Proc. VIIth Digital Image Computing: Techniques and Applications, Sun C., Talbot H., Ourselin S. and Adriaansen T. (Eds.), 10-12 Dec. 2003, Sydney

Unsupervised Segmentation of Moving MPEG Blocks Based on Classification of Temporal Information

Ofer Miller[1], Amir Averbuch[1], and Yosi Keller[2]

[1]School of Computer Science,Tel-Aviv University,
Tel-Aviv 69978, Israel
[2]Dept. of Computer Science, Yale University,
New Haven 208285, CT, USA
{millero,Amir}@post.tau.ac.il

Abstract. Most codecs such as MPEG-1,2 are based on processing pixels in predetermined tiles of square blocks. The processing of each block is independent of the object-based contents that may be present in these blocks. Segmentation of moving object blocks enables, for example, efficient distribution of bits among consecutive frames in video sequences by allocating more bits to moving object blocks at the expense of the background blocks. This paper proposes an algorithm which determines whether the MPEG blocks contain static or moving objects. The proposed algorithm is based on temporal statistical information obtained by multiple comparisons of consecutive frames. A dynamic process of gathering statistical information is converged into three clusters with different characterizations. Automatic classification determines which blocks belong to either a moving object (foreground) or a static object (background). The number of look-ahead frame comparisons, which is required to determine the type of each block, is automatically and adaptively determined according to the nature of the processed sequence.

1 Introduction

The information whether a MPEG block contains moving or static objects is important to many applications such as image sequence restoration, motion compensation, bit rate distribution and contents interaction. A block that contains a moving part or a complete moving object is called a moving block (foreground) while one that contains a static object is called a static block (background). If we can identify the moving MPEG blocks then we can derive a more efficient and smarter bit distribution among the frames to achieve better compression in the video-encoding phase. One immediate use is the improvement of the compression ratio within the framework of many existing video standards, like H.263, H26L or MPEG-1/2/4.

In this paper we develop an automatic adaptive algorithm that analyzes video sequences in order to separate between background and foreground MPEG blocks. One of the problems in conventional block-matching algorithms, which is indirectly addressed in this paper, is that the distortion measure being used is unable to distinguish between errors introduced by object motion or by luminance and/or contrast changes. Errors caused by luminance changes result in motion vectors, which may negatively affect the image coherency. Luminance correction tries to reduce the matching-error by modifying the distortion criteria with the introduction of illumination correction parameters (see 1,2,3). However, much work deal with motion segmentation, ignore the question of depth ordering of layers (which are in front of which), the problem of occlusion, and the problem of luminance changes. When they are considered, it is by identifying those outliers of the motion estimation which could be due to the occlusion of one layer by another 4,5. Segmentation between static and moving MPEG blocks enhances the bit rate distribution between the blocks, and thus, the coherency of the image is preserved.

Our proposed segmentation algorithm compares a frame to several of its consecutive frames. Each comparison provides information on the intensity changes across the sequence. Two temporal statistical tests are used. The tests are performed as an incremental calculation along the comparisons results between the pairs. After each comparison we classify the block according to the relations between these statistical tests. We show that after *sufficient* iterative comparisons between the input frame and its consecutives, the moving object blocks are constantly classified into pre-known clusters, while others do not have any periodic or constant cluster pattern. The number of comparisons needed is automatically determined and depends on the nature of the sequence.

The rest of this paper is organized as follows. Section 2 describes the main algorithm, this section contains description for the comparison modeling technique and for the classification process. In section 3 we present experimental results.

2 The MPEG Block Segmentation Algorithm

The segmentation algorithm contains three main steps: multiple temporal comparisons, temporal analysis and classification. The first is the 'multiple temporal comparisons', which is an extension of the change detection technique 7 into a temporal comparison among a series of consecutive frames. We compare a given frame I_k (reference frame) with its next successive frames I_t, $t=k+1,...,T_k$, where T_k is the number of required comparisons. The results from each comparison are held in a temporal comparison matrix (*TCM*). The change detection algorithm assumes that the images, which participated in the comparison process, are registered.

