

# Video Segmentation Utilized in Visual Data Mining Applications

Mark Smith and Alireza Khotanzad

Department of Electrical Engineering, Southern Methodist University  
Dallas, Texas 75275

## Abstract

*A novel approach utilized in video database object-based queries is proposed. This new method segments an example video sequence into real world objects using a combination of color image segmentation techniques along with MPEG-1/2 motion vectors. First, the initial frame in the sequence undergoes a color/texture segmentation algorithm that divides the frame into homogenous regions of color and texture. Features extracted from the co-occurrence matrix provide the texture measurement utilized by this system. The MPEG-1/2 motion vectors are then exploited in predicting the objects of the next frame based on the initial frame's segmentation. A robust technique for the identification of new objects as they enter the scene is proposed along with a measurement pertaining to the object segmentation accuracy is introduced. This new method has a host of visual data mining applications including image database retrieval and Internet image searches. This new algorithm is tested on various image sequences with segmentation results included.*

## 1. Introduction

The central idea behind object-based video segmentation is that each frame in a given video sequence is segmented into meaningful, real-world objects based on a pre-specified criteria (color, texture, motion, etc.). The correspondence between objects must be maintained between all frames in the video clip along with consistent region segmentation pertaining to each object's boundary. The wide range of multimedia applications currently being developed has emphasized the need for object-based video segmentation. Video surveillance applications must distinguish and track objects for security purposes. Video databases can utilize objects for content-based indexing and retrieval applications. The area of video compression benefits

directly from object-based segmentation as discussed in the MPEG-4 coding standards [2].

In general, video segmentation is a difficult task. Many existing techniques attempt to segment frames into objects from estimated motion based on MPEG motion vectors, optical flow, or affine motion transform [3,4,5]. Moving regions are often labeled as foreground objects while regions undergoing minimal motion are regarded as background objects. The following problems often occur when attempting to perform motion segmentation:

- Motion vectors are not always reliable in non-textured regions (due to correlation techniques implemented by most MPEG encoders)
- Estimated motion based on the affine transform is often inaccurate in close-up views.
- Motion pertaining to occluded and revealed objects is often estimated incorrectly.

Many algorithms, such as JSEG[1] and the Watershed algorithm [11] exist for the segmentation of color images into independent regions of color and texture, but it is a very difficult job to apply these same applications to video segmentation problems. There are millions of segmentation possibilities for each image, thus making consistent object regions difficult to maintain between successive frames. This proposed system utilizes a color variance measurement first introduced in the JSEG algorithm but significantly improves on JSEG's video-object segmentation consistency using the co-occurrence matrix texture features

## 2. Gray-Level Co-occurrence Matrix

The Gray-Level Co-Occurrence Matrix (GLCM) has provided texture features useful in the segmentation of images [7]. The following features have commonly been extracted from the GLCM:

- Contrast
- Homogeneity
- Entropy

- Energy
- Mean
- Standard Deviation
- Coefficient of Correlation

The Mean measurement is the primary feature utilized by this system and is given by the following equations below:

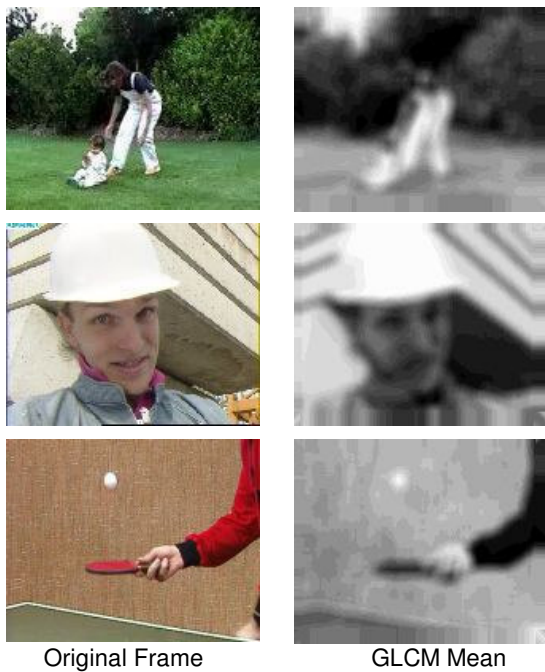
$$(1) \mu_i = \sum_j \sum_i i (P_{ij})$$

