



Universidad
Carlos III de Madrid



This is a postprint version of the following published document:

Expert Systems with Applications (2015). 42(1), 488–502.
DOI: <http://dx.doi.org/10.1016/j.eswa.2014.08.001>

© 2014 Elsevier Ltd.



This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International License.

Temporal Segmentation and Keyframe Selection Methods for User-Generated Video Search-Based Annotation

Iván González-Díaz^a, Tomás Martínez Cortes^a, Ascensión Gallardo-Antolín^a,
Fernando Díaz-de-María^a

^a*Department of Signal Theory and Communications,
Universidad Carlos III de Madrid, Leganés, 28911, Madrid
e-mail: {igonzalez,tmcortes,gallardo,fdiaz}@tsc.uc3m.es*

Abstract

In this paper we propose a temporal segmentation and a keyframe selection method for User-Generated Video (UGV). Since UGV is rarely structured in shots and usually user's interest are revealed through camera movements, a UGV temporal segmentation system has been proposed that generates a video partition based on a camera motion classification. Motion-related mid-level features have been suggested to feed a Hierarchical Hidden Markov Model (HHMM) that produces a user-meaningful UGV temporal segmentation. Moreover, a keyframe selection method has been proposed that picks a keyframe for fixed-content camera motion patterns such as *zoom*, *still*, or *shake* and a set of keyframes for varying-content *translation* patterns.

The proposed video segmentation approach has been compared to a state-of-the-art algorithm, achieving 8% performance improvement in a segmentation-based evaluation. Furthermore, a complete search-based UGV annotation system has been developed to assess the influence of the proposed algorithms on an end-user task. To that purpose, two UGV datasets have been developed and made available online. Specifically, the relevance of the considered camera motion types has been analyzed for these two datasets, and some guidelines are given to achieve the desired performance-complexity tradeoff. The keyframe selection algorithm for varying-content *translation* patterns has also been assessed, revealing a notable contribution to the performance of the global UGV annotation system. Finally, it has been shown that the UGV segmentation algorithm also produces improved annotation results with respect to a fixed-rate keyframe selection baseline or a traditional method relying on frame-level visual features.

Keywords: User Generated Video, Video Annotation, Video Temporal Segmentation, Camera Motion Analysis, Keyframe Selection

1. Introduction

The amount of multimedia content that is generated daily has dramatically grown during recent years. This is particularly true in the case of User Generated Content (UGC), due to the massive access of users to mobile devices with recording capabilities [Cricri et al. \(2011\)](#). Consequently, algorithms providing automatic content annotation and content-based search are more and more demanded by both multimedia hosting services and users.

Although the automatic annotation problem has been traditionally posed as that of object/concept recognition [Smeaton et al. \(2006\)](#); [Everingham et al.](#); [Deng et al. \(2009\)](#), this approach has not yet reached a suitable solution due to the large amount of visual concepts to detect, including not only general visual categories such as car, street, or chair, but also particular places, people, artworks, and other objects of special interest for users.

Alternatively, the problem of content annotation can be approached by taking advantage of valuable user-provided metadata (tags, titles, and descriptions) that are available through online repositories such as Panoramio¹, Flickr² or Picasa³. Such a vast amount of (noisy) annotated contents opens the possibility of annotating a particular image or video by propagating tags from visually similar content. This approach has been referred to as search-based annotation in the literature [Wang et al. \(2006, 2010, 2012b\)](#).

Most successful methods make use of some kind of contextual information to pre-select a candidate set of images/videos that show some aspect in common with the query content. In [Soderberg & Kakogianni \(2010\)](#) a set of tags is suggested by combining the context in which the photo was captured with prior knowledge about popular anno-

¹<http://www.panoramio.com/>

²<http://www.flickr.com/>

³<http://www.picasa.com/>

tation concepts. In [Moxley et al. \(2008\)](#) GPS coordinates are used together with image features to propose labels that are chosen by considering both geographical distances and visual similarities. Similarly, in [Lee et al. \(2011\)](#) a Fuzzy ARTMAP network was used to map images and their visual features to geographic nouns. The main disadvantage of these methods is that they are not applicable to non-geolocated contents, which has resulted in the creation of mechanisms for automatic geotagging [Sevillano et al. \(2012\)](#); [Schindler et al. \(2008\)](#).

Although the initial approaches for search-based annotation were restricted to images, in the last few years some effort has been directed towards video content and, in particular, towards the development of methods that exploit redundancy among videos. In [Siersdorfer et al. \(2009\)](#) a system combining video copy detection and tag propagation techniques used redundancy between videos as a key to annotate new ones. The work in [Shang et al. \(2010\)](#) focused on real-time video retrieval over large-scale web datasets by developing efficient spatio-temporal features. In [Li et al. \(2011a\)](#), the authors proposed a system that relied on user global tags to further analyze the video content at shot level. The approach in [Tang et al. \(2013\)](#) went beyond and, besides identifying the particular segments associated with a tag, also generated spatio-temporal segmentations of the object representing the tag. The work in [Li et al. \(2011a\)](#) was later extended in [Wang et al. \(2012a\)](#) to detect events and automatically generate video summaries. Finally, [Ulges et al. \(2008\)](#) proposed a system that identified relevant frames in a video from its global tags and then used these frames to train concept detectors.

All the mentioned methods focus on similar aspects of the annotation system such as the development of robust and efficient visual search methods for near-duplicate contents, the application of these search methods to the more challenging tasks of video segment alignment and matching, the design of methods for tag propagation, or the deployment of systems for large-scale datasets with even billions of images. We have found, however, that little or no effort has been devoted to study how the video temporal segmentation and the subsequent keyframe selection affect the video annotation performance. In fact, all aforementioned methods employ very basic techniques to select the frames being analyzed: they run a shot-boundary detector to identify abrupt cuts in

videos and then represent each shot by means of one keyframe, normally sampled at the middle of the temporal segment.

As we will discuss in the section devoted to related work, although several methods can be found in the literature proposing smart techniques for video segmentation, all of them have been assessed just in terms of segmentation quality, thus obviating how they may influence subsequent end-user tasks, such as video content annotation.

In this paper, we present a video segmentation algorithm that analyzes the camera motion using a Hierarchical Hidden Markov Model (HHMM) and provides a fine-grain temporal segmentation of the video content. Moreover, a strategy for keyframe selection is proposed that considers a camera motion-based model of the interests of the person who is recording the video. Finally, we embed these subsystems on a complete system for automatic annotation of User Generated Video (UGV) and prove that our model for video segmentation not only achieves successful segmentation results, but also contributes to improve the performance of a high-level tasks such as specific object/place recognition and search-based video annotation.

Furthermore, as a by-product of our experimental evaluation, two video datasets for specific object/place recognition have been developed and made publicly available, which also becomes an important contribution of our work, and hopefully will help future developments in the field.

The rest of the paper is organized as follows. Section 2 introduces related work on temporal video segmentation and keyframe selection. Section 3 explains in detail the proposed method for automatic video segmentation and keyframe selection. Section 4 describes the complete system for video annotation. Section 5 is devoted to the experimental results, assessing both the segmentation performance and its impact on a higher-level task. Finally, Section 6 summarizes our conclusions and outlines some future work directions.

2. Related Work

Temporal video segmentation aims to split a video sequence into homogeneous subsequences, in such a manner that the properties of each subsequence are different

enough from those of its temporal neighbors. When dealing with edited video, most temporal segmentation techniques rely on shot boundary detection, which entails detecting both abrupt or gradual changes in the video and/or audio signal properties [Yuan et al. \(2007\)](#), [Smeaton et al. \(2010\)](#).

By contrast, user generated videos are usually continuous recordings taken with a mobile phone or a digital camcorder, where (frequently) only one shot is present. Thus, in order to divide UGVs into meaningful semantic units, segmentation must be performed at sub-shot level. According to the definition given by [Petersohn \(2009\)](#), a sub-shot is *an unbroken sequence of frames within a shot only having a small variation in visual content*. Some sub-shot detection methods are based on the comparison of color histograms between video frames [Petersohn \(2009\)](#), [Cahuina & Camara Chavez \(2013\)](#). However, since the type of camera motion (such as pan, tilt, or zoom) can be an indicator of the user's interests in the scene, and therefore of the video content, recently, several temporal video segmentation techniques have been proposed which use features derived from the camera motion information. In this sense, [Abdollahian et al. \(2010\)](#) define the so-called *camera view* as the basic unit of UGV.

Camera motion-based segmentation approaches involve, as first stage, the extraction of a set of features that allow for discriminating among the different types of camera motion considered. Typically used features include region-based correlation between consecutive frames [Aggarwal et al. \(2008\)](#), parameters derived from a 2D affine motion model [Bouthemy et al. \(1999\)](#), [Mei et al. \(2013\)](#), motion vectors [Abdollahian et al. \(2010\)](#), or even parameters provided by auxiliary motion-sensors, such as accelerometers [Cricri et al. \(2011\)](#).

Once the relevant features have been extracted, the temporal segmentation can be approached in different ways. In some works a simple thresholding method is used to detect the different camera motions [Bouthemy et al. \(1999\)](#), [Luo et al. \(2009\)](#), [Mei et al. \(2013\)](#). The main drawback of this method being the difficulty to find threshold values suitable for all kinds of video sequences. On the contrary, supervised machine learning methods, such as Support Vector Machines (SVM) or hidden Markov models, do not require any threshold adjustments. A SVM-based segmentation method was

proposed in [Abdollahian et al. \(2010\)](#), where binary SVMs are used to classify the camera motion of each video frame, and the final segmentation is obtained by grouping together neighboring frames that exhibit the same type of camera motion. HMMs have been used for shot detection and segmentation due to their ability for modeling time varying sequences [Bae et al. \(2004\)](#), [Zhang et al. \(2006\)](#). Nevertheless, the complexity of multimedia data might make HMM not suitable for this kind of tasks. To address this limitation, more recently, it has been proposed the use of hierarchical hidden Markov models (HHMM) for the indexing of daily living activities in videos acquired from wearable cameras [Karaman et al. \(2011\)](#), [Karaman et al. \(2014\)](#).

Following the concept of camera view-based segmentation proposed in [Abdollahian et al. \(2010\)](#) and the use of HHMM for video analysis suggested in [Karaman et al. \(2014\)](#), in this paper we propose a camera motion-based video segmentation method for UGC that uses mid-level features derived from the motion vectors and a hierarchical hidden Markov model that performs the temporal segmentation.