The second step is the 'temporal analysis', which analyzes the *TCM* by two statistical tests: *distribution (dst)* and *mean*. The results will be the inputs to the classification step which the third step. The classification step separates between moving blocks and the rest of the blocks. It is called after each iterative comparison, and only when there is sufficient information to classify the MPEG blocks, the algorithm is terminated.

The phases of the algorithm depend on each other. As long as the results of the statistical tests do not converge into a preset cluster, the second phase continues to compare more frames and the temporal analysis continues to generate updated tests.

2.1 Multiple Comparisons Modeling

Let k be the index of the reference frame. Each of its successive I_t, $t=k+1,...,T_k$, will be compared with the reference frame I_k, where T_k is the index of the last frame that participates in the comparisons. It is adaptively determined (section 2.4) according to the nature of the examined sequence. We assume the background is static after the motion of the camera is compensated 9,11 . Then each frame I_t can be written as:

$$I_t = bk_{(t)} + obj_{(t)}\,,\ \ t = k+1,\ldots,T_k$$

(1)

where $bk_{(t)}$ and $obj_{(t)}$ represent the background and object pixels in frame I_t, respectively. Each comparison result (the comparison technique is described in section 2.2) between I_k and I_t is held in a temporal comparison image (TCI_t). We divide the comparison results between I_k and I_t into four different groups of pixels: $bk_{(TCI)}$, $cv_{(TCI)}$, $uncv_{(TCI)}$ and $ol_{(TCI)}$ (shown in Figure 1), which are the background, the covered, the uncovered and the overlapped groups of pixels, respectively. Thus, the TCI_t can be written as:

$$TCI_{(t)} = bk_{(TCI)} \bigcup cv_{(TCI)} \bigcup uncv_{(TCI)} \bigcup ol_{(TCI)} \qquad \textbf{(2)}$$

where

$$bk_{(TCI)} = bk_{(k)} \bigcap bk_{(t)} \qquad \textbf{(2.1)}$$

is the common background (in pixels) of I_k and I_t ;

$$cv_{(TCI)} = obj_{(t)} - \{obj_{(k)} \bigcap obj_{(t)}\} \qquad \textbf{(2.2)}$$

and

$$uncv_{(TCI)} = obj_{(k)} - \{obj_{(k)} \bigcap obj_{(t)}\}$$

(2.3)

are the covered and uncovered regions, respectively, produced by the object's movement and;

$$ol_{(TCI)} = obj_{(k)} \bigcap obj_{(t)}$$

(2.4)

is the overlapping pixels between the groups $obj_{(t)}$ and $obj_{(k)}$.

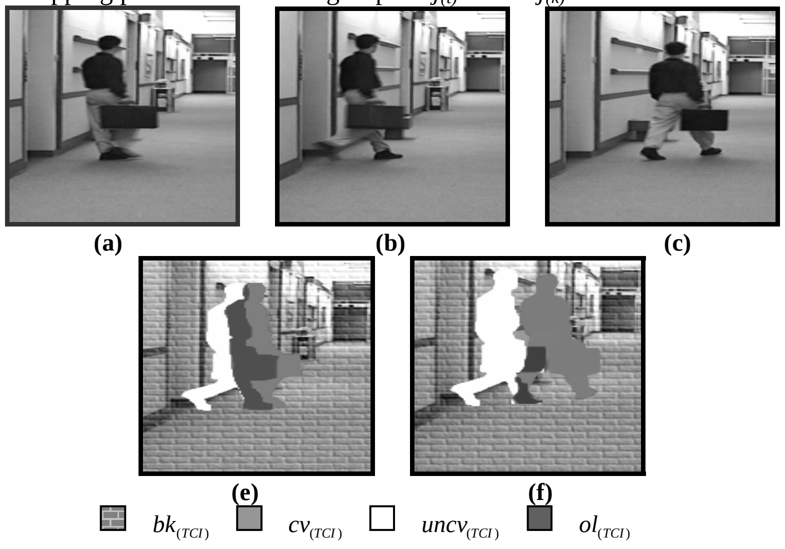

Figure 1: (a), (b), (c) are the original frames where the lag between them is five. (e) and (f) represent the temporal comparison images ($TCI_{(t)}$) between (a),(b) and (a),(c) respectively. $bk_{(TCI)}$ is the common background. $uncv_{(TCI)}$ and $ol_{(TCI)}$ contain the object pixels in their original position (image (a)). $cv_{(TCI)}$ and $ol_{(TCI)}$ contain the object pixels in I_t (images (b) and (c)).

