Index picture selection for automatically divided video segments

Gábor Szűcs
Inter-University Centre for Telecommunications and Informatics H-4028 Kassai út 26., Debrecen; and Department of Telecommunications and Media Informatics, BME, Hungary,
e-mail: szucs@tmit.bme.hu.

Abstract**—**The methods described in this paper are capable of analyzing and processing videos without any meta-information to divide continuous videos into segments without human interaction, and to select index pictures from segments. In the automatic video segmentation procedure pictures are sampled, the differences between them are measured, and 1-dimensional clustering – as contribution of the paper – is used to filter out the non-adequate segment border candidates in the segmentation. Further contribution of the paper is the construction of two factors of similarity among pictures, the common similarity indicator taking the semantic information into consideration, and the last step of the index picture selection: the adaptation of k-means++ clustering for the similarity-based picture set, where the distances among images are not available.

Keywords**—** k-means++ clustering, index picture, similarity, video segmentation.

I. Introduction

NOWADAYS, there is an oversupply of multimedia content, such as video, which is one of the most significant type. A great amount of video content can be found online as well as on devices of almost every user. It is a challenge to handle, index, sort of these contents without manual human help. The indexing of videos is particularly based on parts, so called segments of them, so the focus of this paper is automatic segmentation without any meta-information and the index picture selection of segments.

There are some MPEG-7-compatible systems, like BilVideo-7 [19][18] to support queries in a video indexing and retrieval framework. If a video possessing MPEG-7 metadata, then these can be used in retrieval, so the aim of indexing is to help the search, and the goal is to automatize the indexing procedure.

Similar to categories of image search [29] three broad categories of video search can be defined: (1) search by association, where there is no clear intent at a video, but instead the search proceeds by iteratively refined browsing; (2) aimed search, where a specific video is sought; and (3) category search, where a single video or index picture representative of a semantic class is sought.

The first and the third categories of video search are on focus of this paper, video segmentation and selection of semantic index pictures from each segment could help in both categories (in browsing and category search as well).

A video contains audio and series of images, so before planning an indexing procedure of multimedia content, like video, the audio and image indexing should be reviewed. There are some audio indexing solutions [26][37], where more types of typical web audio content, podcasts, video clips, and online lectures are handled for phrase spotting and relevance ranking.

An image indexing can be hierarchical regarding the different resolution of the images. In this hierarchical system the search starts in the subspace with the lowest resolution of the images, where set of all possible similar images is determined. In the next subspaces with higher resolution the retrieved set can be reduced by new similarity comparisons, and this process can be repeated until the similar images can be found [2]; but this indexing is not a semantic one.

For a long while there is a need for semantic video indexing at handling video collections [1]. Within a semantic index hierarchy five different levels can be defined [4]. The first three levels (purpose, genre, sub-genre) are related to the video document as a whole, the next levels (logical unit, named event) are related to parts of the content:

1) Purpose is defined as set of video documents sharing similar intention. In this high level “entertainment”, “information”, and “communication” can be used for purpose.
2) Genre is defined as set of video documents sharing similar style. There are some examples for genre: films, sports, news broadcasts, or commercials.
3) Sub-genre is defined as a subset of a genre where the video documents share similar content, e.g. in genre of film may be romantic, horror, science fiction, or drama films, and in genre of sport may be football, basketball, or tennis as sub-genre.
4) Logical unit is defined as continuous part of a video document’s content consisting of a set of named events or other logical units which together have a meaning. E.g. a film can contain different logical units: dialogues and actions, etc.

5) Named event is defined as short segment which can be assigned a meaning that doesn't change in time. E.g. action logical unit may be car chase, karate, shooting, etc.

Finding the interesting or representative images [9][16][32] from collections is a sorting task, in which the indexing is also important.

Approach of generic multimedia indexing (GEMINI) [3] is to find a feature extraction function that maps the high dimensional objects into a low dimensional space. Objects that are very different (dissimilar) in the low dimensional space are expected to be very dissimilar in the original space. In this way the non-qualifying objects can be discarded in low dimensional space.