Since the focus of the paper at application level is the annotation of UGV, an optimal representation of the video temporal segments that maximizes the tradeoff between annotation/retrieval performance and computational complexity is also desirable. When dealing with video content, the computational efficiency has been traditionally achieved by means of a keyframe selection mechanism, in such a manner that a video temporal segment is represented by one or more keyframes that properly represent its visual content. Several works can be found in the literature addressing this problem, from simple approaches selecting one frame per video shot [Shang et al. \(2010\)](#); [Li et al. \(2011a\)](#); [Su et al. \(2010\)](#), to more advanced techniques extracting several keyframes per shot. In general, this problem is known as Video Abstraction and represents a preprocessing step required in several applications such as video browsing, video summarization, video retrieval, or video event detection. Most methods rely on low-level visual descriptors to represent frame content, such as color histograms [Li et al. \(2011b\)](#); [Zhang et al. \(1997\)](#); [Ciocca & Schettini \(2006\)](#), histograms of gradient orientations [Li et al. \(2011b\)](#), motion features [Liu et al. \(2003\)](#); [Chang & Chen \(2007\)](#); [Li et al. \(2009\)](#), or even visual attention-based features [Peng & Xiaolin \(2010\)](#). These features are then used to select the most informative keyframes within each video shot

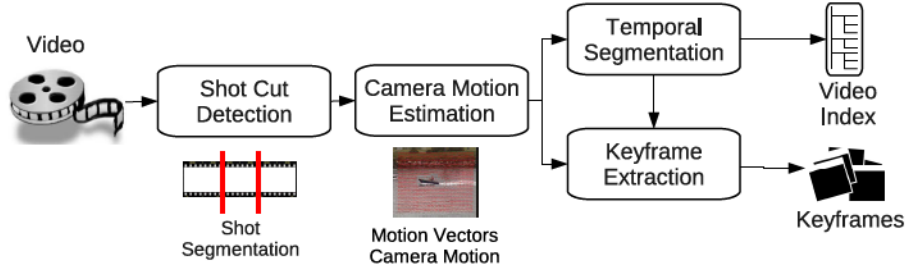


Figure 1: Processing pipeline of the proposed UGV Temporal Segmentation and Keyframe Selection

by applying sequential comparisons [Zhang et al. \(1997, 2003\)](#), global comparisons such as maximum coverage [Rong et al. \(2004\)](#), keyframe correlation [Li et al. \(2011a\)](#) or graphs [Porter et al. \(2003\)](#); [Gao et al. \(2009\)](#), or clustering methods such as that proposed in [Girgensohn & Boreczky \(1999\)](#).

In contrast to all these methods, our proposal relies on detected camera motion patterns to select the most informative keyframes in each video segment resulting from the previous video segmentation stage. From our point of view, these mid-level camera motion patterns are more meaningful to represent the user’s interests in UGV than the aforementioned traditional visual features. Hence, our work is more aligned with the previous proposal in [Abdollahian et al. \(2010\)](#), but besides developing a more robust approach for temporal video segmentation, we evaluate how video segmentation and keyframe selection methods contribute to improve the performance of a UGV annotation system.

3. Proposed UGV Temporal Segmentation and Keyframe Selection

In most previous approaches, the temporal video segmentation task is restricted to the detection of shot changes and a simple selection of one keyframe per shot. However, as already mentioned, UGV is rarely structured in shots. A much more common scenario is a single long-duration shot, in which the camera moves from one object to another (typically by panning) and stands still in front of or zooms in on the most interesting ones. Therefore, in this particular scenario, using camera motion is probably the most reliable source of information to temporally segment a video. In any case, as it is

possible to find several shots in a UGV, our system also includes a shot-cut detection mechanism.

We have divided this subsystem into three processing blocks: a first block that detects abrupt shot changes; a second block that estimates global camera motion by means of a parametric model inferred from the motion vectors directly extracted from the bitstream; and a third block that classifies motion into one of five motion patterns (pan/tilt, zoom, still, shaky, or fast), thus providing a suitable temporal segmentation. Once the segmentation has been performed, a set of keyframes is extracted representing each temporal segment. Figure 1 shows the block diagram of the proposed temporal video segmentation and key frame selection process.

The traditional shot-cut detection divides the UGV into several very long shots delimited by either an abrupt cut between two scenes or by very fast camera motions that completely change the view of the camera (and therefore are also detected by this kind of algorithms). However, representing a video by such a long segments and, subsequently, each segment by just one keyframe, might cause dramatic losses of useful information and, consequently, would prevent the system from properly annotating the video content. Therefore, a finer-grain segmentation is required to produce more meaningful video segments. With this purpose in mind, an advanced temporal segmentation approach is proposed that analyzes the camera motion at frame level, and produces a video segment for each group of consecutive frames exhibiting a stable motion pattern (pan/tilt, zoom, still, shaky, or fast).

The abrupt shot detector was implemented using the second derivative of the difference between gray-level histograms of consecutive frames, a simple and adaptive measure that allows to perform a fast scene segmentation with a high rate of success. The next blocks are described in the following subsections.

3.1. Camera Motion Estimation

Several models and methods for camera motion estimation have been proposed in the literature Liu (2008), Weng & Jiang (2011). In our system, we have adopted an efficient global camera motion model that considers three parameters, each corresponding

to one major direction: horizontal (H), vertical (V) and radial (R), as proposed in [Abdollahian et al. \(2010\)](#). Henceforth, we will refer to this model as HVR.

Since the model parameters are estimated from a dense local Motion Vector Field (MVF), we need to generate this motion field for every frame in a video. To that end, and with the objective of minimizing the computational burden, we use the motion vectors available in the coded bitstream.

Our implementation generates a dense block-based MVF in which a motion vector is assigned to each 8x8 block in the image, which dramatically reduces the computation time of a pixel-wise dense motion estimation and still provides enough data points to robustly estimate the camera motion. Since modern video coding standards handle different sizes of block (from 4x4 to 16x16 in H.264 [Wiegand et al. \(2003\)](#)), vector replications or interpolations are made when necessary. The obtained MVF is denoted as $\mathbf{V} = \{\mathbf{v}_t(\mathbf{x})\}_{M \times N}$, where t represents the time instant, M and N are the frame dimensions in number of blocks, and $\mathbf{x} = \{x, y\}$ are the block coordinates. This matrix relates the block locations in subsequent frames of the video, such that ideally $\mathbf{x}_t = \mathbf{x}_{t-1} + \mathbf{v}_t(\mathbf{x})$. Most of the vectors in this map will follow the camera motion whereas a small portion will show other motion patterns associated with objects that are moving independently in the scene.

Nevertheless, vectors coming from the bitstream do not always represent real motion due to the fact that cost functions in video coding standards consider a tradeoff between quality and bitrate (rate-distortion optimization). These non-real motion vectors usually happen in low textured regions, where the bits associated with the motion vectors become an important part of the total amount of bits allocated to the block, producing motion fields with static vectors. To avoid processing misleading non-real motion vectors, we have implemented a simple texture classification system that decides if a motion vector should be included or not in the camera motion parameterization process by comparing a measure of the texture in a block (we use the standard deviation σ of the grayscale values of pixels within the block) with a predefined threshold th_σ ($th_\sigma = 30$ in our experiments).

Taking this post-processed MVF as reference, the next step consists of estimating

the HVR parameterization that properly approximates the local motion vector map:

$$\begin{pmatrix} \hat{x}_t \\ \hat{y}_t \end{pmatrix} = R_t \begin{pmatrix} x_{t-1} \\ y_{t-1} \end{pmatrix} + \begin{pmatrix} H_t \\ V_t \end{pmatrix} \quad (1)$$

In order to build this approximation an error measure $e(\mathbf{x}_t - \hat{\mathbf{x}}_t)$ is minimized that considers the difference between those block locations \mathbf{x}_t provided by the MVF and those ones $\hat{\mathbf{x}}_t$ estimated by the HVR model. In particular, the HVR parameters are estimated using a robust algorithm for parameter estimation based on RANSAC [Fischler & Bolles \(1981\)](#). The aim of RANSAC is to properly estimate the HVR parameters even in presence of the outliers caused either by objects in motion or by errors in the motion estimation process during video coding. RANSAC is able to provide successful results as long as the background represents the largest element/object in a frame (even though it takes up less than the 50% of the frame). Hence, the output of this module is, therefore, a sequence of camera motion estimations, i.e.: $HVR(t), t = 1 \dots T$, where T is the length of the video sequence.

3.2. Temporal UGV segmentation algorithm

As previously mentioned, in UGV, camera motion information is usually quite relevant to determine how the video is structured and which are the most significant segments. This subsection describes the proposed temporal video segmentation algorithm and is organized in three parts: a) description of the camera motion patterns of interest; b) proposed set of mid-level features used to classify the camera motion into these patterns; and c) suggested probabilistic approach to perform the classification and segmentation. Once the temporal segmentation is available, an appropriate strategy for keyframe extraction is needed, as described in [Section 3.3](#).

3.2.1. Camera motion patterns

A wide set of camera motions can be considered: boom, track, roll, pan, zoom, tilt, still, etc. Furthermore, there are several combinations of those which lead to other more complex ones. However, many of them are hardly found in UGV and can be removed from our study without a significant loss of performance. Hence, in this work, we consider five patterns of camera motion that recurrently appear in UGVs:

1. *Translation (pan, tilt, diagonal)*: the camera follows a linear axis (horizontal, vertical or diagonal) and, although some changes on the motion speed might appear, it remains quite stable (H and V parameters exhibit low variance). It is associated with various scenarios: the camera follows a moving object, as a transitory motion between objects of interest, or to construct panoramic views when the objects are too large to fit into the camera wide.
2. *Zoom*: the camera follows a radial axis. It typically occurs when an interesting area is being focused and usually entails a sequence of consecutive zoom in/out patterns. This motion is well defined by a sequence of non-zero values in the R value of HVR model (zoom-in/zoom-out produce positive/negative values, respectively).
3. *Fast or Blurry*: it is a sudden and fast camera motion that blurs the image. The frames recorded during this motion are normally of no interest for the user. This pattern may be characterized by displacements of large magnitude, with huge variance and sudden changes in speed.
4. *Shake*: small displacements caused by a hand-held camera. It typically occurs when the camera man is walking, traveling in some sort of vehicle or there is simply a wobbly hold or support. It can be described by a large number of direction changes per second in the horizontal and vertical components of motion (H,V in the HVR model).
5. *Still*: represents the absence of motion and indicates that an object of interest is being recorded. It is, along with the zoom, the most relevant camera motion due to its high probability of containing user-relevant video segments.

3.2.2. *Mid-level features for camera motion classification*

These five motion patterns can be properly described by their average speed and acceleration, their variance, the number of direction changes in both axis, and the radial velocity of the motion. We propose to gather this information in a 9-dimensional mid-level feature vector f that can be easily computed from the HVR model. Since some of the features involve more than one frame, we have used a sliding window centered at the frame of interest, so that the parameters of a given frame depend on previous and

subsequent frames of the video. The length of the sliding window, denoted as W , has been experimentally set to $W = 42$ as we will show in section 5.1.

For a particular t frame, the suggested mid-level features are:

1. Average horizontal speed:

$$\bar{V}_x(t) = \frac{1}{W} \sum_{s=t-W/2}^{t+W/2-1} H(s) \quad (2)$$

The speed is a basic feature to discriminate between static and dynamic patterns.

2. Average vertical speed:

$$\bar{V}_y(t) = \frac{1}{W} \sum_{s=t-W/2}^{t+W/2-1} V(s) \quad (3)$$

3. Average horizontal acceleration:

$$\bar{a}_x(t) = \frac{1}{W} \sum_{s=t-W/2}^{t+W/2-1} \frac{dH(s)}{ds} \quad (4)$$

The acceleration is very useful to discriminate between several dynamic patterns: translation patterns are usually associated with constant motion whereas fast or blurry patterns are more random, with continuous changes in speed.