Except for the $bk_{(TCI)}$ group of pixels, all the described regions are assumed to be 'change' areas for a typical change detection algorithm 7, 8. In general, when an object does not stay in a fixed position we expect that for a certain $t, t \in k+1,...,T_k$: $obj_{(k)} \cap obj_{(t)} = \phi$, $|uncv_{(TCI)}| = |cv_{(TCI)}|$ and $|ol_{(TCI)}| = \phi$. In other cases, when the object is moving but remains close to its original location in I_k, we expect that $|ol_{(TCI)}| \approx |obj_{(k)}|$, $obj_{(k)} \cap obj_{(t)} \approx |obj_{(k)}|$ and $|cv_{(TCI)}| \approx \phi$.

2.2 Comparison between Consecutive Frames

Detection of changes between frames can be carried out with a variety of existing change detection algorithms. However, since the *MC* methodology compares frames with growing accumulated time differences, the algorithm has to handle luminance and noise changes that are amplified during the comparisons. In order for the algorithm to be independent of these changes, we extend the model that was described in 7, to handle multiple temporal comparisons between several consecutive frames. First, we choose a moving window $\eta(x,y)$ of $N_\eta = 16$ elements, which belongs to the division image. Let

$$\tau_t^k(i,j) \overset{\Delta}{=} \frac{I_k(i,j)}{I_t(i,j)} \qquad \textbf{(3)}$$

$$\hat{\mu}_{(t)}(x,y) \overset{\Delta}{=} \frac{1}{N_\eta} \sum_{(i,j)\in\eta(x,y)} \tau_{(t)}^{(k)}(i,j) \qquad t \in k+1,...,T_k\,. \qquad \textbf{(3.1)}$$

Then, for each pixel, a statistical difference (SD) is computed between I_k and I_t where the result indicates that a local change between the frames has occurred:

$$sd_t(x,y) = \frac{1}{N_\eta} \sum_{(i,j)\in\eta(x,y)} \left(\tau_t^k(i,j) - \hat{\mu}_t(x,y)\right)^2 \qquad t \in k+1,...,T_k\,; \qquad \textbf{(3.2)}$$

where t is the index of the current compared frame. $Sd_t(x,y)$ will be used as an indication whether or not a change was detected in the (x,y) coordinate of I_k relative to I_t. Binary values of each pair of compared frames are stored in a Temporal Comparison Matrix (TCM) that has the following structure for $t=k+1,...,T_k$:

$$TCM_{(t)}(x,y) = \begin{cases} 1 & sd_t(x,y) > \beta \\ 0 & otherwise \end{cases} \qquad \textbf{(3)}$$

where β is a predefined threshold. Note that analysis among sequence of frame comparisons caused the threshold β to be insensitive to a local error in changes across the sequence. By applying Eq. (**3**) on T_k frames we obtain the (TCM) matrix, as illustrated in Figure 2.

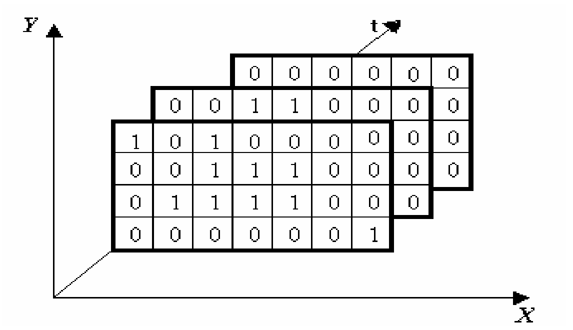

Figure 2: The *temporal comparison matrix,* which is filled using Eq. (3) on T_K frames. Each binary value at (x,y,t) is the output from the comparison between frame I_t and I_k for an 8x8 block whose center is located at (x,y).

