A Robust Video Identification Framework using Perceptual Image Hashing

Francisco Vega*, José Medina†, Daniel Mendoza*, Víctor Saquicela*, and Mauricio Espinoza*
*Computer Science Department, University of Cuenca, Ecuador
{francisco.vegaz, daniel.mendoza, victor.saquicela, mauricio.espinoza}@ucuenca.edu.ec
†Department of Electrical, Electronic Engineering and Telecommunications, University of Cuenca, Ecuador
jose.medina@ucuenca.edu.ec

***Abstract*—This paper proposes a general framework that allows to identify a video in real time using perceptual image hashing algorithms. In order to evaluate the versatility and performance of the framework, it was coupled for a use case about ads tv monitoring. Four Perceptual Image Hashing (PIH) algorithms were subject to a benchmarking process in order to identify the best one for the use case. This process was focused on analyze differences in terms of discriminability (D), robustness (R), time processing (Tp) and efficiency (E). A truth table was used to obtain information about discriminability and robustness, while processing time was directly measured. An efficiency metric based on time processing and identification capacity was proposed. In general terms, DHASH and PHASH algorithms have higher identification capacities than AHASH and WHASH in order to identify a video using only one frame. Moreover, a progressive decrease in robustness with the increment of the Hamming distance is observed in all cases. However, in a specific case of tv monitoring where speed is critical, the processing time becomes the most discriminatory parameter for the selection of the algorithm. So, for this case, a particular type of PIH (Average Hash) is highlighted as the most efficient one among other techniques, reaching an accuracy of 100% and frame rates on processing average of 108 fps with a Hamming Distance of 1. At the end, the proposed framework has remarkable identification skills, and presents an efficient search. Furthermore, presents the steps to select the best algorithm and its more adequate parameters, according to the requirements of each particular case.**

***Index Terms*—video identification, perceptual image hashing**

I. Introduction

During the last years, the proliferation of multimedia content has grown enormously, mainly due to costs savings and ease of use of new tools for generating, producing, storing, distributing, and delivering digital content. However, this has complicated the management of multimedia files, especially in tasks such as: control of copyright infringements, search for content, or custom filtering. From all types of multimedia objects, videos and particularly the process used in their identification, is still one of the major research challenges. A reason is the amount of alterations and variations that can suffer these multimedia objects in their properties, such as resolution, format or codecs. Furthermore, the videos can suffer the adhesion of banners or logos within their original content.

In order to deal with these problems, many studies have focused their attention on the development and application of Perceptual Hashing techniques [1]–[4], which allow the identification of multimedia contents for obtaining short binary strings of robust characteristics present in the multimedia data [1]. These techniques are also known as *fingerprinting* or *content-based media identification*. The application of these techniques is much faster and more efficient than performing a direct comparison of the multimedia content, especially in the case of videos, since this type of multimedia objects is composed by a sequence of images (frames) and generally an audio track.

Most authors agree that Perceptual Hashing based algorithms must ensure the following properties: *robustness* to support distortions in the content, *fast extraction* to generate a hash representation from the multimedia content, *discrimination* to avoid collisions between hash values, *fast searching* to retrieve an element from a database, and *efficiency* to identify the required item. Considering these criteria, several algorithms have been developed that using a measure of similarity allow to compare the hash representation of a multimedia object with the hash representation of multimedia objects previously extracted and stored in a database. The aim of these algorithms is to establish the "equality" of the compared objects.

In a general way, the identification of videos based on Perceptual Hashing techniques can be classified into two main groups:

- *Based on audio fingerprinting [5]–[7].* This type of algorithm allows identifying a song or audio in general from a video, by extracting particular characteristics (fingerprints) of a fragment of the audio track, whether or not to a noisy environment.
- *Based on visual characteristics.* These algorithms are based on the extraction of the particularities of the multimedia object from its visual characteristics. Several algorithms of this type have been proposed in the literature, using techniques of diverse nature to obtain the required representation. According to [8] these algorithms can be classified into three groups:
  1) *Frame-by-frame video hashing [1], [3], [9].* This group applies in an individual way algorithms based

on Perceptual Image Hashing to every frame on the video, finally to combine them in an alone representation (hash) of every video.

2) *Key frame based video hashing [10], [11].* These algorithms identify key frames from the video, then to obtain from them the same particular hash as described in 1).
3) *Spatio-temporal video hashing [12], [13].* This technique uses a normalized vector representation of the video that includes temporal samplings and spatial resizing of the content of the whole video.

The main problems found in these approaches are: i) the lack of audio in some videos, which prevents from executing techniques based on audio fingerprint, ii) the degradation that suffers the algorithms based on visual characteristics (like 1 and 2) in case of frame dropping either intentional or by codec conversion, iii) the time of response both in the process of feature extraction and in the matching process, and iv) the excessive use of computational resources and the reduction of response speed as new videos are included in the database.

In our opinion, these problems can be mitigated, analyzing in depth the relation between the used techniques for storing the fingerprints and the used perceptual hash algorithms. This type of analysis has great relevance because the success of these algorithms is directly linked to the time of fingerprint extraction, the time of the search process, and to the efficiency of the used algorithm. All these characteristics have direct relation to the storage method used to enable the identification of videos.

Considering the above mentioned and emphasizing the challenge that supposes at present the identification of a video, the following question arises: is it possible to determine an approach that allows identifying a video in real time and that in addition consumes few resources?. This work helps to solve this question, by the definition of a framework that enables the identification of videos, using open source perceptual image hashing algorithms, emphasizing in the reduction of the processing time and the consumption of computational resources, without neglecting the efficiency.

The main contributions of this paper can be summarized as follows: i) to propose a framework that allows the robust identification of videos in real time using techniques of Perceptual Image Hashing, ii) to evaluate the importance of the interaction between the process of extraction and search of fingerprints, and iii) to evaluate the behavior of the proposed framework in a case study for television monitoring of advertising.

The rest of this paper is structured as follows. The conceptual basis of Perceptual Image Hashing techniques and the definition of the algorithms used in this work are shown in Section II. In Section III, the proposed framework to support video identification in real time using Perceptual Image Hashing algorithms is presented. Section IV introduces possible applications of the proposed framework. The benchmarking process used to compare different Perceptual Image Hashing algorithms is described in Section V. The results of benchmarking process are presented and interpreted in Section VI. The description of the use case implemented to evaluate the proposed framework is presented and discussed in the Section VII. The paper ends in Section VIII with the conclusions and the suggestions for future works.

