

Delivery of near real-time soccer match highlights to wireless PDA devices

Vladimir Nedovic, Chris Nelson, Scott Bowser, and Oge Marques*

Department of Computer Science and Engineering
Florida Atlantic University
777 Glades Road, Boca Raton, FL 33431-0991
omarques@fau.edu

Abstract — Intelligent abstraction and summarization of video programs containing sports events is a very active field of research. This paper describes the development of a system that enables the users of wireless handheld PDA devices to receive video clips of the goals scored in a soccer match just minutes after they actually happened. The service provider performs the extraction and processing of the video clips and, using wireless and Internet protocols, delivers the data to the user.

Index Terms — Video Processing, Video Summarization, OCR, Wireless Video Communications, Video Streaming, Digital Image Processing.

1. Introduction

Several technological developments in the last decade enabled the creation, compression, and storage of large amounts of digital information, such as images, audio, and video. With increasing usage of wireless communications, consumption of such information poses a problem because of insufficient bandwidth. Although it is expected that new wireless standards (especially 3G) will offer wireless users higher transmission rates, live streaming of complete programs or events may still be unaffordable.

This work focuses on video programs containing soccer matches, a topic that appeals to large audiences. In the case of soccer and other sports events, an elegant alternative to the bandwidth-intensive transmission of an entire match would be a system in which the user could receive summaries of the most relevant events. Those summaries (e.g., goals in a soccer match) could be sent after the game or, as in the work described in this paper, streamed in near real-time. The value of sports video drops significantly after a relatively short period of time and the delivery of the information in near real-time is very important [1].

The purpose of this project is to develop a system that would enable the users of wireless handheld PDA devices to receive video clips of the goals scored in a soccer match just minutes after they have actually

occurred. The service provider, i.e. the main server, performs the extraction and processing of the video clips and, using wireless and Internet protocols, delivers the data to the user.

In the proposed system, a real-time video input of the soccer game is digitized and stored on a server. The server implements intelligent algorithms capable of detecting the occurrence of a goal in a soccer match. Once a new goal has been detected, the portion of the video that contains the action in which it was scored is cut out, processed for delivery to the wireless handheld devices, and sent to the Internet, ultimately to reach the subscribers.

This paper is organized as follows: Section 2 presents an overview of related work in the field of highlight extraction and summarization of video programs containing sports events as well as pattern recognition techniques related to this project. Section 3 describes our approach in more details, both from design as well as implementation points of view. Finally, Section 4 presents our conclusions and suggests directions for future work.

2. Background and related work

In this section we discuss some background issues and related work in the field of semantic-driven video segmentation, summarization, and highlight extraction, particularly for sports events.

2.1 Event Detection in Sports Videos

An intuitive area of research in sports video segmentation is based around the detection of slow motion replays. It is a justifiable assumption that in most sporting telecasts a notable event is followed by a slow motion instant replay. In [2] Pan et al. describe a low-level detection method based on the Hidden Markov Model (HMM) using a Viterbi algorithm. The system assumes that a slow motion replay (*slo-mo*) is preceded and followed by an editing effect (i.e. fade, cut, dissolve). The Viterbi algorithm uses normalization and thresholding methods to find a single field with the highest probability of a *slo-mo* and uses this as the basis

* Contact author

for forward and backward sweeps until the editing effects are discovered. This work addresses the important concept of using context clues within the video stream but leaves room for other non-video clues. The general application for this technology lies in post processing. Real-time bi-directional sweeps through the video stream are feasible.

A much higher-level algorithm is proposed in [5], in which Ahmed and Karmouch propose the three-step process of parsing, analysis and selection. The proposal alludes to similar concepts as in [2], however additional clues are used for the parsing stage. High-energy audio changes, differences in on-screen text overlays, and histogram based shot detection are all triggering events. Each possible event is then analyzed based on a particular subscriber’s selection criteria. This approach is more adaptive to pseudo real-time environments and most importantly addresses additional context clues beyond the actual video stream.

