Multimedia Tools and Applications, 24, 253–272, 2004

A Formal Model for Video Shot Segmentation and its Application via Animate Vision*

MASSIMILIANO ALBANESE malbanes@unina.it
ANGELO CHIANESE angchian@unina.it
VINCENZO MOSCATO vmoscato@unina.it
LUCIO SANSONE sansone@unina.it
Dipartimento di Informatica e Sistemistica, Università di Napoli "Federico II"

Abstract. The first step in a video indexing process is the segmentation of videos into meaningful parts called *shots*. In this paper we present a formal model of the *video shot segmentation* process. Starting from a mathematical characterization of the most common transition effects, a video segmentation algorithm capable to detect both *abrupt* and *gradual* transitions is proposed. The proposed algorithm is based on the computation of an arbitrary similarity measure between consecutive frames of a video. The algorithm has been tested adopting a similarity metric based on the *Animate Vision* theory and results have been reported.

Keywords: video indexing, video segmentation, shot transitions, animate vision

1. Introduction

The rapid advances in video technology, the widespread diffusion of video products—such as digital cameras, camcorders and storage devices—and the explosive growth of the internet have quickly made of digital video an essential component of today multimedia applications, including video-on-demand, video conferences, multimedia authoring systems and so on. Thanks to the development of multimedia compression techniques, we have observed an exponential increase in the amount of available digital video data. The major bottleneck limiting a wider use of digital video is the ability of finding desired information from a huge database using content. A way to enable fast access to video clips is to properly index video sequences.

The first step in an automatic video indexing process is the so called *video shot segmentation*. The objective of video shot segmentation is to partition a video into basic, meaningful parts called *shots*. Each video shot represents a meaningful event, or a continuous sequence of actions, and it corresponds to a sequence of frames captured from a unique and continuous record by a camera. Further scene analysis and interpretation could then be performed on such units. Semantic annotations, determined either by an human or by means of some automatic or semiautomatic content analysis techniques, can be associated to each shot

*This work has been carried out partially under the financial support of the Ministero dell'Istruzione, dell'Università e della Ricerca (MIUR) in the framework of the FIRB Project "Middleware for advanced services over large-scale, wired-wireless distributed systems (WEB-MINDS)".

to complete the indexing process. The segmented video sequences could also be used for browsing, in which only one or a few representative frames of each shot are displayed.

Different kinds of transitions may occur. The basic distinction is between *abrupt* and *gradual* ones. An abrupt transition occurs in a single couple of frames, when stopping and restarting the video camera. A gradual transition is obtained using some spatial, chromatic or spatio-chromatic effects, such as *fade in (out)*—i.e. a gradual increase (decrease) in brightness resulting in a solid color frame—or *dissolve*—i.e. a gradual super-imposition of two consecutive shots.

Abrupt transitions are very easy to detect because the two successive frames involved in the transition are totally uncorrelated. On the contrary gradual transitions are harder to detect from a data-analysis point of view because the difference between consecutive frames is substantially reduced. Considerable work has been reported on the detection of abrupt transitions. All the proposed techniques evaluate some similarity measure between successive frames and assume that a cut occurs when the value returned by the measure is lower than a fixed threshold. The comparison of successive frames is not useful to detect gradual transitions because the difference between them is very small.

One of the main issue in segmenting a video sequence into shots is the ability to distinguish between scene breaks and normal changes into the scene. Moreover camera movements, such as panning, tilting and zooming, present similar features to transition effects such as dissolves. A reliable video segmentation algorithm must be able to recognize dissolve effects without misinterpreting camera movements as gradual transitions.

In this paper we present a formal model for the video shot segmentation process and propose a video segmentation algorithm capable to detect both abrupt and gradual transitions. The paper is organized as follows: in Section 2 related works are discussed. In Section 3 the formal model for the segmentation process is presented and discussed and a mathematical characterization of cuts and dissolves is also proposed. In Section 4 the segmentation algorithm is presented and discussed. Section 5 illustrates the experimental results obtained adopting a metric $\mathcal{M}_{\text{image}}$ founded on the *Animate Vision Theory* and described in Appendix A. Concluding remarks are given in Section 6.

2. Related works

Considerable work has been reported on detecting abrupt transitions both in uncompressed [31] and compressed video [5, 6, 18, 29, 32]. The first methods for shot detection used video features from the uncompressed domain, such as pixelwise difference [31], histogram difference [19], motion vector analysis [6] and edge tracking [30]. All these methods are based on the comparison between successive frames in the video sequence and assume that a cut occurs when a certain physical feature suddenly changes. The introduction of the MPEG standard has redirected the research efforts for the video segmentation in the compressed-domain. The most common approaches in this domain are based on DC-components [18, 32], DC-sequences [29] and number of interpolated blocks [5]. In [7] a unified approach to scene change detection in both uncompressed and compressed video is proposed.

Authors agree that pixel-based methods are highly sensitive to motion of objects, so they generate an high rate of false detection. On the contrary histogram-based methods provide a

better trade-off between accuracy and speed, and the performances of such methods are good for the case of abrupt scene changes. However, color histograms provide information about the color composition of images, but not about the spatial distribution of color, so different images could have similar histograms. In order to overcome the limitation due to the use of single comparison technique, in [23] Philips and Wolf propose a multi-attribute algorithm for detecting cuts in video programs: the algorithm uses a motion metric to identify a set of cuts, then uses luminance histograms to eliminate false cuts.

Also the detection of gradual transitions has been widely investigated. At the best of our knowledge, a lot of techniques and algorithms have been proposed for fading (fade-in, fade-out) regions detection [6], while there are few works about the detection of other special effects such as dissolve and wipe. Lienhart [16] casts the problem of automatic dissolve detection as a pattern recognition and learning problem. Nam and Tewfik [20] use a technique based on B-spline polynomial curve fitting. In [14] Li and Wei propose a dissolve detection method based on the analysis of Joint probability Images. In [3] Bull, Fernando and Canagarajah propose a dissolve detection algorithm based on some statistical features of an image, while in [15] a motion-tolerant algorithm for dissolve detection is presented.

In conclusion, a general video segmentation algorithm independent from a particular frame comparison technique is highly desirable. Furthermore we can state that it is the first time that *Animate Vision* theory is used in a video segmentation process.

3. A formal model for video shot segmentation

3.1. Segmentation

In this section we present a formal model of the video segmentation process. The basic assumption is that a video is a sequence of frames, which are still images. Given this assumption, the problem of detecting where a shot change occurs in a video stream can be approached using some techniques developed in the field of image databases. A key concept of all the researches in this field is the *similarity between two images*. The greater is the similarity between two images, the smaller is the distance between them. Let us formalize the concepts of *image distance* and *image similarity metric*.