Figure 3 demonstrates the binary changes of the *TCM* slices relative to previous slices as a function of the progress of frames comparisons. The video sequence in Figure 3 is characterized by a slow motion of the objects between the frames. Therefore, the advantage of the comparison methodology becomes evident when $t \geq 7$. Image (e), which is the output after three comparisons (t=3) from the reference frame, shows that the change detection algorithm fails to detect many of the interiors of the object's regions. In image (g), which is the output for $t=10$, there is a significant improvement in the change detection mask inside the object regions (bounded by red curve). However, the byproduct of having a large lag (t) between the compared frames is that many of the covered, $cv_{(TCI)}$, areas (outside of the red curves) are also marked as 'change'. These regions are visible because of the accumulated error during the comparison process.

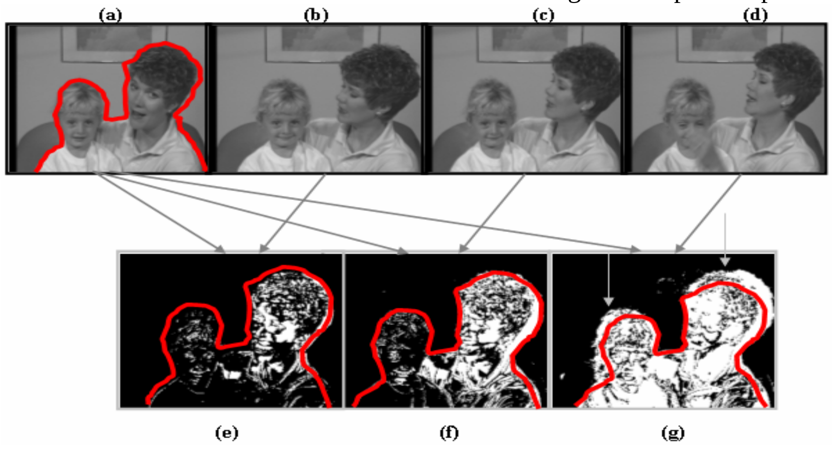

Figure 3: Images (a), (b), (c) and (d) are four frames taken from a video clip. (a) is the reference frame I_k (k=0), (b) is frame $I_{t=3}$, (c) is frame $I_{t=7}$ and (d) is frame $I_{t=10}$. Images (e), (f) and (g) are the three slices taken from the *TCM*. (e) represents the slice of $TCM_{(3)}$. (f) represents $TCM_{(7)}$ and (g) represents $TCM_{(10)}$.

2.3 Temporal Analysis

As was shown, there exist T_k such that the values of $TCM_{(t)}(x,y)$, $\forall(x,y)\in obj_{(k)}$, are classified as 'change' (how T_k is determined will be discussed in section 2.4). However, focusing on a single $TCM_{(t`)}(x,y)$ slice without taking into consideration the previous $t$ slices where $t=k+1,...,t`$, may lead to wrong classifications such as considering background blocks as foreground blocks (in particular, over the occluded and the uncovered area, which become larger as $t`>>k+1$). Change detection algorithm, such as 7, does not design to deal with such lag between compared frame nor to distinguish between occluded and uncovered area. The temporal analysis across the *TCM* slices enables us to differentiate between these regions.