There are some other general frameworks for indexing, e.g. Windsurf [30] library for management of multimedia hierarchical data. The library provides a general framework for assessing the performance of alternative query processing techniques for efficient retrieval of complex data that arise in several multimedia applications, such as image/video retrieval and the comparison of collection of documents. The library includes characteristics of generality, flexibility, and extensibility: these are provided by way of a number of different templates that can be appropriately instantiated in order to realize the particular retrieval model needed by the user. Another prototype tool is VIVO (Video Indexing and Visualization Organizer) [21] which has been developed to help digital video librarians to input, edit and manage video metadata elements on different levels. It is also part of the work in NSF-funded Open Video Digital Library [23]. In video indexing literature the multimodal approaches [4], (e.g. TIME - Time Interval Multimedia Event framework, which handles context and synchronization of video [5]) outperforms the unimodal solutions.

II. Automatic video segmentation

A. Possible video segmentation techniques

Automatic video segmentation techniques consider the sample pictures from video; the simple methods are based on two consecutive pictures p_1 and p_2. There are some elaborated video segmentation algorithms, e.g. for MPEG-4 camera system with change detection, background registration techniques and real time adaptive threshold techniques [34].

A possible solution to find boundaries among segments is the difference measurement between pictures p_1 and p_2. The first step to measure it is the comparison of pixel pairs of grayscale pictures with same size. We can count the number of pixels, where the difference is larger than a predefined threshold (ε), as can be seen in (1), and ratio of difference can be defined as in (2), where $p_i(x,y)$ is the grayscaled value of the pixel on the point (x,y) of the picture *i*

$$\mathrm{diffArea} = \sum_{x}\sum_{y} 1 \text{ if } \left|p_1(x,y) - p_2(x,y)\right| > \varepsilon \qquad (1)$$

$$\mathrm{diffRatio} = \frac{\mathrm{diffArea}}{\sum_{x}\sum_{y} 1} = \frac{\mathrm{diffArea}}{x \cdot y} \qquad (2)$$

Another possible video segmentation technique is histogram based segmentation. A histogram can be constructed for every picture based on the number of a given color value of pixels. The histograms of two consecutive pictures can be compared. In a video segment these histograms will be similar to each other, even if dynamic video, because the histograms are independent from the local place of moving objects in the picture. So the objects can move or the camera can move, the histogram based segmentation will find the connected pictures in a segment and task is to find the boundaries among the segments.

Let *H(p,k)* be the histogram value at *k* in a picture *p*, so the *H(p,k)* shows the number of pixels in picture *p* that possessing pixel intensity value: *k*. The value k can be from 1 to K, where K is the number of possible different pixel intensity values. If the picture is grey-scaled, then there is only one K, but at color pictures three K numbers are defined for each color component.

Definition of difference between two histograms is can be presented in (3).

$$diff_H = \sum_{j=1}^{K} \left| H(p_1, j) - H(p_2, j) \right| \qquad (3)$$

Another definition takes the difference color weights into consideration; because it can occur, that a color is dominant, so in this case this color could get larger weight in the comparison.

$$diff_H = \frac{r}{s} diff_H(p_1,p_2)^{(red)} + \frac{g}{s} diff_H(p_1,p_2)^{(green)} + \\ + \frac{b}{s} diff_H(p_1,p_2)^{(blue)} \qquad (4)$$

The definition of the difference is the sum of three differences as can be seen in (4), where *r*, *g*, *b* are the aggregated luminance values of the whole picture for red, green and blue respectively, and *s* is the average of *r*, *g*, *b* values.