Definition 1 (Distance). Let $\mathcal{X}$ be a nonempty set. A function $\delta : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is called a distance on $\mathcal{X}$ if and only if the following conditions hold, for all $x, y, z \in \mathcal{X}$:

Nonnegative definiteness: $\delta(x, y) \geq 0$, and $\delta(x, y) = 0 \Leftrightarrow x = y$;
Symmetry: $\delta(x, y) = \delta(y, x)$;
Triangularity: $\delta(x, y) \leq \delta(x, z) + \delta(z, y)$.

The real number $\delta(x, y)$ is called the *distance* between x and y.

According to the above definition, an *image distance* can be defined as a distance on the set $\mathcal{X}$ of all the images. In the image realm, an useful but not necessary property is the following:

Normalization: The values of δ lie in the closed interval [0, 1].

Image similarity is usually introduced as the complement to 1 of a normalized image distance.

Definition 2 (Image similarity metric). Given two images f_1 and f_2, an *image similarity metric* $\mathcal{M}_{\text{image}}(f_1, f_2)$ is a function that returns a real number, in the closed interval [0, 1]. The number $\mathcal{M}_{\text{image}}(f_1, f_2)$ is called the *similarity measure* between f_1 and f_2.

By considering Definitions 1 and 2, we can deduce that, given a normalized image distance $\delta(f_1, f_2)$, the corresponding image similarity metric can be defined as $\mathcal{M}_{\text{image}}(f_1, f_2) = 1 - \delta(f_1, f_2)$.

Several metrics have been proposed in the literature to measure image similarity. Such metrics take into account some physical features of the images such as color [9, 25], texture [17] and shape [17]. Given two images f_1 and f_2, let us present some sample distances.

Example 1 (Color histogram distances). Given an image f, let us consider an n-dimensional vector $H_f = (h_1, h_2, \ldots, h_n)$, where each $H_f[i] = h_i$ represents the number of pixels with color i in the picture f. We first consider the concept of δ_1 *distance* given in [25]. The δ_1-norm distance function is

$$\delta_1(f_1, f_2) = \frac{\sum_{i=1}^{n} \min\left(H_{f_1}[i], H_{f_2}[i]\right)}{\sum_{i=1}^{n} H_{f_1}[i]} \tag{1}$$

Another notion of image distances based on color histograms is the δ_2 *distance* given in [9]. The δ_2-norm distance function is

$$\delta_2(f_1, f_2) = \sum_k \sum_t \left(H_{f_1}[t] - H_{f_2}[t]\right) \times a[k, t] \times \left(H_{f_1}[k] - H_{f_2}[k]\right) \tag{2}$$

$a[k, t]$ being the cross-correlation between histograms bins.

Example 2 (Texture distance). The texture distance δ_{text} [17] is defined as follows

$$\delta_{text}(f_1, f_2) = \frac{\sum_{n=0}^{d-1} \sum_{i=1,2,3} \left|\text{Cov}_{wt}^{n,i}(f_1) - \text{Cov}_{wt}^{n,i}(f_2)\right|}{\min_{n=0,\ldots,d-1;i=1,2,3} \left(\left|\text{Cov}_{wt}^{n,i}(f_1)\right|, \left|\text{Cov}_{wt}^{n,i}(f_2)\right|\right)} \tag{3}$$

$\text{Cov}_{wt}^{n,i}(f_i)$ being the Wavelet Covariance Signature of image f_i, evaluated using the Burt-Adelson wavelet transform.

Example 3 (Shape distance). The shape distance δ_{shape} [17] is defined as follows

$$\delta_{shape}(f_1, f_2) = \sum_i \min_{p_j \in I(p_i)} \left|\left(W_{p_i}^{f_1} - W_{p_j}^{f_1}\right)\right| \tag{4}$$

where $I(p_i) = [p_i - \Delta p, p_i + \Delta p]$, $\{(W_{p_i}^{f_1}, p_i)\}$ and $\{(w_{p_i}^{f_2}, p_i)\}$ being respectively shape descriptors for images f_1 and f_2.

Given any metric $\mathcal{M}_{\text{image}}(f_1, f_2)$, let us give the following definition.

Definition 3 (ξ-Similarity). A frame f_i is ξ-similar (not ξ-similar) to a frame f_j if and only if the similarity measure between f_i and f_j is greater or equal (lower) than the threshold ξ, i.e.

$$f_i \sim_\xi f_j \Leftrightarrow \mathcal{M}_{\text{image}}(f_i, f_j) \geq \xi \quad (f_i \not\sim_\xi f_j \Leftrightarrow \mathcal{M}_{\text{image}}(f_i, f_j) < \xi) \tag{5}$$

Before going any further, let us introduce the basic concepts of *video block* and *video*.

Definition 4 (Video block). A *video block* b is an ordered sequence of k frames, i.e.

$$b = \langle f_1, f_2, \ldots, f_k \rangle, \tag{6}$$

where k is said to be the length of video block b.

Definition 5 (Video). A *video* v is an ordered sequence of video blocks, i.e.

$$v = \langle b_1, b_2, \ldots, b_n \rangle, \tag{7}$$

where n is said to be the length of video v. Let us denote with $\mathcal{V}$ the set of all videos.

For the sake of simplicity, given a video block $b = \langle f_1, f_2, \ldots, f_k \rangle$ and a video $v = \langle b_1, b_2, \ldots, b_n \rangle$, let us suppose that there exist some functions *bLength*, *startFrame* and *endFrame*, that respectively return the number of frames, the first and the last frame of b, and a function *vLength* that returns the number of blocks of v.

As mentioned in the previous section, a shot represents a continuous sequence of frames recorded by a camera. Let us formalize the notion of continuity for a video block.

Definition 6 (ξ-Continuity). Given a video block $b = \langle f_1, f_2, \ldots, f_k \rangle$ the ξ-Continuity predicate is defined as follows

$$\xi\text{-Continuity(b)} = \text{true} \Leftrightarrow f_i \sim_\xi f_{i+1} \forall i \in [1, k-1]$$
$$\xi\text{-Continuity(b)} = \text{false} \Leftrightarrow \exists i \in [1, k-1] \,|\, f_i \not\sim_\xi f(i+1)$$

Let us suppose that there exists some predicate called *isShot* which takes a video block as input and returns true or false. Such predicate must satisfy the following axiom.

$$isShot(b) = true \overset{\Rightarrow}{\not\Leftarrow} \xi\text{-}Continuity(b) = true$$

The implication from left to right is required because a shot is a continuous sequence of frames, while the implication from right to left, as we will better explain in the next

subsection, is not required because a continuous sequence of frame could be a sequence of two shots merged together by a dissolve effect.