Each slice in the $TCM_{(t)}$, $t=k+1,...,T_k$, is composed of four different regions: $bk_{(TCI)}$, $cv_{(TCI)}$, $uncv_{(TCI)}$ and $ol_{(TCI)}$ (see section 2.1). Thus, for the reference frame I_k we have:

$$obj_{(k)} = uncv_{(TCI)} \bigcup ol_{(TCI)} \quad \textbf{(4)}$$

$$bk_{(k)} = cv_{(TCI)} \bigcup bk_{(TCI)} \,. \quad \textbf{(5)}$$

Our claim is that by a statistical test, which is applied on the *TCM* slices, both $uncv_{(TCI)}$ and $ol_{(TCI)}$ have the same distinct distribution. The following formulate the basis for the statistical tests:

I. $\forall t > \lambda, \forall(x,y)\in obj_{(k)} \Rightarrow TCM_{(t)}(x,y) > \beta \quad k+\lambda \le t \le T_k\,,\ \lambda \ge 1$

II. $\exists t < T_k, \exists(x,y)\in bk_{(k)} \Rightarrow TCM_{(t)}(x,y) > \beta$

where k is the index of the reference frame and β is a pre-defined change detection threshold. Statement *I* assumes that for any $t \ge \lambda$, $obj_{(k)}$ pixels are considered as a 'change', independent of the object's movements in the sequence. The second statement assumes that there is a time difference between the compared frames such that if an object covers the background region in I_t then this region is considered as a 'change' at this comparison. Recall that both statements rely on the temporal comparison definition in section 2.1.

In addition to the above statements, the following defines the statistical tests, which compute the distribution, *dst*, (Eq. 7) and the mean, *mean*, (Eq. 6) of the comparisons results. Its is done for a single block in I_k with central coordinates (i,j), for t iterative comparisons:

$$mean^{t}_{(i,j)} = \frac{1}{t}\sum_{s=1}^{t}\left(\frac{1}{N^2}\sum_{x=i-\frac{N}{2},y=j-\frac{N}{2}}^{x+\frac{N}{2},y+\frac{N}{2}} TCM_{(s)}(x,y)\right) \quad \textbf{(6)}$$

$$dst^{t}_{(i,j)} = \frac{1}{t}\sum_{s=1}^{t}\left\{\frac{1}{N^2}\sum_{x=i-\frac{N}{2},y=j-\frac{N}{2}}^{i+\frac{N}{2},j+\frac{N}{2}} TCM_{(s)}(x,y) - mean^{t}_{(i,j)}\right\}^2 \quad \textbf{(7)}$$

N^2 is the block size in pixels $i>N/2$, $j>N/2$ and $t=k+1,...,T_k$. The *distribution* test (Eq. 7) calculates the temporal variance of a specific block for all *TCM* slices. After each comparison, a new slice is added to $TCM_{(t)}$, and its block distribution is updated. Similarly, the *mean* test (Eq. 6) computation, which calculates the temporal mean of a specific block for all the $TCM_{(t)}$ slices, $t=k+1,...,T_k$.

2.4 Block Classification

The goal of the classification phase is to separate the statistical results between foreground and background blocks. The results of the two statistical tests are placed in a feature vector (*FV*). Each MPEG block $b_r(i, j)$, $r = 1,..., |I_k| / N \times N$, is associated with a distinct *FV* where (i, j) are the central coordinates of the block. The *FV* is denoted by $FV^t_{b_r(i,j)} = (FV^t_{b_r(i,j)}(1), FV^t_{b_r(i,j)}(2))$ where $FV^t_{b_r(i,j)}(1) = dst^t_{(i,j)}$ and $FV^t_{b_r(i,j)}(2) = mean^t_{(i,j)}$, $t = k+1,...,T_k$. Preset clusters of the background, object and occlusion regions are defined. In general, a simple classification technique, which models the clusters by minimal distances, is required (10, Chapter 16.3). However, since reliability of the $FV^t_{b_r(i,j)}$ depends on the number of MC iterations, we have to incorporate into the classification technique the ability to determine how many *MC* iterations are needed for reliable blocks classification.