2.2 Pattern Recognition Techniques

A secondary area of interest in this research is the use of specific pattern recognition techniques to identify events. One such technique, video OCR is presented in [4]. Text boxes are extracted through filter techniques that identify relatively static regions of the image. This box is then extracted and a character mask swept over the text box to identify potential characters using clustering methods. This approach proves efficient after the initialization phase is completed. The initial filter requires several frames separated temporally to properly determine the region of interest.

3. Our system

3.1 Objectives

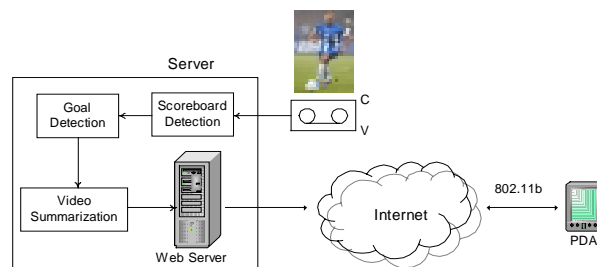
The main goal of our proposed system is to provide a proof-of-concept for a wireless service that allows its subscribers to receive alerts (and subsequent video clips) related to relevant events in a soccer match. A collection of intelligent algorithms, running on the main server, behave as if they were watching the game on the user’s behalf and selecting the most relevant events (e.g., goals) for eventual compilation and delivery.

3.2 Basic definitions, assumptions, and design criteria

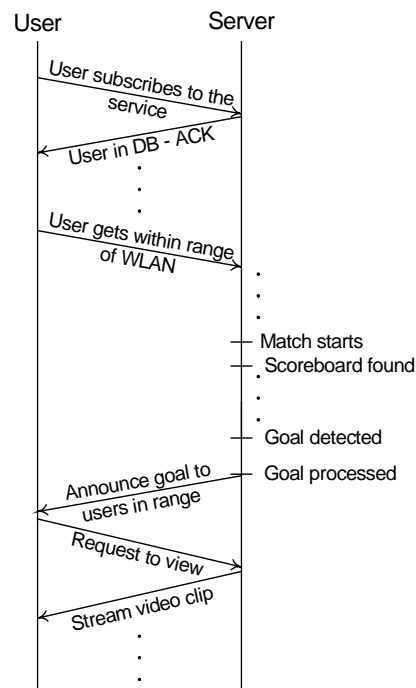
“A goal is scored when the whole of the ball passes over the goal line, between the goal posts and under the crossbar. Unfortunately, it is difficult to verify these conditions automatically and reliably by video processing algorithms [1]”. The occurrence of a goal is

generally followed by a special sequence of frames that could be used in future versions of our project for error-checking, however we make a simplifying assumption about the constant presence of a superimposed scoreboard for the most part of the game, one that will reflect the change in the score shortly after the goal has been scored. We also assume that the video input we have is a game broadcast on ESPN channel that always has its *BottomLine* section present, and we disregard that portion of the frame altogether.

Another assumption we make is that the goal does not occur in the first minute of the match, which would enable us to accumulate enough frames needed for the algorithm to locate the scoreboard on the screen.



(a)



(b)

Figure 1. (a) The overall architecture of the system (b) The sequence of events in the overall process

3.3 Architecture

The architecture of the system (Figure 1a) shows its main components: a video input, a processing module, a web server, and a user's wireless device. Within the server, the main processing tasks are: scoreboard detection, goal detection, and video summarization. Each of these tasks will be explained in more detail later in this paper.

The system was conceived under the assumption that the wireless user would initially subscribe to a service that would send alert messages reporting relevant events (e.g., "a goal has occurred in a match to which you subscribed and a video clip summary is available") and providing options on which actions to take (e.g., watch it now, watch it later, or dismiss it). The sequence of events that may take place in the overall process, from the moment the user subscribes to the service to the streaming of a video clip containing a goal, is depicted in Figure 1b.

3.4 Goal detection

The video input comes from a live game through a commercial video capture card into the server. The processing of video and its streaming to wireless devices is briefly described in the flowchart shown in Figure 2.