We are now able to give the formal definition of the *Video Shot Segmentation* process.

Definition 7 (Video shot segmentation). Given a video $v \in \mathcal{V}$ and an *isShot* predicate, a *Video Shot Segmentation* process *Segm* is an application that transforms v in a new video $Segm(v) \in \mathcal{V}$ constituted by a sequence of video blocks satisfying the *isShot* predicate and not included in other shots, i.e.

$$\text{Segm} : v \rightarrow \text{Segm}(v) = \langle b_1^s, b_2^s, \ldots, b_r^s \rangle, \tag{8}$$

where $r = vLength(Segm(v))$. The following properties must be satisfied

$$\begin{cases} isShot(b_i^s) = true \, \forall i \in [1, r] \\ \forall i \in [1, r] \;\; \nexists b^s \mid isShot(b^s) = true \wedge b_i^s \subset b^s \end{cases} \tag{9}$$

In other words, using an analogy with the minimization of boolean functions, the segmentation process must provide a minimum shot coverage of the video, i.e. a decomposition of the video in the minimum number of shots.

3.2. *Characterization of cuts and dissolves*

In this section we present a mathematical characterization of cuts and dissolves, that will be the starting point for the definition of the segmentation algorithm.

In a cut the last frame of a shot is very dissimilar to the first frame of the next one, so the similarity measure between such frames is always very low. The segmentation process can be based on the assumption that the sequence of frames belonging to the same shot is continuous according Definition 6.

Definition 8 (Cut). Given two consecutive video blocks b_i and b_{i+1}, a cut occurs between blocks b_i and b_{i+1} if and only if

$$endFrame(b_i) \not\sim_{\xi} startFrame(b_{i+1})$$

Harder is the detection of gradual transitions between shots, such as dissolve effects. During a dissolve, since the the difference between consecutive frames is very low, a frame results to be similar to the next one according any metric $\mathcal{M}_{\text{image}}$. Let us distinguish between different kinds of dissolve effects: fade-in, fade-out and cross dissolve. During a fade the visual information gradually appears from a solid color frame (fade-in) or disappear, leaving a solid color frame (fade-out). Cross dissolve is a combination of fade-out and fade-in. In other words, cross dissolve is used to gradually move from a video block b_1 to a video block b_2. The frames of the transition interval are obtained as a weighted sum of frames of blocks b_1 and b_2. As the contribution of frames of b_1 varies from 100% to zero, the contribution

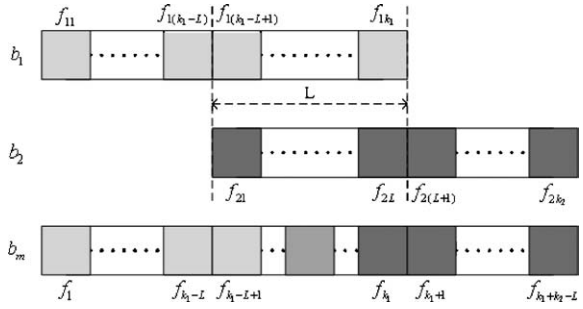

Figure 1. Cross dissolve.

of frames of b_2 varies from zero to 100%. If the frames of b_1 or b_2 are solid color frames, the cross dissolve respectively reduces to a fade-in or a fade-out.

Let us introduce a mathematical model of cross dissolve.

Definition 9 (Cross dissolve). Given 2 video blocks $b_1 = \langle f_{11}, \ldots, f_{1k_1} \rangle$ and $b_2 = \langle f_{21}, \ldots, f_{2k_2} \rangle$, a merge of b_1 and b_2 by means of a cross dissolve of length L is a block $b_m = \langle f_1, f_2, \ldots, f_{k_1+k_2-L} \rangle$, whose frames are defined as follows

$$f_i(p) = \begin{cases} f_{1i}(p) & i \in [1, k_1 - L] \\ \dfrac{(k_1 - i)}{L - 1} f_{1i}(p) + (1 - \dfrac{(k_1 - i)}{L - 1}) f_{2(i-k_1+L)}(p) & i \in (k_1 - L, k_1] \\ f_{2(i-k_1+L)}(p) & i \in (k_1, k_1 + k_2 - L] \end{cases}$$

where $f_i(p)$ is the color of the pixel p of frame f_i. Let us denote *dissolve region* the frame sequence $\langle f_{k_1-L+1}, \ldots, f_{k_1} \rangle$

Figure 1 shows as two video blocks are merged together by means of a dissolve transition. Starting from Definition 9 we can define fade-in and fade-out as particular cases of cross dissolve. A fade-in occurs if $f_{1i}(p) = C \; \forall i \in [1, k_1]$, $\forall p$, while a fade-out occurs if $f_{2i}(p) = C \; \forall i \in [1, k_2], \forall p$, C being a color.

The above mathematical characterization of dissolve confirms our previous considerations about the detection of shot changes in the presence of dissolve transitions, so we need to better characterize a *dissolve region*, to make feasible the detection of dissolve transitions. First of all, let us introduce the following function.

Definition 10 (Similarity function). Given a sequence $\langle f_1, \ldots, f_n \rangle$, the similarity function $f_{\mathcal{M}_{\text{image}}}$ returns, for each index $i \in [1, n-1]$, the similarity measure $\mathcal{M}_{\text{image}}$ between frames f_i and f_{i+1}, i.e.

$$f_{\mathcal{M}_{\text{image}}} : i \in [1, n-1] \rightarrow f_{\mathcal{M}_{\text{image}}}(i) = \mathcal{M}_{\text{image}}(f_i, f_{i+1}) \tag{10}$$

It is clear that the similarity function is strictly dependent on the particular image similarity metric $\mathcal{M}_{\text{image}}$ that we have adopted. However, given Definition 9, we expect that any

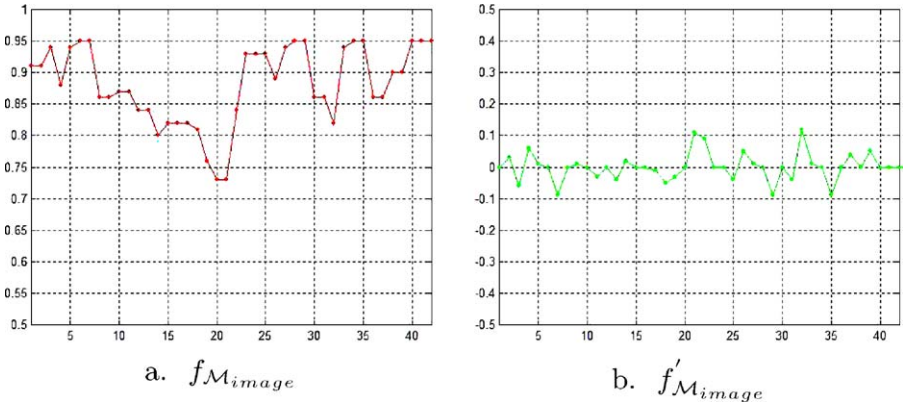

Figure 2. Typical trends of $f_{\mathcal{M}_{\text{image}}}$ and $f'_{\mathcal{M}_{\text{image}}}$.