Comparison of two histograms can be based on intersect of the charts, so using minimum operator (5) defines similarity between two pictures, from which the difference can be declared.

$$sim_H = \sum_{j=1}^{K} \min(H(p_1, j), H(p_2, j)) \qquad (5)$$

The difference can be accounted emphatically for as can be seen in (6).

$$diff_H = \sum_{j=1}^{K} \frac{(H(p_1, j) - H(p_2, j))^2}{\max(H(p_1, j), H(p_2, j))} \tag{6}$$

A sophisticated difference is the Bhattacharyya distance, which measures the differences of two discrete or continuous probability distributions, this can be seen in (7) for histograms [12].

$$diff_H = \sqrt{1 - \sum_{j=1}^{K} \frac{\sqrt{(H(p_1, j) \cdot H(p_2, j))}}{\sqrt{\sum_{j=1}^{K}(H(p_1, j) \cdot \sum_{j=1}^{K}(H(p_2, j)}}} \tag{7}$$

B. Applied video segmentation technique

In this paper a new idea is applied for the video segmentation technique. The first part of the segmentation procedure is usual as in other works: analysis of the sampled pictures from the video, measurement of the difference between them, and comparing it to a predefined threshold. The new part is the 1-dimensional clustering, which is able to help to refine the inner results. The block diagram of the constructed procedure can be seen in Fig. 1, where the input is the continuous video content and the output are the segments, the larger steps of segmentation can be seen in the middle rectangle. The idea of 1-dimensional clustering aims the filtering of non-adequate candidates in the segmentation, which is presented in more details later.

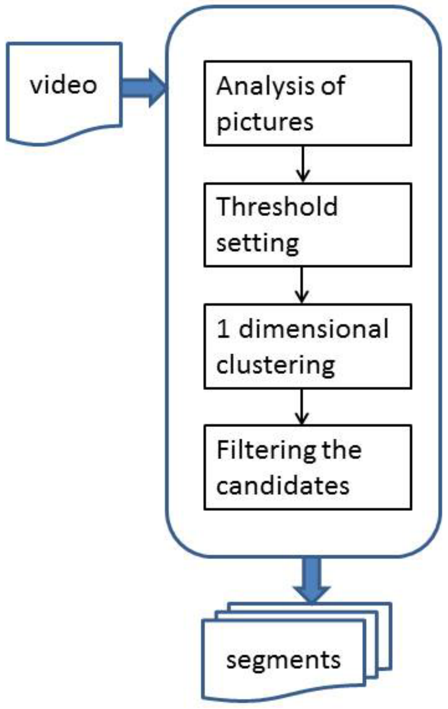


Fig. 1 Block diagram of elaborated video segmentation

C. Shot-boundary candidates

In this paper the following formula (8) has been used for measurement the difference of two consecutive pictures p_1 and p_2.

$$\text{diff}_{\text{color}} = \sum_x \sum_y |p_1(x, y) - p_2(x, y)| \tag{8}$$

where color = r, g, b. In this case three values represent the difference between pictures. A simple aggregated indicator for measure the difference is the sum of these values.

There is other possibility to define the difference between two pictures: instead of colored image the grayscale pictures should be used. So the images should be converted into grayscale, and the difference between them can be calculated, as can be seen in (8), but only 1 value (*diff*) will show the result.

If *diff* is greater than a threshold, then the second picture will be candidate for shot-boundary. We can define shot-boundary candidates in alternative way as well: if the total sum of pixel intensities in picture 1 is greater than zero, then the ratio of *diff* and *total sum* can be calculated and can be compared a *ratio threshold* as can be seen in (9).

$$\frac{\text{diff}}{\sum_x \sum_y |p_1(x, y)|} > \varepsilon \tag{9}$$

D. Final shot-boundaries by 1-dimensional clustering

A shot-boundary candidate possesses a point of time, and investigating these values it can be drawn, that too close neighboring candidates determine too brief segment, which can not be real shot. In order to eliminate inappropriate candidates a 1-dimensional neighborhood-based clustering method is constructed. The clustering algorithm is the following:

```
1. Let $D_n$ is the neighborhood limit.
2. Let $T_{SBCi}$ is the i^th shot-boundary candidate.
3. Create a cluster for $T_{SBC1}$ and put it to the cluster.
4. For i = 2 to Number_of_shot-boundary candidates
{
    5. If $T_{SBCi} - T_{SBCi-1} < D_n$, then put them into common cluster,
    6. else create a new cluster for $T_{SBCi}$ and put it to this new cluster.
}
```

After this clustering the median of each cluster is selected as representative element in the cluster, other shot-boundary candidates are eliminated. Based on final shot-boundaries the video is divided into segments, and the next phase is the semantic index picture selection from each video segment.

