Multimed Tools Appl
DOI 10.1007/s11042-011-0948-1

SHIATSU: tagging and retrieving videos without worries

Ilaria Bartolini · Marco Patella · Corrado Romani

Abstract The dramatic growth of video content over modern media channels (such as the Internet and mobile phone platforms) directs the interest of media broadcasters towards the topics of video retrieval and content browsing. Several video retrieval systems benefit from the use of semantic indexing based on content, since it allows an intuitive categorization of videos. However, indexing is usually performed through manual annotation, thus introducing potential problems such as ambiguity, lack of information, and non-relevance of index terms. In this paper, we present SHIATSU, a complete system for video retrieval which is based on the (semi-)automatic hierarchical semantic annotation of videos exploiting the analysis of visual content; videos can then be searched by means of attached tags and/or visual features. We experimentally evaluate the performance of SHIATSU on two different real video benchmarks, proving its accuracy and efficiency.

Keywords Content-based video annotation · Video segmentation · Hierarchical semantic video annotation · Visual features

1 Introduction

The recent boom of video net traffic, sparkled by the widespread availability of broadband technology and the emergence of multimedia sharing web portals, such as

This work is partially supported by the CoOPERARE MIUR Project.

I. Bartolini (✉) · M. Patella · C. Romani
DEIS, Alma Mater Studiorum, Università di Bologna, Bologna, Italy
e-mail: i.bartolini@unibo.it

M. Patella
e-mail: marco.patella@unibo.it

C. Romani
e-mail: corrado.romani@unibo.it

Published online: 10 December 2011

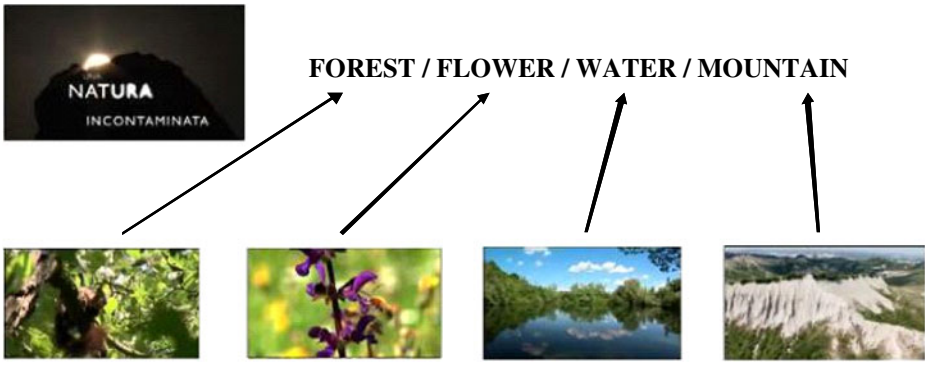

Fig. 1 Tagging videos using textual labels

YouTube (www.youtube.com) and Google Videos (video.google.com), and of pay-per-view services over the Internet, mobile phones, and television introduces a set of new challenges. According to a recent survey,[1] video traffic over the Internet amounts nowadays to over 40% of globally transmitted data, surpassing peer-to-peer traffic for the first time. Besides the obvious technological advances needed to store and transmit such ever-increasing amount of data, the problem of correctly indexing and categorizing video data is particularly important, in order to allow the development of efficient and user-friendly browsing tools. The main challenge is to find an effective way to obtain needed information (such as visual content and genre) from large collections of videos in a (semi-)automatic way, in order to allow a generic user to reliably retrieve videos of interest. The two commonly used approaches, exemplified by the above mentioned systems, exploit user-defined tags (YouTube) or the text surrounding the video (Google). Since it is not always possible to analyze the context where the video is enclosed (which could be missing or simply unrelated to the information contained in the video), many studies focus on video retrieval based on visual content [16]. At present, the most common way to describe the video content is represented by annotation (tagging) techniques, aiming to express such content through a series of textual labels (refer to Fig. 1 for an example). In this way, the semantic concept conveyed by each textual label can be transferred to the video and used later for retrieval purposes. This approach has however some serious drawbacks, preventing its use in several applications: (1) tagging could be incomplete, because some concept contained in a video may be missing from the description of such video (e.g., since the user who tagged that video was not interested in that particular concept); (2) the existence of synonymy makes it hard to retrieve videos relevant for a given concept, because they might have been labeled with synonymous labels; (3) on the other hand, the existence of homonymy/polysemy does not allow to distinguish between the different meanings a single label may have. A common approach to avoid the above problems is to have videos indexed by experts, who are allowed to tag resources by only using a controlled set of labels,

[1] http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481360.pdf, retrieved on Nov. 17, 2011.

with the understanding that no synonymy and homonymy/polysemy are included in the vocabulary. Such solution, which is commonly used by media producers, however puts a burdensome load on experts, since a lot of effort is required to manually index large data collections.

To overcome the previous limitations, the approach we pursue here is the (semi-)automatic tagging of videos using concepts drawn from multidimensional taxonomies. Bounding labels to be part of a common knowledge (represented by taxonomies) alleviates problems of ambiguity and synonymy, while the exploitation of automatic techniques provides the possibility of labeling large collections of videos in a complete way. Towards this end, we apply image tagging techniques on video frames to generate an appropriate description of the content of video scenes [12, 34]. Existing approaches for video tagging [14, 27, 42] are usually able to achieve satisfying levels of accuracy only for specific domains or for short video sequences. Indeed, videos included in large general purpose collections usually are several minutes long and have a heterogeneous content, making it hard to automatically derive significant labels for such videos. For example, a news broadcasting program contains several individual "stories", each focusing on a particular subject: such stories would be different among each other and also with respect to the intermediate sequences depicting the anchorman.

Our approach to solve the above problem is to first divide a video into sequences of visually coherent frames: each sequence, being homogeneous in content, could then be automatically labeled using the previously described techniques. Finally, tags associated with frames sequences can be, in a way, propagated to the whole video, thus obtaining a hierarchical technique for video tagging.

The issue of segmenting a video into scenes sharing the same visual features, usually called shot detection, has been an important topic of video retrieval research throughout the last 20 years [41]. This goal is commonly achieved by detecting hard cuts and gradual transitions (such as fade-in/fade-out, wipe, and dissolve), a task which involves the computation of visual differences between consecutive frames and the subsequent application of criteria to declare the presence of a shot cut (see Fig. 2 for an example).

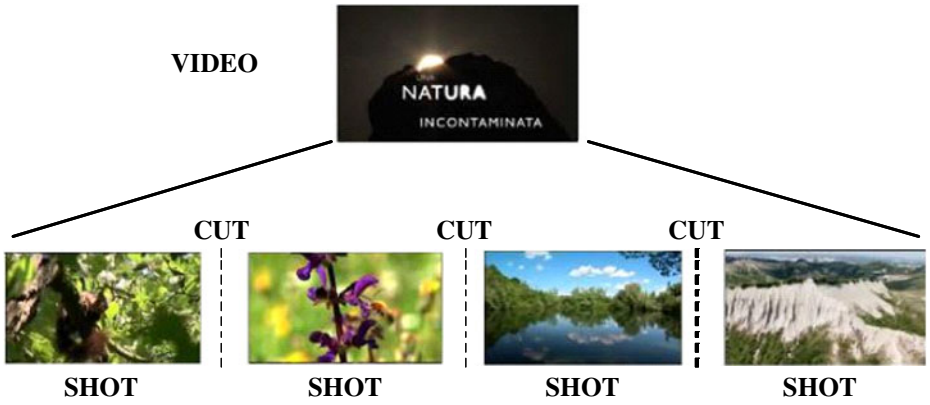

Fig. 2 Segmenting a video in shots, i.e., visually coherent sequences of frames

In this context, we note that segmenting a video into a sequence of frames for video tagging purposes is inherently different with respect to other video segmentation tasks. In fact, some scenarios, like video editing, need a great amount of accuracy in detecting the precise timing of a transition. On the other hand, automatic tagging does not require an extremely precise detection of shot cuts, since we basically want to identify the frames carrying the visual "content" of a sequence. A quicker, but slightly less accurate, shot detection algorithm could be then preferred over a very precise, yet slower, one. Our solution to the problem of shot detection offers a balanced trade-off, giving very fast, yet fairly accurate, results.

In this paper, we present SHIATSU (Semantic HIerarchical Automatic Tagging of videos by Segmentation Using cuts), a complete video processing/retrieval system which offers an accurate description of video content by means of textual labels.[2] Contributions of this manuscript are the following:

1. we present the overall integrated architecture of SHIATSU (Section 2), including a description of the multidimensional taxonomies that convey the semantics of video annotations;
2. we detail the process of video annotation (Section 3), including the tasks of shot detection, shot tagging, and hierarchical video tagging;
3. we introduce the functionality of video retrieval based on high-level concepts and/or visual features (Section 4);
4. we provide experiments (based on two video benchmarks) to evaluate performance of shot detection, shot tagging, and shot/video retrieval facilities and to asses accuracy of our hierarchical approach to video tagging (Section 5);
5. after reviewing the literature on video labeling and retrieval (Section 6), we discuss future improvements and real use scenarios for our prototype system (Section 7).

2 SHIATSU architecture and principles

SHIATSU offers two different use modalities: when a video is first submitted to the system, it can be (semi-)automatically annotated by exploiting semantic tags extracted from several concept taxonomies; alternatively, the user can search for videos of interest by using different retrieval paradigms, some of which are based on semantic tags. The user is allowed to search for both videos and shots. In the example of Figure 3, the top video would be retrieved as result for the video query "forest", while one of the bottom shots would be returned for the shot query "bee".

The architecture of SHIATSU is composed of three software layers. The top layer consists of the Graphical User Interface of the two main tools available in SHIATSU:

- The Annotation Tool is used to semi-automatically assign textual labels to shots and whole videos.
- The Retrieval Tool is exploited to search for videos/shots of interest for a particular information need.

[2]This article is a substantially extended and revised version of [6], where only the annotation component of SHIATSU was described.