4. Average vertical acceleration:

$$\bar{a}_y(t) = \frac{1}{W} \sum_{s=t-W/2}^{t+W/2-1} \frac{dV(s)}{ds} \quad (5)$$

5. Variance of horizontal acceleration:

$$\sigma_{a_x}^2(t) = \frac{1}{W-1} \sum_{s=t-W/2}^{t+W/2-1} (a_x(s) - \bar{a}_x(t))^2 \quad (6)$$

6. Variance of vertical acceleration:

$$\sigma_{a_y}^2(t) = \frac{1}{W-1} \sum_{s=t-W/2}^{t+W/2-1} (a_y(s) - \bar{a}_y(t))^2 \quad (7)$$

7. Average number of horizontal direction changes: very useful to detect shaky motion patterns presenting large number of direction changes.

$$\bar{d}_x(t) = \frac{1}{W} \sum_{s=t-W/2}^{t+W/2-1} |\text{sgn}(H(s)) - \text{sgn}(H(s-1))| \quad (8)$$

8. Average number of vertical direction changes: as the previous one, very useful to detect shaky motion patterns.

$$\bar{d}_y(t) = \frac{1}{W} \sum_{s=t-W/2}^{t+W/2-1} |\text{sgn}(V(s)) - \text{sgn}(V(s-1))| \quad (9)$$

9. The R parameter of the HVR model: since it models the zoom pattern.

3.2.3. An HHMM for motion classification and segmentation

Once we have computed the mid-level feature vector for each frame, segmentation can be approached as a classification task where each vector gets associated with one of the five types of camera motion considered. In previous works, this task has been tackled using a cascade of SVM classifiers [Abdollahian et al. \(2010\)](#). However, when dealing with UGV, camera movements tend to extend over tens or even hundreds of frames. Since SVMs do not properly handle this temporal structure of video recordings, our temporal video segmentation system relies on the well-known hidden Markov models.

HMMs [Rabiner \(1989\)](#) are statistical generative models that allow for capturing the temporal structure of the camera motion in UGV. Basically, HMMs consist of a set of hidden variables, called states, which follow a temporal sequence or Markov chain. At regularly spaced times, the system undergoes a change of state according to the corresponding transition probabilities and an observation is emitted according to the set of emission probabilities of the current state. To explain how the HMM-based segmentation method works, let us consider the example illustrated in the central part of [Fig. 2](#) (gray states). A M -state HMM is shown where S_i , with $i = 1, 2, \dots, M$, represents each state, a_{ij} is the probability of transition from state i to state j , and b_i is the observation symbol probability distribution for state i . Thus, in general, an HMM model is characterized by the number of states N and a set of model parameters $\lambda = (A, B, \pi)$, where A is a matrix that contains all the transition probabilities (a_{ij}), B represents the set of observation probabilities (b_i) and π is the probability distribution of the initial state.

The mechanism for generating a Markov observation sequence of length T , $O = [O_1, O_2, \dots, O_T]$ is as follows:

1. Choose an initial state S_i in the initial instant $t = t_0$ according with π .
2. Generate a symbol O_t in the chosen state using b_i .
3. Do $t = t + 1$ and make a transition from state i to j according to a_{ij} .
4. If $t < T$ return to step 2), otherwise finish the process.

The likelihood of the sequence O given the model λ can be computed as:

$$P(O/\lambda) = \sum_{q_1, q_2, \dots, q_T} \pi_{q_1} b_{q_1}(O_1) a_{q_1 q_2} b_{q_2}(O_2) \dots a_{q_{T-1} q_T} b_{q_T}(O_T) \quad (10)$$

where q_t is a state S in a time t and $b_{q_t}(O_t)$ is the emission probability of the observation O_t associated with state q_t . There are plenty of state sequences that could lead to this observation sequence. However, it is possible to measure which is the most likely path along the states by using the Viterbi algorithm [Rabiner \(1989\)](#).

Unfortunately, we can not rely on an HMM in which each motion pattern is represented by just one state. The rationale behind is that one state is not enough to represent the statistical complexity of a motion pattern when the videos have been recorded by multiple users. Instead, it is necessary to build a full individual HMM model to represent each type of motion and then design an appropriate way for integrating all of them. In our system, this is accomplished by using hierarchical hidden Markov models [Fine et al. \(1998\)](#). In particular, we have designed a two-level HHMM, in which the bottom level models the different camera motions, and the top level is intended to represent the temporal video structure, i.e., transitions between motion types.

[Fig. 2](#) illustrates the proposed HHMM architecture for a simplified case in which only two different motion types are considered. Note that dashed circles represent non-emitting states, whereas dashed arrows indicate forced transitions (with probability 1). The bottom level HHMM is composed by five HMM submodels, one for each type of camera motion. Specifically, each motion type is represented by a M -state HMM (in our case, M was set to 5 following the procedure explained in [Section 5.1](#)), where the corresponding emission probabilities (B matrix) are modeled by Gaussian mixture distributions with diagonal covariances. These individual submodels (A, B and π matrices) are estimated separately using the training partition of the video database described

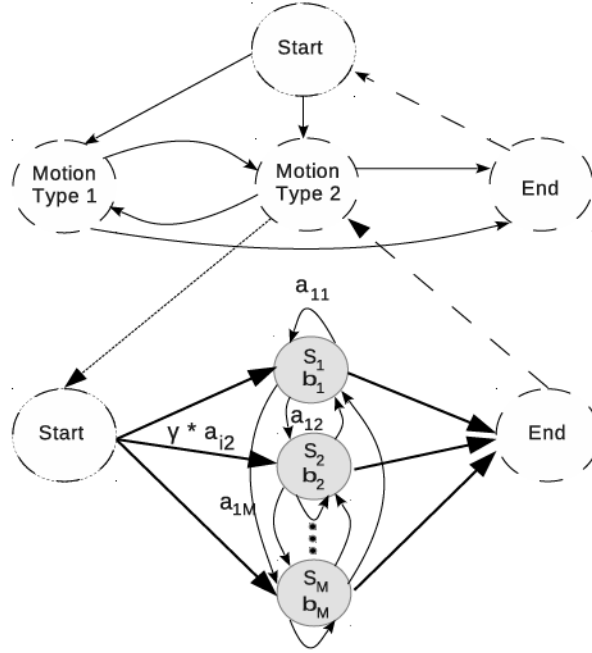


Figure 2: Hierarchical Hidden Markov Model. Only two of the five motion types are shown in the figure for simplicity.

in Section 5.1 through an iterative scheme based on the Expectation-Maximization (EM) algorithm [Dempster et al. \(1977\)](#).

Since an UGV can contain several types of camera movements, the top level of the full HHMM is built in such a way that transitions between different submodels are allowed. As it can be observed in Fig. 2, it is possible to jump from state i of one motion type to state j of any other motion with a probability $\gamma * a_{ij}$. The top level transition probabilities a_{ij} are estimated using the Viterbi Path Counting (VPC) algorithm [Davis & Lovell \(2003\)](#) during the training stage of the system, whereas the constant γ is determined in the validation stage and it controls the submodel insertion probability, preventing the appearance of short duration segments.

Once the HHMM is trained, the temporal segmentation is performed by executing the Viterbi algorithm on the video sequence, so that the most likely path of hidden states

is obtained. By analyzing this sequence of states, each frame is classified into one of the five considered camera motion types. From the resulting segments, keyframes are extracted following the procedure explained in next subsection.

3.3. *Keyframe extraction*

Since we aim to solve a higher-level task, such as a video annotation task, the optimal strategy will be that one which maximizes the performance of the annotation system analyzing the lowest number of keyframes. Therefore, the keyframe extraction strategy proposed in this paper relies on the previous camera motion classification and, in particular, on the meaning of each motion pattern for the annotation task. Specifically, the keyframe extraction process depends on the detected camera motion as follows:

- For *Still* and *Zoom* camera motion patterns, the visual contents should remain fixed during the whole video segment. It is worth noting that, although important changes in scale and even viewpoint may occur with respect to annotated reference images, our annotation approach is invariant to several geometric transformations and, therefore, one keyframe should contain all the information in the shot.
- *Shake* camera motion patterns are actually similar to *Still* patterns from the user point of view, the only difference being that the user fails to hold the camcorder steady. In other words, during shake camera motion segments the user focuses on a concept of interest. Therefore, one keyframe is enough again to capture the concept shown in the shot.
- *Fast* motion patterns normally appear when the user intends to change the objective of the recording (pointing to another place). Thus, frames associated with this motion pattern are of little interest for the user. Additionally, they may appear severely blurred due to the fast motion, which would dramatically decrease the performance of the annotation system. Therefore, no keyframes are extracted from shots associated with *Fast* patterns.

- The keyframe extraction model for the segments exhibiting a *Translation* motion pattern deserves a brief explanation. The perception accuracy of the visual content when this type of motion is involved varies according to the camera motion speed. Previous studies [Daly \(1998\)](#) have established that the human eye is specially attracted by motions of a certain velocity (between 6 and 30 deg./s. of the visual field), then decreasing its accuracy for faster motions. Therefore, instead of analyzing just one keyframe per video segment, as done for other patterns such as stills or zooms, when this motion pattern happens, the actual camera motion speed should be taken into account to decide on the keyframe sampling rate. In order to measure the camera speed, we consider the average translational speed (\bar{T}_s) in a video segment s as the average of the components $[H V]$ of our HVR motion model (see sec. 3.1). In particular, we propose a keyframe sampling rate defined by means of a shifted (T_{ref}) and scaled (r_{max}) triangular function:

$$r_s = r_{max} Tri\left(\frac{\bar{T}_s - T_{ref}}{T_{ref}}\right) \quad (11)$$

with r_{max} being the maximum sampling rate that depends on the desired computational complexity of the application; and T_{ref} has been heuristically estimated as the 25% of the largest dimension of the image. Intuitively, the keyframe extraction rate for translation segments grows with the camera speed \bar{T}_s until the value T_{ref} , when it starts to decrease according to the presumed lower user's interest on the content being recorded. In the experimental section, we will compare this strategy with a baseline fixed rate of keyframe sampling.

In summary, we are taking one keyframe from *Still*, *Zoom*, and *Shake* segments where we assume that the video content does not vary; we are taking several keyframes from *Translation* segments where we assume that the user is exploring a varying content, modulating the keyframe sampling rate according to a presumed perception accuracy derived from the camera speed; and finally, we are discarding *Fast* segments because we assume that they are meaningless for the user.

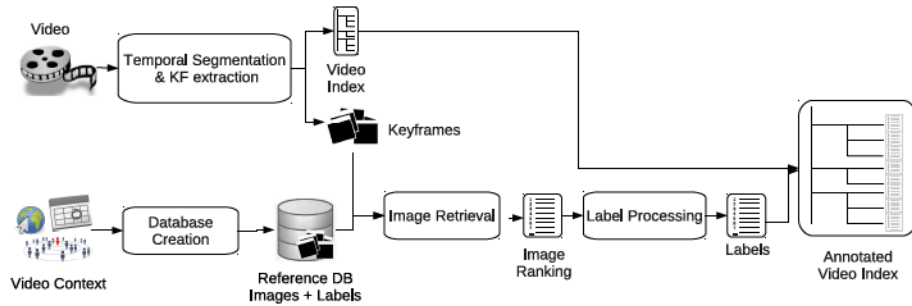


Figure 3: Processing pipeline of the proposed UGV annotation system.

