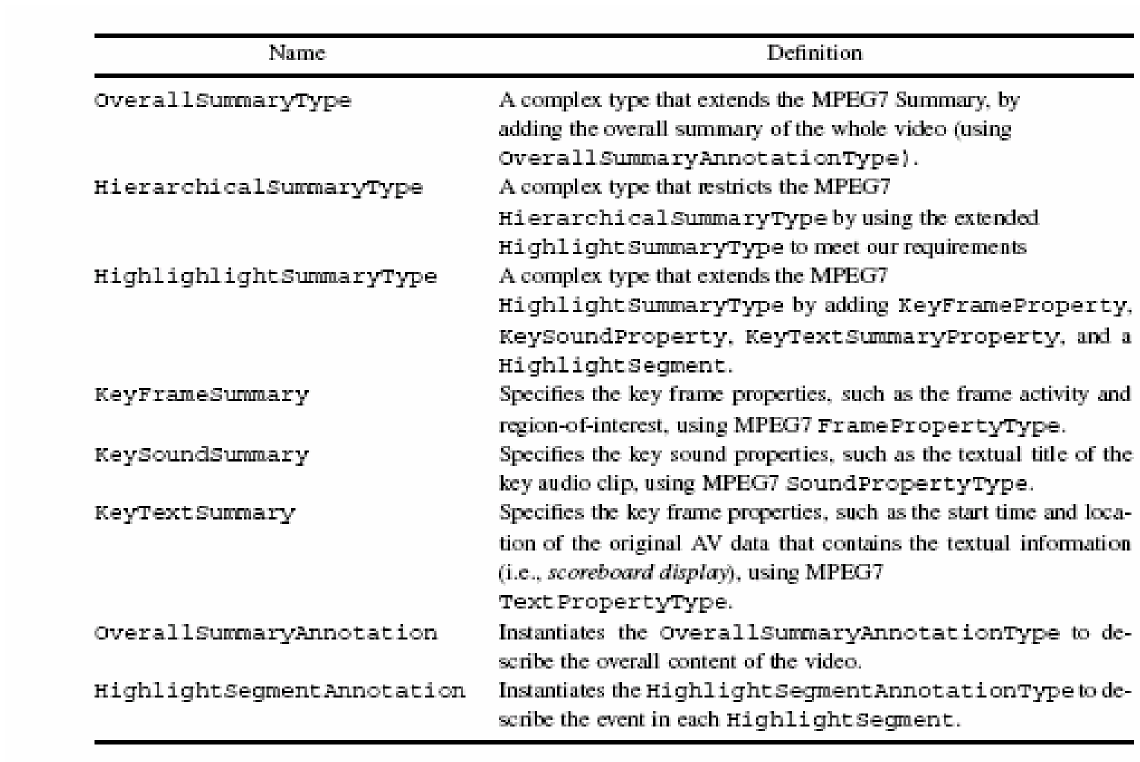
Figure 2. Proposed video summary components.

Overall Summary and Hierarchical Summary type

```
<complexType name="OverallSummaryType">
     <complexContent>
          <extension base="mpeg7:SummaryType">
               <element name="OverallSummaryAnnotation"
                    type="vid:OverallSummaryAnnotationType"/>
               </extension>
     </complexContent>
</complexType>

<complextType name="HierarchicalSummaryType">
     <complexContent>
          <restriction base="mpeg7:HierarchicalSummaryType">
               <element name="HighlightSummary"
                    type="vid:HighlightSummaryType"/>
               </restriction>
          </complexContent>
</complextType>

<complexType name="HighlightSummaryType">
     <complexContent>
          <extension base="mpeg7:HighlightSummaryType">
               <element name="KeyFrameProperty"
                    type="mpeg7:FramePropertyType"
                    minOccurs ="0" maxOccurs = "unbounded"/>
               <element name="KeySoundProperty"
                    type="mpeg7:SoundPropertyType"
                    minOccurs ="0" maxOccurs = "unbounded"/>
               <element name="KeyTextProperty"
                    type="mpeg7:TextPropertyType"
                    minOccurs ="0" maxOccurs = "unbounded"/>
               <element name="HighlightSegmentAnnotation"
                    type="vid:HighlightSegmentAnnotationType"
                    use = "required">
          </extension>
     </complextContent>
</complexType>
```