II. Perceptual Image Hashing Algorithms

An image comparison algorithm of this type is based on the extraction of characteristics derived from the content of an image to form a representation in short binary strings, sufficiently flexible and robust enough to effectively discriminate similarities and differences between two images subject to alterations of diverse nature. This representation is obtained through hash functions, which are mathematical tools that allow the coupling of data of different sizes into a single fixed-sized representation. The required flexibility is obtained using a mathematical function known as Hamming Distance (Hd), which returns a result that denotes how many elements are different between two binary strings. This means that while closer to 0 is this value the greater the similarity between them. It should be considered that the hash representations obtained with these algorithms are not unique, although they are different.

The algorithms used in this study, were selected due to their ease of implementation and can be obtained free from the Python library called "Image Hash"[1]. All these methods obtain a fingerprint of 64-bit as a result. Generally, these values are presented and stored as hexadecimal values. The following is a brief description of the algorithms selected in this work, describing the steps used to execute them:

A. Average Hashing (AHASH).

As described in [14]the main idea of AHASH algorithms is to obtain the mean value of all the low frequencies of an image and then use it in the construction of the hash value. The steps to follow are: (1) image compression to a size of 8x8 pixels; (2) grayscale conversion; (3) obtaining the mean value of the image intensities; (4) comparison the 64 intensities of the image one by one with the mean value, one value of 1 is applied if the intensity is greater and 0 otherwise, and (5) building a 64 bits hash representation with the bits obtained.

B. Difference Hashing (DHASH).

According to [15], this technique is defined for obtaining a fingerprint (hash) from the number of different pixels and its gradient. The steps in this case are: (1) image conversion to a grayscale, (2) image reduction to a common size (here, 9x8 pixels), (3) comparison of each intensity with the adjacent values for each row –if the value is greater, a 1 is applied, otherwise 0–, and (4) obtain the representation with binary values.

[1]https://pypi.python.org/pypi/ImageHash

C. Perceptive Hashing (PHASH).

In [16], is mentioned that PHASH instead of using intensities to execute the process of comparison, uses a range of frequencies obtained from the technique Discrete Cosine Transform (DCT). So, the steps to follow for fingerprint extraction are: (1) image reduction to 32x32 pixels, (2) grayscale conversion, (3) DCT calculation –is to say the obtaining of a collection of frequencies and scalars represented by a square matrix–, (3) DCT coefficient (1,1) becomes the upper left corner, and DCT (8,8) becomes lower right corner, (4) calculation of average value of these values, (5) comparison of the 64 DCT coefficients with the mean value, should be placed 1 if the intensity is greater and 0 if it is smaller, and (6) construction of the hash representation with the bits obtained (64 bits).

D. Wavelet Hashing (WHASH).

In [17] the author describes that this technique is similar to PHASH. The main difference is the use of Discrete Wavelet Transformation (DWT) instead of DCT as technique of comparison. The steps for extraction are similar to those of PHASH.

The next section describes our approach to support video identification in real time using Perceptual Image Hashing algorithms.

III. General Architecture of Proposed Framework

The proposed video identification framework is shown in Figure 1. It adopts a client-server model. The main purpose of the components considered in the client-side is to save processing time to the server, by the previous extraction of each fingerprint frame for its query on the server. In addition, the client is responsible for managing the requests of video identification, and the answers obtained from the server. The client can be included in any companion device (e.g., a smart phone or a personal computer). However, logically it could be inside the same computer considered as server.

The communication channel between the server and the client can be established through a Web service, Socket or Rest service. The implementation of this channel and the most appropriate security method depend on the use case. It should be emphasized that the proposed framework is generic and should be tailored for the needs of each particular use case.

Server-side components are responsible for managing i) the data population process in order to add new elements to the database and ii) the search process with asynchronously managed requests. Regarding data layer, it is recommended that this component be included within the server to reduce the data transfer time on the network. Although, it could be placed inside an equipment exclusively prepared for data management, if necessary.

In the following subsections, the processes that define the server-side are described in detail:

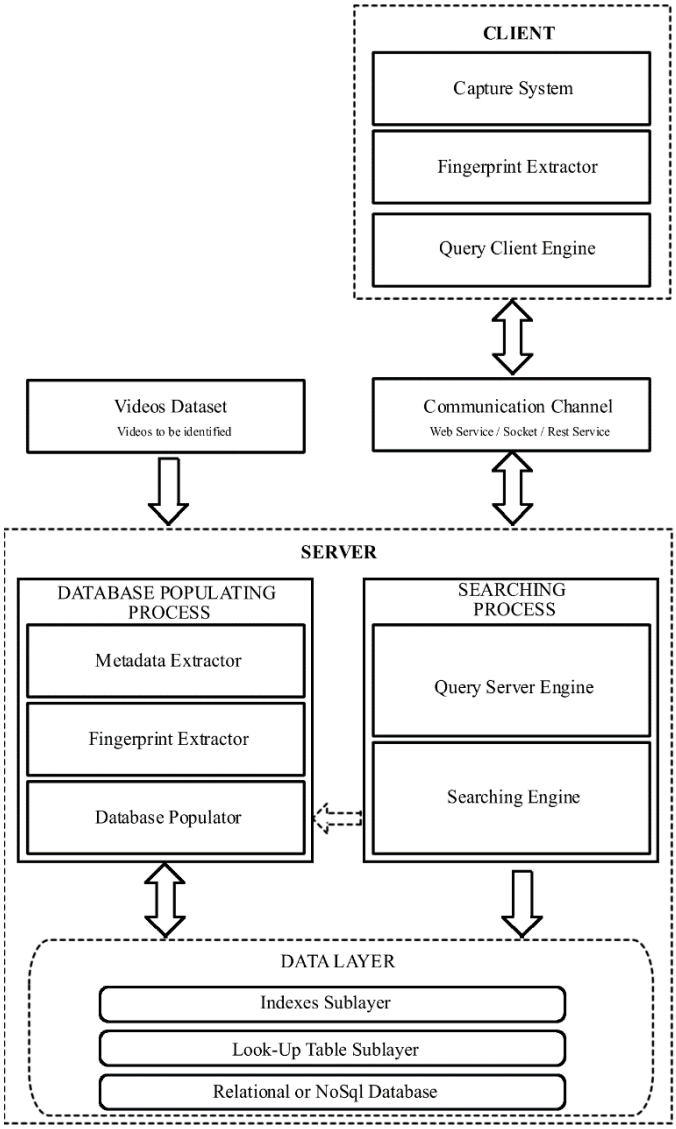

Fig. 1. General Architecture of Proposed Framework.