function $f_{\mathcal{M}_{\text{image}}}$ will present a slow trend in dissolve regions. Our expectation has been confirmed by experimental results, since we have experimentally demonstrated that

$$\left| f_{\mathcal{M}_{\text{image}}}(i) - f_{\mathcal{M}_{\text{image}}}(i+1) \right| < \delta \tag{11}$$

δ being a threshold, and i the index of any frame of a dissolve region.

Furthermore, in figure 2(a) we can observe that, during a dissolve, in a first phase the similarity function decreases very slowly, as suggested by mathematical model, till a minimum point (this behavior is called *fade-out effect*), then it increases so much slowly (*fade-in effect*).

Starting from the above considerations and experimental observations, we can attempt to better characterize dissolve regions considering the first derivative of the time-discrete function $f_{\mathcal{M}_{\text{image}}}$, defined as the divided difference. As a consequence of Eq. (11), the first derivative of $f_{\mathcal{M}_{\text{image}}}$ should be almost constant and close to zero in all the intervals corresponding to dissolve regions. However such feature is not enough to completely characterize a dissolve region. In fact we could have a similar trend of $f'_{\mathcal{M}_{\text{image}}}$ also in a slow zoom.

Another feature necessary to characterize a dissolve region is the not-similarity between the first and last frame of the region. This feature intuitively descends from the fact that a cross dissolve is used to gradually move from a video block to another one, so the first and last frame of the dissolve region, belonging to two different blocks, are presumably quite different. Eventually, let us formalize the property that a video block must satisfy to be considered a dissolve region.

Property 1 (Dissolve region)**.** *A video block* $b = \langle f_1, f_2, \ldots, f_n \rangle$ *is a dissolve region if and only if*

1. $|f'_{\mathcal{M}_{\text{image}}}(i)| \leq \delta \ \forall i \in [1, n-1]$
2. *startFrame*$(b) \not\sim_{\xi}$ *endFrame*(b)

δ *being a threshold.*

4. Segmentation algorithm

In the following we describe an algorithm (Algorithm 1) which, starting from the above definitions, detects both abrupt and gradual transitions, using some similarity metric. The algorithm, as explained in the following, is organized into two phases, in which abrupt and gradual transition are respectively detected.

Let us denote with v the video we wish to segment; with b_i^c the blocks determined by detecting only abrupt transitions; with d the vector in which the frames of a dissolve region are stored for further processing; with N the minimum number of consecutive zeros of $f'_{\mathcal{M}_{\text{image}}}$ that characterizes a dissolve region (to avoid false detections); and with b_i^s the video shots determined at the end of the segmentation process.

In the first phase an initial block-decomposition $v = \langle b_1, \ldots, b_n \rangle$ of the video v is considered. During this phase, the algorithm verify if blocks $\{b_1, \ldots, b_n\}$ satisfy the ξ-*Continuity* predicate. If there exists some block b such that ξ-*Continuity*$(b) =$ *false*, it is disassembled into two or more output blocks satisfying the ξ-*Continuity* predicate. Algorithm 2 is an example of block-disassembly function. The output of this step is a new video $v_{\text{temp}} = \langle b_1^t, \ldots, b_m^t \rangle$, such that ξ-*Continuity*$(b_i^t) =$ *true* $\forall i$. Then the algorithm compares the last frame of each video block with the first frame of the next one. If these two frames are not similar then, according Definition 8, they are marked as a cut, else the corresponding blocks are merged. The output of this step is a new video $v_c = \langle b_1^c, \ldots, b_m^c \rangle$.

In the second phase of the algorithm, the dissolve regions are detected. For each block b_i^c produced by the first phase, the similarity function $f_{\mathcal{M}_{\text{image}}}$ and the first derivative of its median filtered version are evaluated. The frames belonging to a candidate dissolve region are detected by counting the number n of consecutive values of $f'_{\mathcal{M}_{\text{image}}}$, such that $|f'_{\mathcal{M}_{\text{image}}}| \leq \delta$. If $n \geq N$ then the current block contains a candidate dissolve region. If the first frame of this region is not similar to last one, according to Property 1, then it is marked as a dissolve region and stored in the the vector d.

After the dissolve detection the video is reorganized into blocks taking into account the detected dissolve regions. A feasible strategy could be to group the frames belonging to a dissolve region into two blocks corresponding respectively to the intervals before and after the minimum point of $f_{\mathcal{M}_{\text{image}}}$ in such region.

Algorithm 1: Video segmentation algorithm

{First Phase: Abrupt Shot-Change Detection}
  $v = \langle b_1, \ldots\ldots, b_n \rangle$
  {Verify the continuity condition and build the video $v_{\text{temp}} = \langle b_1^t, \ldots\ldots, b_m^t \rangle$}
  for $i = 1, \ldots, n$
    $v_{\text{temp}} = v_{\text{temp}} \cup$ *dissasembly*$(b_i, b_i, bLength(b_i))$
  end for
  {Detect the cuts according to definition 8 *and build the blocks* b_i^c of video v_c}
  $k = 1$
  $b_k^c = b_1^t$
  for $i = 1, \ldots, m - 1$