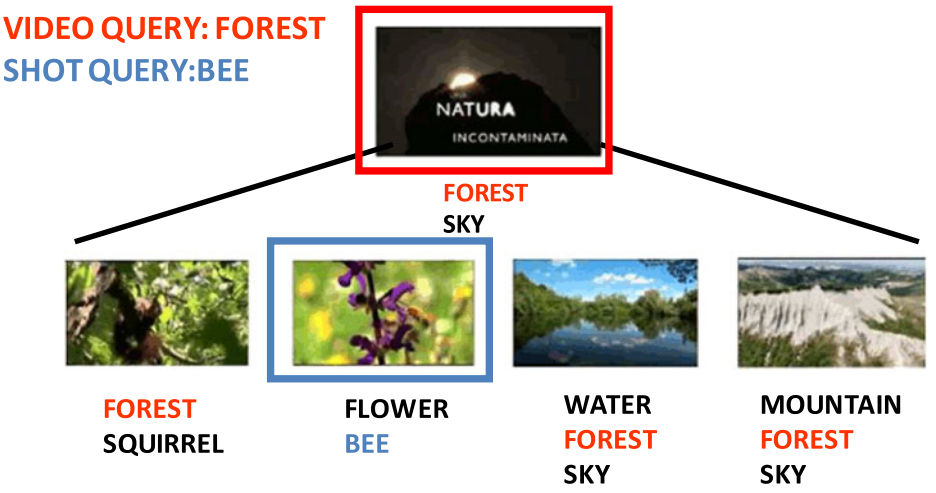

Fig. 3 Retrieval of videos/shots using associated tags

The bottom (data) layer consists of the three databases containing data of interest:

- The Video DB contains raw video data together with other meta-information like shots timestamps, and so on.
- The Feature DB stores the visual features of frames representing shots.
- The Tag DB collects the associations between labels and videos/shots.

Finally, the middle (engine) layer consists of three main components:

- The Visual Features Extractor is used to automatically extract visual features from video frames so as to assess similarity between two given frames.
- The Annotation Processor implements algorithms for the automatic tagging of shots/videos, by exploiting video features.
- The Query Processor contains the logic for retrieval of videos/shots based on semantics (tags) and/or similarity (features).

Figure 4 depicts the interaction between components of the SHIATSU architecture. Solid arrows represent a flow of data between components (e.g., the Visual Features Extractor retrieves data from the Video DB storing results into the Feature DB), while dashed arrows denote that a component requests a service to another one (e.g., the Annotation Processor may ask to the Visual Features Extractor to assess the similarity between two given video frames).

SHIATSU is based on three main principles, which will be discussed in the following: hierarchical annotation, similarity-based labeling, tagging and retrieval based on multidimensional taxonomies.

2.1 Hierarchical annotation

The basic assumption at the core of automatic tagging in SHIATSU is that frames with a similar visual content also convey the same semantic content. According

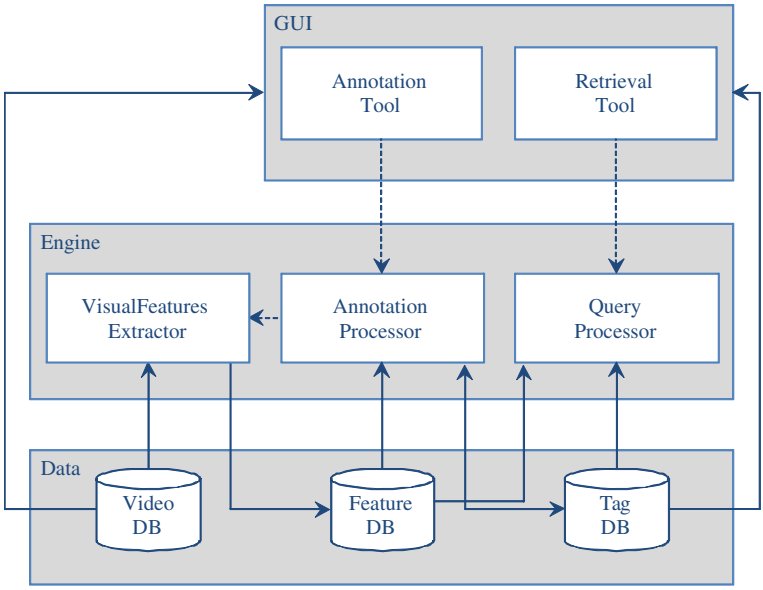

Fig. 4 Architecture of the SHIATSU system

to this principle, tagging of a video in SHIATSU is performed in a hierarchical way:

1. First videos are automatically segmented into shots, i.e., sequences of consecutive frames that share a common visual content.
2. So-obtained shots are then (semi-)automatically labeled using high-level concepts, according to the above mentioned similarity principle.
3. Finally, tags assigned to videos shots are used to appropriately annotate the whole video.

SHIATSU exploits this two-level tagging procedure (see Section 3) so as to provide accurate labels also for long videos. Indeed, the visual content of long sequences could be so diverse to make automatic summarization an extremely hard task. On the other hand, the segmentation process of SHIATSU ensures that shots have a visually coherent content, which can be effectively labeled using automatic tools originally designed for the realm of still images (see below). A real example, drawn from one of the benchmarks used in our experiments, of hierarchical annotation in SHIATSU is depicted in Fig. 5: labels automatically suggested for frames are propagated first at the shot level; then, shot labels are summarized into video tags.

2.2 Similarity-based labeling

As said, labeling of shots (and thus videos) in SHIATSU is based on the principle that objects that share a similar visual content also have the same semantic content, thus similar objects should be tagged using the same labels. This paradigm, that has been proficiently used for the automatic annotation of images [4, 22], is exploited in SHIATSU for automatically suggesting to the user labels at the shot level. For this, the Annotation Processor (see Fig. 4) retrieves the tags assigned to shots containing

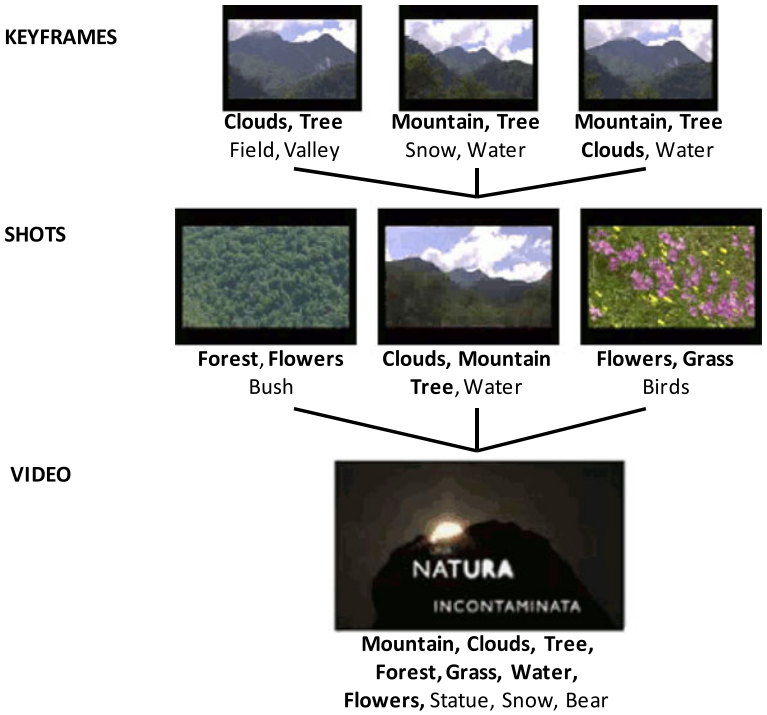

Fig. 5 Hierarchical annotation in SHIATSU: labels suggested for frames are summarized into shot and video tags

frames that are most visually similar to frames contained in the shot currently under examination. This is performed by automatically extracting from frames a number of visual features that are compared to visual features contained in the Feature DB (a metric index allows to efficiently retrieve only the most relevant features for each frame). The whole process of tagging is described in details in Section 3.

2.3 Tagging and retrieval based on multidimensional taxonomies

The typical approach for annotating multimedia objects exploits user-defined textual labels [4, 24, 38]. However, this is commonly performed by drawing tags from a unstructured set, thus not taking into account their intended meaning (i.e., the meaning such tags convey in the context where their associated multimedia data are found: this gives a sort of meaning vagueness to labels [29]). To deal with this, in order to connect each tag with its intended meaning, we exploit the coexistence of multiple, independent classification criteria [15]. According to this "multidimensional" approach, labels belonging to different dimensions may have separate meanings, while each dimension will represent the meaning of high-level concepts contained therein, providing a disambiguation of their semantics.[3] Moreover, each

[3]Our dimensions are therefore quite different with respect to those used in other semantic indexing approaches, like bag-of-words or Latent Semantic Indexing (LSI), where each dimension corresponds to a single, basic concept.

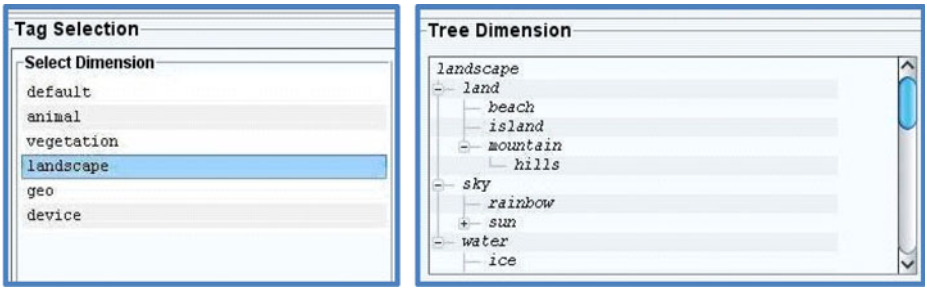

Fig. 6 A 6-dimensional scenario: the "landscape" dimension is selected on the left and (part of) its concept tree is shown on the right-hand side

dimension takes the shape of a tree, where each concept is represented within a taxonomy node and terms are linked with a parent/child relationship. More precisely, each concept is denoted as a *semantic tag*, represented as a path in a tree. To each tree node is therefore associated a single label; the label of the root node corresponds to the name of the dimension itself. This model of hierarchical faceted categories has been successfully used in a variety of applications to provide a coherent and complete description of data [3, 19, 40].