4. Complete UGV Annotation System

In order to assess how the proposed temporal UGV segmentation and keyframe extraction subsystems contribute to improve the performance of a higher-level system, we have implemented a complete UGV annotation system that is described in this section.

The general scheme of our UGV annotation system implemented is illustrated in Fig. 3. Once the UGV is available, the system generates a video temporal index and selects the appropriate keyframes for each video segment according to the inferred camera motion-based relevance, as described in the previous section. In parallel, a set of reference images is selected according to the context information associated with the query. As we will describe in the experimental section, these reference datasets may be generated in a variety of ways: in some cases, we use a general name of the place where the video was recorded (a museum, a city, a region, etc.) to automatically generate a list of artworks or landmarks from Wikipedia; in others, we use the GPS coordinates obtained from the built-in GPS receivers of the acquisition devices. This information is used to perform searches over web-available image datasets such as Flickr to obtain a dataset of potentially similar images enriched by annotations in the form of tags, descriptions, or titles.

Once both keyframes and reference images are available, the image retrieval subsystem is in charge of computing a similarity measure between each keyframe and the reference images and, therefore, looking for near-duplicate contents. Although it is out

of the scope of this paper, it is worth noting that the experiments conducted in this paper have used a robust image retrieval system described in [Gonzalez-Diaz et al. \(2012, 2014\)](#), a probabilistic approach that incorporates robust methods of geometric verification for rigid objects. Given a video keyframe and set of R reference images related to the query, the output of this module is a vector containing the global similarity measure χ_r , with $r = 1 \dots R$, between the corresponding keyframe and each of the reference images.

Finally, based on this similarity measure, the system generates the set of annotations associated to each keyframe. For that end, two scenarios are envisaged: a) in some particular scenarios as those presented in the section [5.2](#), each reference image belongs to one of a pre-defined set of concept categories or vocabulary and max-pooling has provided the best results in our tests. Hence, the category of the most similar image is used as annotation. b) In a more general and unconstrained scenario, labels are given in the form of tags or title words provided by users, so that the final annotation process becomes more complex as we will explain in section [5.3](#).

5. Experiments and Results

In this section we describe the experiments and discuss the results concerning the assessment of the proposed UGV temporal segmentation and keyframe selection algorithms. We have conducted experiments at two levels. At subsystem level, we have assessed the performance of the HHMM-based UGV temporal segmentation subsystem in comparison to a state-of-the-art SVM-based system. At system level, besides the annotation performance, we have assessed the influence of the UGV indexing method on the automatic annotation process over two novel datasets for particular object/scene recognition in video. Finally, an online demo for automatic User Generated Image and Video annotation is also introduced that shows how our proposal can be applied to UGC annotation in very general and unconstrained scenarios.

5.1. Experiments on UGV Temporal Segmentation

In order to show the performance of the video temporal segmentation module, we have created a video database with clips showing contents generated by non-professional

users. In total, the database contains 2.12 hours of video, distributed in 106 different files with an average length of 1.2 minutes, which gives place to approximately 230K frames to be analyzed. The videos have been collected from Youtube and the average quality in terms of image resolution and bitrate is:

- resolution: 640 x 480
- bitrate: 400 kbps

The segmentation dataset has been manually annotated at frame level, so that each frame belongs to a specific category of camera motion. Furthermore, we have split the database into three sets of approximately equal size, namely: train, val and test. It is noteworthy that the split was carefully made to provide a balanced sub-division for every camera motion pattern, so that, for each camera motion pattern, a similar number of video segments was contained in each set. In any case, the split also resulted quite balanced in terms of video total duration.

Our approach to train the HHMM was as follows:

1. For the individual HMMs devoted to each motion pattern, we used the train set to generate models for several values of the hyperparameters (number of states M , length of the temporal window W) and evaluated those models over the validation set, thus selecting the optimal set of values $M=5$, $W=42$. The result of this validation is shown in Fig. 4(a).
2. Once the optimal individual HMMs were set, we built a global model, and validated the global hyperparameter γ over the validation set. As mentioned in Section 3.2.3, γ affects the submodel insertion probability and controls how likely our segmentation model allows changes between different motion patterns (balance between under and oversegmentation). Therefore, to optimize this parameter with some generality, we selected that value that minimizes over the validation set the relative difference between the number of temporal segments provided by our model and the true number of segments in the ground truth. As can be observed in Fig. 4(b), large values of γ lead to very unstable segmentations, with very short segments, and a lot of changes between motion patterns, producing an increase in the relative error. Nevertheless, for a wide range of values of

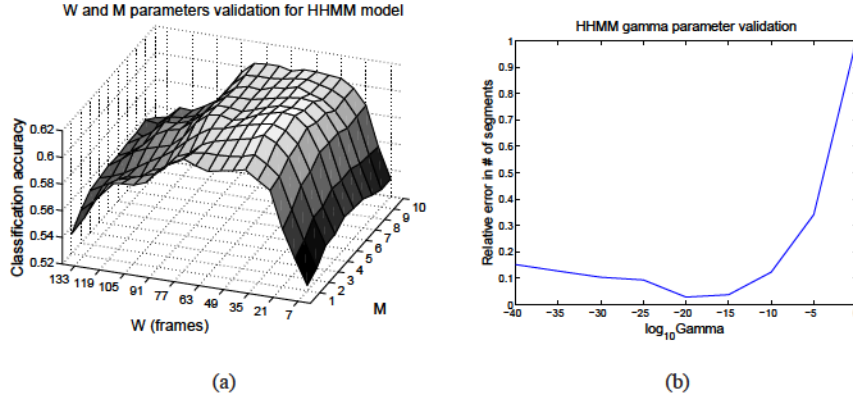


Figure 4: Results of the cross-validation procedure for HHMM hyperparameters. a) Joint 2D cross-validation of M (number of states) and W (length of temporal window); b) Validation of the γ hyperparameter.

Table 1: Huang and Dom (HD) segmentation error for the HHMM and SVM-based segmentation systems

Model	HD error w/o post.	HD error with post.
HHMM	0.4781	0.4747
SVM	0.5842	0.5150

γ , the achieved error remains low and quite stable. Finally, from the results in Fig. 4(b), we chose $\gamma = 1e - 20$ for the remaining experiments.

- Once we had all the optimal parameters, we trained a global HHMM model for video segmentation using the train+validation set, and assessed it using the test set.

We have assessed the temporal segmentation module according to two different system capabilities: 1) *Temporal Segmentation*; and 2) *Camera Motion Classification*. The former evaluates the system performance at partitioning a video into a sequence of segments with a stable motion pattern and, consequently, at extracting keyframes of interest that are automatically analyzed and annotated by subsequent stages in the processing pipeline. From our point of view, this represents the main functionality of the

segmentation module and, although the segmentation is computed based on the camera motion, the identified motion pattern is actually not considered in the evaluation. In other words, the alignment between the proposed and the ground truth segmentations is computed without taking into account the class of each segment.

The latter, meanwhile, aims to provide a complementary measure of the system performance at classifying the camera motion into one of the considered patterns. This classification, although initially conceived as a way to tailor the temporal segmentation, additionally allows us to assign a camera motion-based relevance to each video segment (e.g. segments with zooms tend to be more relevant for the user than ones with fast motion).

We have compared our approach to a SVM-based segmentation technique. This reference approach is very similar to the one proposed in [Abdollahian et al. \(2010\)](#) except for two differences: a) in order to ensure a fair comparison between the SVM- and HHMM-based segmentation approaches, the input features for the SVM are the same ones used in our proposal; and b) in [Abdollahian et al. \(2010\)](#) the multiclass problem was solved by means of a cascade of binary classifiers whereas, in our case, a multi-class implementation of the SVM was used to select the most appropriate motion pattern for each frame. Let us note that, similarly as we did for our HHMM, we have also validated the hyperparameters involved in the SVM-based model, getting the following optimal values: $W=42$, and $C=0.125$.

To evaluate the temporal segmentation performance we have used the well known Huang and Dom segmentation error [Huang & Dom \(1995\)](#), a measure in which bidirectional Hamming distances are computed (and later combined) between the Ground Truth (GT) and the Estimated Segments (ES). Each direction of the distance takes into account (and therefore penalizes) either oversegmentation (GT→ES) or undersegmentation (ES→GT). Segmentation results are provided in [Table 1](#), in terms of the alignment error between the two segmentations (0 means a perfect alignment). For each approach, two alternatives have been evaluated: a) the basic one, which uses the direct outputs of the classifier/segmenter (either the HHMM or the SVM outputs); and b) a post-processed approach, in which segments of less than one second (30 frames) are removed from the results. As it can be easily noticed, whereas post-processing

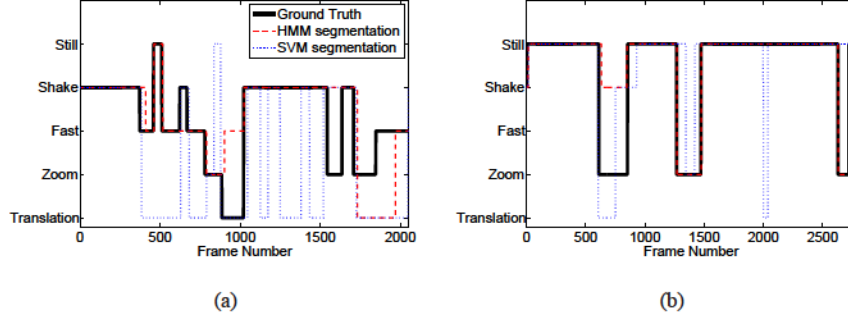


Figure 5: Two examples of video temporal segmentations based on camera motion classification. The red dashed line stands for our proposal, the blue dotted one for the SVM-based, and the ground truth is represented with a black solid line.

becomes critical for the SVM approach, the results of our approach remain unaltered when postprocessing is activated. The explanation is that the HHMM-based solution partitions the input video in a similar number of segments than those ones of the ground truth, whereas the SVM, due to the fact that it does not explicitly considers the transitions between different camera motions, notably suffers from oversegmentation. This observation is supported by the results shown in Fig. 5, which displays two examples of video temporal segmentation using camera motion classification for both the SVM-based approach and our proposal. Furthermore, even if we activate this post-processing step, our proposal notably outperforms the SVM-based solution, with an improvement of about a 8% (18% if we remove the postprocessing step). In the following, we are comparing the approaches that incorporate the post-processing stage.