if $(endFrame(b_i^t) \not\sim_\xi startFrame(b_{i+1}^t))$
$b_{k+1}^c = b_{i+1}^t$
$k++$
else
$b_k^c = b_k^c \cup b_{i+1}^t$
end if
end for
{Second Phase: Gradual Shot-Change Detection}
$f = 0$
for $i = 1, \ldots, vLength(v_c)$
if $(bLength(b_i^c) \geq N)$
{Compute $f_{\mathcal{M}_{\text{image}}}$ on the current block and apply median filtering}
$b_i^c \longrightarrow f_{\mathcal{M}_{\text{image}}}$
$f_{\mathcal{M}_{\text{image}}}^{mf} = Median\ Filtering(f_{\mathcal{M}_{\text{image}}})$
{Compute the $f_{\mathcal{M}_{\text{image}}}^{mf'}$ function}
$f_{\mathcal{M}_{\text{image}}}^{mf'} = diff\,(f_{\mathcal{M}_{\text{image}}}^{mf})$
{Detect the candidate dissolve regions}
$counter[1, \ldots, bLength(b_i^c)] = 0$
$init = 1$
for $k = 1, \ldots, bLength(b_i^c)$
if $(\neg(-\delta \leq f_{\mathcal{M}_{\text{image}}}^{mf'}(i) \leq +\delta))$
$init = k$
else
$counter[init] = counter[init] + 1$
end if
end for
{Build actual cross-dissolve region and store it in d_f}
for $j = 1, \ldots, bLength(b_i^c)$
if $((counter[j] \geq N) \wedge (b_i^c(j) \not\sim_\xi b_i^c(j + counter[j] - 1)))$
$f = f + 1$
$d_f = [b_i^c(j), \ldots, b_i^c(j + counter[j] - 1)]$
end if
end for
end if
end for
{Reorganize the blocks b_i^c taking into account the detected dissolve regions}
$b_i^c, d_f \longrightarrow Segm(v) = \langle b_1^s, \ldots, b_n^s \rangle$

Algorithm 2: Example of disassembly function

global $v_{\text{temp}} = \emptyset$
global $n = 0$
function $disassembly(b_{\text{start}}, b_{\text{temp}}, len)$

if (ξ-*Continuity*(b_{temp}) $= true$ || *bLength*($b_{\text{temp}} \leq 1$)
  $v_{\text{temp}} = v_{\text{temp}} \cup b_{\text{temp}}$
  $n = n + bLength(b_{\text{temp}})$
  if ($n == len$)
    return v_{temp}
  else *disassembly*($b_{\text{start}}, b_{\text{start}} - v_{\text{temp}}, len$)
  end if
else
  $b_{\text{temp}} = [b_{\text{temp}}(1), b_{\text{temp}}(2), \ldots, b_{\text{temp}}(ceil(\frac{bLength(b_{\text{temp}})}{2}))]$
  disassembly($b_{\text{start}}, b_{\text{temp}}, len$)
end if

We can notice that the threshold ξ have a key role in the algorithm. It used to verify the similarity between video frames and can be set in adaptive manner by means of a statistical analysis of the trend of function $f_{\mathcal{M}_{\text{image}}}$ in the presence of cuts dissolve regions [10].

In the following we show a brief example of the application of the algorithm on a video sequence.

Example 4. Let us consider the video $v = \langle b_1, \ldots, b_5 \rangle$ and the following initial decomposition of it (no matter how it has been determined):

$$b_1 = \langle f_1, \ldots, f_4 \rangle, b_2 = \langle f_5, \ldots, f_7 \rangle, b_3 = \langle f_8, \ldots, f_{11} \rangle, b_4 = \langle f_{12}, \ldots, f_{15} \rangle,$$
$$b_5 = \langle f_{16}, \ldots, f_{32} \rangle$$

Let us suppose that there exist two abrupt transitions (one between frames f_4 and f_5 and the other one between frame f_{10} and f_{11}) and a cross dissolve effect involving the frames from f_{11} to f_{32}. By running Algorithm 1, in a first step we obtain the temporary video $v_{\text{temp}} = \langle b_1^t, \ldots, b_6^t \rangle$ composed by the following blocks satisfying the ξ-$Continuity$ predicate:

$$b_1^t = \langle f_1, \ldots, f_4 \rangle, b_2^t = \langle f_5, \ldots, f_7 \rangle, b_3^t = \langle f_8, \ldots, f_{10} \rangle, b_4^t = \langle f_{11} \rangle,$$
$$b_5^t = \langle f_{12}, \ldots, f_{15} \rangle, b_6^t = \langle f_{16}, \ldots, f_{32} \rangle$$

In the second step the remaining abrupt shot changes are detected and the new video $v_c = \langle b_1^c, b_2^c, b_3^c \rangle$ is produced:

$$b_1^c = \langle f_1, \ldots, f_4 \rangle, b_2^c = \langle f_5, \ldots, f_{10} \rangle, b_3^c = \langle f_{11}, \ldots, f_{32} \rangle$$

In the second phase of the algorithm, the cross-dissolve region between frames f_{11} and 32 is detected and the final video shot segmentation $v = \langle b_1^s, b_2^s, b_3^s, b_4^s \rangle$ is produced:

$$b_1^s = \langle f_1, \ldots, f_4 \rangle, b_2^s = \langle f_5, \ldots, f_{10} \rangle, b_3^s = \langle f_{11}, \ldots, f_{21} \rangle, b_4^s = \langle f_{22}, \ldots, f_{32} \rangle$$

5. A sample application

5.1. Video segmentation via animate vision

The described shot detection algorithm relies on the computation of a generic frame similarity metric $\mathcal{M}_{\text{image}}$.

From a general standpoint, shot detection is feasible by assuming that a significant change of visual content occurs across a shot boundary. Human ability to recognize significant changes in the visual content of a video stream is related to the mechanisms of visual attention. The term *attention* means the cognitive functions that are responsible for filtering out unwanted information and bringing to conscious what is relevant for the observer [11]. Visual attention is related to how we view scenes in the real world: moving our eyes (saccade) three to four times per second and integrating information across subsequent fixation points [28]. Saccades are overt shifts of spatial attention that can be either voluntarily performed (top-down) or automatically induced (bottom-up) by salient targets suddenly appearing in the visual periphery and allow us to bring visual targets of interest onto the fovea, the retinal region of highest spatial resolution. An example of foveation sequence is shown in figure 3.

In other words, eye movements, though being characterized by some degree of randomness, are likely to occur along a specific path, the *scanpath* [21], so as to focus areas that are deemed to be important. During a scan we have a rich visual experience from which we abstract the meaning or gist of a scene. During next scan, if the gist is the same our perceptual system assumes that the details are the same.

In this framework, it is reasonable that, in the presence of similar visual configurations, the observer should consistently fix attention to similar regions of interest and follow a similar scanpath. Thus, if a measure of attention consistency $\mathcal{M}$ is defined, when the gist of the observed world undergoes a significant change, $\mathcal{M}$ should decrease down to a minimum value. Moreover during a gradual transitions, $\mathcal{M}$ should exhibit a behavior similar to that experienced in change blindness experiments, where subjects fail to detect a slow spatio-chromatic editing of a sequence presenting the same image. In this case the attention consistency function should be approximately constant across the interval, but minimum

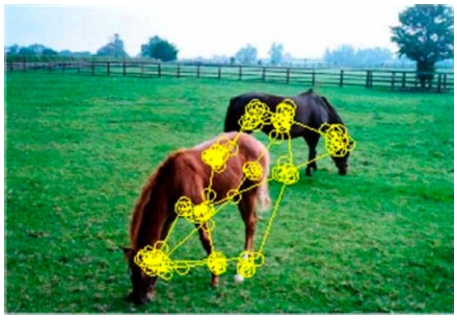

Figure 3. Scanpaths eye-tracked from a human observer. Circles and lines respectively represent fixations and trace displacements (saccades) between fixations.

when measured on a subsequence only formed by the first and the last frame of the interval (according to Definition 9).