Assume $FV^t_{b_r(i,j)}$ is assigned to each block $b_r(i, j)$ for $t = k+1,...,T_k$ iterative comparisons. Each comparison adds essential information to $FV^t_{b_r(i,j)}$ that fits the object temporal movement in I_t. As mentioned (section 3.2), the more iterative comparisons that taking place, the more reliable information on object movements is contained in $FV^t_{b_r(i,j)}$. To use this fact, we first define a classification space Ω:

$$\Omega = \{(x, y) \mid 0 \le x \le 100 \quad 0 \le y \le 100\}.$$

This space is partitioned by three different subspaces. Each subspace is called a preset cluster, denoted by C_p, $p = 1,2,3$. The preset clusters are determined to indicate the high probability of belonging to one of the following:

$$C_1 \overset{\Delta}{=} \{(x, y) \mid (x, y) \in (ol_{(TCI)} \cup uncv_{(TCI)})\},$$

$$C_2 \overset{\Delta}{=} \{(x, y) \mid (x, y) \in cv_{(TCI)}\},$$

$$C_3 \overset{\Delta}{=} \{(x, y) \mid (x, y) \in bk_{(TCI)}\},$$

and $C_1 \cap C_2 \cap C_3 = \phi$. As a result, a void space (*VS*), denoted by C_{VS}, is obtained (see Figure 5):

$$C_{vs} \overset{\Delta}{=} \{(x, y) \mid (x, y) \in \Omega - (C_1 \cup C_2 \cup C_3)\}.$$

Recall that (as defined in section 2.1) $ol_{(TCI)} \cup uncv_{(TCI)}$ represents the object's region in I_k while $CV_{(TCI)}$ and $bk_{(TCI)}$ represent the background region. The preset clusters were determined (see Figure 5) after applying the classification technique on one hundred video sequences.

Then, for a certain t, $t = k+1,...,T_k$, a given block $FV^t_{b_r(i,j)}$ is classified to belong to either one of the preset clusters C_p, $p \in 1,2,3$, or to C_{VS}. Based on *MC* methodology, we claim that if after t iterative comparisons $FV^t_{b_r(i,j)} \subseteq C_{VS}$, then there is insufficient information to classify this block; otherwise $FV^t_{b_r(i,j)} \subseteq C_p$ for one *p*, $1 \le p \le 3$. In this case, more iterative comparisons are required and *t* is increased by one.

However, an exceptional situation has to be taken into consideration when $FV^t_{b_r(i,j)} \subseteq C_{VS}$ for all $t = k+1,...,T_k$. This may occur when an *MO* becomes static in its movements after λ frames where $k+1<\lambda \le T_k$. In such a case, the following minimum distance technique is computed.

Each cluster C_p, $p=1,2,3$, is defined by its center of gravity $m_p(x,y)$ where $x=\frac{1}{|C_p|}\sum_x C_p^x$, $y=\frac{1}{|C_p|}\sum_y C_p^y$, C_p^x , C_p^y are the *x* and *y* coordinates of C_p , and $|C_p|$ is the size of C_p in pixels. $FV^t_{b_r(i,j)}$ can be classified by searching its minimal distance (*md*) from the $m_p(x,y)$, $p=1,2,3$, by:

$$md_{b_r(i,j)} = \min_{p=1,2,3}\left\{\sqrt{(FV^t_{b_r(i,j)}(1)-m_p(x))^2+(FV^t_{b_r(i,j)}(2)-m_p(y))^2}\right\} \qquad \textbf{(8)}$$

where $FV^t_{b_r(i,j)}(1)$ and $FV^t_{b_r(i,j)}(2)$ are the $dst^t_{(i,j)}$ and $mean^t_{(i,j)}$ of the block, respectively.

