

# Fast Client-Server Video Summarization for Continuous Capture

John Dixon  
Media and Entertainment Technologies  
Laboratory  
Department of Computer Science and  
Engineering  
Michigan State University  
3115 Engineering Building  
East Lansing, MI 48824  
dixonjoh@cse.msu.edu

Charles B. Owen  
Media and Entertainment Technologies  
Laboratory  
Department of Computer Science and  
Engineering  
Michigan State University  
3115 Engineering Building  
East Lansing, MI 48824  
cbowen@cse.msu.edu

## ABSTRACT

This paper describes a keyframe summarization method for client-server applications. This technique is designed for applications where a camera is collecting content on a continuous basis that must be transmitted in a summarized form to a remote database server over wireless network. The system combines three keyframe selection methods including a novel fast motion-based selection method, keyframe pooling and clustering for bandwidth control, and network bandwidth estimation.

## Categories and Subject Descriptors

**H.3.1 [Information Systems]:** Content Analysis and Indexing

## Keywords

Video summarization, keyframe selection

## 1. INTRODUCTION

This paper presents a method for keyframe summarization in client-server environments where video cameras continuously collect content and transmit it to a remote server for later analysis and/or retrieval. Often, a simple summarization of the salient content of the video is sufficient in these applications. As an example, the video camera equipped Unmanned Aerial Vehicles (UAV) commonly utilized for remote reconnaissance purposes by the military utilize wireless communications links that are subject to noise, signal degradation over distance, and jamming. Given shared communications channel usage and retransmission due to packet loss, the bandwidth of the channel can vary substantially.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or to publish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

*MM'01*, Sept. 30-Oct. 5, 2001, Ottawa, Canada.

Copyright 2001 ACM 1-581 13-394-4/01/0009...\$5.00

Transmission of the actual video from a UAV may not be possible given the bandwidth constraints.

The approach described in this paper is designed for raw, unedited video and real-time performance. It utilizes parallel keyframe extraction so as to provide three measures of saliency. A new motion-based keyframe selection method estimates the amount of de-occlusion and selects keyframes appropriately. Also implemented is a simple color/intensity histogram-based keyframe selection method [7]. The purpose of this component is to select new keyframes when the camera is not moving or has moved very little. The third method is a simple temporal keyframe selection that forces a new keyframe at some minimum interval (typically 2-10 seconds) if neither of the other methods trigger. Bandwidth control estimates effective network bandwidth and limits the keyframe generation.

There have been numerous research efforts with respect to keyframe extraction. Zhang, Kankanahalli, and Smoliar [8], and Nagaska and Tanaka [4] are examples of color-based histogram implementations. Wolf proposes a motion based keyframe selection algorithm based on optical flow [6].

## 2. METHOD

Figure 1 is a block diagram of the system, a hybrid of motion-based, color histogram-based, and temporal-based keyframe extraction methods that provide keyframes to a pool. Selection from the keyframe pool is based on clustering for saliency within the constraints of the network bandwidth.

### 2.1 Motion-based Keyframe Selection

The motion-based technique keyframe selection technique used in our system can accurately select keyframes with significant global motion between frames. Imagine a camera moving over a fixed scene. As the camera moves, the new image overlaps the previous images by varying amounts. Given a starting reference frame (keyframe), 1 *–overlap* represents the amount of new content uncovered by the camera motion. The approach presented in this paper is illustrated in Figure 2. All computations are done on adjacent pairs of frames<sup>1</sup>. The previous frame and current frame are compared using a block matching algorithm commonly used to MPEG

<sup>1</sup>In our system we are performing this computation once every five frames to achieve real-time results without ma-