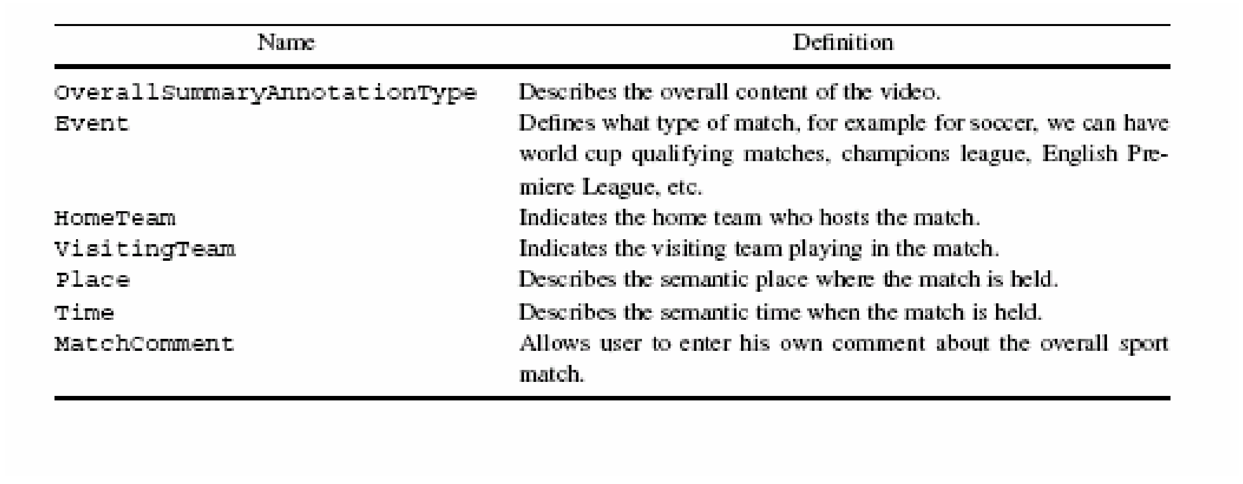


Name	Definition
`OverallSummaryType`	A complex type that extends the MPEG7 Summary, by adding the overall summary of the whole video (using `OverallSummaryAnnotationType`).
`HierarchicalSummaryType`	A complex type that restricts the MPEG7 `HierarchicalSummaryType` by using the extended `HighlightSummaryType` to meet our requirements
`HighlighlightSummaryType`	A complex type that extends the MPEG7 `HighlightSummaryType` by adding `KeyFrameProperty`, `KeySoundProperty`, `KeyTextSummaryProperty`, and a `HighlightSegment`.
`KeyFrameSummary`	Specifies the key frame properties, such as the frame activity and region-of-interest, using MPEG7 `FramePropertyType`.
`KeySoundSummary`	Specifies the key sound properties, such as the textual title of the key audio clip, using MPEG7 `SoundPropertyType`.
`KeyTextSummary`	Specifies the key frame properties, such as the start time and location of the original AV data that contains the textual information (i.e., *scoreboard display*), using MPEG7 `TextPropertyType`.
`OverallSummaryAnnotation`	Instantiates the `OverallSummaryAnnotationType` to describe the overall content of the video.
`HighlightSegmentAnnotation`	Instantiates the `HighlightSegmentAnnotationType` to describe the event in each `HighlightSegment`.