In addition to this segmentation-based evaluation, we have also assessed the motion classification performance of our approach by computing the confusion matrices for the SVM and the HHMM-based systems, which are shown in Fig. 6 (a) and (b), respectively. In both figures, the rows correspond to the correct class, the columns to the hypothesized one and the cell colour indicates accuracy values (white corresponds to the highest classification rate and black to the lowest one). Table 2 shows the diagonals of these two confusion matrices, which provide the accuracy of each system for each class of camera motion, and the average classification rate computed as the mean of the

Table 2: Classification accuracy (%) for the compared HHMM- and SVM-based segmentation systems

Model	Trans.	Zoom	Fast	Shake	Still	Avg.
HHMM	29.48	56.11	51.42	53.73	89.91	56.13
SVM	69.15	36.72	38.48	44.41	78.90	53.53

diagonal. From this table, it can be observed that very similar average accuracies are obtained, with a slightly better performance of our approach. However, the HHMM-based system achieves better performance for almost every motion pattern (specially for *Zoom* which has higher semantic relevance than other motion types) except for the translational one, in which SVM obtains notably better results. Analyzing the confusion matrices, it can be observed that for the SVM-based system, *Fast* and *Shake* (and in less degree *Zoom* and *Still*) are frequently classified as *Translation* whereas HHMM reduces, in general, the confusability between classes. In other words, HHMM seems to discriminate better between the different motion patterns than SVM. Also, it is worth mentioning that in the HHMM system, *Translation* segments are mainly confused with *Still* because these segments present a very low camera motion speed and, according to the keyframe extraction strategy proposed in Section 3.3, they will produce almost the same results in terms of extracted keyframes as if they were considered as *Still* segments. For all the aforementioned reasons, and as we will see in the following sections, the HHMM is expected to perform better than SVM in the evaluation of the complete UGV annotation system.

Hence, we can conclude that, for a similar average classification accuracy, our proposal achieves more suitable segmentations in the sense that the number of detected video segments is much closer to the ground truth, and notably lower than that one provided by the SVM-based solution, which is prone to over-segmentation. This leads not only to better system performance (in terms of a more precise video indexing), but also to important savings in computational time for video segments showing *Zoom*, *Still*, and *Shake* patterns, for which our system extracts just one keyframe.

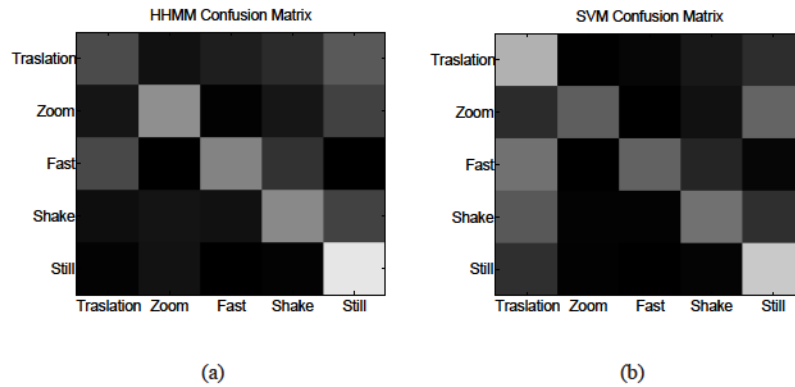


Figure 6: Confusion matrices [%]: (a) HHMM-based system; (b) SVM-based system.

5.2. Experiments on Video Annotation

We have assessed the influence of the UGV temporal segmentation subsystem on a higher level task. To that purpose we have built a complete system (see Section 4 that performs an automatic annotation of each video segment generated by the segmentation subsystem. For this assessment, we have generated two datasets that have been made available online⁴: the *Louvre Artworks dataset* and the *Madrid Landmarks dataset*. In the following paragraphs we will thoroughly describe both datasets.

The *Louvre Artworks dataset* contains 80 minutes of user generated video footage divided into 10 sequences with lengths ranging from 2 to 20 minutes. These video sequences have been recorded by non-professionals users either inside or outside the Louvre Museum and mostly show raw content with no or little edition. In parallel, using information automatically retrieved from Wikipedia, we have generated a list of 165 artworks (paintings and sculptures) that can be potentially detected in the video dataset. Furthermore, we have downloaded from Flickr an image dataset showing the selected artworks. It should be emphasized that the whole process for the dataset generation was completely automatic. On the one hand, this fact proves the usefulness of the proposed application since a database for a similar task could be readily generated in the same way. On the other hand, an automatic procedure inevitably entails diverse

⁴<http://cerceta.tsc.uc3m.es:9090/dbsLouvreMadrid.zip>

artifacts: some of the images may be incorrectly associated with an artwork due to noisy annotations by Flickr users; some images may show more than one artwork, etc, which makes things more challenging to our approach. The reference image database consists of 610 images and, in average, it contains between 4 and 5 images per artwork. We have manually annotated the video dataset, finding 123 occurrences of 49 distinct artwork categories. Let us note that, for simplicity, labels have been assigned at video level and, therefore, the evaluation is also performed on a video basis. Moreover, we have checked that the probability of assigning a correct label at video level because of a false positive detection at a given segment is negligible.

The *Madrid Landmarks dataset* contains 115 minutes of video divided into 20 sequences with lengths between 30 seconds and 30 minutes. These videos were recorded by tourists visiting Madrid and exhibit properties similar to those of the Louvre dataset. We have automatically identified 40 landmarks of Madrid and we have downloaded a total of 1998 reference images (approximately 50 images per landmark). After manually annotating the video dataset, we have found a total of 65 occurrences of 28 distinct landmarks. As in the previous case, the dataset has been generated following a completely automatic process. This dataset is more challenging than the Louvre dataset due to various reasons: the variability of the landmarks is higher than that of the artworks (a city vs. a museum); the visual appearance of an artwork, specially a paint that is a planar surface, exhibits lower variations than that of buildings and places of interest, which can be recorded from quite different points of view; the variability of photos taken in very large places, such as parks, is so high that makes very hard even to apply classical image retrieval techniques; etc.

In our experiments we have roughly extracted and analyzed 2000 video keyframes per dataset. Therefore, we can say that the size of these datasets (queries and reference images) is comparable to that of the most well known datasets for near-duplicate image search (e.g. Oxford DB [Philbin et al. \(2007\)](#), Holidays DB [Jegou et al. \(2008\)](#), or Paris DBJ. [Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman \(2008\)](#)).

In this scenario, since both datasets require detecting instances of from a predetermined list of concepts (either artworks or landmarks, depending on the task), we have preferred to assess the system performance in terms of its concept recognition ability,

by computing the Average Precision (AP). Specifically, we have manually annotated every video by listing the concepts that actually appear in it and we have computed the AP of the system by comparing these concepts to those automatically annotated by the system (one per video segment). Moreover, we have assessed the performance at different complexity levels, thus simulating systems with different computation time requirements. To that end, we have used the proposed camera motion-based UGV segmentation and keyframe selection procedures to generate a complete keyframe set, which has been then randomly subsampled, thus varying the computational complexity (the lower the number of evaluated keyframes per second, the lower complexity, and viceversa).

The proposed method for keyframe selection involves selecting one keyframe per each *shake*, *zoom*, or *still* segment, and several keyframes for (potentially varying-content) *translation* segments. The first experiment was devoted to assess the performance of the method proposed in Section 3.3 for varying the keyframe sampling rate in *translation* motion patterns according to the camera motion speed. In order to evaluate the proposed method, the reference keyframe subset for translations was built in two different ways, which were compared: a) as suggested by the proposed camera speed-based keyframe selection method; and b) using a fixed-rate keyframe selection method. For this experiment, all the keyframes associated with non-translation segments were used, while those of translation segments were sub-sampled, for the two compared reference subsets, to achieve results at several complexity levels.

In Fig. 7 the proposed keyframe selection method (green line) is compared to the fixed-rate keyframe selection method (blue line) for the Louvre Artwork detection task. The results in AP terms are shown as a function of the number of analyzed keyframes (expressed in keyframes per second). Let us note that this dependence has been achieved by varying the value of the r_{max} parameter in eq. (11). As can be seen, the proposed camera speed-based keyframe selection method notably contributes to improve the concept recognition results. Furthermore, the improvement remains approximately constant with the number of used keyframes, which means that the proposed speed-based keyframe selection process turns out to be more effective for concept recognition purposes. For the subsequent experiments, we have set the value of

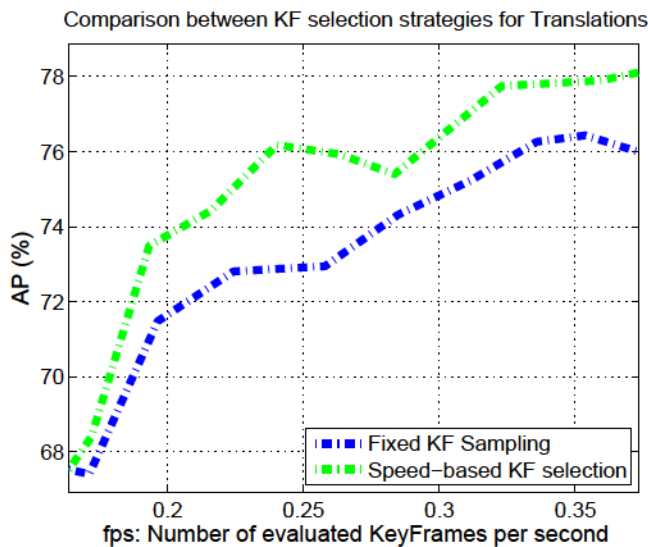


Figure 7: A comparison of Keyframe selection techniques for *translation* segments in the Louvre Artwork detection task.

the maximum frame rate $r_s = 1$ kfps, which approximately produces an effective frame rate of 0.25 kfps in Fig. 7.

Our second experiment aims to gain some insight into the relevance of the considered motion patterns for the video annotation task. In Fig. 8 we compare several system configurations using the HHMM segmentation approach for the Louvre Artwork detection task. In particular, each configuration uses only keyframes from segments exhibiting one particular motion pattern. As we did it previously, to compute the performance at different complexity levels, the corresponding reference keyframe sets were subsampled. The results highlight the *zoom* as the most relevant motion pattern for video annotation, which agrees with our initial hypothesis that this camera motion pattern is used to focus on concepts of special interest for the user. Nevertheless, the *zoom* pattern is not so commonly used and, therefore, limiting the analysis to this type of segments would probably lead to a large number of missed detections. Similarly, we have found that, although they represent a very small portion of the analyzed frames, the relevance of *shake* patterns is also quite notable. One could think of shaky seg-

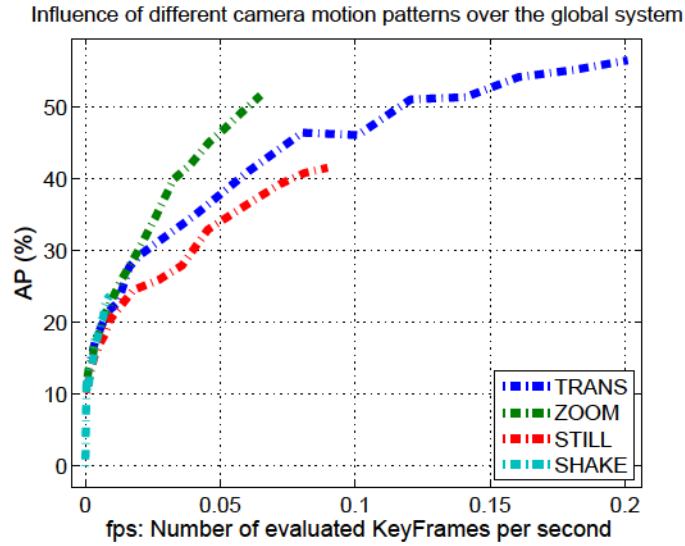


Figure 8: A study of the relevance of different motion patterns in the Louvre Artwork detection task.