III. INDEX PICTURE SELECTION

A. Previous works for selecting representative picture

There are some works have been proposed for index picture selection from video. The works in [35][33]employ the spatial, color, and motion information to select index picture in the video segment. DuFaux [10] has used high level features such as face detection into this selection process. Another work in [20] has proposed to use the thematic criteria to order the sampled pictures for index pictures selection.

A recent work [17] has elaborated a personalized approach to dynamically generate the web video thumbnails according to user’s query, which not only consider the video visual content, but also meet the requirement of the user.

Although some approaches [35][33][10], have been proposed for index picture selection in videos, none of them addresses a robust, quick and general semantic aspect.

Our previous paper [13] has been concerned with managing image album, where the picture set in each album has been given. The goal has been to select the most representative pictures from the album, and the solution has been based on the clustering of the images. The central pictures of the largest clusters have been selected for representing the album. In picture selection solution the pixel information (statistics of RGB values of the picture points: mean, variance, mode, range, quartiles) and metadata features (EXIF data of pictures) has been taken into consideration, but semantic information has not been taken into account.

B. Steps of index picture selection procedure

In this task (index picture selection procedure in video segments) there are no metadata features, so only content information is available.

The steps of overall procedure are feature extraction, finding common key points, common similarity calculation from two types of similarity indicators, clustering using k-means++ method, and selection the most appropriate index pictures as can be seen in Fig 2.

The task of feature extraction is to extract the emphatic information (in this case key points with serial of determinative values) from image. After finding common key points, similarity could be declared between two images. In the overall procedure an index picture could be selected based on similarities among the pictures, but the solution described in this paper contains a previous step: clustering for improvement the index picture selection solution. The next sub-sessions will explain these in more details.

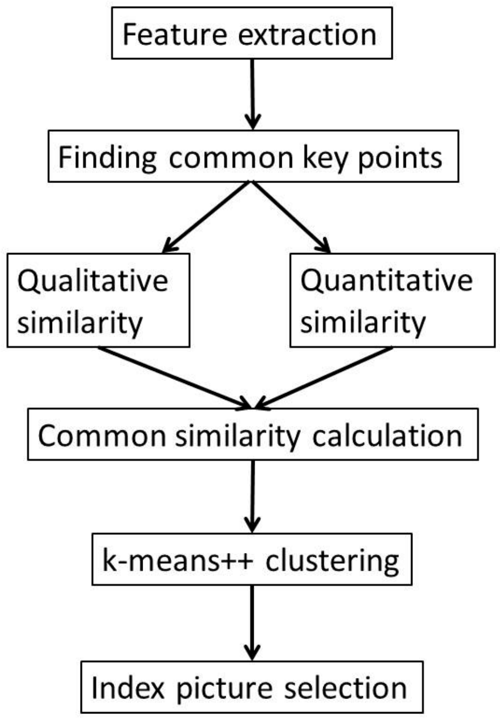


Fig. 2 Block diagram of elaborated index picture selection

C. Feature extraction by SURF

For semantic information some important objects or key points should be found in the image; in this paper the lower level solution is investigated, so the second (key points) is chosen. There are several feature extraction method for key values of an image, and the most frequent used are the SIFT [7][8] and SURF.

SURF (Speeded Up Robust Features) [14][15] is a robust local feature detector, that can be used in computer vision tasks. The standard version of SURF is several times faster than SIFT, and the descriptor is based on sums of Haar wavelet components.