Following the multidimensional approach, SHIATSU is able to describe each video by using labels drawn from several independent dimensions. For instance, Fig. 6 shows some dimensions for a real-world scenario: these include "animal", "landscape", "geographic location", and so on. Each node in a tree path corresponds to a more specialized concept with respect to its parent node, so that moving up/down within a tree a user encounters more/less abstract semantic concepts. This means that if a video is tagged with a given semantic concept t, it is also associated to all ancestors of t. In the example of Fig. 6, the label "landscape/sky/rainbow" also includes the semantic concept "landscape/sky".

SHIATSU allows the user to either use already available dimensions, e.g., (parts of) established ontologies (like those included in Swoogle[4]), or to create her own "custom" dimension. At any time, the user can also add new semantic tags to existing dimensions: every time a new tag is added, it becomes available to be used as a label for a video. We also note that, in line of principle, a same label could appear in several different trees: this allows to distinguish between the different uses and/or meanings that different occurrences of a same label could convey. For example, it is possible that the label "turkey" will appear as a node within both the dimension "geographic location" (e.g., used to describe videos according to the location they were shot) and the dimension "animal", associated to videos about the gallinaceous bird. Extending the example, we could conceive the existence of a "sports" dimension (associated to videos related to sport events), where the same "turkey" label could appear in several places, for example, "sports/soccer/turkey" and "sports/basketball/turkey", representing, respectively, the soccer and basketball Turkish national teams. Clearly, the fact that each semantic tag corresponds to a single complete path within a tree

[4] http://swoogle.umbc.edu/

allows the disambiguation of the different uses of a same label, as illustrated by the previous example.

In order to ensure compatibility with unstructured ("flat") dimensions, such as those used in systems like YouTube or Flickr, SHIATSU also supports 2-level taxonomies, with all tags appearing as children of the single tree root node (in Fig. 6, this is represented by the "default" dimension). This fact allows SHIATSU to also exploit another technique to solve label ambiguity, i.e., label co-occurrence. On the other hand, label co-occurrence is not able, alone, to solve problems of homonymy/polysemy. For example, suppose the user wants to retrieve videos about the animals of the Eurasian country of Turkey: it is quite likely that querying the system using the *flat* concepts "animal" and "turkey" would primarily return videos concerning the gallinaceous bird, due to the polysemy of the term "turkey". The system is therefore not able to satisfy this particular information need of the user by using label co-occurence only.

Each video can be assigned a variable number of semantic tags. If a dimension is not relevant for a video, then no semantic tag from such dimension is used to characterize content. On the other hand, a video could be characterized by multiple semantic tags from the same dimension, if this is appropriate. For instance, a shot containing a dog and a cat might be assigned the two semantic tags "animal/dog" and "animal/cat", both from the "animal" dimension. Thus, although each dimension provides a means to classify videos/shots, this classification is not exclusive at the instance level, a fact that provides the necessary flexibility to organize videos.

Before proceeding with details on the main components of SHIATSU, we would like to highlight here the fact that, in our context, videos are usually medium-long sequences (more than 1 min for each video) containing heterogeneous visual content, so that several high-level concepts could be attached to each video. For tagging, videos are internally segmented into shots (containing keyframes with a same visual content), tags are automatically assigned to shots and are finally propagated to the whole video. Then, such tags could be exploited by the user during the video retrieval phase. As a matter of fact, a user could be as well unaware of the existence of shots. Since SHIATSU models videos as sequences of shots, it offers to the user the *additional* feature of retrieving shots using tags and/or visual features, but this should not be viewed as the main ingredient of the system.

3 Annotation

In this section we detail how the Annotation Processor is able to derive a number of high-level concepts (semantic tags) that can be attached to shots/videos. This is depicted by Fig. 7, where the three main modules included in the Annotation Processor are listed: the Shot Detection Module (in charge of segmenting the video into shots), the Shot Tagging Module (that suggests labels for video shots), and the Video Tagging Module (which performs the hierarchical propagation of shot tags to the whole video).

It is worth noting that, although the process described here is fully automatic, the GUI of SHIATSU allows the user to check proposed tags at both shot and video level, so as to discard tags that were inaccurately suggested (see Fig. 8). The lower right panel in the SHIATSU GUI includes the complete path of suggested tags in

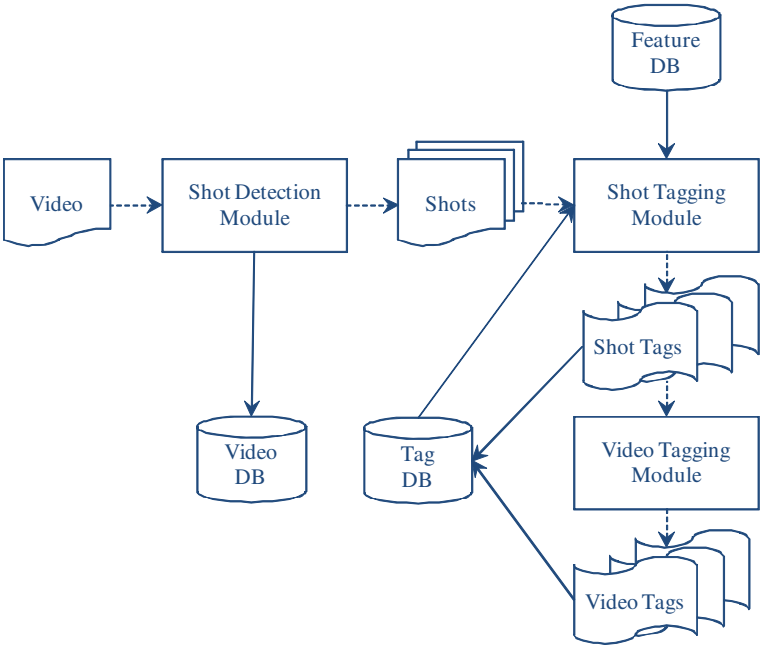

Fig. 7 Labeling of videos by the Annotation Processor

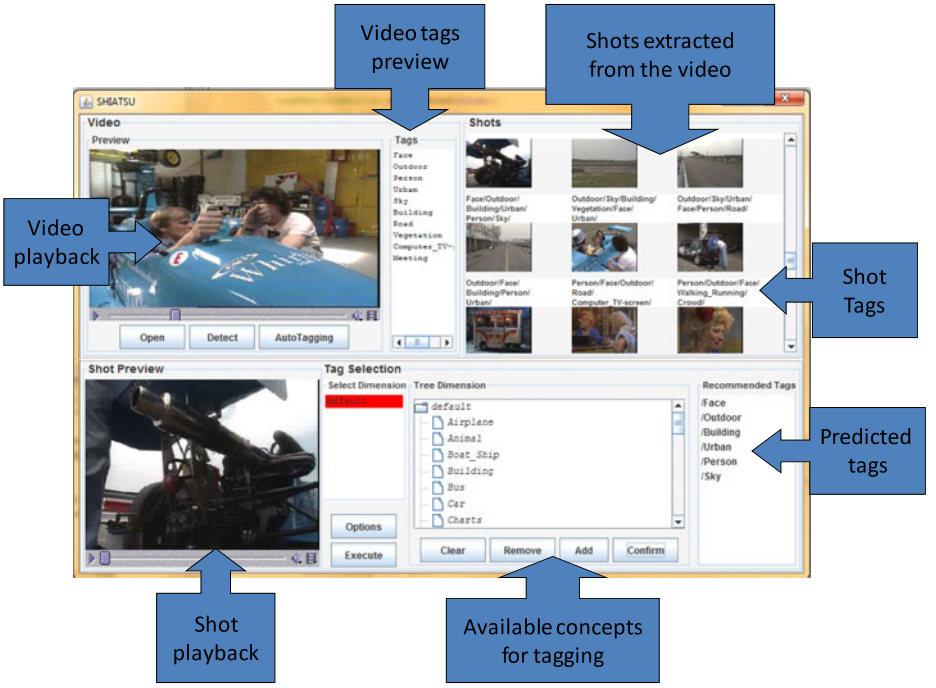

Fig. 8 The Annotation Tool in SHIATSU

the currently selected dimension. In the example of Fig. 8 six tags are suggested: all of them are child nodes of the dimension root node (for ease of display, the name of the root node is excluded from the tag path). When clicking on one of such tags, the dimension panel shows the position of such tags into the taxonomy. Clearly, the GUI also gives the user the possibility of manually inserting new tags, by typing or clicking on them in existing semantic taxonomies.

3.1 Shot detection

The SHIATSU Shot Detection Module exploits color histograms and object edges to compare consecutive frames by applying two different distance metrics: this is because usually a shot transition produces a change in both the color and the texture structure of the frames. A double dynamic threshold system is then used in order to take into account possible dissimilarities in the content of different video types. A frame cut is detected by first using differences in color features and then in edge features. For the shot detection process, we chose a balanced approach in order to mediate between effectiveness, efficiency, and ease of implementation.

3.1.1 HSV color histogram

The color information of each frame is represented using the distribution of HSV values of its pixels. Each frame is characterized using three histograms (we have 12 bins for Hue, 5 for Saturation and 5 for Value). The difference between histograms of two frames h and h' is computed using a L_1 bin-to-bin metric [23]:

$$d_{b2b}(h, h') = \frac{1}{2N} \sum_i |h[i] - h'[i]| \tag{1}$$

where N denotes the number of frame pixels. From (1), the HSV distance between two consecutive frames k and $k+1$, $d_{HSV}(k, k+1)$, is defined as the sum of bin to bin differences for the three HSV histograms:

$$d_{HSV}(k, k+1) = \frac{1}{6N} \sum_i |h_k[i] - h_{k+1}[i]| \tag{2}$$

With respect to [31], where an histogram intersection formula is used to measure frame similarity, (2) allows a better discrimination between cut and non-cut frames. This happens because distance values between non-cut frames generally remain in the same range throughout the whole video: distances computed using (2) will, thus, maintain the same average value in different parts of a video, making it easier to correctly select video cuts.