3.4.1 Scoreboard and digit detection

The processing starts by capturing 30 still frames every 2 seconds at the beginning of the game. After all 30 have been collected, the first algorithm, called **findDigits**, is invoked. Based on the assumption that every other pixel on the screen changes a lot compared to the portion where the scoreboard is located, the algorithm accumulates the intensity differences of successive frames. Figure 3 shows the surface representation of the matrix that contains the values of those differences. As it can be seen in the lower right corner of Figure 3, the scoreboard portion consists of small values compared to the rest of the image; the stripe on the left represents ESPN's *BottomLine*.

The primary assumption in the algorithm was that the pixels within the scoreboard do not change too much compared to the others, but it turns out that the peak of the above surface (i.e. the small number of pixels that change the most) is located inside the scoreboard portion of the frame. That is because the scoreboard contains the timer, which is updated every second. So the task of the algorithm is to find the region of pixels that don't change too much around the smaller region of pixels that change the most. For that purpose, adapted Breadth First Search algorithm was used, called

SeekFunc. Based on the thresholds in the analysis of the game feeds, the function will extract only the rectangular portion of the frame containing the scoreboard. Figure 4 shows the result of this part of the algorithm.

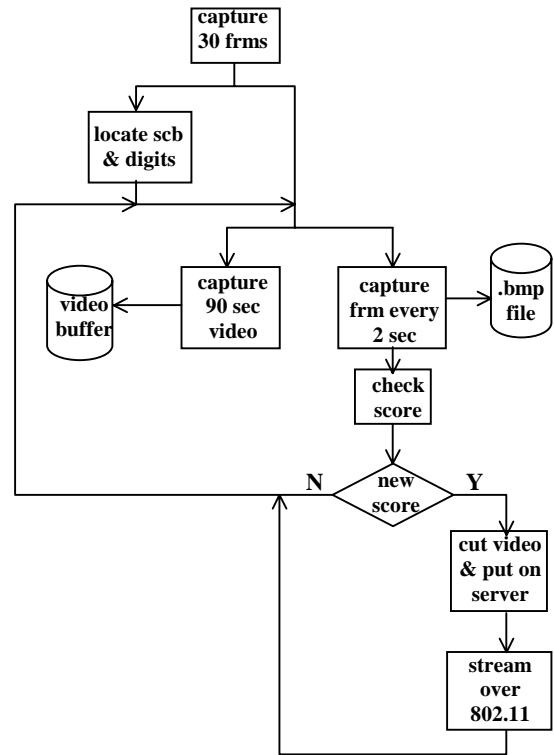


Figure 2. Process flowchart

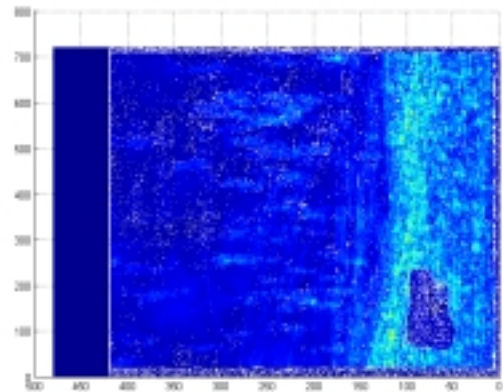


Figure 3. Top view of the cumulative frame differences after the first 30 seconds

The algorithm now focuses on the isolated scoreboard and tries to find the digits in one of the frames that have been captured. A technique called Normalized Cross-correlation was used in this part of the process to locate the parts of the image that best

match the image of a zero that is input. Since at the outset of the match there are two zeros in the scoreboard, we expect to have two peaks in the surface representation of the correlation matrix. Figure 5 shows the results.