A. Database Population Process

Metadata Extractor: This component is responsible of obtaining the related information of a video, previous to be stored into the database. This information can be extracted from the metadata included in the multimedia file or using information provided by a user. Additional information can be added from the Web through the use of Web scrapping tools. The mandatory data for the correct operation of the framework are the video title and the number of image frames. The Metadata Extractor component obtains each of the frames that compose the image, previous to the fingerprint extraction. The FFMPEG library offers different tools for the execution of this task. Once metadata are collected, the video title is associated with a unique Id, and this information is stored in the database.

Fingerprint Extractor: The aim of this component is to obtain a hash representation from each of the frames processed in

the Metadata Extractor component. In spite of used perceptual hashing algorithm, the output must be a 64 bit hexadecimal hash. Of course, it is necessary to use the same algorithm on both the client-side and the server-side.

Database Populator: This component relies on four activities: (1) to give format to each one of video data obtained in the Metadata Extractor and Fingerprint Extractor, (2) to add the new information to a backup file, (3) to populate the data layer with the new elements, and (4) in case of server restart, to populate the data layer with the information kept in memory in the backup file.

B. Searching Process

The Query Server Engine and the Searching Engine are the key components of this module. These components allow to manage asynchronous queries on the server and to determine the existence or not of fingerprints in the data layer. The process has the following activities: (1) to accept requests from the client, (2) to manage the information to be sent as a response to the client, (3) to manage the search and filtering tasks within the data layer, and (4) generate logs with information about processing times, errors, security breach attempts, etc.

In the next sections we explain the associated components to data layer.

C. Data Layer

The framework core is the database structure. It consists of three major components: indexes sublayer, look up table sublayer, and database sublayer.

Indexes Sublayer

In order to optimize the search process, given the huge number of fingerprint elements that can be stored in the database, it is necessary to manage indexes and a look up table directly on RAM, as shown in Figure 2. However, for huge data sets it is recommended to use NoSql indexing tools. Indexes are no more than the same fingerprints obtained during the data population process. A Bk-Tree allows to keep in RAM the data in any moment. This approach can be very useful for small or medium scale databases. A Bk-Tree is a structure based on discrete metrics proposed by [18]. Originally, it emerged as a data structure that allows to identify the number of changes necessary to convert one word to another, this with the purpose of applying it in an orthographic corrector.

In the proposed framework, the distance chosen between the hexadecimal hash values is the Hamming distance. In [19], Hamming Distance (Hd) is described as a natural measure of similarity between two binary codes of equal size. In other words, this value represents the total number of binary differences found between two fingerprints. It should be emphasized that the fingerprints are the indices of Bk-Tree, so, these values are unique in the tree. The repetition of fingerprints is handled by the Look Up Table. The main reason for having these indexes in a Bk-Tree, outside of the Look Up Table indexes, is to improve the speed, when executing the search process and

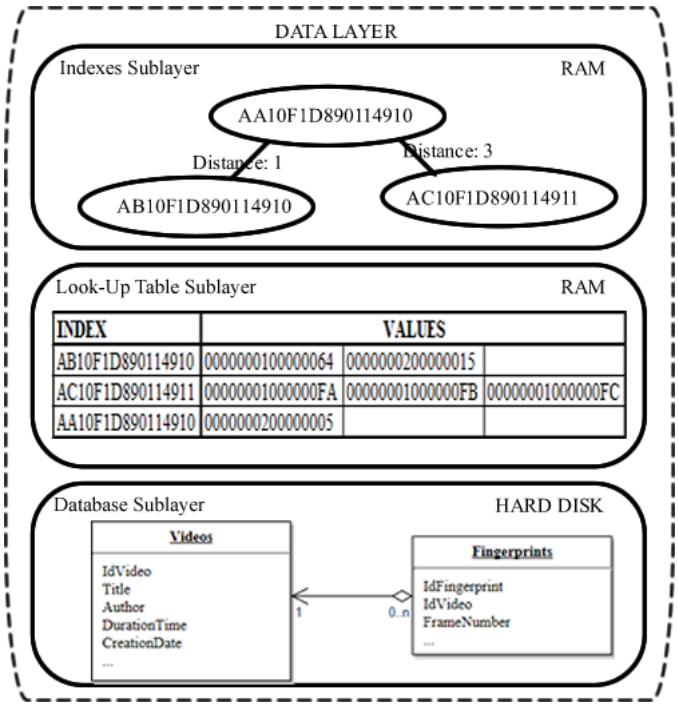

Fig. 2. Data Layer of the proposed framework.

the filtering process using different values of the Hamming Distance (Hd).

Look Up Table Sublayer

It is a data dictionary that allows having more than one column associated with an index. Each column presents a hexadecimal value of 64 bits. Where, the first 32 bits represent the video Id and the remaining bits represent the frame number of the video associated with the fingerprint. For specific scenarios, it is possible to use only the video Id, since, in the case of large data sets could be used few fingerprints of a shot to populate the database. A shot is the name that receives a fragment of a video scene, where several contiguous frames maintain a similar sequence.

An ambiguity can occur in two cases: (1) when a frame is very similar to another contiguous frame within the same video and, (2) when the same image appears in more than one video. For example, in commercials about car models belonging to a same brand are usual to place at the end of ad, some frames with the same logo. Ambiguities can be handled differently according to the use case. Thus, if the intention is to use the framework to identify a commercial from a screen capture, in the first case, the only commercial that is repeated can be returned as response. While, in the second case a brand can be returned as response and all commercials corresponding to that brand can be listed, or a new screen capture can be requested to the device until there is no ambiguity

Database Sublayer

Depending on the amount of data managed, a relational or NoSQL database can be used. The main goal of this

component is to store more detailed information about the video, without consuming the RAM of the equipment. The database structure clearly depends on the intended use, for example, in the case of television monitoring, it is necessary to add one or several tables that allow registering information such as channel, date, or detection time.

Some possible applications that can take advantage of the existence of a video identification algorithm that works in real time is described below.