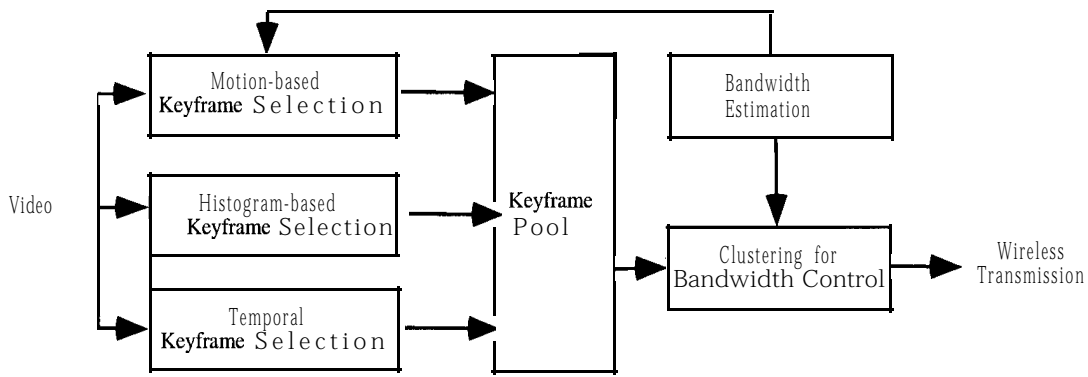


Figure 1: System block diagram

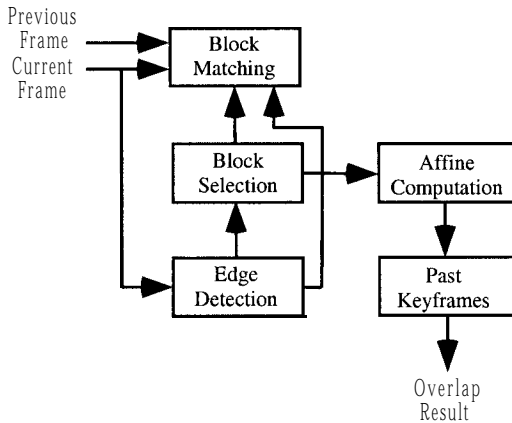


Figure 2: Motion-based keyframe selection



Figure 3: Motion vectors for a frame

motion vector computation. However, motion vectors are not selected arbitrarily. A set of motion vectors is selected based on presence of edge detail as indicated by a statistical edge detection algorithm.

The result of the block matching algorithm is a set of displacements and their associated block centers in the current frame. This computation is based on searching the previous image for matching blocks. These displacements are converted to  $(x, y)$  to  $(u, v)$  correspondences and used to compute a least-square estimate of the affine transformation from the previous image to the current image. This affine transformation is then used to warp all recent **keyframe** locations to correspond with the current motion. The warped keyframes are tested for percentage overlap with the current frame. When the overlap drops below a preset threshold, a new **keyframe** is selected.

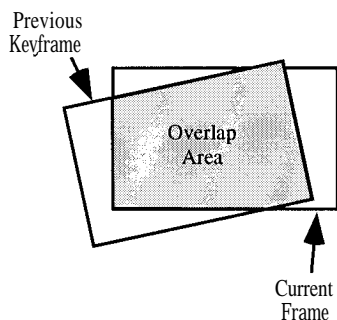
In order to characterize the background motion we implemented a global two-dimensional affine parametric model. The goal of this model is to approximate the motion of the camera platform relative to the scene. The content clearly includes rotation, scaling, and translation, so an affine solution is a minimum **parameterization** for this application. A large distance of the camera from the image content and a long camera focal length significantly limit the value of

a projective model. The affine model is commonly used to characterize motion in video sequences [1, 5]. The affine model was considered because of its resilience to noise and sparse motion vector conditions [1]. It is assumed that the affine model is characterizing the underlying motion flow of the background from one image to the next (or within a sequence of images).

Given two input image frames, a robust statistical based edge detection algorithm adapted from Kundu [2] is used to compute the edge maps of both frames. Motion vectors are computed from the edge maps based on a block correspondence algorithm proposed by Kundu [3]. A block correspondence algorithm was chosen for this application in order to decrease the computation requirements relative to methods such as optical flow and because only sections of the image with significant edge detail will be examined for correspondences. Block correspondence is clearly not a good choice when a significant rotation component is evident. However, for most video the rotation rate is physically constrained, so the method remains effective. In order to increase efficiency and performance, motion vectors are only computed where edge content is evident. Figure 3 is an example set of computed motion vectors and illustrates the grouping of vectors where complex content is located.

Based on the computed motion vectors, the affine parameters are estimated using linear least-square decomposition. The previous keyframes image corner coordinates are con-

major optimization and because its motion in 1/30 second is typically insignificant.