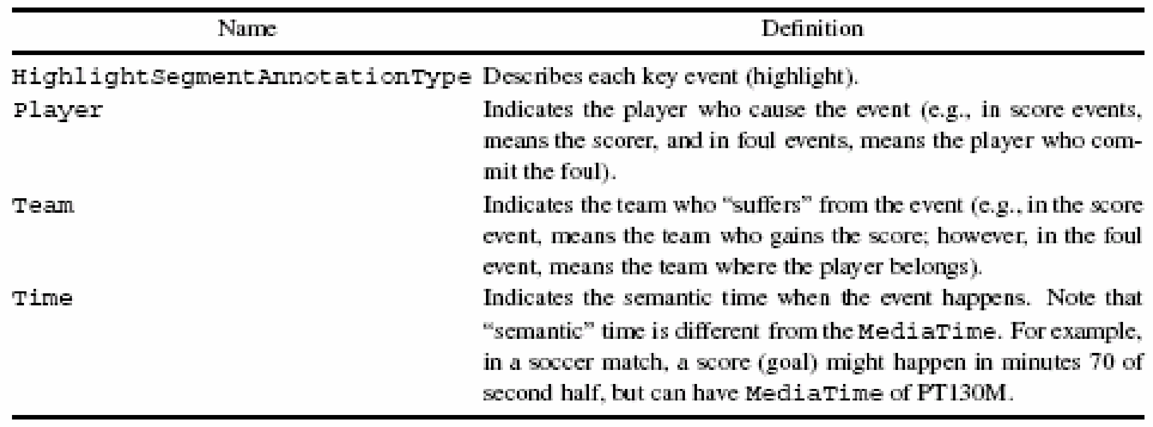
Overall Summary Annotations and Highlight Segment Annotations

```
<complexType name="OverallSummaryAnnotation">
      <sequence>
            <element name="WhatAction" type="mpeg7:TermUseType"/>
            <element name="HomeTeam" type="mpeg7:TextualType"/>
            <element name="VisitingTeam" type="mpeg7:TextualType"/>
            <element name="Where" type="mpeg7:TermUseType"/>
            <element name="When" type="mpeg7:TermUseType"/>
            <element name="Winner" type="mpeg7:TextualType"/>
            <element name="FinalScore" >
                  <simpleType>
                        <restriction base="string">
                              <pattern value="(H)\d - (V)\d"/>
                        </restriction>
                  </simpleType>
            </element>
            <element name="MatchComment" type="mpeg7:TextualType"
                  use="optional"/>
      </sequence>
</complexType>

<complexType name="HighlightSegmentAnnotation">
      <sequence>
            <element name="Who" type="mpeg7:TermUseType"/>
                  <!-- scorer -->
            <element name="Team" type="mpeg7:TextualType"/>
                  <!-- team -->
            <element name="Time" type="mpeg7:TextualType"/>
                  <!-- when scored -->
      <element name="Why" type=" mpeg7:TermUseType use="optional"/>
      <element name="How" type="mpeg7:TermUseType" use="optional"/>
            <element name="HighlightComment" type="mpeg7:TextualType"
                  use="optional"/>
      </sequence>
</complexType>
```



Name	Definition
`OverallSummaryAnnotationType`	Describes the overall content of the video.
`Event`	Defines what type of match, for example for soccer, we can have world cup qualifying matches, champions league, English Premiere League, etc.
`HomeTeam`	Indicates the home team who hosts the match.
`VisitingTeam`	Indicates the visiting team playing in the match.
`Place`	Describes the semantic place where the match is held.
`Time`	Describes the semantic time when the match is held.
`MatchComment`	Allows user to enter his own comment about the overall sport match.



Name	Definition
`HighlightSegmentAnnotationType`	Describes each key event (highlight).
`Player`	Indicates the player who cause the event (e.g., in score events, means the scorer, and in foul events, means the player who commit the foul).
`Team`	Indicates the team who "suffers" from the event (e.g., in the score event, means the team who gains the score; however, in the foul event, means the team where the player belongs).
`Time`	Indicates the semantic time when the event happens. Note that "semantic" time is different from the `MediaTime`. For example, in a soccer match, a score (goal) might happen in minutes 70 of second half, but can have `MediaTime` of PT130M.