SURF is based on integral images, and it uses an integer approximation to the determinant of Hessian blob detector, which can be computed extremely quickly with an integral image. Not only Hessian detector, but other the leading existing detectors and descriptors are used in SURF by building on the strengths of them and by simplifying these methods to the essential. This leads to a combination of novel detection, description, and matching steps.

The above mentioned speed gain is due to the use of integral images, which drastically decrease the number of operations for simple box convolutions, independent of the chosen scale. Investigating the performance of Hessian approximation (with 3 integer operations) in an integral image is comparable and sometimes even better than the state-of-the-art interest point detectors.

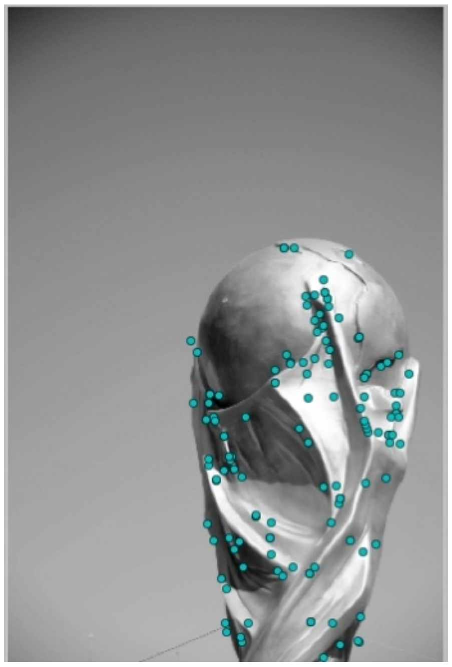


Fig. 3 Example: a picture and key points

In video analysis SURF has been chosen for feature extraction step, because without any dedicated optimizations, an almost real-time computation is possible, which is an important advantage for this task. A picture example and SURF points can be seen in Fig. 3.

D. *Defining similarity between images*

In this paper new similarity measurement is introduced, where similarity depends on two factors, area and density of common key points in the pictures. The common key points (as can be seen in Fig. 4) represent the common details (may be common objects) of the images, which can be found in well determined area in the picture. The size of this area shows similarity volume, but instead of accurate area we can use an estimated value. The estimation is based on the rectangle (frame) that encompasses of the whole area. The ratio of the size of this rectangle and the total size of the picture is a good indicator for area factor of similarity measurement.

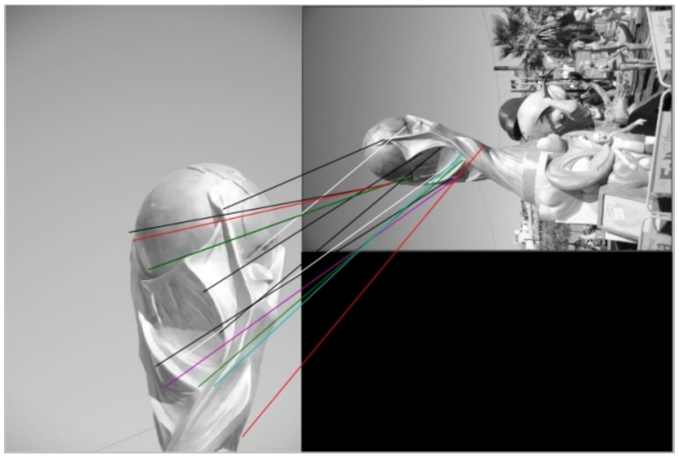


Fig. 4 Two pictures and common key points

The sizes of the rectangles should be calculated in both investigated images. Let denote a common key point by P(*i*), where *i* goes from 1 to number of common key points. Two functions, GetX and GetY will return with coordinates x and y of key point (as input parameter of the function) respectively. The coordinates can be in pixel or other measurement units, this is not important, because only the ratio of the sizes will be into account. The width (w) (10) of the rectangle is the difference of the maximal and minimal x values of the points:

$$w = \underset{i}{\mathrm{Max}}\{\mathrm{GetX}\{P(i)\}\} - \underset{i}{\mathrm{Min}}\{\mathrm{GetX}\{P(i)\}\} \tag{10}$$