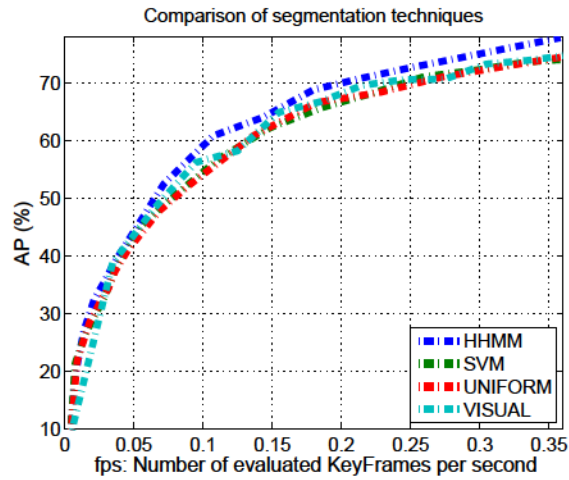
ments as *still* segments in which some kind of camcorder shake happens. However, the surprisingly poor results achieved for the *still* motion pattern put into question this hypothesis. After visualizing the videos, we found two reasons for these results: 1) there are many short still segments at the beginning or the end of translations. In all these cases, the segment does not contain any valuable content since the user either has not pointed yet to the object of interest or has already left it; and 2) still segments also happen in scenarios where the camera is attached to a tripod and records a person talking, without any concept of interest in the scene. We would like to note that this second finding could be generally arguable since people faces, although useless in our datasets, are usually of great interest in many problems. Finally, *translation* segments, although less relevant than *zoom*, are so popular in our video datasets that become the main source of information for video annotation.

Therefore, a smart keyframe selection method that relied on the type of camera motion could be used to properly balance complexity vs. performance in a given application. For instance, if complexity were the toughest requirement, the system would focus on *zoom* and *shake* segments; while if very high annotation rates were mandatory

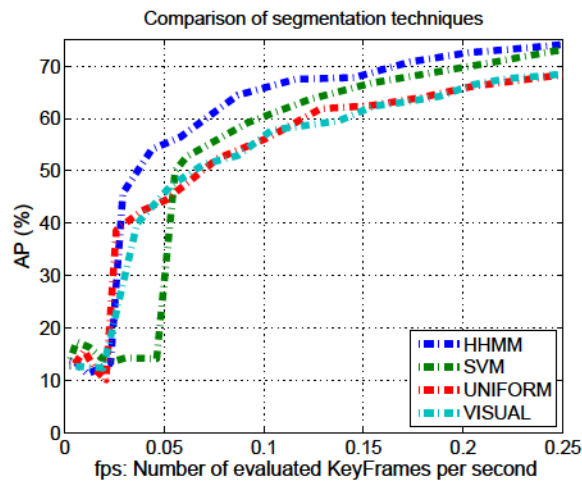
in our application, *translation* segments would be analyzed in detail at the expense of a notable increase in processing time. To prove this last insight, we have designed a simple *relevance-based keyframe subsampling* scheme, where the keyframe subsampling has been made according to a non-uniform probability density function that assigns the following heuristic relevance indexes to the different camera motion types: *fast*: 0, *still*: 1, *translation*: 2, *shaky*: 3, and *zoom*: 4. It is worth mentioning that these values have been obtained from the previous experiment by simply computing and rounding the average ratio between the AP and the number of analyzed frames for each particular motion pattern. Obviously, these relevance indexes could have been more carefully estimated, but we decided not to address this task because this simple set of values allowed us to prove our claim.

In Figs. 9(a) and 9(b) we show detection results for the *Louvre Artworks* and *Madrid Landmark* datasets, respectively. In each figure, we compare the performance of three strategies to select keyframes; namely: 1) a keyframe selection based on the camera motion-based segmentation achieved by the proposed hierarchical HMM (referred as to HHMM); 2) the same using a SVM for camera motion-based segmentation [Abdollahian et al. \(2010\)](#) (referred as to SVM); 3) a representative approach of the family methods relying on visual frame descriptors (VISUAL); and 4) a baseline keyframe selection method consisting on a fixed-rate sampling that does not take segmentation into account (UNIFORM). In particular, the visual frame descriptor method firstly performs a shot boundary detection and then, within each shot, clusters frames based on visual descriptors and selects as keyframes those frames that best represent each cluster. In particular, following previous approaches [Li et al. \(2011b\)](#); [Zhang et al. \(1997\)](#), we have considered color histograms, histograms of gradient orientations, and camera motion parameters as visual descriptors.

Let us remind that, in order to provide results at several complexities, both HHMM and SVM approaches use non-uniform sampling based on motion-based segment relevances, whereas the VISUAL approach sets the number of clusters (keyframes) per shot depending on the desired complexity. As shown in the figures, in both datasets, our proposal consistently outperforms the rest of the approaches at any given rate of evaluated keyframes. Furthermore, in the Madrid Landmarks dataset, both segmentation-



(a)



(b)

Figure 9: Comparison of recognition results for several camera motion-based segmentation methods in (a) the Louvre Artwork dataset, and (b) the Madrid Landmark dataset

based mechanisms clearly outperform those ones without segmentation as the number of evaluated keyframes increases. The rationale behind the poor results achieved by the SVM method for low rates of analyzed keyframes in Madrid dataset is the following: at these rates, this particular segmentation method analyzes a higher proportion of shake

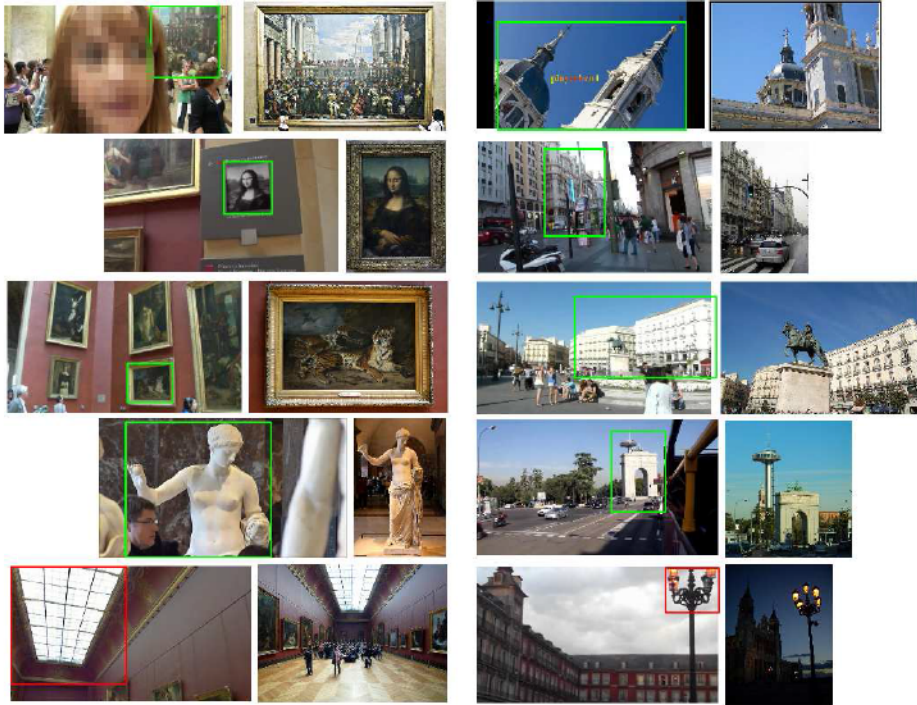


Figure 10: Some examples of outputs of the annotation over Louvre dataset (left column) and Madrid dataset (right column). The first 4 rows show relevant retrieved documents that give place to correct annotations (detected objects are shown in a green bounding box). The last row shows the first non-relevant retrieved image that gives place to a wrong annotation.

motion segments, as opposed to our method, that analyzes many more keyframes of zoom segments. This fact is due to probably wrong camera motion labels produced by the SVM, which leads to misalignment between the user preferences and the relevance-based sampling. Finally, the more traditional approach relying on frame visual features, although provides notable results in the Louvre dataset (probably due to the very distinct visual nature of each artwork in the museum), shows quite poor performance in the Madrid dataset. In summary, we can conclude that these results highlight the importance of suitable temporal segmentation and keyframe selection mechanisms in a video annotation system. This fact is particularly noticeable in the Madrid dataset, where both segmentation-based methods notably outperform the other techniques.

The proposed system for search-based UGV annotation achieves remarkable AP values: 77.82% and 74.86% for the Louvre Artwork and the Madrid Landmark datasets, respectively. In Fig. 10 we show some examples of video annotation. The first 4 rows show successful cases, while the last one illustrates annotation errors. As we can see in the first rows, our system is able to identify either artworks or landmarks of interest under a variety of transformations, such as scale, viewpoint change, varying illuminations, or severe occlusions. After an in-depth study of the results, we can conclude that the main source of error in the system are the missed detections due to the lack of keyframes showing the concept. Moreover, this type of errors can be attributed to the segmentation and keyframe selection modules, which again stresses their importance. On the contrary, false alarms are much less common since it is highly unlikely to find non-relevant images showing high similarity values with a given keyframe. In any case, in the last row of Fig. 10 we show two examples of false alarms that produce wrong annotations of the content. For the example in the Louvre dataset, the error is due to a noisy annotation in the reference dataset, since the reference image (right) was annotated by a Flickr user with the name of a particular painting (The Massacre at Chios) that is actually located in the room shown in the image. The example in the Madrid dataset is even more striking, since our system is matching the lamppost appearing in the two images. Although the places are completely different, the lamppost model is in fact quite similar in many streets of the town.

5.3. An online demo for Video Annotation

In order to demonstrate the system performance, an online demo of our annotation system is available online⁵. In this demo, our system provides automatic annotations and ROI segmentations given a geo-located query image or video.

The demo allows users to upload multimedia contents and geo-locate them using an intuitive web interface based on Google Maps. The user may select different values for the geo-location (from very precise, simulating capturing devices with a built-in GPS receiver, to very rough, covering the whole city/region where the content was taken),



⁵<http://cerceta.tsc.uc3m.es:9090/apps/AFICUS/web/AficusDemo.html>

VIDEO SUMMARY

Video tags
madrid
espanha
spain
monumentoaalfonsoxiii
retiro
lake
parque
statue
park



(a)

Shot no 3					
	Tags	Camera Motion Pattern	Relevance	Start frame	End frame
	Retiro	ZOOM	HIGH	518	652
	monument				
	spain				
	parque				
	madrid				
	park				
	Estanque				
	lake				
	Alfonso				
Monumento					
Similar content					

(b)

Figure 11: Online annotation demo: a) Video Summary (4 video segments) and annotation; and b) example of a segment annotation.

and for the number of retrieved images from Flickr. In addition, several demonstrative samples are also available for direct use. Once the input data are uploaded, our system retrieves from Flickr a set of reference images taken around the same location. It is worth mentioning that the database is always built in real-time and, thus, never stored in our servers. Consequently, this process entails an important overhead to the system operation with respect to a scenario in which reference images have been previously retrieved and processed. However, the objective of this demo is not to get a very fast annotation system but to show up its capabilities.

Compared to the previous experiments, in this case labels are given in the form of user tags/images titles provided by users uploading the reference content. Hence,

in order to provide a video annotation, the system computes a ranking of proposed labels in such a way that those labels associated with more similar images are better candidates to become part of the automatic annotation of the corresponding keyframe. Then, all the individual sets of labels (one per keyframe) are properly fused to provide a high level annotation for the whole video.