This is an example of how this schema can be instantiated for the actual descriptions.

```
<SportVidLib>
     <Video genre="sport" type="soccer">
          <MatchSummary>
               <Name>soccer: ManchesterUnited vs Deportivo2001</Name>
               <SourceLocator>
                    <MediaUri>http://vidlib.org/soccer1.mpg</MediaUri>
                    <<MediaTime>
                         <MediaRelTimePoint>PTOS</MediaRelTimePoint>
                         </MediaDuration>PT105M</MediaDuration>
                    </MediaTime>
               </SourceLocator>
               <MatchAnnotation>
                    <WhatAction>Soccer: Champions League Qualifying
                         <WhatAction>
                    <HomeTeam>Manchester United soccer team</HomeTeam>
                    <VisitingTeam>Deportivo La Coruna soccer team
                         </VisitingTeam>
                    <Where>Old Trafford, England</Where>
                    <When><Y>2001</Y><M>8</M><D>8</D></When>
                    <Winner>Deportivo La Coruna soccer team</Winner>
                    <FinalScore>H2-V3</FinalScore>
                    <MathchComment>A very thrilling lots of goals and
                         exciting plays!</MatchComent>
               </MatchAnnotation>
          <MatchSummary>
<HierarchicalSummary>
     <SummaryThemeList>
```

```
        <SummaryTheme xml:lang="en" id="EO">soccer
        </SummaryTheme>
        <SummaryTheme xml:lang="en" id="E01" parentId="EO">goals
        </SummaryTheme>
   </SummaryThemeList>
   <HighlightSummary id="MU\_goals" themelds="E01">
        <HighlightSegment id="MU\_Goal\_videoclip\_l" level="0"
             duration="PT56S">
             <KeyAVclip>
                  <MediaTime>
                      <MediaRelTimePoint>PT8M31S</MediaRelTimePoint>
                           <MediaDuration>PT56S</MediaDuration>
                  <MediaTime>
             <KeyAVclip>
             <HighlightSegmentAnnotation>
                  <Who>Van Nilestrooy</Who>
                  <Team>Manchester United soccer team</Team>
                  <Time>7M</Time>
                       <HighlightComment>Ryan Giggs displays a great
                          control, then assists Van Nilestrooy. It’s
                          Van Nilestrooy first goal in Champions
                          League since he joined Manchester United
                          in 2001
                  </HighlightComment>
             </HighlightSegmentAnnotation>
   </HighlightSegment>}
```

5. Implementing XQuery to retrieve an MPEG-7 based video summary

Since MPEG-7 uses XML, we need to search for a standard query language to retrieve our proposed video content descriptions. Since there is no standard language yet, we have compared the three most recent XML query languages proposed toW3C: XML-QL, XQL, and XQuery (see Table 1 for the comparison). Our study has found that XQuery is superior compared to the older ones, because it supports our requirements, as well as proposing some important extra features that we can use for our complex queries.
We found some benefits of XQuery in terms of meeting our requirements for a video retrieval:
• XQuery supports our video summary schema:

– Able to combine information from different parts of a document or multiple documents.
– Supports existing database management operations, such as conditions, quantifiers, aggregation, sorting and null values.
– Supports all operations on all data types represented by the data model (e.g., simple and complex types, references, and collections).
– Supports hierarchies and sequence of document structures and must be able to preserve it to the output document.

• XQuery produces an XML data model that supports MPEG-7.
• XQuery provides some powerful extra features that we can benefit for building complex queries:
- For-where-let-return or FLWR expressions (usually used for Iteration).
– Support for multiple XML documents (e.g., for $v in (“video.xml”)//video, $b in (“bib.xml”)//bib.
– Constructor (construct new element and attribute during query).
– Arithmetic expressions (additive, multiplicative, unary).

– Conditional expressions (if, then, else).
– Function definitions.
– Datatype expressions.
– Query prolog.

In addition to these benefits, Table 1 shows the comparison between XQL, XQuery, and XML-QL.