The height (h) (11) of the rectangle is the difference of the maximal and minimal y values of the points:

$$h = \underset{i}{\mathrm{Max}}\{\mathrm{GetY}\{P(i)\}\} - \underset{i}{\mathrm{Min}}\{\mathrm{GetY}\{P(i)\}\} \tag{11}$$

The sizes of the rectangles (12) are calculated in both investigated images using the width and height values.

$$R_j = w_j \cdot h_j \qquad j = 1, 2 \tag{12}$$

The total sizes of both images (13) are calculated by the width and height values of the picture.

$$R_{Tj} = w_{Tj} \cdot h_{Tj} \qquad j = 1, 2 \tag{13}$$

The area factor of similarity (14) is determined by the sum of the ratios, and the sum is divided by two in order to get normalized values.

$$AreaSim_2 = \frac{R_1}{2R_{T1}} + \frac{R_2}{2R_{T2}} \tag{14}$$

The above general formula can be used in any images, even that the sizes (widths and heights) of two pictures are different (The number two at the index shows that sizes of two pictures can be different). But in our special case, when we investigate the pictures coming from video, the sizes are equal, so $R_{T1} = R_{T2} = R$ (briefly) as can be seen in (15).

$$AreaSim = \frac{R_1}{2R} + \frac{R_2}{2R} \tag{15}$$

The density factor of similarity depends on number of common key points – denoted by K_C as can be seen in (16) – and the number of all key points in both images (K_1 and K_2).

$$K_C = |P(i)| \tag{16}$$

The density factor of similarity (17) is determined by the largest ratio of common key points.

$$DensitySim = \max\left(\frac{K_C}{K_1}, \frac{K_C}{K_2}\right) \tag{17}$$

The similarity between pictures f_1 and f_2 (18) is defined by product of two factors of similarity.

$$s(f_1, f_2) = AreaSim(f_1, f_2) \cdot DensitySim(f_1, f_2) \tag{18}$$

Two factors of similarity is investigated in Fig. 4 and Fig. 5. Good ”density similarity” can be found between images in Fig. 4, but the other factor of similarity will not be high, because the common object is only a part of the picture. We will get “area similarity” between upper images in Fig. 5, but the other factor of similarity will not be so high, because the images are only similar, but not the same. High similarity (with both of the factors) can be found between bottom images in Fig. 5, because the images semantically are same, they are only the color and gray-scaled version of a picture.