For the above theoretical reasons in this paper we have chosen a similarity metric based on the concepts of the *Animate Vision* theory (see appendix 6 for further details). We can state that this image similarity metric satisfy all the properties asserted in Definition 1, but we omit demonstration for brevity's sake. Our choice was also driven by computational reasons: the similarity metric deriving from such theory compares not the entire images, but only a few significant portions of them and the comparison combines the advantages of all the classical metrics of Examples 1, 2, 3 and some spatial and temporal information.

5.2. Experimental results

To evaluate the performances of the video segmentation algorithm using the above mentioned metric, a certain number of videos from a database of movie clips and documentaries have been segmented. Videos were captured at a rate of 30 frames/sec, at a pixel resolution of 640 × 480. For each sequence the ground-truth information was obtained by three experienced humans, using frame-by-frame inspection [8]. The database represents a total of 630 cuts and 90 cross dissolves in approximately 50 minutes of video stream. The data set is described in Table 1.

The comparison between algorithm's output and the ground truth is based on the number of not detected shot-changes *MD* (Missed Detections) and the number of wrongly detected shot-changes *FA* (False Alarms), expressed as *recall* and *precision* [8]:

$$recall = RD/(RD + MD) \quad precision = RD/(RD + FA)$$

where *RD* is the number of right detected shots. In other words *recall* is the ratio between the number of rightly detected shot changes and the total number of changes in a video, while *precision* is the ratio between the number of rightly detected shot changes and the total number of changes detected by the algorithm. Since we want to detect two types of shot transitions, cuts and dissolves, the critical parameters of the algorithm are the similarity thresholds ξ_c and ξ_d.

Table 1. Data set.

Sequence	Duration (sec.)	No. of cuts	No. of cross dissolves
Desert storm war operation	300	80	10
Moulin rouge	700	200	10
Matrix	600	170	0
The patriot	500	80	40
The life is beautiful	600	100	30
Total	45 min	630	90

Table 2. Cut detection performance on testing set.

Threshold	Detected	Correct	Recall	Precision
0.40	68	62	0.098	0.911
0.50	237	188	0.298	0.792
0.60	642	437	0.693	0.681
0.70	1495	592	0.941	0.316
0.80	21379	620	0.984	0.029
0.90	56909	626	0.993	0.011
1.00	80999	630	1.000	0.007

For evaluating the performances of cut detection, *recall* and *precision* have been evaluated varying the similarity threshold ξ_c, while in the case of cross dissolves, the threshold ξ_d has been dynamically obtained. Table 2 shows the cut detection performances of the algorithm. In figures 4(a) and (b) *recall* and *precision* behaviors are reported.

As described in [8], a *recall* value of 0.90 probably is enough for a video indexing system. This assumption allows to evaluate the optimal value of similarity threshold ξ_c for cuts detection. On the basis of Table 2, the optimal threshold value results 0.66, corresponding to a *precision* rate of 0.50.

In the case of cross dissolve detection, the threshold ξ_d is dynamically calculated. Note that an high value of ξ_d would increase the false alarms, while a low value increases missed detections. ξ_d is determined on the basis of the minimum and maximum values of the $f_{\mathcal{M}_{\text{image}}}$ function in a candidate dissolve region. Denoting with $\hat{m}(i)$ and $\hat{M}(i)$ respectively the minimum and maximum of $f_{\mathcal{M}_{\text{image}}}$ in a candidate dissolve region i, the dynamic threshold $\xi_d(i)$ is evaluated as:

$$\xi_d(i) = \frac{\hat{m}(i) + \hat{M}(i)}{2} \tag{12}$$

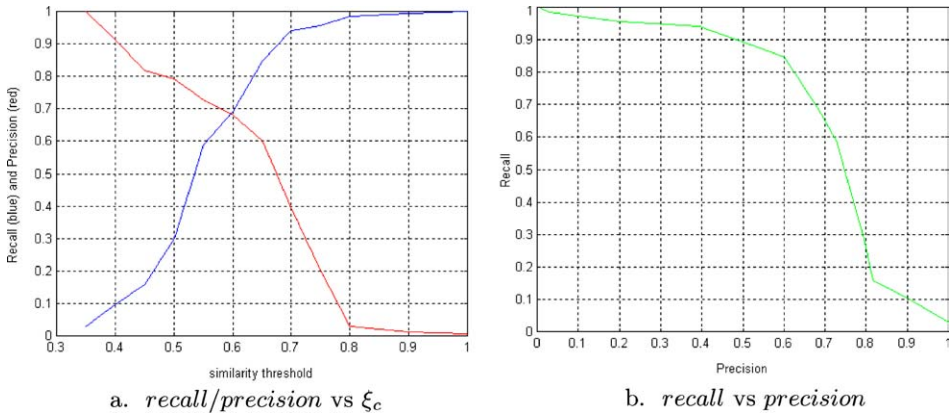

Figure 4. (a) *recall/precision* vs. ξ_c and (b) *recall* vs. *precision*.

Table 3. Dissolve detection performances.

Video sequence	No. of cross dissolves	No. of detections	MD	FA
Desert storm war operation	10	12	0	2
The patriot	40	43	2	5
Moulin Rouge	10	11	0	1
The Life is beautiful	30	35	1	6

Detection results for movies having cross dissolves are listed in Table 3 (for $\delta = 0.185$ and $N = 15$). Summing up, a 97% average recall rate and an average precision rate greater than 85% has been observed.

The video segmentation process using the above described *IP* matching algorithm has lower performances in the case of video sequences characterized by some strong motion phenomenons. In such situations the visual attention moves itself on the moving object giving rise to possible false alarms in the cuts detection process. To increase the algorithm precision some motion detection methodology should be applied.

6. Final remarks

In this article we have presented a formal model for the video segmentation process and a video segmentation algorithm capable to detect both abrupt and gradual transitions.

Experiments have been carried on, adopting a particular image similarity metric. Experimental results confirm the reliability of our approach and encourage to go on in this direction. Further research can be conduced with the aim of making the algorithm robust with respect to more types of transitions. A still unsolved problem is the presence of rapid objects or camera movements, that could be misinterpreted as cuts or transition effects, so further research efforts can be addressed to extract motion information from video sequences in order to improve the performances of the segmentation algorithm.