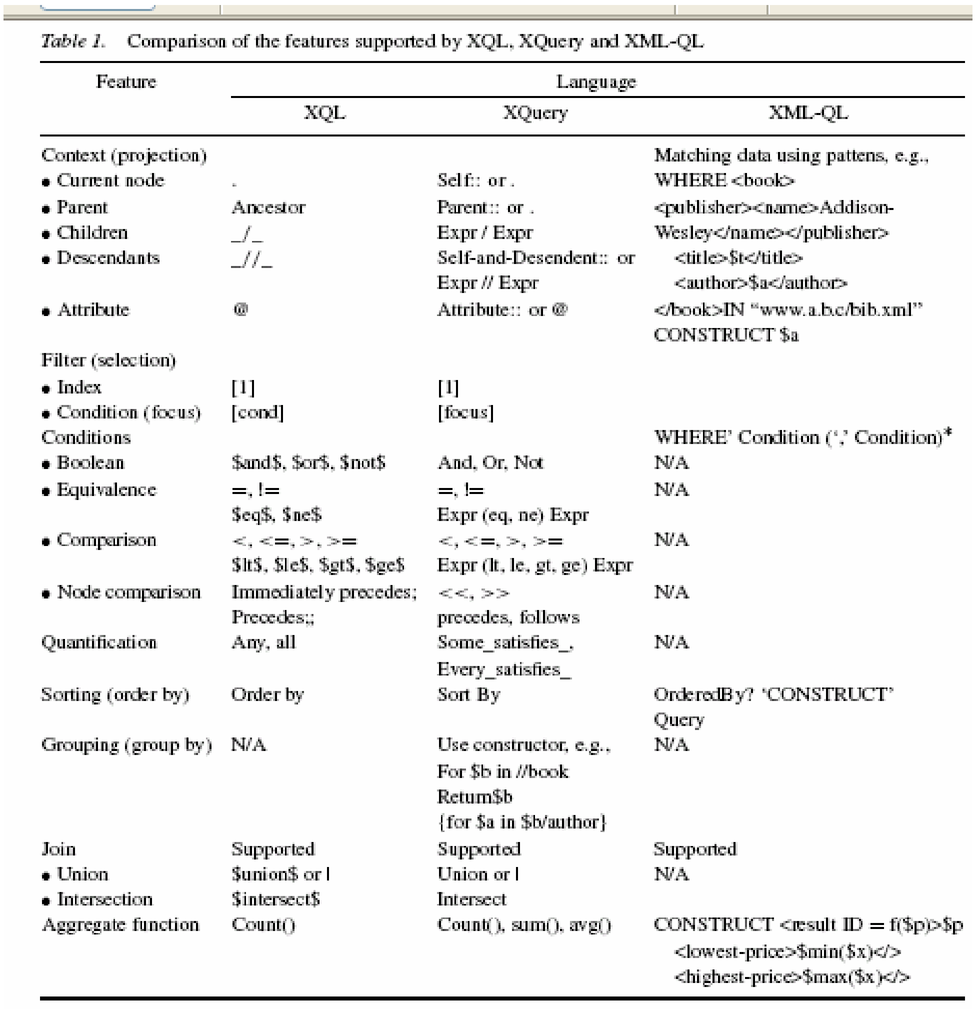


Table 1. Comparison of the features supported by XQL, XQuery and XML-QL

Feature	Language		
	XQL	XQuery	XML-QL
Context (projection)			Matching data using pattens, e.g.,
• Current node	.	Self:: or .	WHERE <book>
• Parent	Ancestor	Parent:: or .	<publisher><name>Addison-
• Children	_/_	Expr / Expr	Wesley</name></publisher>
• Descendants	_//_	Self-and-Desendent:: or Expr // Expr	<title>$t</title> <author>$a</author>
• Attribute	@	Attribute:: or @	</book>IN "www.a.b.c/bib.xml" CONSTRUCT $a
Filter (selection)			
• Index	[1]	[1]	
• Condition (focus)	[cond]	[focus]	
Conditions			WHERE' Condition (',' Condition)*
• Boolean	and, or, not	And, Or, Not	N/A
• Equivalence	=, != eq, ne	=, != Expr (eq, ne) Expr	N/A
• Comparison	<, <=, >, >= lt, le, gt, ge	<, <=, >, >= Expr (lt, le, gt, ge) Expr	N/A
• Node comparison	Immediately precedes; Precedes;;	<<, >> precedes, follows	N/A
Quantification	Any, all	Some_satisfies_, Every_satisfies_	N/A
Sorting (order by)	Order by	Sort By	OrderedBy? 'CONSTRUCT' Query
Grouping (group by)	N/A	Use constructor, e.g., For $b in //book Return$b {for $a in $b/author}	N/A
Join	Supported	Supported	Supported
• Union	$union$ or |	Union or |	N/A
• Intersection	$intersect$	Intersect	
Aggregate function	Count()	Count(), sum(), avg()	CONSTRUCT <result ID = f($p)>$p <lowest-price>$min($x)</> <highest-price>$max($x)</>

The following are some of our implemented (XQuery) queries, to show how we can retrieve our video summaries, using XQuery features.

Query1. Retrieve all highlight segments

The simplest query example is to access a particular element. For example, users want to display the highlight segments from all videos. We can do the query by using relative-path expression to access elements:

```
/SportVidLib/Video/HierarchicalSummary/HighlightSummary/HighlightSegment
```

However, we can alternatively use absolute-path expression. Absolute path reduces the query length; it is less error-prone, and we do not have to know the path to access the element we need. Moreover, an absolute path does not get affected if the actual path to the element needs to be changed:

```
//HighlightSegment
```

Projection can also be done using FLWR expression (which looks more like SQL), however, path expression is easier and more compact compared to FLWR expression:

```
For $h in //HighlightSegment
Return $h
```