$$(2) \mu_j = \sum_i \sum_j j (P_{ij})$$

$$(3) \|\mu\| = (\mu_i^2 + \mu_j^2)^{1/2}$$

Where  $\mu_i$  is the horizontal mean,  $i$  is a given row value,  $P_{ij}$  is an element of the glcm,  $\mu_j$  is the vertical mean, and  $j$  is a given column value. The two values are equal because of the glcm's symmetry. The magnitude of the mean is expressed in equation (3).

Examples of plotting the mean's magnitude as a 2-dimensional textured image is shown in Figure 1.



**Figure 1. GLCM Mean Feature**

The resulting mean textured image is very smooth and almost all micro-textures have been removed. The individual region interiors possess consistent gray-scales throughout this image; therefore the region boundaries can be identified from

a basic edge detection filter. A  $N \times N$  filter computing the variance as

$$\text{var} = \frac{\sum_{i=0}^N \sum_{j=0}^N (\mu - g_{ij})^2}{(N)(N)}$$

Where  $\mu$  is the mean within the windowed region and  $g_{ij}$  is the average gray scale. For this system,  $N=9$  provides good results in most instances. A plot of the edge image generated as a result of applying the above filter is shown below in Figure 2. The images on the left will be referred to as the glcmv image in later discussion.

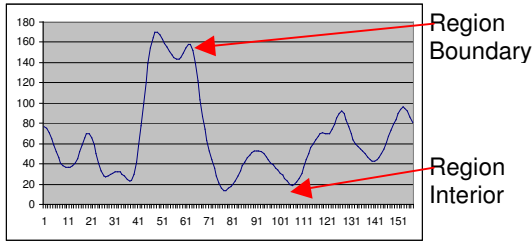


**Figure 2. GLCM Variance Feature**

### 3. Image Segmentation

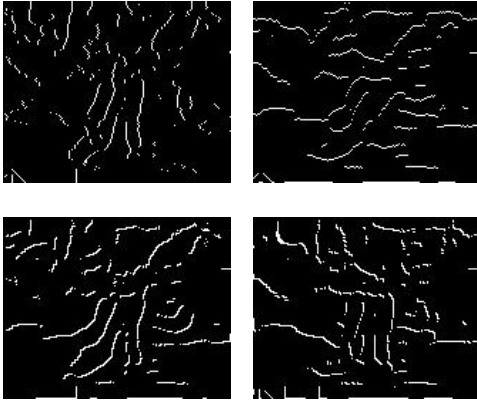
Although the glcmv images provide edges which distinguishes region boundaries, these edges are greatly blurred. The resulting images have been heavily smoothed due to the averaging effects of using the windowed region to compute the color variance features. Attempting to apply simple thresholding techniques (e.g. applying 1 global threshold for the entire image) provides unsatisfactory results because the interior object pixel values are non-zero. To illustrate this point consider the plot of the color variance values pertaining to a single

horizontal scan line extracted from the glcmvm image in Figure 3 is shown below:



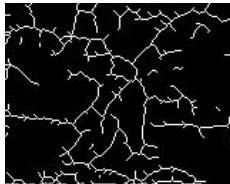
**Figure 3. GLCM Mean**

Contours are generated from the glcmvm by extracting the peaks of each of its scan lines taken in the horizontal, vertical, diagonal, and off diagonal directions. The peaks extracted from each scan line represent region's boundaries while the minimas represent region's interiors. The resulting contours from this step is shown below in Figure 4.:



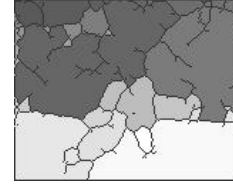
**Figure 4. Contours**

The corresponding horizontal, vertical, diagonal, and off diagonal scan line entries are then added together to form one image of optimal boundary points. This image is then thinned using the algorithm described in [9]. The resulting edges of this step is shown below in Figure 5:



**Figure 5. Contour Summation**

A gap-filling algorithm is applied to all contour endpoints resulting in a closed set of contours. A basic component-labeling algorithm [10] is then applied to the closed contours resulting in a region segmentation illustrated below in Figure 6:



**Figure 6. Component Labeling**

#### 4. MPEG -1/2 Motion Vectors

The first frame in the video sequence undergoes the image segmentation process described in section 3, resulting in a collection of initial objects to be tracked in the subsequent frames. The video sequences utilized in this system were extracted from either an MPEG-1 or MPEG-2 [2] video clip which provides motion vectors [2] which are then associated with the objects segmented in the initial frame of the sequence. These motion vectors are generated by the MPEG encoder based on the matching of 16x16 regions (aka as macroblocks) between the first and second frames. In other words, the motion vectors supply the location where a given pixel will appear in the next frame. A 16x16 region is intra-encoded [2] whenever a match cannot be determined between the first/second frames. Intra-encoded simply means the contents of the region cannot be predicted from the previous frame and must be encoded directly with the contents of the current frame. The following examples in Figure 7 illustrate the prediction of subsequent frames based on the previous frames by using MPEG motion vectors.



**Figure 7. Motion Vectors**

The solid black boxes represent the 16x16 regions that have no motion vectors (i.e., intra encoded regions). Notice the increased number of intra blocks that are generated when the girl steps back quickly. The segmentation of the intra blocks is unknown and

requires additional processing in order to obtain the correct region labels.

## 5. Intra Block Segmentation

The intra blocks (see Figure 7) can match any of the following segmentation possibilities:

- One object from the previous frame
- Multiple objects from the previous frame
- A new object entering the scene for the first time

The intra blocks undergoes the image segmentation algorithm described in section 3. The intra block's segmented regions are then matched with the adjacent regions of the previous frame. The regions are matched based on the following criteria:

- Common Quantized Color Difference
- Texture Difference

A standard k-means algorithm [4] quantizes all colors of the previous frame's regions and intra blocks into 16 clusters. The seven co-occurrence matrix texture features are also extracted for the previous frame's regions and the intra blocks of the current frame. The following rules are applied when matching intra block to assure the optimal segmentation:

- If 67% of the quantized colors match between a segmented region of the intra block and an adjacent region, and
- If the normalized Euclidean distance between the intra block region and the adjacent regions is within 33 then

The adjacent region is selected as the best matching region and the intra block is merged into this region.

If no adjacent object matches the following criteria, then the intra block is identified as a new object and is assigned the next available label. The completed intra block segmentation results are shown below in Figure 8:

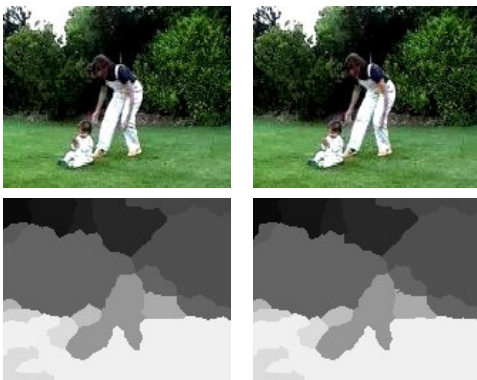


Figure 8. Intra Block Segmentation

## 6. Object Accuracy

A measurement pertaining to the accuracy of the segmented video is needed to assure quality before storing the object information into a database management system. This accuracy measurement is based on detecting color/texture differences between corresponding objects in adjacent frames and is summarized by the following two equations:

$$cdiff = \sum_i^N (qci - qpi) \quad (1)$$

$$tdiff = \|vc - vp\| \quad (2)$$

Where in equation (1)  $cdiff$  represents the quantized color difference,  $N$  is the number of quantized colors in the previous frame,  $qci$  is the  $i$ th quantized color in the current frame, and  $qpi$  is the  $i$ th quantized color in the previous frame. In equation (2),  $tdiff$  represents the texture difference,  $vc$  is the seven feature texture vector extracted from the co-occurrence matrix (see section 2) in the current frame, and  $vp$  is the seven feature texture vector extracted from the co-occurrence matrix in the previous frame. Equations (1) and (2) apply to only one object located in both the current/previous frames. These equations are applied to the objects tracked in the sequences shown below in Figure 9 (tracked object outlined in purple – only 5 are shown for 20 frame sequence) along with the color/texture differences between adjacent frames plotted