Appendix A: Animate vision: Information path matching

In most visual biological systems, only a small fraction of the information registered at any given time reaches levels of processing that directly influences behavior. In other words the points of an image have not the same importance: human eye attention is captured only from certain points, called *saliency points* [2]. *Animate Vision* [2] is the visual biological system capacity of quickly detecting interesting regions of visual stimulus.

The *Itti-Koch* algorithm [11], based on *Koch-Ullman* model [12, 13], allows the selection and extraction of image saliency points on the base of their luminosity, color and orientation information. These features are extracted using a computational strategy based on a *center-surround* elaboration [4]. At the end of the extraction phase three *Conspicuity Maps*, one for each feature, are derived. These maps are then merged into a unique *Saliency Map* and processed by the *WTA (Winner Takes All)* neural network [26], which determines the winner

points corresponding to the image saliency points. An inhibition mechanism avoids that a winner point is considered in the next steps.

The regions which surround the saliency points are also called *FOA (Focus Of Attention)*. We have considered 10 *FOAs* for each image, because this number has been proved to be enough to completely characterize an image and because of the bottom-up importance of earliest *FOAs* [22]. An image can be represented through its *FOA* sequence, also called *scanpath* [21]. Given the scanpaths of two images, they can be compared in order to provide a similarity measure of the two images. To this end, each scanpath undergoes further processing, so that relevant information/features are derived. Such features are of two kinds: static and dynamic. The static information are based on the classic physical features of image *FOAs* (color, texture and shape) and on their spatial position; the dynamic feature are based on the time that human eye spends, for each *FOA*, to detect the saliency point [24].

We denote the *flow* of such features, *Information-Path*(*IP*). An *IP* can be seen as a dynamic *signature* of the image. The attentive selection process of the human eye on two similar images will generate similar *IPs* and the image-matching problem can be reduced to an *IP* matching problem [27].

The *IP* must represent a compact characterization of an image, so we have introduced the following syntectic descriptors of the image features: a color descriptor (based on the *HSV* color histograms [25]) and a texture-shape descriptor (based on the *Wavelet Transform* [17]) both evaluated for each *FOAs*, a spatial descriptor based on the coordinates of each *FOAs* and a temporal descriptor based on the firing times of the *WTA* [24]. In particular the *FOA* similarity measures in terms of color, texture and shape are obtained respectively using the normalized metrics in the definitions of Examples 1, 2 and 3, Euclidean metrics are used to define the difference in terms of spatial position and firing time between two *FOAs*. Overall *FOA* similarity is given by a weighted mean of all these terms. Only in the case of two identical images, the *FOAs extraction process* gives the same results and the matching returns the maximum value, since the selection of saliency points depends on the entire image.

In [1] we proposed a matching algorithm between two *IPs*, the *Information-Path Matching Algorithm*, that takes into account all the above mentioned features. Given a generic *FOA* i of the first image, it is compared with the *homologous* one of the second image. The *homologous FOA* i of the second image is selected among those belonging to the local window $[i-W, \ldots, i-1, i, i+1, \ldots, i+W]$. The choice is performed by the computation of static and dynamic *FOAs* features and choosing the *FOA* that maximizes the described *FOAs* similarity. Algorithm 3 is a short pseudo-code description of the *IPM* algorithm.

Algorithm 3: Information path matching algorithm

Compute the *IPs* of the two frames
$\mathcal{M} = 0$
for $(i = 1, \ldots, 10)$
    Select *FOA* i from the first frame
    Select *homologous FOA* i from the second frame using a *best fit* strategy
    Compute the color similarity measure $\mathcal{C}(f_i^1, f_i^2)$ between i-*FOAs*

Compute the texture similarity measure $\mathcal{T}(f_i^1, f_i^2)$ between *i-FOAs*
Compute the shape similarity measure $\mathcal{S}(f_i^1, f_i^2)$ between *i-FOAs*
Compute the spatial similarity measure $\mathcal{D}(f_i^1, f_i^2)$ between *i-FOAs*
Compute the temporal similarity measure $\tau(f_i^1, f_i^2)$ between *i-FOAs*
if (movements toward next *FOAs* not in the same direction)
Penalization of spatial measure
end if
$\mathcal{M} = \mathcal{M} + \mathcal{C}(f_i^1, f_i^2) + \mathcal{T}(f_i^1, f_i^2) + \mathcal{S}(f_i^1, f_i^2) + \mathcal{D}(f_i^1, f_i^2) + \tau(f_i^1, f_i^2)$
end for
$\mathcal{M} = \mathcal{M}/i$

The application of the above algorithm on a couple of images returns a real number in the interval [0, 1]. We can assume the similarity measure returned by this algorithm as the image similarity metric $\mathcal{M}_{\text{image}}$ to be used in the segmentation algorithm.

References

1. M. Albanese, G. Boccignone, V. Moscato, and A. Picariello, "Image similarity based on animate vision: Information-path matching," in Proc. 8th Workshop Multimedia Information System, Tempe, AZ, USA, Nov. 2002, pp. 66–75.
2. D. Ballard, "Animate vision," Artificial intelligence, No. 48, pp. 57–86, 1991.
3. D.R. Bull, W.A.C. Fernando, and C.N. Canagarajah, "Fade and dissolve detection detection in uncompressed and compressed video sequences," IEEE Int. Conf. on Image Processing, 1999, pp. 299–303.
4. P.J. Burt and E.H. Adelson, "The Laplacian pyramid as a compact image code," IEEE Trans. on Communication, Vol. 9, pp. 532–540, 1983.
5. W.A.C. Fernando, C.N. Canagarajah, and D.R. Bull, "Sudden scene change detection in MPEG-2 video sequences," in Proc. IEEE International Workshop on Multimedia Signal Processing, Copenhagen, Denmark, Sep. 1999, pp. 259–264.
6. W.A.C. Fernando, C.N. Canagarajah, and D.R. Bull, "Video segmentation and classification for content based storage and retrieval using motion vectors," in Proc. SPIE Conference on Storage and Retrieval for Image and Video Databases VII, San Jose, CA, USA, Jan. 1999, pp. 687–698.
7. W.A.C. Fernando, C.N. Canagarajah, and D.R. Bull, "A unified approach to scene change detection in uncompressed and compressed video," IEEE Trans. on Consumer Electronics, Vol. 46, No. 3, Aug. 2000.
8. U. Gargi, R. Kasturi, and S.H. Strayer, "Performance characterization of video-shot change detection methods," IEEE Trans. on Circ. Sys. for Video Tech., Vol. 10, No. 1, pp. 1–13, 2000.
9. J. Hafnerm, H. Sawhney, W. Equitz, M. Flickner, and W. Niblack, "Efficient color histogram indexing for quadratic form distance functions," IEEE Trans. on Pattern Analisys and Machine Intell., Vol. 17, No. 7, pp. 729–736, 1995.
10. A. Hanjalic, "Shot-boundary detection: Unraveled and resolved," IEEE Trans. on Circ. Sys. for Video Tech., Vol. 12, pp. 90–105, Feb. 2002.
11. L. Itti and C. Koch, "Computational modelling of visual attention," Natire Reviews-Neuroscience, Vol. 2, pp. 1–11, 2001.
12. L. Itti, C. Koch, and E. Niebur, "A model of saliency based visual attention for rapid scene analysis," IEEE Trans. on PAMI, Vol. 20, pp. 1254–1259, 1998.
13. C. Koch and S. Ullman, "Shifts in selective visual attention: towards the underlying neural circuitry," Hum Neurobiol, No. 4, pp. 219–227, 1985.