Query 2

Using a similar technique (path expression), we can do a more specific query as follows. Display all key-segments of soccer matches, where Manchester United hosts the game and Van Nilestrooy scores:

```
//video[@genre = "sport"] [@type = "soccer"]
[.//homeTeam = "Manchester United soccer team"]
/hierarchicalSummary/highlightSummary/
highlightSegment[.//player = "Van Nilestrooy"]
```

Query 3. A complex example

Please note that the XQuery engine we used has not supported this query yet, however, XQuery has proposed the features, to enable this type of query.
For each player, display all his/her key-segments (where they appear), and we count the number of the key-segments belonging to that player within our video library:

```
FOR $p IN DISTINCT-VALUE(//Who/data( ))
LET $c := COUNT (//Video/hierarchicalSummary
        highlightSummary/highlightSegment[.//Who/data( ) =$p] )
RETURN
        <result>
                <player>{$p}</player>
                <number of goals>$c</number of goals>
                { FOR $v in //Video, $p2
                        $v/HierarchicalSummary/HighlightSummary
                /HighlightSegment/HighlightSegmentAnnotation/Who/data( )
                WHERE $p = $p2
                RETURN $v/HierarchicalSummary/
                        HighlightSummary/highlightSegment
                }
                </result>
```

6. The overall system

To summarise our discussions about key-segments extraction, video content description (video summarisation), and its retrieval, we present the overall data-management architecture in this section.

• Video input is digitised and stored in the video buffer.
• Video buffer is filtered according to user preference during the classification process. For example, only sport and news video are stored in the video storage.
• Full-length video, which are stored in video storage are analysed by the customisable video segmentation module, which produces the key-segments that link to the actual data of each video. In our diagram, we simplify our concept by assuming key-segment 1 links to video 1 and key-segment 2 links to video 2.
• Each of (full-length) video and each of its key-segment is described using MPEG-7 DS (by the video summarisation module) to form video summaries. In our diagram, we show that video summary 1 contains the descriptions of key-segment 1 and video 1 and the same with video summary 2.
• Retrieval module queries the video summaries collection (e.g., sport video library, news video library) to obtain the query result according to user preference or user requests. The result of querying video summaries is called query result, which contains the link to video summary(s) or specifically links to full-length video(s) and/or key-segment(s). To clarify our retrieval module concept, let us have a look at an example of when a user is querying the sport video library.

• When user query asks to display all the soccer matches, in which Manchester United team hosts the game, the query result would return to users the video summaries of the soccer matches. Each video summary contains the links to the full-length video (in the Overall Summary), as well as the links to each key-segment (in the Hierarchical Summary).
• When user query asks to display only the overall summary of soccer videos in the video storage, the query result would return only the Overall Summary of each video, and thus, contains the links to each full-length video.
• However, when user query asks to display only the goal segments of the soccer videos in the video storage, the query result would return only the Hierarchical Summary of each video, and thus, contains the links to each key-segments.

When the video storage is about to run out of space, user can have the option to keep only the key-segments of the video, and may even get compressed to so save more space. Thus, instead of being a link to the full-length video, key-segments “download” the actual data, before the full-length video is deleted. After such a process is done, video summary needs to be updated: the overall summary marks that the link to the full length video has been deleted (e.g., by simply deleting the source ID, locator and information descriptions). As a result, when user tries to query the deleted full-length video (e.g., by querying the source locator of the overall summary), the query result will not find anything or an error message can be displayed to say that it is not available (depending on the implementation of the query engine). The diagram in Figure 3 depicts our proposed architecture for a content-based video data-management system, as well as tries to give readers an overview of how the components of the architecture works together as a system.

7. Summary and future works

This study alone has not presented a complete solution for the development of contentbased VDMS. The major obstacle is on the effort of making the whole progress automatic, while still trying to achieve 100% accuracy. Computer still cannot work as a human, who is able to fully capture the semantic content of the video, locate the key-segments, and finally describe the content of the key-segments for effective video retrieval. However, we believe that we have shown that such a system can be achieved progressively from a smaller variety of video types to eventually employ any types of video. Our study has shown that by concentrating on a particular type of video, we can eliminate the complexities of trying to directly find the solution for all type of videos. The main obvious reason lays in the different characteristics of video that we can benefit. For example, during the key-segment extraction of sport videos, we can benefit the limited camera operations and its repeated events, which contain typical sound, pictures and texts characteristics. During video content descriptions, we found that by using domain specific annotations, we can be more precise in describing the content of the video and its key-segments. For retrieval, users can benefit the prior knowledge of the specific type of videos to retrieve the video more precisely (e.g., instead of retrieving the key-segments in general, users can specifically ask the key-segments, in which a certain player or team appears). Furthermore, it is important

Figure 3. The overall architecture of our content-based VDMS.