The typical tags that can be found in databases such as Flickr are noisy and do not follow any predefined vocabulary or taxonomy, which prevents from a straightforward fusion of labels. In particular, it happens that different, but very redundant labels, appear as a result of slight typographical variations. For example, “towerof ondon”, “tower_of.london”, and “thetowerof ondon” represent the same semantic concept and should be merged into just one tag. In our implementation, the Levenshtein distance [Levenshtein \(1966\)](#) has been used to evaluate the similarity between labels and merge those ones showing very low distances.

Then, each label associated with a reference image is weighted according to its visual similarity with the query; specifically, the weight is computed as follows:

$$w_r = \left(\frac{\chi_r}{\chi_{max}} \right)^\alpha \quad (12)$$

where the normalization factor χ_{max} is the maximum similarity measure in the reference set, and α , which verifies that $0 \leq \alpha < \infty$, is a parameter that was heuristically chosen ($\alpha = 2$ in our experiments). Higher values of α lead to annotations dominated by labels belonging only to the most similar images, whereas lower values lead to annotations exhibiting labels from many images of the reference set.

Finally, based on these weights, a label histogram H is computed by accumulating the weight of each label l of the reference set as:

$$H(l) = \sum_r^R w_r n_{lr} \quad (13)$$

where n_{lr} is equal to one if the label l was found in the reference image r and zero otherwise. A list of tags ordered according to their relevance to the query content is easily generated by sorting the histogram of labels $H(l)$ in descent order.

In summary, a fully annotated video index is obtained by taking into consideration two types of relevance measures extracted by the system. The first relevance measure,

called *camera motion-based relevance*, is inferred from the fact that certain camera motion patterns (such as zooms, shake, or stills) are actually relevant indicators of the user’s interest. The second, referred to as *visual similarity-based relevance*, is obtained from the visual similarity measures between each keyframe and the set of annotated reference images gathered by the image retrieval subsystem.

6. Conclusions and Further Work

In this paper we have proposed a temporal segmentation and a keyframe selection method for User Generated Video (UGV). Specifically, an efficient UGV temporal segmentation system based on camera motion has been proposed which partitions a video into user-meaningful segments. Additionally, a suitable keyframe selection algorithm has been proposed that maximizes the concept annotation performance for a target complexity level. Moreover, a complete search-based UGV annotation system has been developed to prove the contribution of the proposed methods to two different UGV annotation tasks.

The segmentation algorithm is based on a frame-level camera motion classification method that has specifically designed for UGV. A frame-level camera motion classifier has been proposed that relies on a carefully selected set of mid-level features designed to capture the proper clues for solving the camera motion classification problem. Subsequently, a novel HHMM-based system that uses these mid-level features as input has been proposed for segmenting the video according to the camera motion type. We have compared the proposed approach to a state-of-the-art SVM-based system. The experimental results allow us to conclude that our proposal provides more suitable segmentations, while the SVM-based system clearly incurs in over-segmentation.

The proposed keyframe selection method has been proved to be an essential tool to optimize the complexity-performance tradeoff. In particular, we have performed a relevance analysis of the different camera-motion types in the annotation system performance, which allows the application developer to choose a design that suitably balances complexity and performance.

The impact of both subsystems, the UGV temporal segmentation and the keyframe

selection, has been assessed on a complete annotation system over two video datasets. The complete system has shown notable performance in video annotation tasks: average precisions of 77.82% and 74.86% for Louvre artwork and Madrid landmark detection tasks, respectively. Furthermore, our experimental results have revealed that such a high performance exhibits a substantial dependence of both the temporal segmentation and the keyframe selection methods, as proved by the experimental results showing how the proposed system outperforms both a fixed-rate keyframe selection baseline and a more traditional method relying on frame-level visual features. It is also worth noticing that two UGV datasets, the *Louvre Artworks dataset* and the *Madrid Landmarks dataset*, have been made publicly available to support future developments on the field.

Moreover, in order to show the applicability of our method, an integrated system for UGC annotation has been developed and made available online. The demo allows users to play with different parameters of the method, showing the annotation results under different scenarios.

Since this work has focused on two particular elements of a search-based video annotation system, many research lines remain open. For example: those related to a more real-time oriented implementation; those concerning the incorporation of additional context-information (such as user preferences or social network-related data); or those associated with more elaborated tag propagation approaches.

7. Acknowledgments

This work has been supported by the National Grant TEC2011-26807 of the Spanish Ministry of Science and Innovation.

References

Abdollahian, G., Taskiran, C., Pizlo, Z., & Delp, E. (2010). Camera motion-based analysis of user generated video. *IEEE Transactions on Multimedia*, 12, 28–41. doi:[10.1109/TMM.2009.2036286](https://doi.org/10.1109/TMM.2009.2036286).

- Aggarwal, N., Prakash, N., & Sofat, S. (2008). Detecting camera movements and production effects in digital videos. In *Proceedings of the 1st Bangalore Annual Compute Conference (COMPUTE '08)* (pp. 13:1–13:5). New York, NY, USA: ACM. doi:[10.1145/1341771.1341785](https://doi.org/10.1145/1341771.1341785).
- Bae, T., Jin, S., & Ro, Y. (2004). Video segmentation using hidden markov model with multimodal features. In P. Enser, Y. Kompatsiaris, N. OConnor, A. Smeaton, & A. Smeulders (Eds.), *Image and Video Retrieval* (pp. 401–409). Springer Berlin Heidelberg volume 3115 of *Lecture Notes in Computer Science*. doi:[10.1007/978-3-540-27814-6_48](https://doi.org/10.1007/978-3-540-27814-6_48).
- Bouthemy, P., Gelgon, M., & Ganansia, F. (1999). A unified approach to shot change detection and camera motion characterization. *IEEE Transactions on Circuits and Systems for Video Technology*, *9*, 1030–1044. doi:[10.1109/76.795057](https://doi.org/10.1109/76.795057).
- Cahuina, E., & Camara Chavez, G. (2013). A new method for static video summarization using local descriptors and video temporal segmentation. In *26th Conference on Graphics, Patterns and Images (SIBGRAPI)* (pp. 226–233). doi:[10.1109/SIBGRAPI.2013.39](https://doi.org/10.1109/SIBGRAPI.2013.39).
- Chang, I.-C., & Chen, K.-Y. (2007). Content-selection based video summarization. In *International Conference on Consumer Electronics (ICCE 2007)* (pp. 1–2). doi:[10.1109/ICCE.2007.341528](https://doi.org/10.1109/ICCE.2007.341528).
- Ciocca, G., & Schettini, R. (2006). An innovative algorithm for key frame extraction in video summarization. *J. Real-Time Image Processing*, *1*, 69–88. doi:[10.1007/s11554-006-0001-1](https://doi.org/10.1007/s11554-006-0001-1).
- Cricri, F., Dabov, K., Curcio, I., Mate, S., & Gabbouj, M. (2011). Multimodal event detection in user generated videos. In *IEEE International Symposium on Multimedia (ISM)* (pp. 263–270). doi:[10.1109/ISM.2011.49](https://doi.org/10.1109/ISM.2011.49).
- Daly, S. J. (1998). Engineering observations from spatiovelocity and spatiotemporal visual models. *Proc. SPIE*, *3299*, 180–191. doi:[10.1117/12.320110](https://doi.org/10.1117/12.320110).

- Davis, R. I. A., & Lovell, B. C. (2003). Comparing and evaluating hmm ensemble training algorithms using train and test and condition number criteria. *Pattern Anal. Appl.*, 6, 327–336. doi:10.1007/s10044-003-0198-6.
- Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39, 1–38. doi:10.2307/2984875.
- Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 248–255). doi:10.1109/CVPR.2009.5206848.
- Everingham, M., Van Gool, L., Williams, C. K. I., Winn, J., & Zisserman, A. (). The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. <http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html>.
- Fine, S., Singer, Y., & Tishby, N. (1998). The hierarchical hidden Markov model: Analysis and applications. *Mach. Learn.*, 32, 41–62. doi:10.1023/A:1007469218079.
- Fischler, M. A., & Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24, 381–395. doi:10.1145/358669.358692.
- Gao, Y., Tang, J., & Xie, X. (2009). Key frame vector and its application to shot retrieval. In *Proceedings of the 1st International Workshop on Interactive Multimedia for Consumer Electronics (IMCE '09)* (pp. 27–34). New York, NY, USA: ACM. doi:10.1145/1631040.1631046.
- Girgensohn, A., & Boreczky, J. (1999). Time-constrained keyframe selection technique. In *IEEE International Conference on Multimedia Computing and Systems* (pp. 756–761). volume 1. doi:10.1109/MMCS.1999.779294.
- Gonzalez-Diaz, I., Baz-Hormigos, C., Berdonces, M., & Diaz-de Maria, F. (2012). A generative model for concurrent image retrieval and roi segmentation. In *In-*

- ternational Workshop on Content-Based Multimedia Indexing (CBMI)* (pp. 1–6). doi:[10.1109/CBMI.2012.6269844](https://doi.org/10.1109/CBMI.2012.6269844).
- Gonzalez-Diaz, I., Baz-Hormigos, C., & Diaz-de Maria, F. (2014). A generative model for concurrent image retrieval and ROI segmentation. *IEEE Transactions on Multimedia*, *16*, 169–183. doi:[10.1109/TMM.2013.2286083](https://doi.org/10.1109/TMM.2013.2286083).
- Huang, Q., & Dom, B. (1995). Quantitative methods of evaluating image segmentation. In *IEEE International Conference on Image Processing (ICIP)* (pp. 53–56). volume 3. doi:[10.1109/ICIP.1995.537578](https://doi.org/10.1109/ICIP.1995.537578).
- J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman (2008). Lost in quantization: Improving particular object retrieval in large scale image databases. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 1–8). doi:[10.1109/CVPR.2008.4587635](https://doi.org/10.1109/CVPR.2008.4587635).
- Jegou, H., Douze, M., & Schmid, C. (2008). Hamming embedding and weak geometric consistency for large scale image search. In *Proceedings of the 10th European Conference on Computer Vision: Part I (ECCV '08)* (pp. 304–317). Berlin, Heidelberg: Springer-Verlag. doi:[10.1007/978-3-540-88682-2_24](https://doi.org/10.1007/978-3-540-88682-2_24).
- Karaman, S., Benois-Pineau, J., Dovgalecs, V., Mégret, R., Pinquier, J., André-Obrecht, R., Gaëstel, Y., & Dartigues, J.-F. (2014). Hierarchical hidden markov model in detecting activities of daily living in wearable videos for studies of dementia. *Multimedia Tools and Applications*, *69*, 743–771. doi:[10.1007/s11042-012-1117-x](https://doi.org/10.1007/s11042-012-1117-x).
- Karaman, S., Benois-Pineau, J., Megret, R., Pinquier, J., Gaestel, Y., & Dartigues, J.-F. (2011). Activities of daily living indexing by hierarchical HMM for dementia diagnostics. In *International Workshop on Content-Based Multimedia Indexing (CBMI)* (pp. 79–84). doi:[10.1109/CBMI.2011.5972524](https://doi.org/10.1109/CBMI.2011.5972524).
- Lee, C.-H., Yang, H.-C., & Wang, S.-H. (2011). An image annotation approach using location references to enhance geographic knowledge discovery. *Expert Systems with Applications*, *38*, 13792 – 13802. doi:[10.1016/j.eswa.2011.04.182](https://doi.org/10.1016/j.eswa.2011.04.182).