Figure 4. Rectangular portion of the frame containing the scoreboard cut by the algorithm

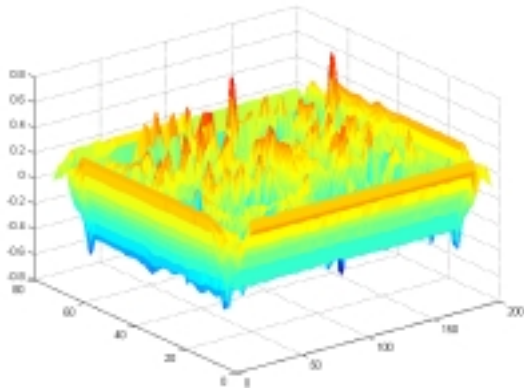


Figure 5. The result of the search for two zeros in the frame clearly showing two peaks

After the digits have been found, the algorithm remembers their positions and, since we assume that the scoreboard is always positioned in the same place (even if removed temporarily), it will only look for the score digits at those positions from this time on.

3.4.2 OCR – Training phase

When the digits have been found within the frame, another task is to recognize their values, which requires some character recognition techniques. For that purpose, we adapted the Eigenfaces algorithm proposed in [3]. It can be split up into two logically distinct parts: one will extract and save the features of the digits in our sample database, while the other will find the best match between the input image and the one in our database. We want to find the components of the distribution of digits, which in mathematical terms translates into finding the eigenvectors of the covariance matrix of the vectors representing those digits (the eigenvectors are a set of features that together characterize the variation between different digits). So the training algorithm, called

TrainDigits, reads in the training set of digit images and calculates the eigenvectors that actually define the digit subspace.

3.4.3 Detecting score changes

The process of recognizing the digits in the scoreboard begins with the matching part of the Eigenfaces algorithm. When a new frame has been read and digit images extracted from the scoreboard, they are projected into the digit subspace and then the best match is found between them and the digits in our training set.



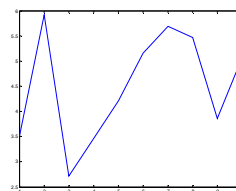
(a)



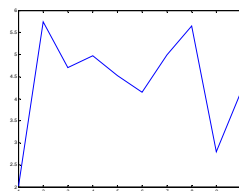
(b)



(c)



(d)



(e)

Figure 6. (a) Sample frame. (b) Digit 1 as extracted from the scoreboard. (c) Digit 2 as extracted from the scoreboard. (d) Differences relative to the digit ‘2’ found in the frame; the graph has a minimum at 3. (e) Differences relative to the digit ‘0’ found in the frame; the graph has a minimum at 1

In the case of a soccer match, the matching is much simplified because we know that the score can go up only by one, so we have to compare the input digit only with two other ones: the one representing the current score and the other one representing the updated score

(current score + 1). The code can be found in the file called `ChkDigFunc`, which takes three input parameters: current score for team one, current score for team two, and the time elapsed from the beginning of the game. The last parameter is relevant because we want to know when the half-time period started, so we can suspend our program. It is also useful for the case when there is no close match between the part of the frame where the digits should be and the digits in the training set. In that case, if the timer shows that we should be in one of the halves, we know that the scoreboard has been temporarily removed. Figure 6 shows a sample frame, the extracted digits, and the two graphs representing the difference values returned by the algorithm; the minimum value shows the best match (but since the differences are stored in an array, the indices are always one more than the value of a digit, i.e. if a good match has been found with digit 1, the graph will have a minimum at 2):

3.4.4 Responding to score changes

After the score has been checked and compared to the previous one, three different cases can occur. If there is no change in score, nothing happens; if a new score has been detected, the algorithm calls another function that will capture a video following the goal and send it over the network; and if a big enough difference has been detected between the digits, then the scoreboard is not present on the screen (it was temporarily removed, the half-time is in progress, or the match is over).

3.5 Communication

3.5.1 Detection of a WLAN by the PDA

When the video clip is ready for delivery, it is sent out to all the subscribers that are currently within range. Every handheld device automatically broadcasts a packet from time to time asking about the active servers in its area (i.e. hotspots). When some server responds, it assigns a new dynamic IP address to the device (DHCP), and that IP address is used for the communication between them from then on (for the duration of the current session, i.e. until the PDA goes out of range again). Our video server establishes a connection with the wireless devices through the router and knows about all its subscribers that are currently within the range of some hotspot.