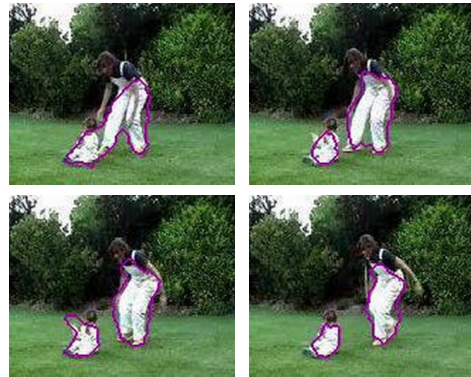


Figure 9. Object Tracking Results

## 7. Results

This object-based video segmentation system has been applied to 13 different sequences including conditions often found in videos - foreground object motion, motion of multiple objects, camera motion and editing effects. The following table lists the test sequences and the number of objects observed and

tracked, and the average object color/texture difference computed for each sequence:

**Table 1. Video Sequences**

Video Name	Frames	object	cdiff	tdiff
foreman.mpg	60 - 80	28	12.4813	4.1928
foreman.mpg	126 - 146	23	15.1206	7.4891
foreman.mpg	174 - 184	19	17.3884	11.9742
carphone.mpg	168 - 188	22	8.7793	3.7733
hall.mpg	36 - 56	27	9.2319	4.7031
container.mpg	246 - 270	18	5.1013	2.9350
tennis.mpg	0 - 20	8	13.8148	12.0103
happygranny	90 - 120	15	8.3057	3.9584
bike.mpg	78 - 98	9	10.7717	4.4899
anni005.mpg	0 - 20	23	9.8985	4.9592
anni005.mpg	164 - 184	31	11.1678	3.8342
anni005.mpg	1000-1020	25	12.5442	5.7909
anni005.mpg	2727 - 2747	3	19.5778	8.6516

This tool can be used as a substrate layer for video retrieval systems where the video accuracy measurement can estimate the likelihood of retrieving objects from the sequence.

## 8. References

- [1] Yining Deng and B.S. Manjunath, "Unsupervised Segmentation of Color-Texture Regions in Images and Video," IEEE Trans on Pattern Analysis and Machine Intelligence, vol. 22, no. 6, p.939-954, 2001.
- [2] Leonardo Chiariglione - Convenor . (June 1996) MPEG-1 ISO/IEC JTC1/SC29/WG11 NMPEG 96.
- [3] M.M. Chang, "Simultaneous motion estimation and segmentation," IEEE Trans. on Image Processing, vol. 6, no. 9, p.1326-33, 1997.
- [4] A Jain, M Murty, and P.Flynn, "Data Clustering - A Review", ACM Computing Surveys, 31(3), September 1999.
- [5] J. Shi , "Motion segmentation and tracking using normalized cuts," Proc. Of IEEE 6th Intl. Conf. on Computer Vision, p.1154-60, 1998.
- [6] R. Jain and R. Kasturi, Machine Vision, New York, McGraw Hill, 1995, pp. 236-239.
- [7] L.S. Davis, S. Johns, Texture analysis using generalized co-occurrence matrices, IEEE Trans. Pattern Anal. Machine Intell. 1 (3), 251-259 (1979).
- [8] M. Gelgon and P. Bouthemy, "A region-level motion-based graph representation and labeling for tracking a spatial image partition," Pattern Recognition, vol. 33, no. 4, pp. 725-740, 2000.
- [9] A. Rosenfield and A.C. Kak, Digital Picture Processing, Vol 2, New York Academic, 1982, pp. 232- 236.

[10] A. Rosenfield and A.C. Kak, Digital Picture Processing, Vol 2, New York Academic, 1982, pp. 240- 244.

[11] L. Shafarenko, M. Petrou, and J. Kittler, "Automatic watershed segmentation of randomly textured color images," IEEE Trans. on Image Processing, vol. 6, no. 11, p. 1530-44, 1999.