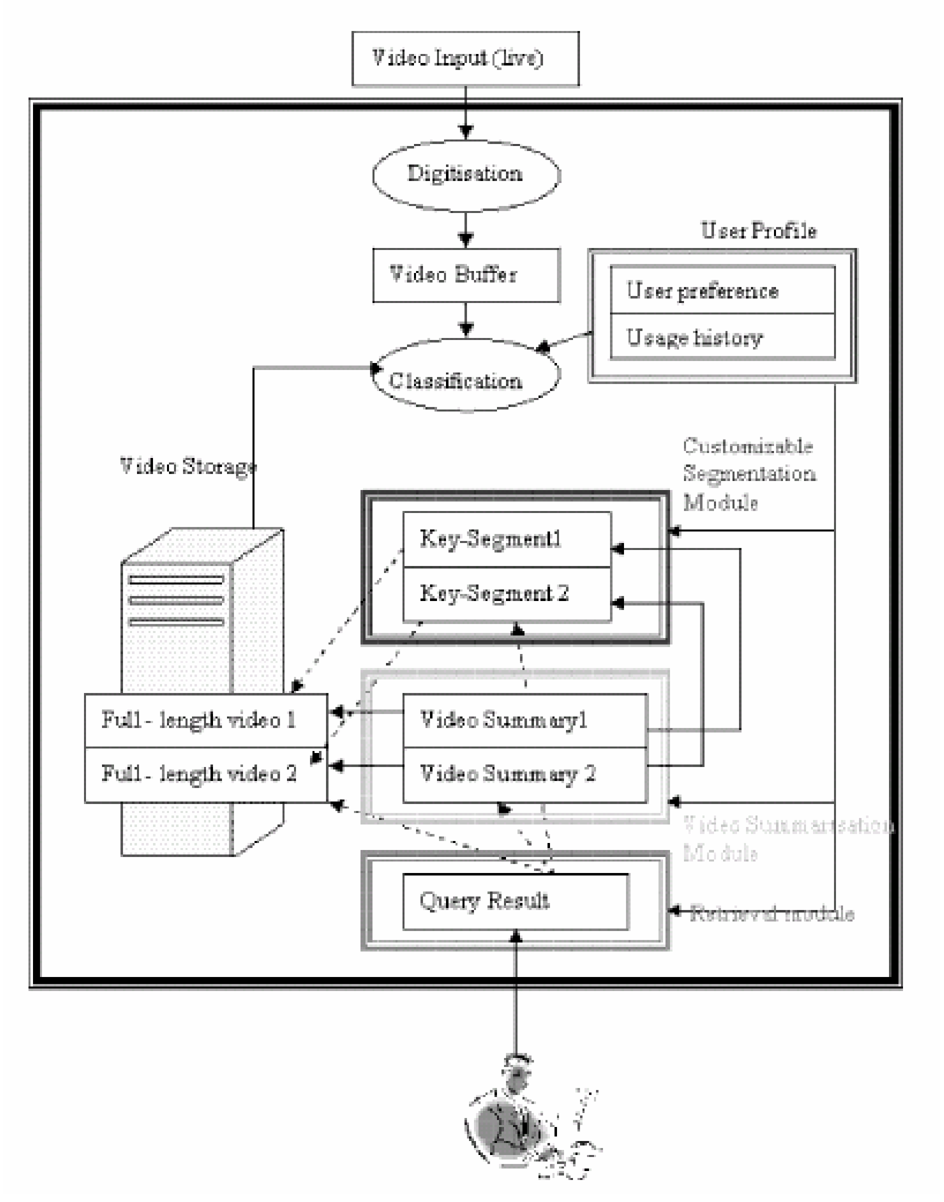


Figure 3. The overall architecture of our content-based VDMS.