To combine the minimal distance, we set a constant T_{max} to be the maximum number of iterative comparisons allowed. $FV^t_{b_r(i,j)}=(FV^t_{b_r(i,j)}(1),FV^t_{b_r(i,j)}(2))$, $r=1,\ldots,|I_k|/N\times N$, is classified only if $FV^t_{b_r(i,j)}\subseteq C_p$ for one *p*, $1\le p\le 3$. Otherwise, if $t\ge T_{\max}$ and $FV^t_{b_r(i,j)}\subseteq C_{VS}$ the block will be classified according to its minimal distance from $m_p(x,y)$, $p\in 1,2,3$, using Eq. (8). In this case, $T_{\max}=T_k$ and the algorithm is terminated. The complete classification per single *t* iterative comparison consists of the following steps:

```
Notation:
  $List_{(FV)}$  list of $FV^t_{b_r(i,j)}$ , $r=1,\ldots,|I_k|/N\times N$
CLASSIFY ($List_{(FV)}$ ,$t$, $T_{max}$)
  1. for $r=1,\ldots,|I_k|/N\times N$ do:
       a. for $p=1,2,3$ do:
            if $FV^t_{b_r(i,j)}\subseteq C_p$ then $b_r(i,j)\leftarrow p$
            else $b_r(i,j)\leftarrow VS$
  2. if $t\ge T_{\max}$ do:
  3. for $r=1,\ldots,|I_k|/N\times N$ do:
       a. If $b_r(i,j)=VS$ do:
            i. compute the minimum distance between
               $FV^t_{b_r(i,j)}=(FV^t_{b_r(i,j)}(1),FV^t_{b_r(i,j)}(2))$ and $m_p(x,y)$ $p=1,2,3$ using
               Eq.(8).
            ii. $b_r(i,j)\leftarrow p$ ($p$ is closest cluster to $FV^t_{b_r(i,j)}$).
  4. return $List_{(FV)}$
```

3 Experimental Results

The following illustrates the algorithm output at four different iterative classification. In Figure 4, the objects (sign by the gray arrows) blocks moved differentially one to each other, not all the objects blocks move at the same time. Some move after one or two frames (from the reference frame) while others move after five frames or even more. Figure 4.0 is the reference frame I_k to be segmented.

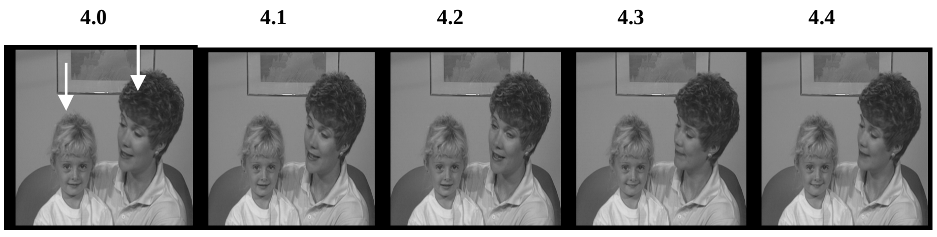

Figure 4: Five consecutive frames taken from "mother-daughter" video sequence

Figure 5(a,b,c,d) presents the modeled C_p clusters, $p \in 1,2,3$, the black dots are the *FV* of each iterative comparison. Each cluster is shown by a black polygon that surrounds it. The three presented clusters (at Figure 5 a,b,c and d) were obtained from a training group of one hundred different video sequences. Images a, b and c present the classification results for t=3, t=7 and t=12 iterations. Figure 5d presents the final output of the classification step, which was obtained after T_k=15 iterations. Since after the final iteration there are still several blocks located inside the C_{vs} subspace, these blocks are classified by the minimum distance technique (see section 2.4).