IV. Possible Applications

The need for a video identification algorithm that responds in real time is mainly due to the great demand for novel and various applications that are currently being developed, for example:

- *Second Screen Synchronizing*: There consists of identifying a particular video reproduced by a main device and offering a set of exclusive services through the use of a companion device, known as second screen device [20]. Its applicability ranges as presented in [21], [22], from social messaging and offering audio for films in different languages, providing narratives for blind people; to present additional cameras at different angles of a sporting event.
- *TV Monitoring*: Continuing the analysis of [23]–[25], is denoted that the interest to develop and improved different techniques that allows identifying any TV content transmitted by a Broadcaster at any time and in real time, is currently latent. Once the content is identified, this type of techniques allows the display of detailed information about the television event, such as: the precise time and date of detection, source or source of the video, type of video, among others. Its benefits are so important that at present this activity demands intense research in order to meet the broad demand for needs such as: publicity monitoring or political advertising.
- *Extended Information for Ads*: The authors [24], [26], [27], proposes different systems that use a mobile device to obtain more information about a brand or product presented on television, in advertising banners, videos on the web, etc.
- *Video Semantic Enrichment*: The studies done by [26], [28], [29], present solutions to combine pattern recognition algorithms to identify objects and people within the video and semantic-based techniques to formally annotate objects.
- *Parental Control*: The authors [30], [31], highlight the importance of this set of applications to filter automatically videos not suitable for minors in real time, according to the characteristics and the content of each video.
- *Live Channel Detection*: It identifies a television channel that is being watched by a user in real time. It can be used by broadcasters to organize contests, surveys, etc. In [32], [33], are enhanced some steps to deal with the job of video identification.
- *Advertising Avoidance*: These systems allow identifying a commercial and blocking their presentation to users who do not wish to observe them. In addition, it could be used to customize the commercials sent to each user during the presentation of the advertising strip. In [34], [35], the advertising avoidance is treated from the point of view economic and statistical.
- *Audience Measurement*: The paper [36], presents the challenges of audience measurement and how the applications can help to identify automatically the videos observed by a user and to generate statistics such as the observation time before the user decides to change channels.
- *Personalized TV Recommender*: These applications use techniques of extraction and identification of videos to generate personalized recommendations of programming according to the likes of each viewer and recommendations of tv viewers [37], [38].
- *Content Based Video Retrieval*: This type of applications allows you to obtain videos similar to an image or sequence of images within a database. The works done by [39]–[41], deal with video indexing and retrieval.
- *Video Database Organization*: The applications in this group allow to organize and categorize videos according to their content. The great amount of the media produced currently makes this job very complicated and tedious to be done manually. In [42], are showed a solution to deal with this in an automatically manner.

V. Benchmarking Process

This section describes the benchmarking process that was carried out with the objective of comparing the four different Perceptual Image Hashing algorithms described in Section II. Two factors were considered in the evaluation: processing time (extraction and search) and identification capacity. The process was divided into three steps: (1) to determine the relative number of successes and failures in the identification of videos under different alterations (considering variations in quality and source of the video), (2) to evaluate the performance of the PIH algorithm (identified in section two) (3) to evaluate the performance of the framework considering computational conditions of low performance (1 processor core).

The machine used to run the evaluation is a PC with a 2.7Ghz i7 processor, 1600Mhz 8GB RAM, 500GB SATA hard drive and a 64bit Ubuntu 14.04LTS operating system.

The dataset used to populate the test database consists of 165 automobile commercial videos obtained from YouTube – a total of 246145 frames are used–. The videos are in MP4 format and encoded with H.264. The characteristics of the videos are variable, with resolutions ranging from 640x480 to 1280x1080 pixels; on the other hand, the frame rate goes from 24 to 30fps; and finally the duration of the videos has variations between 30 seconds to 5 minutes.

To begin the evaluation process, the database was populated with all the fingerprints of the videos in the dataset. However, in a real application, due to ambiguities case 1, the storage of

TABLE I
TRAINING/VALIDATION DATASET

Feature	**Parameters**					
Dataset	**Training**	**Validation**				
Video Format	MP4	FLV	FLV	MP4	OGV	OGV
Video Codec	H.264	H.264	X264	MPEG-4 Video	Theora	Theora
Resolution	variable	variable	shrinked	variable	static	variable
Min. Resolution	640x360	640x360	320x240	640x360	1280x720	640x360
Max. Resolution	1280x720	1280x720	320x240	1280x720	1280x720	1280x720
Frame Rate	24-30fps	24-30fps	24-30fps	24-30fps	24-30fps	24-30fps
Added Banners/Logos	No	No	No	Yes	Yes	No
Video Format		WEBM	MKV	WEBM	WEBM	MP4
Video Codec		VP8	FFMpeg v1	VP9	VP9	X265
Resolution		variable	variable	shrinked	shrinked	variable
Min. Resolution		640x360	640x360	320x240	240x120	640x360
Max. Resolution		1280x720	1280x720	320x240	240x120	1280x720
Frame Rate		24-30fps	24-30fps	24-30fps	24-30fps	24-30fps
Added Banners/Logos		No	No	Yes	No	No

very similar fingerprint frames could be omitted. Prior to the evaluation process, all videos were subject to changes, using the FFMPEG library, in codecs, format, resolution and quality; as shown in Table I. Getting 16500 videos with different characteristics to the originals, giving a total of 2461319 frames available to execute random queries. To these videos were added additionally 2182988 frames from 45 videos with similar variations to the previous videos and duration times of 2 to 8 minutes.

The benchmarking process used 30000 random frames of the 4644307 total frames in order to evaluate the identification process of the four PIH algorithms described in Section II. The results were interpreted as follows: i) a result is right when the platform returns a correct video Id (true positives) or it detects an video not existing in the database (true negatives) and ii) a result is incorrect when the platform returns ambiguities (as case 2) and erroneously detected videos (false positives) or when it not detects a video that is sure contained in the database (false negatives).

TABLE II
IDENTIFICATION PROCESSING TIME

Method	**Hamming Distance**						
	0	**1**	**2**	**3**	**4**	**5**	**6**
AHASH (0.0091)[a]	1.09[b]	,5.57[b]	21.25[b]	67.59[b]	163.65[b]	329.81[b]	635.42[b]
DHASH (0.0099)[a]	1.01[b]	7.91[b]	43.01[b]	162.52[b]	450.96[b]	971.01[b]	1786.99[b]
PHASH (0.0808)[a]	1.24[b]	1.23[b]	68.33[b]	68.97[b]	795.62[b]	806.65[b]	3104.11[b]
WHASH (0.0737)[a]	1.16[b]	1.94[b]	23.76[b]	26.91[b]	154.65[b]	166.56[b]	530.81[b]

[a] Fingerprinting Extraction Time (Te): average in seconds per frame.
[b] Searching Process Time (Ts): duration in seconds for the total number of samples.