14. Z.N. Li and J. Wei, “Spatio-temporal joint probability images for video segmentation,” in Proc. IEEE Int. Conf. on Image Processing, Vancouver, BC, Canada, Sep. 2000, pp. 295–298.
15. H.Y.M. Liao, L.H. Chen, C.W. Su, and H.R. Tyan, “A motion-tolerant dissolve detection algorithm,” in Proc. IEEE Int. Conf. on Multimedia and Expo, Lausanne, Switzerland, Aug. 2002, pp. 225–228.
16. R. Lienhart, “Reliable dissolve detection,” in Proc. SPIE: Storage and Retrieval for Media Databases, 2001, pp. 219–230.
17. S. Mallat, A wavelet Tour of Signal Processing, Academic Press, NY, 1998.
18. J. Meng, Y. Juan, and S.F. Chang, “Scene change detection in an MPEG compressed video sequence,” SPIE, Alg. and Tech., Vol. 2419, Feb. 1995.
19. A. Nagasaka and Y. Tanaka, “Automatic video indexing and full-video search for object appearence,” Visual Database Systems, Vol. II, pp. 113–127, 1992.
20. J. Nam and A.H. Tewfik, “Dissolove transition detection using B-splines interpolation,” IEEE Int. Conf. on Multimedia and Expo, July 2000.
21. D. Noton and L. Stark, “Scanpaths in the saccadice eye movements during pattern perception,” Vision Research, No. 11, pp. 929–942, 1990.
22. D. Parkhurst, K. Law, and E. Niebur, “Modeling the role of salience in the allocation of overt visual attention,” Visual Research, Vol. 42, pp. 107–123, 2002.
23. M. Philips and W. Wolf, “A multi-attribute shot segmentation algorithm for video programs,” Telecomm. Sys., No. 9, pp. 393–402, 1998.
24. R.P.N. Rao and D.H. Ballard, “Dynamic model of visual recognition predicts neural response properties in the visual cortex,” Neur. Comp., Vol. 9, 1997.
25. M.J. Swain and D.H. Ballard, “Color indexing,” Int. J. of Computer Vision, Vol. 7, No. 1, pp. 11–32, 1991.
26. J.K. Tsotsos, S.M. Culhane, W.Y.K. Wai, Y.H. Lai, N. Davis, and F. Nuflo, “Modeling visual-attention via selective tuning,” Artif. Intell., Vol. 78, pp. 507–545, 1995.
27. G.J. Walker-Smith, A.G. Gale, and J.M. Findlay, “Eye movement strategies involved in face perception,” Perception, Vol. 6, pp. 313–326, 1997.
28. A.L. Yarbus, Eye Movements and Vision, Plenum Press: New York, USA, 1967.
29. B.L. Yeo and B. Liu, “Rapid scene analysis on compressed video,” IEEE Trans. on Circ. Sys. for Video Tech., Vol. 5, No. 6, pp. 533–544, 1995.
30. R. Zabih, J. Miller, and K. Mai, “A feature based algorithm for detecting and classifying scene breaks,” in Proc. of ACM Multimedia ’95, 1995, pp. 189–200.
31. H.J. Zhang, “Automatic partitioning of full-motion video,” ACM/Spinger Multimedia Systems, Vol. 1, No. 1, pp. 10–28, 1993.
32. H.J. Zhang, L.Y. Ghong, and S.W. Smoliar, “Video parsing using compressed data,” in Proc. IS&T/SPIE Conf. on Image and Video Processing II, 1994.

Massimiliano Albanese received the Laurea degree in Computer Science and Engineering from the University of Napoli, Italy, in 2002. In 2002 he joined the Dipartimento di Informatica e Sistemistica, University of Napoli “Federico II”, where he’s currently a Ph.D. student in Computer Science and Engineering. His current

research interests lie in Knowledge Extraction and Management, Multimedia Integration and image and video databases.

Angelo Chianese received the Laurea degree in Electronics Engineering from the University of Napoli, Italy, in 1980. From 1984, he joined the Dipartimento di Informatica e Sistemistica, University of Napoli "Federico II", as an Assistant Professor. Since 1992 to present he is *Associate Professor* of Computer Science and Engineering. Since 2003 he leads the e-Learning Laboratory of the University of Napoli. He has been active in the field of Pattern Recognition, Optical Character Recognition, Medical Image Processing and Object-Oriented models for image processing. His current research interests lie in Multimedia Data Base and Multimedia Content Management for e-learning. He is a member of the IAPR.

Vincenzo Moscato received the Laurea degree in Computer Science and Engineering from the University of Napoli, Italy, in 2002. He is currently a Ph.D. student in Computer Science and Engineering at he Dipartimento di Informatica e Sistemistica, University of Napoli "Federico II". His research interests lie in the area of image processing (active vision) and multimedia database systems (image databases, video databases and architectures for multimedia data sources integration).

Lucio Sansone received a Laurea degree in Electronic Engineering in 1965 from the University of Naples "Federico II". From 1976 to 1980 he has been chair of "Centro di Studio sui Calcolatori Ibridi" (Hybrid Computer Research Center, The National Research Council of Italy, C.N.R.) and he has been Project Leader of C.N.R.

research project “Information Systems and Parallel Computing”. He joined the Department of Computer Science and Systems of the University of Naples (Italy) in 1965. Since 1980 to present he is a *Full Professor* of Information Systems. Lucio Sansone is the author/coauthor of more than 80 reseaerch papers in the field of Information Systems and he has been active in the field of distributed object systems, information retrieval, multimedia information systems and general techniques for the management of multimedia data, image analysis and description, object-oriented models for image processing. His current research interests lie in image and video databases.