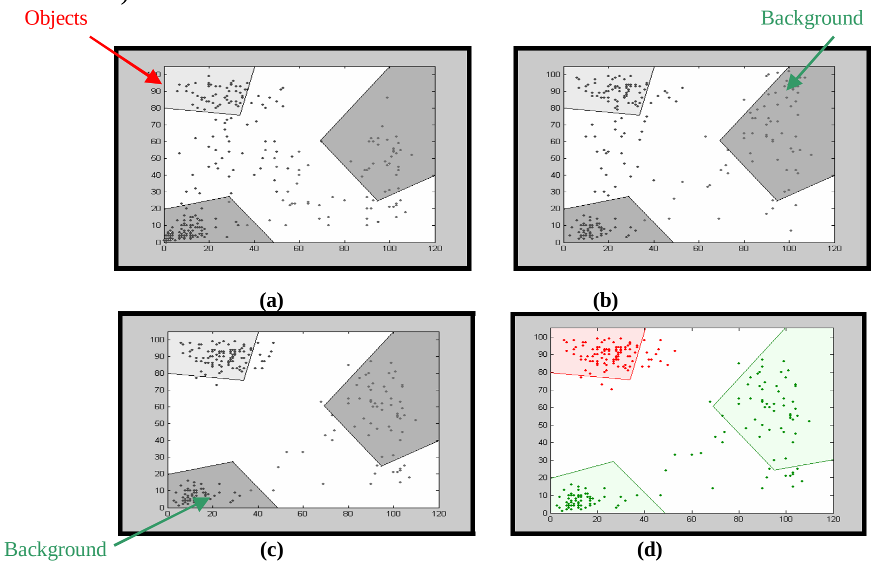

Figure 5: *FVs* distribution of the blocks in image 4.0. (a), (b) and (c) illustrate the segmentation results after t=3 t=7, t=12 iterative comparisons. (d) is the final block classification.

According to the final classification in Figure 5d, we present in Figure 6a the grid of the block segmentation results such that the red and the green blocks belong to the red and green clusters (in Figure 5d), respectively.

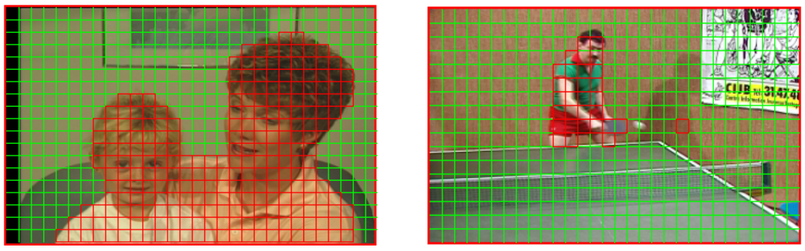

Figure 6: (a) The final segmentation output for Figure 4.0 after the $T_k=15$ comparisons. (b) The final output for single frame from the Tennis movie after the $T_k=12$ comparisons

References

1. M. Gilge. "Motion estimation by scene adaptive block matching (sabm) and illumination correction". *Image Processing Algorithms and Techniques*, volume 1244 of SPIE Proceedings, pages 355-366, (1990).
2. P. Eisert and B. Girod. Model-based 3D-motion estimation with illumination compensation. *Proceedings of Image Processing and its Applications* (IPA'97), VOL 443, pages 194-198, Dublin, Ireland, July (1997).
3. P.M. Kuhn and W. Stechele. VLSI architecture for variable size motion estimation with luminance correction. In F.T. Luk, editor, *Advanced Signal Processing: Algorithms, Architectures, and Implementations VII, volume 3162 of SPIE Proceedings*, pages 497-508, October (1997).
4. L. Bergen and F. Meyer. Motion segmentation and depth ordering based on morphological segmentation. *Proc. 5th ECC*V, volume II, pages 531–547, Freiburg, Germany, June (1998).
5. Y. Weiss and E. H. Adelson. A unified mixture framework for motion segmentation: Incorporating spatial coherence and estimating the number of models. In *Proc. CVPR '96*, pages 321–326, San Francisco, CA, June (1996).
6. P. H. S. Torr, R. Szeliski, and P. Anandan. An integrated Bayesian approach to layer extraction from image sequences. In *Proc. 7th ICC*V, volume II, pages 983–990, Kerkyra, Greece, September (1999).
7. K. Skifstad R. Jain, "Illumination independent change detection for real world image sequences", *Computer Vision, Graphics, and Image Processing*, Vol. 46, pp. 387-399, (1989).
8. Til Aach, Andre Kaup, Rudolf Mester, "Statistical model-based change detection in moving video", *Signal Processing*, Vol. 31, pp. 165-180, (1993).
9. F. Dufaux and J. Moscheni and A. Lippman. "Efficient, robust and fast global motion estimation for video coding". *IEEE Trans on Image Processing*, Vol .9, Mo. 3, pp. 497-500, (2000).
10. B. Jahne. "Digital Image Processing concepts algorithms and scientific application" 4th edition. (1997).
11. A. Averbuch, Y. Keller, "Fast motion estimation using bidirectional gradient methods", to appear in *IEEE Trans. on Image Processing.*