The measured values are shown in Figure 3. The processing time (Tp) is the sum of the extraction time of fingerprint (Te) and the search process (Ts). These times are measured independently within the evaluation process, and their results are shown in Table II. The extraction time also considers the load from the hard disk of the frame to consult with the FFMPEG library. On the other hand, the Hamming Distance is subject to variations in the range of 0 to 6 units.

Comparative criteria

As mentioned in Section I and by different authors in [43]–[46], a suitable algorithm must ensure the following properties: robustness, discrimination, speed, and efficiency. To evaluate the four properties, the truth table of each algorithm was analyzed. The following metrics were designed for each property:

- Robustness: it is represented by the false negative rate (FN). This metric is inverse to the robustness parameter. In such a way, the smaller the FN the more robust the algorithm.
- Discrimination: It is the capacity of image recognition in each case. Its measure is directly related to SR, which involve the sum of TP and TN.
- Speed: This parameter is measured in two parts: (1) the speed of the fingerprint extraction, and (2) the search speed within the database.
- Efficiency: is the ability to identify the images in a shorter time. At first glance, the fastest and the most efficient algorithm should be selected. However, due to the inherent interaction between the type of PIH algorithm, the Hamming distance, and the search process, the preference for one of these parameters generally has an impact on the other. Therefore, it becomes necessary to find a metric that combines the processing time and the algorithm type, discriminating the compromise between both. To achieve this, an identifiable requirement must be set for each algorithm, and then under these conditions, evaluate the time with which it is possible to meet this requirement. In other words, if somehow the identification of a video under any algorithm could be assured in some degree, then the time that takes to each algorithm to guarantee that requirement would become an efficiency discriminator parameter. Fortunately, a probabilistic perspective (Bernoulli processes) can be considered in terms of video identification, in which, the number of successes corresponds to $SR \cdot (TN + TP)$, and the number of

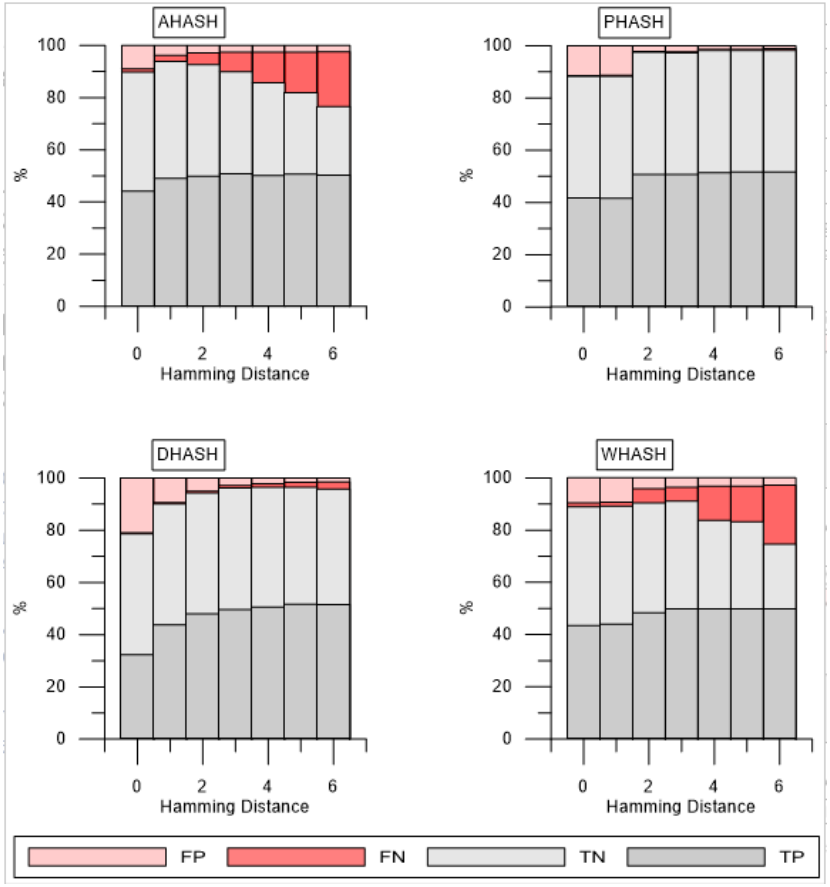

Fig. 3. Graphical representation of the True Table.

samples needed to obtain at least one success is the condition of the identifiable requirement.
If the probability is set at a high fraction ($P(X \geq x) = 0.9999$), then the probability for that within of a specified number of samples (N) be having at least one success in the identifying of videos will be:

$$P(X \geq x) = 1 - (1 - SR)^N$$

If we calculate N with the degree of assertion proposed, then each algorithm would require a different number of samples to identify a video (due to the differences between each algorithm). In this way, a uniform requirement is established to evaluate the algorithms. So, this number of samples must be multiplied by the processing time of each algorithm in order to find the proposed efficiency indicator E.

$$E = N \cdot (T_e + T_s)$$

This parameter is then interpreted as a reflection of the inherent interaction between the processing time (especially T_s), related to the type of algorithm and its Hamming Distance. The parameter searches a reconciliation between the recognition capability and the search time. At less time, the algorithm is more efficient because under the same requirement, it takes less time to identify a video.

VI. Results and Interpretation

As can be observed in Figure 3, the AHASH and WHASH algorithms are less robust than the DHASH and PHASH algorithms, according to the proportions of FN. Similarly, in general terms the latter algorithms have higher identification capacities than the other two. On the other hand, Figure 4 and Figure 5 indicate that faster and more efficient algorithm corresponds to AHASH, closely followed by the DHASH algorithm.

A progressive decrease in robustness with the increment of the Hamming distance is observed in all cases. However, this decrease is more accentuated in AHASH and less visible in PHASH. In addition, it is possible to notice a decrease of the discrimination in the AHASH and WHASH algorithms in relation to the Hamming distance. Moreover, the optimal point of the hamming distance for greater discrimination is higher in the DHASH (DH = 5) and PHASH (DH = 6) algorithms than in AHASH (DH = 1) and WHASH (DH = 3).