3.1.2 Edge change ratio

A scene transition usually exhibits a modification in edges of frame objects. We can therefore compute how many entering (new) edges and exiting edges exist between two consecutive frames, so as to detect the occurrence of a shot cut. We define as entering edges those new edges which have appeared with respect to the previous frame and as exiting edges those edges which are present in the actual frame but not in the next frame. We compute edge pixels using a Canny filter [7] and determine entering and exiting edges by analyzing the difference of edge pixels between frames:

edge pixels which are at most three pixels away from edges in the previous/next frame are not counted as changed edges to compensate for object motion into the scene. The Edge Change Ratio (ECR) [21] between frames k and $k+1$ is calculated as follows:

$$ECR(k, k+1) = \max\left(\frac{X_k^{\text{out}}}{\sigma_k}\frac{X_{k+1}^{\text{in}}}{\sigma_{k+1}}\right) \tag{3}$$

where σ_k is the number of edge pixels in frame k, X_k^{out} and X_{k+1}^{in} are the exiting and entering edge pixels in frames k and $k+1$ respectively. In [31] edge density and average gray variation are used to detect object edges: we believe that the use of ECR could help in better defining frame differences and avoiding a large amount of false positive cuts in the result.

3.1.3 Cut selection process

The process of cut selection, i.e., determining the timestamps where a scene cut occurs, can start once all the M frames have been processed. Note that this has an $O(M)$ complexity, since each frame is only compared with the next frame. Distances between consecutive frames are compared with threshold values: whenever a distance exceeds the threshold a cut is declared. The choice of the threshold value is therefore of utter importance: a too small value would produce over-segmentation, while a too high value would result in no cuts. Three alternatives currently exist in literature: fixed thresholds [21, 28, 31, 35, 43], clustering [2, 26], and dynamic thresholds. The use of fixed threshold values generally cannot deal well with videos of different types, while clustering performs poorly when differences calculated around probable cuts are not very high, compared to the mean of differences between non-cut frames. In such case, grouping the frames in two clusters (normal frames and cut frames) could lead to miss a large number of shots (false negatives). The best available option seems therefore to be dynamic thresholding, i.e., the computation of a threshold value based on the content of each processed video. An interesting approach is the one described in [31], where edge features and HSV histograms are used with a double dynamic threshold system. Although the performance exhibited by the system are adequate in most cases, the similarity metrics used in [31] to evaluate HSV histograms and the proposed edge features lead in some cases to an excessive number of false positives (detecting a transition between shots belonging to a same scene). For SHIATSU, we appropriately modified the approach presented in [31] to improve the accuracy of shot detection, while keeping the complexity of the process at reasonable levels.

To determine the HSV threshold θ_{HSV}, SHIATSU computes the mean of the highest (M/f) HSV distances:

$$\theta_{HSV} = \frac{\beta_{HSV}}{M/f}\sum_{i=M-M/f+1}^{M} L_{HSV}(i) \tag{4}$$

where β_{HSV} is a sensitivity parameter (the default value is 1), M is the total number of video frames, f is the video framerate, and L_{HSV} is the ascendingly ordered list of

HSV distances among all consecutive shots. To determine the ECR threshold θ_{ECR}, the average of the highest $2(M/f)$ ECR values is computed:

$$\theta_{ECR} = \frac{\beta_{ECR}}{2M/f} \sum_{i=M-2M/f+1}^{M} L_{ECR}(i) \tag{5}$$

where β_{ECR} is a sensitivity parameter (the default value is 1) and L_{ECR} is the ascendingly ordered list of ECR values. We consider a larger window of data for the computation of θ_{ECR} because ECR distances are usually sparser than HSV distances. Our algorithm filters out all frames having HSV distance $\leq \theta_{HSV}$ and considers ECR values of the remaining candidates: they are again excluded from the result if ECR $\leq \theta_{HSV}$; the frames that exceed both thresholds are considered as shot cut frames. The approach described in [31] uses two different dynamic thresholds to detect abrupt shot cuts and gradual cuts: as we will show in the experimental section, SHIATSU is able to detect both hard and gradual cuts and is also easier to implement.

3.2 Video tagging

For tagging purposes, SHIATSU exploits the Imagination system [4], using a set of pre-annotated images/frames as a knowledge base. Every image/frame is used by the system as an example of the semantic concepts attached to it; in this way, all and only concepts included in the knowledge base could possibly be suggested as relevant for a given video. The system extracts a set of visual features from each image and saves the information in a database, indexing them efficiently with an implementation of the M-Tree metric index [9]. When provided with a frame to be labeled, the tagging module extracts its visual features, exploits the M-Tree index to efficiently retrieve images having similar features and proposes semantic concepts depending on the similarity of the shot with the images in the knowledge base (more details can be found in [4]). Every time a new image/shot is processed and tagged, its information is inserted into the database, hence improving the system accuracy and quality. For simplicity, what follows assumes the existence of a single classification dimension although, as said, SHIATSU supports multiple dimensions; the GUI allows the user to specify, at tagging time, which dimension(s) she is interested in for the particular video at hand.

3.2.1 Shot tagging

For each shot, SHIATSU extracts a set of representative *keyframes*, by using the cuts timestamps. Visual features of such keyframes are then computed and compared with those contained in the knowledge base. We experimented with three different policies for selecting keyframes representative of a shot:

- The first alternative is to select just the first video frame as representative for the whole shot.
- Another possibility is to select three keyframes, namely the first, the middle, and the last frame of the shot.
- The last alternative is to select a number of keyframes N_k depending on the shot length $l(s)$, $N_k(s) = c \cdot l(s)/f$, where c is a constant value.

We experimented with all the three strategies, with results included in the experimental Section 5.2.

Each keyframe is then fed to the Visual Feature Extractor module that extracts its visual features. In details, this module exploits the Windsurf features [1, 5], i.e., each keyframe is first segmented into visually coherent regions, then color/texture features are extracted and stored for each keyframe region (details about the segmentation process and features can be found in [1]). Features of each keyframe are then exploited by the annotation module to retrieve, from the Feature DB, images that are visually similar to the given keyframe; again, this is efficiently carried out by way of an M-tree index built over the Feature DB, thus avoiding a costly sequential scan of the whole DB. Tags associated to retrieved images are then suggested for the given keyframe and this procedure is repeated for all keyframes of a given shot: only terms recurring in the majority of keyframes are selected as suitable concepts to describe the whole shot. To avoid producing an overwhelming number of tags for each shot, only the most frequent tags retrieved for each keyframe in the sequence are presented to the user as relevant tags. The proposed tags can then be reviewed by the user and, if she is satisfied with them, these are finally stored into the Tag DB. Note that the frequency of each tag t in shot s is also stored within the Tag DB, so as to maintain the relevance of t for s; this is exploited both in the tagging of whole videos (as will be shown in the following paragraph) and in the tag-based retrieval of shots (see Section 4).

3.2.2 Hierarchical tagging

Shot tags can be used to browse frame sequences across different videos. However, for video indexing purposes, they could be too specific, particularly in the case of long videos, containing a wide range of different visual content. Propagating tags from shots to videos is an activity of summarization, i.e., the description of the video is a compact sum of the tags associated with its shots. A simple criterion to select video tags from the set of shot tags is to weigh every tag depending on its frequency and the length of the shot it is associated with. We first compute the relevance of each shot s as the length of s with respect to the whole video V:

$$W(s) = \frac{l(s)}{l(V)} \tag{6}$$

Then, we define the rank $R(t)$ for every shot tag t as:

$$R(t) = \frac{1}{N_s} \sum_s W(s) A(t, s) \tag{7}$$

where N_s is the total number of shots and $A(t, s)$ is the relevance of tag t for shot s ($A(t, s) \in [0, 1]$ with 0 meaning that shot s does not contain tag t). Tags are ordered by descending $R(t)$ values and the first 10 tags (if available) become video tags. The rationale behind the proposed propagation method relies on the fact that concepts extracted from long shots and/or that appear in several shots are probably more relevant, to describe the content of a whole video, than concepts occurring rarely or in short sequences. Finally, video tags are stored in the Tag database (together with shot tags), thus the user can exploit them when searching for videos of interest (see Section 4).

For tag propagation, we also considered the use of tf-idf weights in (7). The use of inverse document frequency would exclude from video description tags that are commonly found in the DB, introducing rarer tags, which are however less representative of the content of the video. Although using idf could favor searching videos containing less frequent tags, we note here that such tags are nevertheless associated with shots, thus the search would succeed when looking for shots containing such tags.

4 Retrieval modalities

The Query Processor is the component of SHIATSU that is in charge of managing user requests so as to efficiently return the videos of interest. The user should first choose whether she is interested in whole videos or in just shots. Then, results can be retrieved according to one of three available query paradigms: *keyword*-based (*KS*), *feature*-based (*FS*) and *keyword&feature*-based (*KFS*) searches.

The *KS* paradigm is the easiest and most popular query modality, used by traditional search engines, where the user enters a set of keywords as query semantic concepts. According to this query paradigm, the result list only includes those videos/shots containing at least one tag among those submitted by the user. The list of returned objects is sorted according to the number of tags that *co-occur* in each video/shot. This means that, if q tags are specified in the user query, the first ranked videos/shots would be those (if any) that include all q tags, followed by those that only contain $q-1$ of such tags, and so on. Among videos/shots that contain the same number of tags, results are ranked for decreasing relevance of such tags. Finally, the user could specify a maximum result cardinality, k, so that the ranked list is limited to the top-k results. Due to the very hierarchical nature of semantic tags, a user requesting videos/shots labeled with a given tag t, would also receive videos/shots annotated with any concept descendant of t. For instance, considering the sample dimensions shown in Fig. 6, a *KS* query requesting for the semantic tag "landscape/sky" would also retrieve objects labeled with tags "landscape/sky/rainbow" and "landscape/sky/sun", if any.

With the *FS* modality, the user is looking for those shots whose representative keyframes are similar to an input query image. Since an entire video generally holds a large range of heterogeneous visual content, we believe it is not useful for the user to perform *FS* at the video level. Again, the user could specify a limit on the number of retrieved results, k. It has to be noted, at this stage, that performing a simple *k-nearest-neighbor* (k-NN) search on the Feature DB could lead to miss some relevant results. Indeed, since a shot is usually represented by several keyframes, only retrieving the k keyframes that are most similar to the query image could lead to retrieve less than k distinguished shots. To overcome this problem, we exploit another feature of the underlying M-tree index, retrieving keyframes in decreasing order of similarity to the query image. Such *sorted access* modality is efficiently performed by the index thanks to algorithms of general applicability [20]. Then, a number $k' \geq k$ of sorted accesses is performed until it is guaranteed that the k' retrieved keyframes belong to k different shots.