to note thatMPEG-7 has provided a standard, powerful and extensible description schemes
that we can use for any type of video and to meet our specific requirements. For retrieval,
we can be confident that we can query the video according to our requirements, by using the
features provided by the newest XML query language. However, we still can expect more
to come from those two solutions since MPEG-7 and XQuery are still moving objects.
In the relation to prior studies, we have designed the concept of integrating various techniques
to automate key-segment extraction by introducing the concept of a "customisable
video segmentation module," which utilises available segmentation techniques, while allowing
addition of new techniques or modification of existing techniques. We have also
extended previous works of utilising MPEG-7 for video summarisation, by introducing domain
specific annotations that allow more specific queries, and we have proposed the
possibility to integrate some sequential-summary descriptions into hierarchical-summary
to benefit useful information, such as the region-of-interest of a key-frame. Lastly, we
have analysed the (currently most current) working drafts of XQuery, in order to experiment
with the language to retrieve our MPEG-7 video summary, using a set of simple and
complex queries that reveal user requirements in practical situations.
We have planned the future works for this research; firstly we need to implement our
"customisable segmentation module" to test its effectiveness in practical situations, including
the implementation of the smart and complex "segmentation agent." Secondly, we
will examine more complex queries to discover how the XQuery engine can be improved.
Thirdly, we also need to develop the document parser for our video summaries. Furthermore,
we aim to apply our solutions to other video types, such as news and movies (in
terms of the segmentation, labelling, and retrieval).

References

[1] O. Avaro and P. Salembier, "MPEG-7 systems: Overview," IEEE Transactions on Circuits and Systems for
Video Technology 11(6), June 2001, 760–764.
[2] N. Babaguchi, N. Y. Kawai, Y. Yasugi, and T. Kitahashi, "Linking live and replay scenes in broadcasted
sports video," in Proceedings of ACM Multimedia 2000 Workshops, November 2000.
[3] S. Boag, D. Chamberlin, M. Fernandez, D. Florescu, J. Robie, J. Siméon, and M. Stefanescu, "XQuery 1.0:
An XML Query Language," W3C Working Draft, 20 December 2001.
[4] N. Fatemi and O. A. Khaled, "Indexing and retrieval of TV news programs based on MPEG-7," in ICCE,
International Conference on Consumer Electronics, 2001, pp. 360–361.
[5] Y. Gong, L. T. Sin, C. H. Chuan, H. Zhang, and M. Sakauchi, "Automatic parsing of TV soccer programs,"
in Proceedings of the International Conference on Multimedia Computing and Systems, 1995, pp. 167–174.
[6] J. Hunter and R. Iannella, "The application of metadata standards to video indexing," in Second European
Conference on Research and Advanced Technology for Digital Libraries ECDL'98, Crete, Greece, 19–23

September 1998.
[7] V. Kobla, D. DeMenthon, and D. Doermann, “Detection of slow-motion replay sequences for identifying
sports videos,” in IEEE 3rd Workshop on Multimedia Signal Processing, 1999, pp. 135–140.
[8] R. Lienhart, “Automatic text recognition for video indexing,” in Proceedings of the Fourth ACM International
Conference on Multimedia, February 1997.
[9] S. Nepal, U. Srinivasan, and G. Reynolds, “Automatic detection of goal segments in basketball videos,” in
Electronic Proceedings of ACM Multimedia 2001 Conference on Authoring Support, October 2001.
[10] S. Pfeiffer and U. Srinivasan, “TV anytime as an application scenario for MPEG-7,” in Proceedings of ACM
Multimedia Workshop, Los Angeles, CA, 2000.
[11] H. Pan, P. Van Beek, and M. I. Sezan, “Detection of slow-motion replay segments in sports video for
highlights generation,” in IEEE International Conference on Acoustics, Speech, and Signal Processing,
Proceedings, Vol. 3, 2001, pp. 1649–1652.
[12] Y. Rui, A. Gupta, and A. Acero, “Automatically extracting highlights for TV baseball programs,” in Proceedings
of the 8th ACM International Conference, October 2000.
[13] P. Van Beek, A. B. Benitez, J. Heuer, J. Martinez, P. Salembier, Y. Shibata, J. R. Smith, and T. Walker,
“Text of 15938-5 FCD information technology – Multimedia content description interface – Part 5, Multimedia
description schemes,” International Organisation for Standardisation, Coding of Moving Pictures and Audio, ISO/IEC JTC 1/SC 29/WG 11/N3966, Singapore, March 2001.
[14] W. Zhou, A. Vellaikal, and J. Kuo, “Rule-based video classification system for basketball video indexing,”
in Proceedings of ACM Multimedia 2000 Workshop, November 2000.