This variability of properties with respect to the parameters considered, and their interaction with each other indicates that the proposed framework must make use of the appropriate algorithm according to the particularity of the application. For

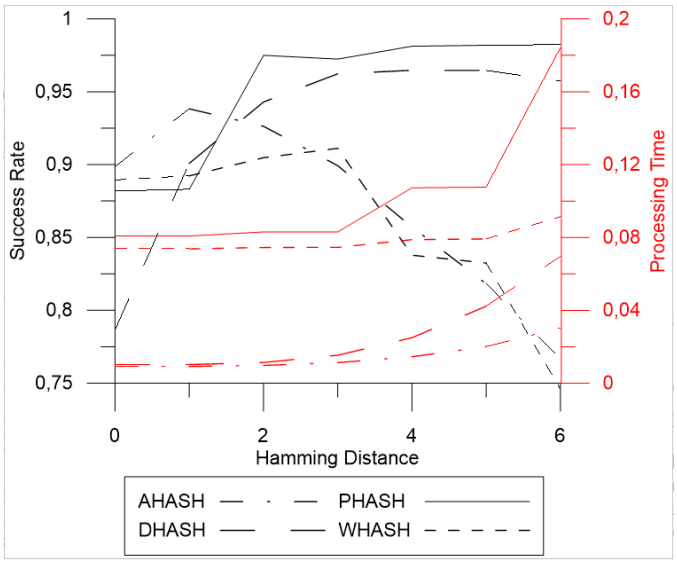

Fig. 4. Measures of Success Rate and Processing Time.

example, if the framework was applied in TV monitoring, where speed and efficiency are needed, then the Efficiency metric (E) should be taken into account (AHASH with Dh = 1 according to Figure 5). On the other hand, if it is required to recognize a video from a single sample, it is considered necessary to use an algorithm whose discriminability is higher than the others (PHASH with Dh = 6, according to Figure 3).

Continuing the analysis of the interactions between processing time, hamming distance and algorithm type, Figure 4 shows an increase in processing time with respect to the increase of Dh. This increase is due to the computational delay of the search process caused by the increase in permissiveness; That is, the greater HD, the greater the number of branches to be traversed by the Bk-Tree (this increase obeys an exponential form, according to Figure 4). Additionally, the time and RAM space consumed by each algorithm are closely linked to the number of indexes and to the distribution of similar fingerprints in both the tree and the Look Up Table. On the other hand, it can be seen in Table II, Table III and Figure 4 that while more singular they should be the fingerprints obtained in the process of extraction, slower will be the search process and the consumption in RAM of the tree of indexation will be greater. However, the detection capacity will be greater.

Another interesting factor, that can be seen in Figure 4 and Table III is the tolerance that appears to have both DHASH and PHASH as the Hd increases. This happens, because the number of indices representing the singularity of each algorithm is greater, and even when it is tolerant to variations, they remain highly robust. Otherwise, for the rest of algorithms, this robustness is altered for the remaining algorithms because, when trying to improve the tolerance, the Hamming distance is increased; however, by having fewer singular fingerprints and very close to each other, there is an exponential increase in the number of FN values. This type of effects shows the importance of the interactions between the search process and the PIH algorithm.

TABLE III
MEMORY CONSUMPTION

Method	RAM Memory Consumption			
	Number of Indexes[a]	Look Up Table (LUT)	LUT + Bk-Tree[b]	LUT + Bk-Tree-2[c]
AHASH	94367	266.3 MB	1.2 GB	539.6 MB
DHASH	139027	121.1 MB	1.5 GB	274.2 MB
PHASH	129479	121.6 MB	1.4 GB	269.9 MB
WHASH	92114	113.8 MB	1017.6 MB	246 MB

[a] Number of indexes obtained from the 246145 fingerprints belonging to the populated database of the benchmarking process.
[b] Indexes + Look Up Table, employing a Bk-Tree implemented in C (used for the benchmarking process) faster but with more RAM consumption.
[c] Indexes + Look Up Table, employing a Bk-Tree implemented with lists and dictionaries in Python (slower in an average of 35% of Ts but with less consumption of RAM).

Finally, the parameter E can be interpreted as an indicator of the compromise between the identification capacity, and the processing time required by each algorithm. For example, in Figure 5 the number of samples (N) required to reach a 99.99% recognition effectiveness threshold ($P(X \geq x)$) is generally lower in the PHASH algorithm; However, its computational cost of extraction and search causes that the efficiency parameter to be affected (see Figure 4, Table II).

In contrast, the AHASH algorithm has high efficiencies due to its low E. This is justified by the optimal compromise achieved between the N number required for a 99.99% identification capacity and its processing time. In particular, the AHASH algorithm with Hd=1 will be used in the use case about television monitoring.

VII. USE CASE: TV MONITORING

The main objective of this section is to evaluate the behavior of the conditioned framework for the use case about Ads TV monitoring. The monitored videos are from the same database used in the benchmarking process. For the evaluation, 30 random combinations of 10 videos were generated in total.

Considering the results obtained in the benchmarking process, the AHASH algorithm with a Hamming Distance of 1 was used. The following considerations were added to the framework for this use case: (1) to assume that any TV commercial was less than 4 seconds, to avoid possible ambiguities between commercials (2) to use one sample every second, because N in the efficiency parameter denotes that it is highly unlikely that any of the samples be true. In other words, at least one of the samples will be true within the minimum 4 seconds of duration; and (3) the real time is simulated using the FFMPEG library.

The restriction of the 4 seconds is given because most commercials add frames with black background before and after each video, however, it is the opinion among the authors of this work that this restriction does not affect the use case. In a real scenario, there are very few video ads with a duration less than 4 seconds. In addition, in order to perform a comparative analysis, the DHASH algorithm with Hd of 1 was