- Levenshtein, V. I. (1966). *Binary codes capable of correcting deletions, insertions, and reversals*. Technical Report 8.
- Li, C., Wu, Y.-T., Yu, S.-S., & Chen, T. (2009). Motion-focusing key frame extraction and video summarization for lane surveillance system. In *IEEE International Conference on Image Processing (ICIP)* (pp. 4329–4332). doi:[10.1109/ICIP.2009.5413677](https://doi.org/10.1109/ICIP.2009.5413677).
- Li, G., Wang, M., Zheng, Y.-T., Li, H., Zha, Z.-J., & Chua, T.-S. (2011a). Shottagger: Tag location for internet videos. In *Proceedings of the 1st ACM International Conference on Multimedia Retrieval (ICMR '11)* (pp. 37:1–37:8). New York, NY, USA: ACM. doi:[10.1145/1991996.1992033](https://doi.org/10.1145/1991996.1992033).
- Li, P., Guo, Y., & Sun, H. (2011b). Multi-keyframe abstraction from videos. In B. Macq, & P. Schelkens (Eds.), *IEEE International Conference on Image Processing (ICIP)* (pp. 2473–2476). doi:[10.1109/ICIP.2011.6116162](https://doi.org/10.1109/ICIP.2011.6116162).
- Liu, L. (2008). A new method for camera motion estimation in video. In *The 9th International Conference for Young Computer Scientists (ICYCS)* (pp. 886–890). doi:[10.1109/ICYCS.2008.490](https://doi.org/10.1109/ICYCS.2008.490).
- Liu, T., Zhang, H., & Qi, F. (2003). A novel video key-frame-extraction algorithm based on perceived motion energy model. *IEEE Transactions on Circuits and Systems for Video Technology*, *13*, 1006–1013. doi:[10.1109/TCSVT.2003.816521](https://doi.org/10.1109/TCSVT.2003.816521).
- Luo, J., Papin, C., & Costello, K. (2009). Towards extracting semantically meaningful key frames from personal video clips: From humans to computers. *IEEE Transactions on Circuits and Systems for Video Technology*, *19*, 289–301. doi:[10.1109/TCSVT.2008.2009241](https://doi.org/10.1109/TCSVT.2008.2009241).
- Mei, T., Tang, L.-X., Tang, J., & Hua, X.-S. (2013). Near-lossless semantic video summarization and its applications to video analysis. *ACM Trans. Multimedia Comput. Commun. Appl.*, *9*, 16:1–16:23. doi:[10.1145/2487268.2487269](https://doi.org/10.1145/2487268.2487269).
- Moxley, E., Kleban, J., & Manjunath, B. S. (2008). Spirittagger: a geo-aware tag suggestion tool mined from Flickr. In *Proceedings of the 1st ACM international*

- conference on Multimedia information retrieval (MIR '08)* (pp. 24–30). New York, NY, USA: ACM. doi:[10.1145/1460096.1460102](https://doi.org/10.1145/1460096.1460102).
- Peng, J., & Xiaolin, Q. (2010). Keyframe-based video summary using visual attention clues. *IEEE Multimedia*, *17*, 64–73. doi:[10.1109/MMUL.2009.65](https://doi.org/10.1109/MMUL.2009.65).
- Petersohn, C. (2009). Temporal video structuring for preservation and annotation of video content. In *16th IEEE International Conference on Image Processing (ICIP)* (pp. 93–96). doi:[10.1109/ICIP.2009.5414114](https://doi.org/10.1109/ICIP.2009.5414114).
- Philbin, J., Chum, O., Isard, M., Sivic, J., & Zisserman, A. (2007). Object retrieval with large vocabularies and fast spatial matching. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 1–8). Los Alamitos, CA, USA: IEEE Computer Society. doi:[10.1109/CVPR.2007.383172](https://doi.org/10.1109/CVPR.2007.383172).
- Porter, S. V., Mirmehdi, M., & Thomas, B. (2003). A shortest path representation for video summarisation. In *12th International Conference on Image Analysis and Processing (ICIAP)* (pp. 460–465). doi:[10.1109/ICIAP.2003.1234093](https://doi.org/10.1109/ICIAP.2003.1234093).
- Rabiner, L. (1989). A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, *77*, 257–286. doi:[10.1109/5.18626](https://doi.org/10.1109/5.18626).
- Rong, J., Jin, W., & Wu, L. (2004). Key frame extraction using inter-shot information. In *IEEE International Conference on Multimedia and Expo (ICME '04)* (pp. 571–574 Vol.1). volume 1. doi:[10.1109/ICME.2004.1394256](https://doi.org/10.1109/ICME.2004.1394256).
- Schindler, G., Krishnamurthy, P., Lubliner, R., Liu, Y., & Dellaert, F. (2008). Detecting and matching repeated patterns for automatic geo-tagging in urban environments. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 1–7). doi:[10.1109/CVPR.2008.4587461](https://doi.org/10.1109/CVPR.2008.4587461).
- Sevillano, X., Valero, X., & Alias, F. (2012). Audio and video cues for geo-tagging online videos in the absence of metadata. In *International Workshop on Content-Based Multimedia Indexing (CBMI)* (pp. 1–6). doi:[10.1109/CBMI.2012.6269808](https://doi.org/10.1109/CBMI.2012.6269808).

- Shang, L., Yang, L., Wang, F., Chan, K.-P., & Hua, X.-S. (2010). Real-time large scale near-duplicate web video retrieval. In *Proceedings of the International Conference on Multimedia (MM '10)* (pp. 531–540). New York, NY, USA: ACM. doi:[10.1145/1873951.1874021](https://doi.org/10.1145/1873951.1874021).
- Siersdorfer, S., San Pedro, J., & Sanderson, M. (2009). Automatic video tagging using content redundancy. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '09)* (pp. 395–402). New York, NY, USA: ACM. doi:[10.1145/1571941.1572010](https://doi.org/10.1145/1571941.1572010).
- Smeaton, A. F., Over, P., & Doherty, A. R. (2010). Video shot boundary detection: Seven years of TRECVID activity. *Computer Vision and Image Understanding*, 114, 411 – 418. doi:[10.1016/j.cviu.2009.03.011](https://doi.org/10.1016/j.cviu.2009.03.011). Special issue on Image and Video Retrieval Evaluation.
- Smeaton, A. F., Over, P., & Kraaij, W. (2006). Evaluation campaigns and TRECVID. In *Proceedings of the 8th ACM International Workshop on Multimedia Information Retrieval (MIR '06)* (pp. 321–330). New York, NY, USA: ACM Press. doi:[10.1145/1178677.1178722](https://doi.org/10.1145/1178677.1178722).
- Soderberg, J., & Kakogianni, E. (2010). Automatic tag generation for photos using contextual information and description logics. In *International Workshop on Content-Based Multimedia Indexing (CBMI)* (pp. 1–7). doi:[10.1109/CBMI.2010.5529911](https://doi.org/10.1109/CBMI.2010.5529911).
- Su, J.-H., Huang, Y.-T., Yeh, H.-H., & Tseng, V. S. (2010). Effective content-based video retrieval using pattern-indexing and matching techniques. *Expert Systems with Applications*, 37, 5068 – 5085. doi:[10.1016/j.eswa.2009.12.003](https://doi.org/10.1016/j.eswa.2009.12.003).
- Tang, K., Sukthankar, R., Yagnik, J., & Fei-Fei, L. (2013). Discriminative segment annotation in weakly labeled video. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 2483–2490). doi:[10.1109/CVPR.2013.321](https://doi.org/10.1109/CVPR.2013.321).
- Ulges, A., Schulze, C., Keysers, D., & Breuel, T. (2008). Identifying relevant frames in weakly labeled videos for training concept detectors. In *Proceedings of the Inter-*

- national Conference on Content-based Image and Video Retrieval (CIVR '08)* (pp. 9–16). New York, NY, USA: ACM. doi:[10.1145/1386352.1386358](https://doi.org/10.1145/1386352.1386358).
- Wang, M., Hong, R., Li, G., Zha, Z.-J., Yan, S., & Chua, T.-S. (2012a). Event driven web video summarization by tag localization and key-shot identification. *IEEE Transactions on Multimedia*, *14*, 975–985. doi:[10.1109/TMM.2012.2185041](https://doi.org/10.1109/TMM.2012.2185041).
- Wang, X.-J., Zhang, L., Jing, F., & Ma, W.-Y. (2006). Annosearch: Image auto-annotation by search. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 1483–1490). Washington, DC, USA: IEEE Computer Society. doi:[10.1109/CVPR.2006.58](https://doi.org/10.1109/CVPR.2006.58).
- Wang, X.-J., Zhang, L., Liu, M., Li, Y., & Ma, W.-Y. (2010). Arista - image search to annotation on billions of web photos. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 2987–2994). doi:[10.1109/CVPR.2010.5540046](https://doi.org/10.1109/CVPR.2010.5540046).
- Wang, X.-J., Zhang, L., & Ma, W.-Y. (2012b). Duplicate-search-based image annotation using web-scale data. *Proceedings of the IEEE*, *100*, 2705–2721. doi:[10.1109/JPROC.2012.2193109](https://doi.org/10.1109/JPROC.2012.2193109).
- Weng, Y., & Jiang, J. (2011). Fast camera motion estimation in MPEG compressed domain. *IEEE Transactions on Consumer Electronics*, *57*, 1329–1335. doi:[10.1109/TCE.2011.6018891](https://doi.org/10.1109/TCE.2011.6018891).
- Wiegand, T., Sullivan, G. J., Bjontegaard, G., & Luthra, A. (2003). Overview of the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, *13*, 560–576. doi:[10.1109/TCSVT.2003.815165](https://doi.org/10.1109/TCSVT.2003.815165).
- Yuan, J., Wang, H., Xiao, L., Zheng, W., Li, J., Lin, F., & Zhang, B. (2007). A formal study of shot boundary detection. *IEEE Transactions on Circuits and Systems for Video Technology*, *17*, 168–186. doi:[10.1109/TCSVT.2006.888023](https://doi.org/10.1109/TCSVT.2006.888023).
- Zhang, H., Wu, J., Zhong, D., & Smoliar, S. (1997). An integrated system for content-based video retrieval and browsing. *Pattern Recognition*, (pp. 643–658).

Zhang, W., Lin, J., Chen, X., Huang, Q., & Liu, Y. (2006). Video shot detection using hidden markov models with complementary features. In *First International Conference on Innovative Computing, Information and Control (ICICIC '06)* (pp. 593–596). volume 3. doi:[10.1109/ICICIC.2006.549](https://doi.org/10.1109/ICICIC.2006.549).

Zhang, X.-D., Liu, T.-Y., Lo, K.-T., & Feng, J. (2003). Dynamic selection and effective compression of key frames for video abstraction. *Pattern Recognition Letters*, *24*, 1523–1532. doi:[10.1016/S0167-8655\(02\)00391-4](https://doi.org/10.1016/S0167-8655(02)00391-4).