Finally, *KFS* queries combine the *KS* and *FS* modalities, returning shots in the intersection of both *KS* and *FS* results first, followed by shots in the *KS* list only and, finally, by shots in the *FS* result only. This query modality is particularly convenient

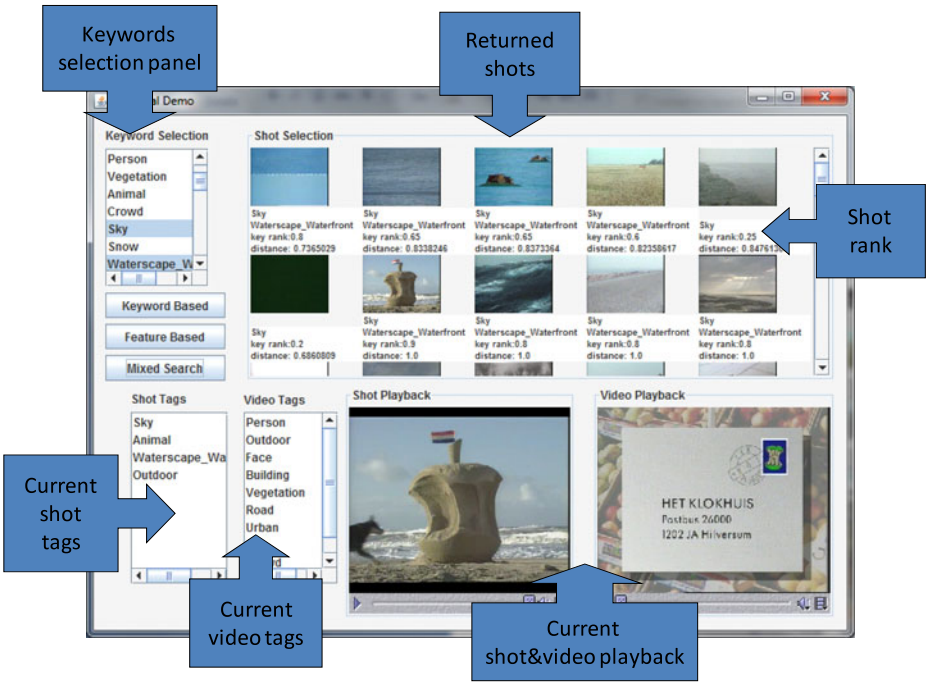

Fig. 9 The Retrieval Tool in SHIATSU

when the keyword-based search involves concepts that are poorly represented in knowledge base, i.e., shots annotated with requested concepts are less than k. In this case, by only applying *KS* we would not be able to return k shots, thus we enrich the result by including also shots that contain keyframes visually similar to the provided query image. Even in the case when *KS* is able alone to provide the desired number of objects, adding features in the query process could still be useful: indeed, it can be exploited to re-rank shots within the result set, e.g., among shots containing all the query tags, those that also contain keyframes visually similar to the query image are listed first. As for *FS*, the use of a query image restricts results to shots (i.e., this query modality cannot be applied to entire videos).

Figure 9 shows the retrieval tool of the SHIATSU GUI. In this example, the result of a *KFS* query concerning the concepts "sky" and "waterscape/waterfront" is depicted. The user has then clicked on the seventh result shot, and the bottom-left part of the GUI is showing tags associated to that shot and the corresponding video. On the bottom-right part of the GUI, previews of the shot and the whole video are shown, so as to allow their playback.

5 Experimental evaluation

In this section, we present results about effectiveness and efficiency of SHIATSU. For this, we used two different real video datasets: the video set for the Item

Segmentation task of the Mammie system[5] and the video set for the High-Level Feature task of the TRECVID-2007 benchmark.[6] The Mammie dataset consists of 43 videos containing a total of almost 2 millions video frames for a playback time of over 1075 min; this dataset also provides a ground truth of 4225 shot cuts, so we used it to test the accuracy of the segmentation module of SHIATSU (Section 5.1). On the other hand, the TRECVID dataset contains 110 videos with about 4.5 millions video frames (resulting in approximately 9000 shot cuts) for a total of around 3000 minutes; each shot in the dataset is described by means of textual tags representing 36 semantic concepts. Such 36 concepts, that were included into a single flat "default" dimension, represent our ground truth for tagging, allowing us to provide non-subjective results on the accuracy of SHIATSU. Experiments on tagging (Section 5.2) and retrieval (Section 5.3) were performed on 7 videos in the Test Set of the TRECVID dataset (about 120 min, 180000 frames, 680 shots) that are tagged using the same 36 labels, so that it is possible to evaluate precision of tagging/retrieval. Since TRECVID only provides a ground truth at the shot level, i.e., no tags are given for full videos, we are not able to give any non-subjective results at the video level. A final experiment (Section 5.4) involves a set of ten real users, who were requested to evaluate the usefulness of video retrieval and the accuracy of our hierarchical approach to video tagging.

Regarding efficiency, SHIATSU was implemented in Java JDK 6.0 and all experiments have been run on a an AMD 3.1 GHz Dual Core processor with 2GB of RAM running the Windows 7 Professional OS; the DBMS used was MySQL Server 5.1.

5.1 Shot detection performance

Performance of the shot detection module of SHIATSU is evaluated using classical precision/recall metrics:

$$P = \frac{CC}{CC + FC} \qquad R = \frac{CC}{CC + MC} \tag{8}$$

where CC denotes the number of cuts correctly detected, MC the number of missed cuts, and FC the number of false (erroneously detected) cuts, respectively. We accept a tolerance of 250 msec from shot boundaries timestamps indicated in the ground truth to declare a correct cut.

We compare results of SHIATSU on the Mammie dataset with those obtained by the technique proposed in [31] (named QLR in the following). Table 1 shows the performance of SHIATSU (using default and optimal sensitivities values) compared with those of the QLR technique.

SHIATSU clearly outperforms the reference algorithm in both recall and precision and achieves good overall values with default and optimal sensitivity values. The use of bin to bin differences for HSV histograms and of ECR for frame object edges dramatically reduce the number of false shot boundaries detected:

[5] http://media.ibbt.be/mammie

[6] http://trecvid.nist.gov

Table 1 Video segmentation results

Parameters	SHIATSU		QLR
	$\beta_{HSV} = 1, \beta_{ECR} = 1$	$\beta_{HSV} = 1.2, \beta_{ECR} = 1$	
CC	3784	3721	3625
FC	672	494	1612
MC	441	504	600
Recall %	89.56	88.07	85.8
Precision %	84.92	88.28	69.22

experimental results prove how our choice of image features leads to significantly better performances. We also evaluated how recall and precision values change when modifying one of the sensitivity parameters β_{HSV} and β_{ECR}, keeping the other at the default value (Fig. 10).

As expected, increasing sensitivity parameters leads to increasing values of precision and decreasing values of recall (since we obtain higher thresholds, thus a lower number of cuts). After tweaking sensitivity parameters, the most balanced setting seems to be $\beta_{HSV} = 1.2$ and $\beta_{ECR} = 1$ where the segmentation algorithm achieves a recall of 88.07% and a precision of 88.28%; we however note that SHIATSU is quite robust in performance since a variation of 20% in sensitivity parameters does not lead to a dramatic drop in neither recall nor precision.

Regarding efficiency of the segmentation task, SHIATSU is quite inexpensive, processing frames at a rate of 15 FPS (note that this is achieved on an average machine, not on a high-end system, where the segmentation task would be even faster).

5.2 Shot tagging performance

We evaluate the accuracy of the annotator in terms of classical *precision* (fraction of suggested tags that are correct for the shot) and *recall* (fraction of relevant tags

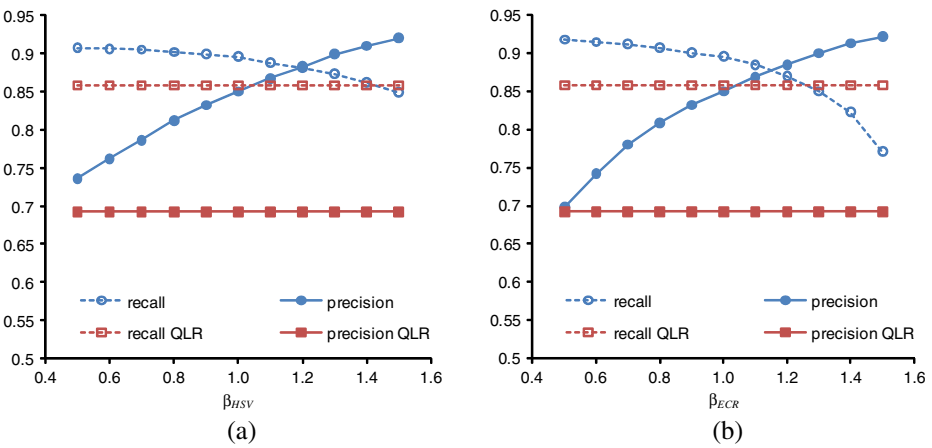

Fig. 10 Precision and recall values when varying (**a**) β_{HSV} and (**b**) β_{ECR}