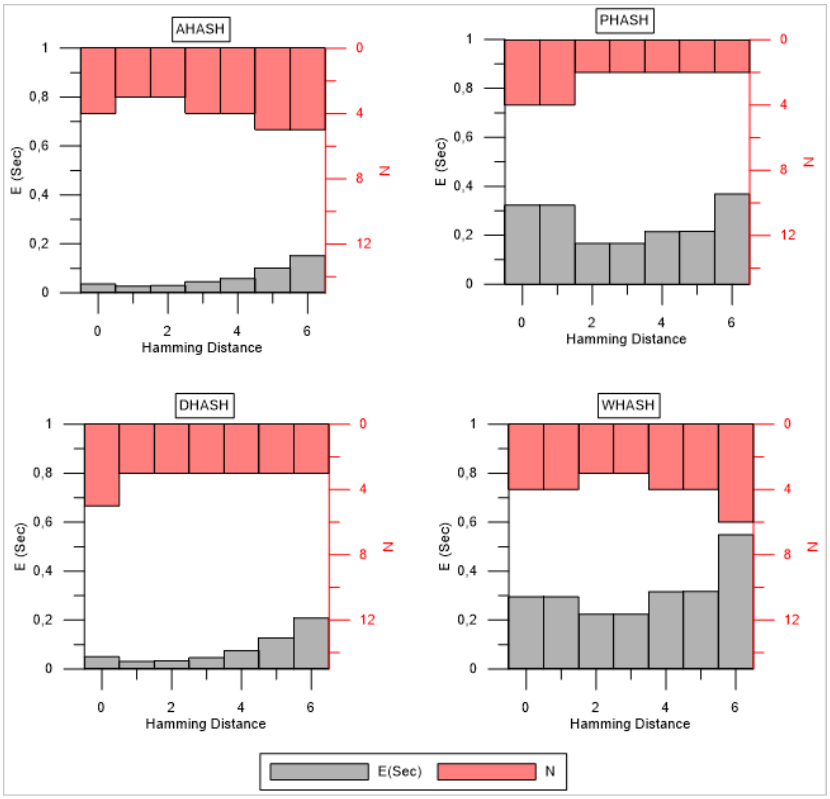

Fig. 5. Graphical representation of the Efficiency Parameter.

used due to its similar efficiency with respect to the selected algorithm (Figure 3).

The results show that the two algorithms evaluated with the previously mentioned considerations are highly efficient identifying the existence or not of the video in 100% of the cases. Moreover, the proposed framework can identify the sequential order within each set of videos.

Taking into account that the results are invariant to the algorithms used, in terms of identification efficiency (100%); then, the efficiency in terms of processing time becomes the discriminatory parameter in the selection of the algorithm. Obtaining in the experimentation, frame rates on average of 108 fps and 98fps for the case of AHASH and DHASH and Hamming Distance 1 respectively.

VIII. Conclusions and Future Work

Based on the experimentation realized for the use case of television monitoring, the following can be concluded:

An important relationship between the search process and the used algorithm type was evidenced in the benchmarking process. However, it is possible to discriminate by means objective metrics a particular type of algorithm that rationally compromises the processing time and the identification capacity.

The benchmarking process is a fundamental step that must be performed prior to the application of the framework for a particular case. Because this stage reveals the optimal combination between the algorithms to be selected, the Hamming Distance to be used and the conditions to which the system must be subject, according to the different needs and limitations of the application considered.

At the end, it is proved that the proposed framework works for the identification of videos in real time with the help of the FFMPEG library, although in order to optimize the process, a joint evaluation of the behavior of each algorithm both in the extraction process and in the search process must be performed. Thus, the type of algorithm that will be part of the framework will depend on the required application type, and its efficiency will be highly related to the relationship between the algorithm, the extraction time and the search time.

In the future, it will be required to perform a detailed evaluation of the application framework for use cases that need to identify videos in large datasets. In the same way, it is necessary to deep the study of the mentioned considerations in the case of television monitoring and its conditioning within the proposed framework, in order to optimize the number of samples necessary to validate the identification of a video.

Acknowledgments

The work presented in this article is part of a research project called “Use of semantic technologies to analyze multimedia content broadcasted by digital television” supported by the Department of Research of the University of Cuenca.

REFERENCES

[1] L. Weng and B. Preneel, "From image hashing to video hashing," in *International Conference on Multimedia Modeling*. Springer, 2010, pp. 662–668.

[2] V. J. Kulkarni, S. Satav, and D. Patil, "Survey on cloud-based multimedia content protection," *International Journal of Engineering Science*, vol. 4004, 2017.

[3] R. Sun, X. Yan, and J. Gao, "Robust video fingerprinting scheme based on contourlet hidden markov tree model," *Optik-International Journal for Light and Electron Optics*, vol. 128, pp. 139–147, 2017.

[4] R. Sandeep, S. Sharma, M. Thakur, and P. Bora, "Perceptual video hashing based on tucker decomposition with application to indexing and retrieval of near-identical videos," *Multimedia Tools and Applications*, vol. 75, no. 13, pp. 7779–7797, 2016.

[5] C. V. Cotton and D. P. Ellis, "Audio fingerprinting to identify multiple videos of an event," in *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*. IEEE, 2010, pp. 2386–2389.

[6] J. Haitsma and T. Kalker, "A highly robust audio fingerprinting system." in *Ismir*, vol. 2002, 2002, pp. 107–115.

[7] J. Medina, F. Vega, D. Mendoza, V. Saquicela, and M. Espinoza, "Audio fingerprint parametrization for multimedia advertising identification," 2017, submitted to Second Ecuador Technical Chapters Meetings (ETCM 2017).

[8] C.-C. Wang, J.-S. R. Jang, and W. Liou, "Speeding up audio fingerprinting over gpus," in *Audio, Language and Image Processing (ICALIP), 2014 International Conference on*. IEEE, 2014, pp. 5–10.

[9] B. Wu, S. S. Krishnan, N. Zhang, and L. Su, "Compact and robust video fingerprinting using sparse represented features," in *Multimedia and Expo (ICME), 2016 IEEE International Conference on*. IEEE, 2016, pp. 1–6.

[10] X. Nie, J. Sun, Z. Xing, and X. Liu, "Video fingerprinting based on graph model," *Multimedia tools and applications*, vol. 69, no. 2, pp. 429–442, 2014.

[11] J. Mao, G. Xiao, W. Sheng, Y. Hu, and Z. Qu, "A method for video authenticity based on the fingerprint of scene frame," *Neurocomputing*, vol. 173, pp. 2022–2032, 2016.

[12] C. Ma, Y. Gu, W. Liu, J. Yang, and X. He, "Unsupervised video hashing by exploiting spatio-temporal feature," in *International Conference on Neural Information Processing*. Springer, 2016, pp. 511–518.

[13] V. E. Liong, J. Lu, Y.-P. Tan, and J. Zhou, "Deep video hashing," *IEEE Transactions on Multimedia*, 2016.

[14] S. F. C. Haviana and D. Kurniadi, "Average hashing for perceptual image similarity in mobile phone application," *Journal of Telematics and Informatics*, vol. 4, no. 1, pp. 12–18, 2016.

[15] V. Bajaja, S. Keluskar, R. Jaisawala, and R. Sawant, "Plagiarism detection of images," *International Journal of Innovative and Emerging Research in Engineering*, vol. 2, pp. 140–144, 2015.