3.5.2 Streaming the video over the Internet

When a change in the score has been detected, the user receives a message about a new goal that has been scored on a particular soccer game of interest. A

window pops-up and the user is offered three choices (Figure 7):

- 1) Download the file, in which case the edited video clip is streamed to his PDA's IP address using PocketPC's PocketTV utility that supports streaming of mpeg files
- 2) Decide to download and watch it later, in which case the URL of the file is saved in the client's memory and the user can decide to view the video at any time
- 3) Choose not to watch the video clip, in which case nothing is saved locally.

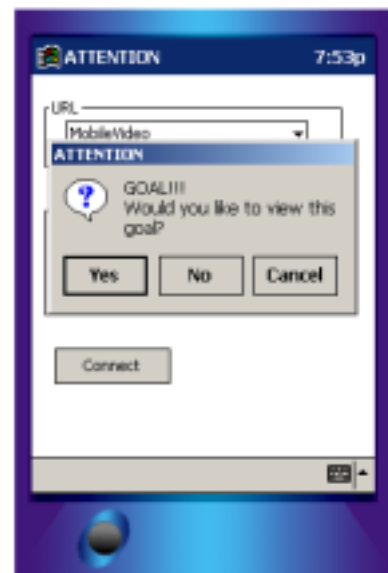


Figure 7. Notification to client that a new goal has been scored

3.6 Implementation aspects

The server software was implemented using Microsoft's Visual Basic (VB) and ActiveX controls. Since the PocketPC had limited memory and the video files generated as a result of processing were quite large (~20 MB), we decided that streaming was the most effective way of transmitting the video to the client. A PocketPC application called PocketTV was chosen to play the resulting MPEG-formatted stream from the Apache server.

4. Conclusion and future work

This paper described the ongoing design and implementation of a system that delivers short video clips of soccer goals to users over a wireless network.

This is a work in progress and many improvements are under way, namely:

- ❑ Implement more reliable and robust image and video analysis algorithms.
- ❑ Support multiple users simultaneously.
- ❑ Provide some aspect of security using username and password pairs.
- ❑ Implement a method to track which clients have successfully received which clips, allowing users to request retransmission of missed or damaged clips.
- ❑ Add notification of which team actually scored a goal. This would allow the user to see goals only for his/her favorite team.
- ❑ Implement error-checking routines after the goal has been detected that would look for the replays, close-up shots of the players, and audience shots that usually follow the goal.
- ❑ Add a virtual scoreboard displayed on the client to keep track of the score in the current game.
- ❑ Develop a web-based utility where users can select games of interest to them.

Acknowledgments

This project was partially supported by an Information Technology grant from the College of Engineering at Florida Atlantic University.

The authors would also like to thank Nitish Barman for his comments and review of an earlier version of the manuscript.

References

- [1] Ahmet Ekin and A. Murat Tekalp, "A generic model and sports video processing for summarization and model-based search," in *Handbook of Video Databases* (ed. Borko Furht and Oge Marques), CRC Press, 2003.
- [2] H.Pan, P. van Beek, and M.I. Sezan, "Detection of slow-motion replay segments in sports video for highlights generation," in *Proc. IEEE Int'l.Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, 2001.
- [3] Matthew A. Turk and Alex P. Pentland, "Face Recognition Using Eigenfaces," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition*, Maui, Hawaii, 1991.
- [4] Dongqing Zhang, Raj Kumar Rajendran, and Shih Fu Chang, "General and domain specific techniques for detecting and recognizing superimposed text in video," *IEEE 2002*

International Conference on Image Processing, Rochester, NY

- [5] Mohamed Ahmed and Ahmed Karmouch, "Uncertainties and event conflict resolution within video analysis service," *Multimedia Modeling 2001*, Amsterdam, Netherlands.