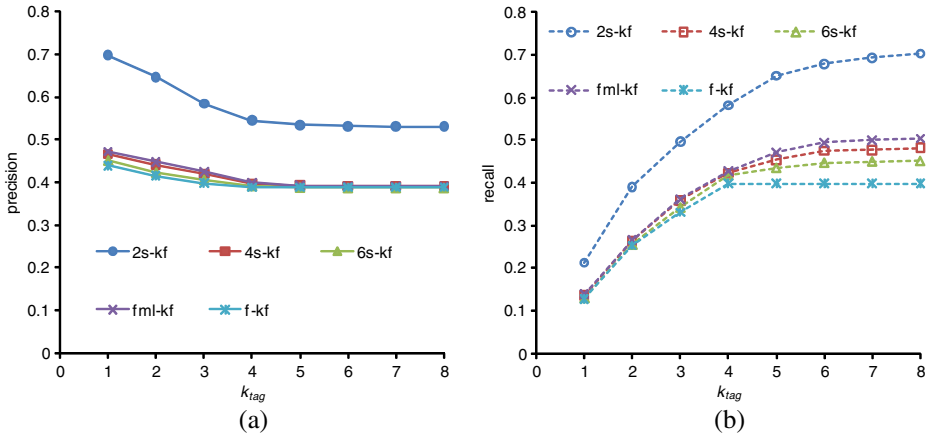

Fig. 11 Average accuracy of the annotator in terms of (**a**) precision and (**b**) recall vs. number of predicted tags

that are suggested by the annotator) metrics over the Test Set of the TRECVID dataset. As noted in Section 3.2.1, we tested three different alternatives for choosing the keyframes that represent a shot. Figure 11 shows the performance of such alternatives: f-kf (first keyframe) denotes the strategy that selects only the first shot frame, fml-kf (first, middle, last keyframe) denotes the strategy that chooses the first, the middle, and the last keyframe, while *c*s-kf denotes the policy that picks a keyframe every *c* sec (e.g., 2s-kf means that a keyframe is selected every 2 sec). For every shot, we request SHIATSU to provide the k_{tag} ($k_{\text{tag}} \in [1, 8]$) most frequent tags, evaluating their precision/recall against the benchmark ground truth. As expected, when requesting for more tags, recall is increased at the expenses of precision, as Fig. 11 shows.

As to efficiency of the tagging task, each keyframe selection alternative has an impact on the time needed to suggest labels for a shot. Since, on average, about 331 msec are required to process a single keyframe and the mean shot length is 10.54 sec, we obtain the average shot tagging times shown in Table 2, which also shows the ratio of processing time to real time, i.e., the average shot length.

Finally, we analyzed the impact of segmentation on tagging accuracy: for this we evaluated how precision/recall measures were affected when varying the sensitivity parameters of our shot detection algorithm (see Section 5.1). Results obtained in this experiment, however, exhibit no statistically relevant difference, with a maximum deviation of around 1%, and are thus not shown here. Again, this should be viewed as a proof of the robustness of both the segmentation and the similarity-based tagging algorithms used in SHIATSU.

Table 2 Average shot tagging time and ratio of tagging time over shot time vs. keyframe selection policy

Keyframe selection policy	2s-kf	4s-kf	6s-kf	fml-kf	f-kf
ms per shot	1744.37	872.18	581.46	993.00	331.00
% of shot time	16.55	8.27	5.52	9.42	3.14

Conclusions of these experiments are the following:

1. The strategy that selects a keyframe every 2 sec is the one that attains the best accuracy, far better than all other alternatives.
2. The same strategy is also the worst when considering tagging costs, five times than the cheapest f-kf alternative.
3. Absolute costs are consistently very low, thus we can easily trade off tagging time for achieving the best accuracy, considering that the processing speed is 6–30× quicker than average video playback.

Figure 12 summarizes precision and recall, when varying the number of predicted tags, for the 2s-kf policy, which is the one that will be used for the remainder of this section. The current implementation of SHIATSU attaches six tags to each shot: we believe that this represents a good trade-off between accuracy (precision is about 55%), recall (over 70%), and usability (a larger number of tags would be hardly manageable by a single user).

Although our experiments on tagging were conducted using a single dimension (due to the fact that there was no correlation among tags available in the ground truth), they would still remain valid also in presence of multiple hierarchies. This is because the result of the tagging process for a given dimension is independent of the co-existence of other dimensions. However, when several hierarchies exist, the user is given much more expressive power. For example, suppose that two different dimensions exist, A and B. The user could ask for tags in both dimensions A and B, or even only in dimension A (or B). Clearly, this freedom is not allowed in the case where such dimensions are included into a new single hierarchy, say C.

5.3 Shot retrieval performance

In order to measure the effectiveness of video retrieval in SHIATSU, we performed tests using a query workload of seven different search labels (namely: *outdoor scenes*, *scenes containing persons*, *faces*, *roads*, *sky views*, *vegetation*, and *waterscape/waterfront views*) drawn for the original 36 concepts in the benchmark. Our first experiment aims at measuring the accuracy of the two main search modalities in

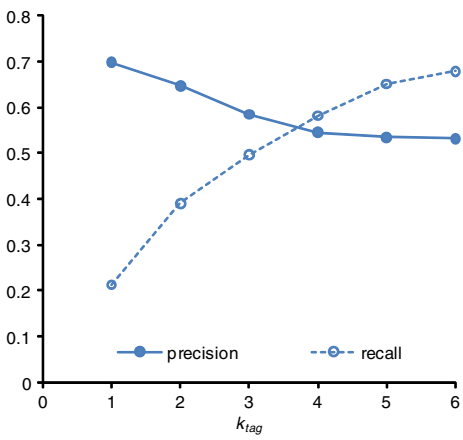

Fig. 12 Average accuracy of the annotator in terms of precision and recall vs. number of predicted tags

SHIATSU: keyword-based and feature-based search. For the former, the retrieval engine was queried using the exact concept, thus performance is expected to reflect the accuracy of the annotator; for the latter, on the other hand, we submitted an image (not drawn from the dataset) exemplifying one of the seven concepts. Retrieval precision is measured as the fraction of returned shots for which the ground truth exhibits the query concept.

Figure 13 (a) shows the average accuracy of the video retrieval vs. the number of returned shots. We recall that for keyword-based (*KS*) searches, the order of returned shots is given by the relevance of the query tag in each shot, while for feature-based search (*FS*) shots are sorted in decreasing order of similarity to the query image. As the graphs show, *KS* performs very well (about 80% of precision for five retrieved shots and 58% for 25 retrieved shots); this again demonstrates the accuracy of the annotation in SHIATSU. On the other hand, performance of *FS* is worse than *KS* (as expected, because of the semantic gap problem), yet it remains on a satisfactory level.

To demonstrate the usefulness of *FS*, we performed a final test assuming a *poor* annotation scenario. For example, in our ground truth the concept "Car" appears in 31 shots, but our annotator is able to only predict it for seven shots (five of which are correct, thus precision is 71%). In this case, recall is clearly limited, reaching $5/31 = 16\%$ and cannot be increased further by *KS*. On the other hand, we can request additional shots using *KFS*. Figure 13 (b) shows that the contribution of the visual features can indeed increase the recall: when 30 shots are retrieved, 12 of them contain cars, giving a recall level of around 40%. The graph also shows the precision of *KFS*: although we have deliberately chosen a situation where the SHIATSU annotator performs not very well, this is never less than 40%, making this a "remarkably good" worst-case scenario.

As to efficiency of the retrieval in SHIATSU, Fig. 14 depicts average retrieval time of *KS* and *FS*. While the latter exploits the M-tree index (see Section 4), the former takes advantage of indexing structures of the underlying DBMS. Indeed, graphs show that retrieval times follow a trend which is linearly dependent on the number of

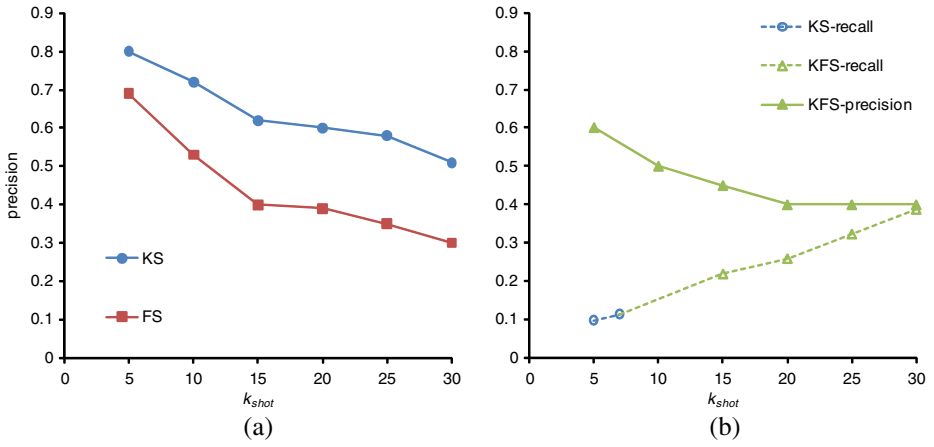

Fig. 13 Average retrieval precision vs. number of returned shots (**a**) and accuracy for the query concept "Car" (**b**)

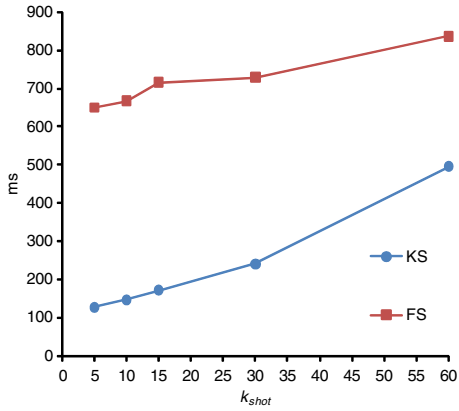

Fig. 14 Average retrieval time vs. number of returned shots

retrieved shots. However, the M-tree index has to pay a major overhead (about half a second) for retrieving the first shot, but its cost per retrieved shot (around 3.5 ms) is cheaper than that exhibited by the MySQL index structure (6.5 ms). These last two results demonstrate that visual features offer a cheap support whenever keyword-based search is unable to provide satisfactory recall levels.

5.4 Evaluation of video tagging and retrieval

Our final experiment concerns the evaluation of tagging and retrieval of full videos. As said, for this we requested ten users to evaluate the performance of SHIATSU when tagging/searching for full videos, as opposed to shots. Every user was shown some videos from the TRECVID benchmark and was then asked to evaluate the accuracy of tags suggested by SHIATSU for each video. Then, we let each user interact with the SHIATSU Retrieval Tool, asking her to perform some keyword-based searches at both video and shot level.