[16] C. Zauner, M. Steinebach, and E. Hermann, "Rihamark: perceptual image hash benchmarking," in *IS&T/SPIE Electronic Imaging*. International Society for Optics and Photonics, 2011, pp. 78 800X–78 800X.

[17] D. Petrov. Wavelethash. [Online]. Available: https://fullstackml.com/wavelet-image-hash-in-python-3504fdd282b5

[18] W. A. Burkhard and R. M. Keller, "Some approaches to best-match file searching," *Communications of the ACM*, vol. 16, no. 4, pp. 230–236, 1973.

[19] M. Norouzi, D. J. Fleet, and R. R. Salakhutdinov, "Hamming distance metric learning," in *Advances in neural information processing systems*, 2012, pp. 1061–1069.

[20] D. Campelo, T. Silva, and J. F. de Abreu, "Beyond the tv borders: Second screen as a tool for audience engagement," in *Iberoamerican Conference on Applications and Usability of Interactive TV*. Springer, 2016, pp. 93–104.

[21] P. Lindgren and T. Olsson, "How true, synchronized live ott can change the second screen and social tv game," Ib 365, Tech. Rep., 2016.

[22] S. Ka, T. H. Kim, J. Y. Ha, S. H. Lim, S. C. Shin, J. W. Choi, C. Kwak, and S. Choi, "Near-ultrasound communication for tv's 2nd screen services," in *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*. ACM, 2016, pp. 42–54.

[23] A. Znotiņš, K. Polis, and R. Darģis, "Media monitoring system for latvian radio and tv broadcasts," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[24] L. Pang, X. Li, J. Chai, and Y. Liu, "A high performance video frame extractor for omni media monitoring system," in *Signal Processing (ICSP), 2014 12th International Conference on*. IEEE, 2014, pp. 399–402.

[25] A. K. Singh, "Improved hybrid algorithm for robust and imperceptible multiple watermarking using digital images," *Multimedia Tools and Applications*, vol. 76, no. 6, pp. 8881–8900, 2017.

[26] F. Vega, J. Medina, D. Mendoza, V. Saquicela, and M. Espinoza, "Towards a multi-screen interactive ad delivery platform," in *43th Latin American Computing Conference, Lat. Am. Symp. on Computer Graphics, Virtual Reality, and Image Processing*, 2017.

[27] D. Aversano, D. Jacobs, P. Marsh, and H. Shalhoub, "Sync apps: Leveraging the power of second screen apps for tv advertising," in *Proceedings of the ARF Re: Think Recap Conference*, 2014.

[28] M. Krug, F. Wiedemann, and M. Gaedke, "Enhancing media enrichment by semantic extraction," in *Proceedings of the 23rd International Conference on World Wide Web*. ACM, 2014, pp. 111–114.

[29] T. Kurz, G. Güntner, V. Damjanovic, S. Schaffert, and M. Fernandez, "Semantic enhancement for media asset management systems," *Multimedia tools and applications*, vol. 70, no. 2, pp. 949–975, 2014.

[30] J. Wehrmann, G. S. Simões, R. C. Barros, and V. F. Cavalcante, "Adult content detection in videos with convolutional and recurrent neural networks," *Neurocomputing*, 2017.

[31] Z. Dong, J. Qin, and Y. Wang, "Multi-stream deep networks for person to person violence detection in videos," in *Chinese Conference on Pattern Recognition*. Springer, 2016, pp. 517–531.

[32] K. Rudman, M. Bonenfant, M. Celik, J. Daniel, J. Haitsma, and J.-P. Panis, "Toward real-time detection of forensic watermarks to combat piracy by live streaming," *SMPTE Motion Imaging Journal*, vol. 125, no. 1, pp. 34–41, 2016.

[33] Y. Volovich, A. Oztaskent, and S. Rowe, "Increasing ad detection reliability by learning per-channel templates and transformation," Technical Disclosure Commons, Tech. Rep., 2017.

[34] K. C. Wilbur, "Advertising content and television advertising avoidance," *Journal of Media Economics*, vol. 29, no. 2, pp. 51–72, 2016.

[35] S. R. Dix and I. Phau, "Predictors of commercial zapping during live prime-time television," *Journal of Advertising Research*, vol. 57, no. 1, pp. 15–27, 2017.

[36] J. Kelly, "Television by the numbers: The challenges of audience measurement in the age of big data," *Convergence*, p. 1354856517700854, 2017.

[37] F. Aisopos, A. Valsamis, A. Psychas, A. Menychtas, and T. Varvarigou, "Efficient context management and personalized user recommendations in a smart social tv environment," in *International Conference on the Economics of Grids, Clouds, Systems, and Services*. Springer, 2016, pp. 102–114.

[38] Y. Park, J. Oh, and H. Yu, "Rectime: Real-time recommender system for online broadcasting," *Information Sciences*, vol. 409, pp. 1–16, 2017.

[39] N. Brindha and P. Visalakshi, "Bridging semantic gap between high-level and low-level features in content-based video retrieval using multi-stage esn–svm classifier," *Sādhanā*, vol. 42, no. 1, pp. 1–10, 2017.

[40] H. Bhaumik, S. Bhattacharyya, M. D. Nath, and S. Chakraborty, "Hybrid soft computing approaches to content based video retrieval: A brief review," *Applied Soft Computing*, vol. 46, pp. 1008–1029, 2016.

[41] L. Jiang, S.-I. Yu, D. Meng, T. Mitamura, and A. G. Hauptmann, "Text-to-video: a semantic search engine for internet videos," *International Journal of Multimedia Information Retrieval*, vol. 5, no. 1, pp. 3–18, 2016.

[42] R. R. Iyer, S. Parekh, V. Mohandoss, A. Ramsurat, B. Raj, and R. Singh, "Content-based video indexing and retrieval using corr-lda," *arXiv preprint arXiv:1602.08581*, 2016.

[43] X. Su, T. Huang, and W. Gao, "Robust video fingerprinting based on visual attention regions," in *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on*. IEEE, 2009, pp. 1525–1528.

[44] J. Lu, "Video fingerprinting for copy identification: from research to industry applications." in *Media Forensics and Security*, vol. 7254, 2009.

[45] S. Lee and C. D. Yoo, "Robust video fingerprinting for content-based video identification," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 18, no. 7, pp. 983–988, 2008.

[46] J. Oostveen, T. Kalker, and J. Haitsma, "Feature extraction and a database strategy for video fingerprinting," in *International Conference on Advances in Visual Information Systems*. Springer, 2002, pp. 117–128.