**Figure 4: Example of polygon overlap**

tinuously warped using the estimated affine parameters. It is not necessary to actually warp the previous keyframe or even keep it around. All that is necessary is a warping of the polygon that the keyframe represents. Figure 4 illustrates the overlap of a previous keyframe and the current frame. If the amount overlap is less than a supplied threshold, the current candidate frame is chosen as the next keyframe.

## 2.2 Histogram and Temporal Keyframe Selection

Motion-based keyframe selection is not sufficient for video summarization. While effective in modeling de-occlusion, it is not effective in modeling content changes that do not involve camera motion. For this reason, we utilize color/intensity histogram-based keyframe extraction operating in parallel with the motion-based method. The temporal-based aspect of the system compensates for missed keyframe selection by the previous two techniques. If a keyframe has not been selected in a predetermined time interval, the current candidate frame is selected as a keyframe and placed in the keyframe pool.

## 2.3 Clustering for Bandwidth Control and Additional Bandwidth Control Features

The keyframe pool will typically contain more images than can be transmitted. Bandwidth control is predominantly accomplished by clustering the available frames into groups that represent similar underlying content using a block region color histogram method [7]. The histogram-based keyframe selection does not utilize regions, whereas the clustering does. This is because regions would induce redundant motion-based keyframe selection into the keyframe pool. Once the pool is created, that is not an issue. For reasons of efficiency, the region-based histogram is computed first in the histogram-based keyframe selection and converted to a non-region-based histogram.

## 3. EVALUATION

In evaluating the system, the primary element of concern is the motion-based keyframe selection. Bandwidth estimation and the keyframe clustering are functional components and need no specific evaluation. Histogram-based keyframe selection mechanisms have been extensively studied elsewhere. The key new element in this system is the motion-based keyframe selection.

**Table 1: Sample overlap computation results**

Frame Number	Computed	Actual
445	0.494	0.502
645	0.468	0.65
845	0.483	0.427
1005	0.475	0.407
1545	0.488	0.11
1865	0.493	0.432
2000	0.485	0.140
2130	0.465	0.36
2540	0.488	0.539

Evaluation of the motion-based keyframe selection is based on human analysis of the results. The frames were loaded into an image editor where they were manually manipulated to align the overlapping regions. Some example test results are included in Table 1. The Frame number is relative to the start of the sequence. The content was sampled at 30 frames per second. The actual column is a manual measurement of overlap with the previously selected keyframe.

The method is effective in most instances, but is less effective when the video quality is poor or there is minimal texture or edges for the block vector search to key to. This is the case for frame such as 1545, which consists mostly of trees and is over-exposed. For this set of data, the mean error was 0.09.

## 4. ACKNOWLEDGMENTS

This work was supported by a grant from Mitre Corporation. Considerable assistance was provided by Amlan Kundu.

## 5. REFERENCES

- [1] J. Kim, H. Chang, J. Kim, and H. Kim. Efficient camera motion characterization for mpeg video indexing. In *ICME00*, page TP11, 2000.
- [2] A. Kundu. Robust edge detection. *Computer Vision and Pattern Recognition*, pages 11-18, 1989.
- [3] A. Kundu. Motion estimation by image content matching and application to video processing. In *ICASSP96*, 1996.
- [4] A. Nagasaka and Y. Tanaka. *Visual Database Systems II*, chapter Automatic Video Indexing and Full-Video Search for Object Appearances, pages 113-127. Elsevier Science Publishers B. V., North-Holland, 1992.
- [5] J. Wang and E. Adelson. Representing moving images with layers. *IP*, 3(5):625-638, September 1994.
- [6] W. Wolf. Key frame selection by motion analysis. In *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 2, pages 1228-1231, Atlanta, GA, 1996.
- [7] H. Zhang. *The Handbook of Multimedia Computing*, chapter Content-Based Video Browsing and Retrieval, pages 255-280. CRC Press, Boca Raton, FL, 1999.
- [8] H. J. Zhang, A. Kankanhalli, and S. W. Smoliar. Automatic partitioning of full-motion video. *Multimedia Systems*, 1:10-28, 1993.