For the first experiment, after watching a video, the user was asked to provide ten tags (drawn from the overall 36 concepts in the TRECVID benchmark) for describing the video and such tags were then compared with the ten tags suggested by SHIATSU. Assessment of tagging at the video level was then averaged over all videos and users (note that in this experiment, since the number of user tags coincides with the number of suggested tags, precision equals recall), obtaining an overall accuracy of 73%. This result is consistent with the average precision of shot tagging, demonstrating the accuracy of hierarchical video tagging.

In the second experiment, each user was first asked to utilize the Retrieval Tool to search for whole videos using semantic tags, assessing the relevance of returned videos. Then, the users were made aware of the existence of shots and were asked to perform keyword-based searches on shots, evaluating their experience. Finally, we requested the users to indicate whether they found useful to search for videos and shots. Each user answered to a questionnaire, giving grades ranging from 0 (“strongly disagree”) to 4 (“strongly agree”).

Table 3 shows individual and average scores for the four statements. We note that there is a strong agreement on the accuracy of the system, with an average score of 3.75/4. The last two questions have the goal to understand whether users

Table 3 Mean satisfaction scores across participants, using 0 as "strongly disagree" and 4 as "strongly agree"

Statement	Agreement					Average
	0	1	2	3	4	score
a) "Retrieved videos are relevant to the query concept"	0	0	1	2	7	3.6
b) "Retrieved shots are relevant to the query concept"	0	0	0	2	8	3.8
c) "Video retrieval is a necessary component of the system"	0	0	1	1	8	3.7
d) "Shot retrieval is a necessary component of the system"	0	1	3	5	1	2.6

deem important to search for videos *and* shots. Considering that videos in the benchmark have an average length of 27.3 min while shots are about 10 sec long, it is not surprising that there is an overall preference of users for video retrieval, because retrieved shots were often considered too short to be interesting. Only one user considered shot retrieval almost unnecessary, grading it 1. When requested to comment on this, she answered: "When using several keywords for searching, very few shots were relevant for all query concepts, while I found it easier to retrieve relevant videos". Indeed, this user has highlighted the fact that shots, being homogeneous in their content, are also necessarily short and usually described with very few concepts. On the other hand, video tags represent a summary of the most representative shot tags, thus it is more likely that, when using several query tags, a video will contain all of them.

We finally remark that although the result of this experiment proves the usefulness of video retrieval, shot retrieval has nevertheless to be considered an important feature of SHIATSU. Since shot retrieval is also significant in a variety of scenarios, several of which recur in the field of media production, we believe that SHIATSU is the first system which is able to provide the users with the best of both (video and shot retrieval) worlds.

6 Related work

Video tagging has become a popular field of study due to the emergence of video sharing platforms such as YouTube. However, in the vast majority of cases, the task of indexing videos by content is usually neglected; this can be only performed by typing meta-information (normally, by means of textual labels) when a user uploads a video. This could generate ambiguity (because a label could carry different meanings due to polysemy or homonymy), lack of information (because a video could miss some of its information), and problems of synonymy/mistyping. On the other hand, the use of automatically extracted visual features is known to lead to misinterpretation of such extracted data. The distance between visual features and the image/video meaning has been termed the *semantic gap* problem, and frequently appears when attempting to automatically interpret complex media [32]. To overcome this, the (semi-)automatic tagging of multimedia objects based on semantic concepts is usually exploited [17, 18]: for the video domain this is commonly carried out using annotation of keyframes [12, 16, 25], so as to detect and categorize objects in video scenes. This also allows the use of annotation tools originally devised for the realm of still images [4, 11, 12].

Concerning the segmentation of videos into shots, one of the basic problems is the selection of the appropriate set of features to be used to detect shots: every frame is described by a vector of properties and differences between those vectors represent differences in content for video frames. Color histograms [21, 23, 31] are widely used to describe frames, because they are easy to compute and mostly insensitive to translational, rotational, and zooming camera motions, hence being quite effective to represent frame content difference. Edges extracted from frame objects [8, 21, 31, 43], although more computational-intensive than color histograms, are also commonly used to detect scene changes with good results, often in conjunction with color features. We chose to limit ourselves to a combination of these two features: the use of other features, like motion vector detectors [21, 35] or wavelet filters [2, 26], might improve the accuracy of video segmentation at the cost of a much higher computational overhead. As said, the shot detection task is, in our case, only half of the picture, our video tagging results being sufficiently accurate even without a perfect segmentation.

The task of automatic shot tagging is commonly carried out either by exploiting a knowledge restricted to a domain [14, 27] or by applying similarity search principles that have a more general applicability [4, 42]. Since the former is usually based on specific descriptors and/or on machine learning tools, their validity is limited to the existence of a very specific understanding of the underlying domain, which is clearly not the case for general purpose systems, like those commonly used by media producers (e.g., TV broadcasters). On the other hand, the basic idea of similarity-based annotation tools is to exploit automatically extracted low-level features to suggest labels for multimedia objects. This aims to overcome the semantic gap by introducing the user in the loop, with the limited burden of only checking the suggested tags.

The final step to overcome the semantic gap is then to unambiguously attach meaning to tags. Free tags, however, are not a viable solution, due to the existence of synonymy (a single concept represented by several, different labels) and homonymy/polysemy (a single label representing several, different concepts). For this, it is well known that concept hierarchies represent a simple, yet powerful, way of organizing concepts into meaningful groups [15, 19]. For example, Yahoo uses a topic-based hierarchy to organize web sites according to their topic and allows users to quickly identify web pages of interest, while Wikipedia contents are based on a hierarchy of categories (http://en.wikipedia.org/wiki/Portal:Contents/Categories). The biggest drawback of this approach is the fact that, while categorization of items can be performed (semi-)automatically, the hierarchies should be manually built, although studies have also focused on the automatically derivation of hierarchies [10].

When the number of categories is large, organizing them into a single taxonomy is detrimental for the usability of the overall structure. To this end, faceted hierarchies [15] are used in a variety of scenarios as a very flexible, and simple, way to represent complex domains [19, 40]. For example, they are successfully exploited in the domain of image retrieval (e.g., in the Catalyst image searching engine[7]) and browsing [3].

Regarding video retrieval, this is one of the most challenging task of nowadays information technology research [16]. Most of the existing video retrieval systems

[7] http://www.gettyimages.nl/Catalyst

exploit either contextual information, like textual information included in the same web page containing the video in Google Videos, or additional user-provided information, like description, comments, categorizations, in YouTube. A rather different approach is represented by content-based video retrieval which uses different types of features to derive annotations to describe videos. The main problem of content-based video search is related to the selection of the essential set of features that best represent each of the video shots (also referred as the video indexing problem). Such features include visual characteristics, like color and texture, temporal features (e.g., motion and audio), and high level concepts, such as event, person, indor/outdoor, etc. Towards this goal, as early as 2001 TRECVID has defined video search similarity as one of the tasks for evaluation, even if the retrieval accuracy achieved by state-of-the-art techniques is still deemed as unsatisfactory [36, 37].

Examples of systems that rely on visual features only to index the data are [33, 39]. Even if the latter approach has the great advantage to be completely automatic, it inherits from the image domain the limits given by the semantic gap problem [11]. Semantic indexing based on textual annotation of the video content seems to be a better option than visual features [16] because users usually search videos based on their meaning and not on visual features. Recent works, such as [13, 30], focus on *multi-modal* video retrieval allowing semantic concept search, feature-based search (through video clustering on low-level features and/or the *query by example* paradigm) and a combination of the two approaches. Since the annotation process exploited to assign labels to the video content usually makes use of visual features itself, it is quite intuitive to derive that the combination of the two search modalities does not bring much improvement to the final retrieval accuracy: low-level information, in fact, are already embedded into the semantic annotations. The keyword&feature-based search we included in SHIATSU aims to alleviate this inconvenience, so as to effectively exploit the feature-based search to expand the query result whenever the semantic (keyword-based) search is not able to return a sufficient number of results (this happens in those cases when the annotator performs poorly for some given concepts).

7 Conclusions and extensions

In this paper, we have presented SHIATSU, a system for the categorization and retrieval of videos. In designing it, our goal was to provide video producers with a tool able to combine a good accuracy with efficiency, so that produced videos can be processed in real time. The main functionalities of SHIATSU are the (semi-)automatic labeling of videos (which is performed by means of segmentation of videos into shots, similarity-based tagging of shots using multidimensional taxonomies, and hierarchical propagation of tags to whole videos) and their retrieval using (semantic) tags, visual features, or a combination of both.

Experimental results on two video benchmarks prove that SHIATSU indeed fulfills its requirements, achieving good precision levels for both annotation and retrieval without sacrificing efficiency: in particular, tagging of a video typically requires a time which is almost equal to the video length, while retrieval is performed in real time.

The segmentation module of SHIATSU, although based on a very simple algorithm, combines a nice accuracy with a good efficiency. We also note that the segmentation of videos is not the key feature in SHIATSU, but just a first step towards the automatic labeling of videos. Replacing the simple segmentation algorithm of SHIATSU with a more sophisticated one could improve the accuracy of shot detection (likely at the expense of speed), but its impact on tagging accuracy would be questionable. Yet, we are considering a modification of our keyframe selection strategy by clustering all frames of a shot, so as to reduce the complexity of tagging (note that two very similar keyframes would be likely tagged in the same way, although the tagging cost has to be paid twice).

As a final consideration, we highlight the fact that, with respect to other existing automatic video labeling tools, SHIATSU only considers the visual content of each video in order to suggest description labels: this can be easily complemented with other (more complex) techniques that analyze audio and speech, so as to provide a more accurate tagging of videos. For the moment, we stress the fact that (as shown in the experimental section) SHIATSU is able to obtain a good tagging accuracy, in spite of its simplicity and efficiency. Clearly, we are interested in extending our labeling technique so as to encompass this, otherwise neglected, video content.

Among the issues that we are currently considering, we would like to highlight:

- Currently, SHIATSU only allows to retrieve either videos or shots; some application scenarios, on the other hand, could benefit in a combined retrieval of videos *and* shots, thus we need to devise a technique for merging such heterogeneous result lists.
- Since the goal of this paper was to present the features of the SHIATSU system and to prove its effectiveness in helping the user retrieving videos of interest in a generic use scenario, we have somehow neglected a thorough analysis of the efficiency of video retrieval. Our preliminary experiments on this (included in Section 5.3) prove that, in our experimental scenario, SHIATSU is able to process queries in real time (<1 sec), but it might be the case that more complex indexing techniques are needed for databases containing large numbers of videos/shots and tags. In tackling this issue, we plan to exploit our decennial experience in query processing algorithms for similarity searching in large multimedia collections [5, 9].
- We are currently working to apply SHIATSU for the tagging/retrieval of videos in a real world application in the domain of cultural heritage. We would also like to experiment with a variety of other different contexts, like media production and personal collections.

References

1. Ardizzoni S, Bartolini I, Patella M (1999) Windsurf: region-based image retrieval using wavelets. In: IWOSS 1999, Florence, Italy, pp 167–173
2. Barbu T (2009) Novel automatic video cut detection technique using Gabor filtering. Comput and Electr Eng 35(5):712–721
3. Bartolini I (2009) Multi-faceted browsing interface for digital photo collections. In: CBMI 2009, Chania, Greece, pp 237–242
4. Bartolini I, Ciaccia P (2007) Imagination: accurate image annotation using link-analysis techniques. In: AMR 2007, Paris, France, pp 32–44

5. Bartolini I, Ciaccia P, Patella M (2010) Query processing issues in region-based image databases. Knowl Inf Syst 25(2):389–420
6. Bartolini I, Patella M, Romani C (2010) SHIATSU: semantic-based hierarchical automatic tagging of videos by segmentation using cuts. In: AIEMPro 2010, Florence, Italy
7. Canny J (1986) A computational approach to edge detection. IEEE Trans Pattern Anal Mach Intell 8(6):679–698
8. Chasanis V, Likas A, Galatsanos NP (2009) Simultaneous detection of abrupt cuts and dissolves in videos using support vector machines. Pattern Recogn Lett 30(1):55–65
9. Ciaccia P, Patella M, Zezula P (1997) M-tree: an efficient access method for similarity search in metric spaces. In: VLDB 1997, Athens, Greece, pp 426–435
10. Dakka W, Ipeirotis PG, Wood KR (2005) Automatic construction of multifaceted browsing interfaces. In: CIKM 2005, Bremen, Germany, pp 768–775
11. Datta R, Ge W, Li J, Wang JZ (2007) Toward bridging the annotation-retrieval gap in image search. IEEE Multimed 14(3):24–35
12. Datta R, Joshi D, Li J, Wang JZ (2008) Image retrieval: ideas, influences, and trends of the new age. ACM Comput Surv 40(2):article 2
13. Diou C, Stephanopoulos G, Dimitrou N et al (2009) VITALAS at TRECVID-2009. In: TRECVID 2009, Gaithersburg, MD, pp 16–17
14. Dorado A, Calic J, Izquierdo E (2004) A rule-based video annotation system. IEEE Trans Circuits Syst Video Technol 14(5):622–633
15. Fagin R, Guha R, Kumar R, Novak J, Sivakumar D, Tomkins A (2005) Multi-structural databases. In: PODS 2005, Baltimore, MD, pp 184–195
16. Geetha P, Narayanan V (2008) A survey of content-based video retrieval. J Comput Sci 4(6):474–486
17. Hauptmann AG, Yan R, Lin WH, Christel MG, Wactlar HD (2007) Can high-level concepts fill the semantic gap in video retrieval? A case study with broadcast news. IEEE Trans Multimedia 9(5):958–966
18. Hauptmann AG, Christel MG, Yan R (2008) Video retrieval based on semantic concepts. Proc IEEE 96(4):602–622
19. Hearst MA (2006) Clustering versus faceted categories for information exploration. Commun ACM 49(4):59–61
20. Hjaltason GR, Samet H (2003) Index-driven similarity search in metric spaces. ACM Trans Database Syst 28(4):517–580
21. Jacobs A, Miene A, Ioannidis GT, Herzog O (2004) Automatic shot boundary detection combining color, edge, and motion features of adjacent frames. In: TRECVID 2004, Gaithersburg, MD, pp 197–206
22. Jin Y, Khan L, Wang L, Awad M (2005) Image annotations by combining multiple evidence & WordNet. In: ACM Multimedia 2005, Singapore, pp 706–715
23. Kasturi R, Strayer SH, Gargi U, Antani S (1996) An evaluation of color histogram based methods in video indexing. In: International workshop on image database and multi media search, Amsterdam, The Netherlands, pp 75–82
24. Kleban J, Moxley E, Xu J, Manjunath BS (2009) Global annotation of georeferenced photographs. In: CIVR 2009, Santorini Island, Greece
25. Lew MS, Sebe N, Djeraba C, Jain R (2006) Content-based multimedia information retrieval: state of the art and challenges. ACM Trans Multimedia Comput, Commun and App 2(1): 1–19
26. Liao J, Zhang B (2008) A robust clustering algorithm for video shots using Haar wavelet transformation. In: IDAR 2007, Beijing, China, pp 81–82
27. Liu PY, Li F (2002) Semantic extraction and semantics-based anotation and retrieval for video databases. Multimedia Tools and Applications 17(1):5–20
28. Liu Z, Zavesky E, Gibbon D, Shahraray B, Haffner P (2007) AT&T research at TRECVID 2007. AT&T Labs - Research, Middletown, NJ
29. Navigli R (2009) Word sense disambiguation: a survey. ACM Comput Surv 41(2):article 10
30. Ngo C-W, Jiang Y-G, Wei X-Y, Zhao W, Liu Y, Wang J, Zhu S, Chang S-F (2009) VIREO/DVMM at TRECVID 2009: high-level feature extraction, automatic video search and content-based copy detection. In: TRECVID 2009, Gaithersburg, MD, pp 16–17
31. Qu Z, Liu Y, Ren L, Chen Y, Zheng R (2009) A method of shot detection based on color and edge features. In: SWS 2009, Lanzhou, China, pp 1–4
32. Rasiwasia N, Moreno PJ, Vasconcelos N (2007) Bridging the gap: query by semantic example. IEEE Trans Multimedia 9(5):923–938

33. Shanmugam TN, Rajendran P (2009) An enhanced content-based video retrieval system based on query-clip. Int J Research and Reviews in Applied Sci 1(3):236–253
34. Smeulders AWM, Worring M, Santini S, Gupta A, Jain R (2000) Content-based image retrieval at the end of the early years. IEEE Trans Pattern Anal Mach Intell 22(12):1349–1380
35. Su C-W, Liao H-YM, Tyan H-R, Lin C-W, Chen D-Y, Fan K-C (2007) Motion flow-based video retrieval. IEEE Trans Multimedia 6(9):1193–1201
36. TRECVID (2001) Guidelines for the Trec-2001 video track. http://www-nlpir.nist.gov/projects/trecvid/revised.html. Accessed 17 Nov 2011
37. TRECVID (2008) Guidelines for the TRECVID 2008 evaluation. http://www-nlpir.nist.gov/projects/tv2008/tv2008.html. Accessed 17 Nov 2011
38. Wang L, Khan L (2006) Automatic image annotation and retrieval using weighted feature selection. Multimedia Tools and Applications 29(1):55–71
39. Wu C-J, Zeng H-C, Huang S-H, Lai S-H, Wang W-H (2006) Learning-based interactive video retrieval system. In: ICME 2006, Los Alamitos, CA, pp 1785–1788
40. Yee K-P, Swearingen K, Li K, Hearst MA (2003) Faceted metadata for image search and browsing. In: CHI 2003, Ft. Lauderdale, FL, pp 401–408
41. Yuan J, Wang H, Xiao L, Zheng W, Li J, Zhang B (2007) A formal study of shot boundary detection. IEEE Trans Circuits Syst Video Technol 17(2):168–186
42. Zavřel V, Batko M, Zezula P (2010) Visual video retrieval system using MPEG-7 descriptors. In: SISAP 2010, Istanbul, Turkey, pp 125–126
43. Zhao H, Hu B, Zheng M, Li X (2009) Shot boundary detection based on mutual information and canny edge detector. J Commun and Comput 6(10):17–22

Ilaria Bartolini is currently an Assistant Professor with the DEIS department of the University of Bologna (Italy). She graduated in Computer Science (1997) and received a Ph.D. in Electronic and Computer Engineering (2002) from the University of Bologna. In 1998 she spent six months at CWI (Centrum voor Wiskunde en Informatica) in Amsterdam (The Netherlands) as a junior researcher. In 2004 she was a visiting researcher for three months at NJIT (New Jersey Institute of Technology) in Newark, NJ, USA. In January-April 2008 and in September-November 2010 she was visiting professor at the Hong Kong University of Science and Technology (HKUST). Her current research mainly focuses on collaborative filtering, learning of user preferences, similarity and preference-based query processing in large databases, and retrieval and browsing of image and video collections. Ilaria Bartolini has published about 40 papers in major international journals (including ACM TODS, IEEE TPAMI, IEEE TKDE, DKE, KAIS, and MTAP) and conferences (including VLDB, ICDE, PKDD, and CIKM). She served in the program committee of several international conferences and workshops. She is a member of ACM SIGMOD and IEEE.

Marco Patella got the "Laurea" degree in Electronic Engineering from the University of Bologna, Italy. He received a PhD in Electronic and Computer Engineering (1999) from the same University. Since 2006 he has been Associate Professor at University of Bologna with DEIS. His current research interests include similarity-based query processing in multimedia databases and Data Mining techniques. He is one of the designers of the M-tree, an index for metric data, which is used by several multimedia and data mining research groups in the world. He has published about about 40 papers, in the area of database systems, in major international journals (including IEEE TPAMI and ACM TODS) and international conferences (including VLDB, ACM-PODS, EDBT, and ICDE). He has also been part of the program committee of several international conferences and workshops. He is a member of ACM.

Corrado Romani received the "Laurea specialistica" degree in Informatics Engineering from the DEIS Department of the University of Bologna (Italy) in 2009. He is currently a fellow at DEIS with research interests related to image and video tagging, including shot detection, content-based retrieval, and semantic-based retrieval.