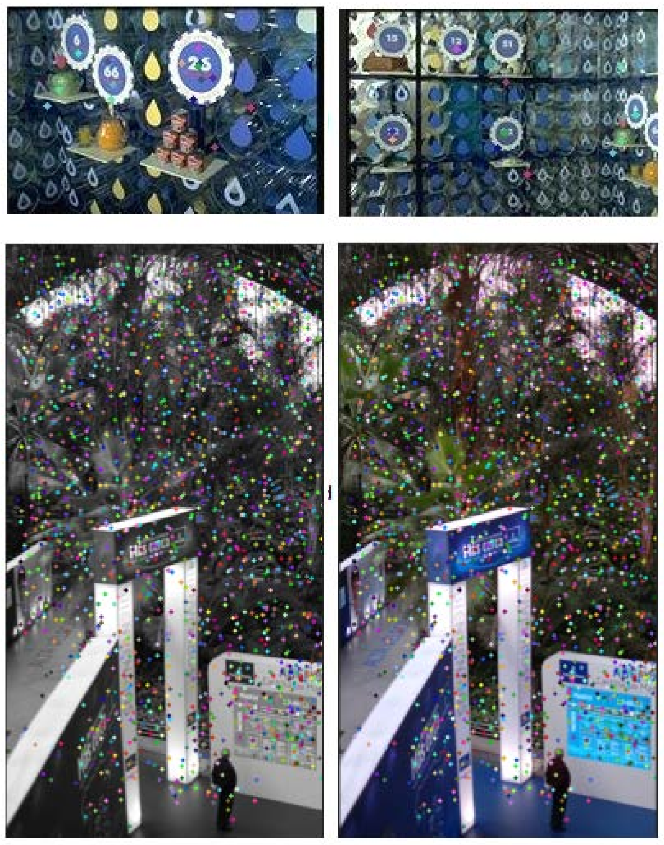

a) b)

Fig. 5 Comparison of two picture pairs

E. Similarity-based k-means++ clustering

The common similarity indicator can be calculated for every image pair that takes the semantic information into consideration. We would like to get the most similar image as index picture based on these similarities, so the task is to find the middle element. This could be solved by simple center calculation, but distances among the images are not defined. Further problem is the burst phenomena in the picture set: some of them are similar to each other, but different from others, so these groups may bias the search of the most similar picture. The idea is finding the groups and the index pictures should be searched in the largest group. Creating groups can be constructed by clustering, but only the similarity values are available for the algorithm, because Euclidean distances among the images are not defined

The k-means method is a widely used clustering technique that seeks to minimize the average squared distance between points in the same cluster. Although it offers no accuracy guarantees, its simplicity and speed are very appealing in practice (it is standard practice to choose the initial centers uniformly at random from X space). By augmenting k-means with a simple, randomized seeding technique, a new algorithm, so called k-means++ [6] has been outlined with the optimal clustering. Preliminary experiments show that the augmentation improves both the speed and the accuracy of k-means. Although this improvement is better than original k-means, this solution can not be used for own purpose, because only similarity (pair wise) values are available, and there are no coordinates, no distance values. For this reason new similarity-based k-means++ algorithm is defined and described in this paper.

The k-means algorithm begins with an arbitrary set of cluster centers, but k-means++ (and similarity-based k-means++) algorithm uses a specific way of choosing these centers. At any given time, let $s_L(f)$ denote the largest similarity from a picture f to the centers we have already chosen; so similarity-based k-means++ algorithm is the following:

- 1a. Choose an initial medoid center c_1 uniformly at random from F (set of all investigated picture).
- 1b. Choose the next medoid center c_i, selecting $c_i = f' \in F$ with probability p, where p can be calculated by (19).

$$p = \frac{1 - s_L(f')^2}{\sum_{f \in F} s_L(f)^2} \tag{19}$$

- 1c. Repeat Step 1b until we have chosen a total of k medoid centers.
- 2. For each $i \in \{1, \dots, k\}$, set the cluster C_i to be the set of pictures in F that are higher similarity with c_i than they are with c_j for all $j \neq i$.

- 3. For each $i \in \{1, \ldots, k\}$, set c_i to be the medoid center of middle (largest similar) point of all points in C_i, based on the sum of similarity values, as can be seen in (20), where $s(f_1,f_2)$ is the value of similarity between pictures f_1 and f_2.

$$c_i = \arg\max_{f_2 \in C_i} \sum_{f_1 \in C_i} s(f_1, f_2) \tag{20}$$

- 4. Repeat Steps 2 and 3 until clusters no longer changes.

Choosing the number of the clusters in k-means++ algorithm is a sensitive parameter for the goodness of the results. We have used the rule of thumb formulated in (21) for the determination of the clusters, where *n* is the number of all pictures.

$$k = \sqrt{n/2} \tag{21}$$

F. Selecting the pictures

The solution would able to select more than 1 picture for representing the whole segment with choosing the second, third, etc. most similar picture, but in in this task the middle (largest similar) picture is enough for the most representative image. So at the end of the clustering the picture corresponding to medoid center of the largest cluster will be selected for index picture, because the largest cluster gives the majority of the images.

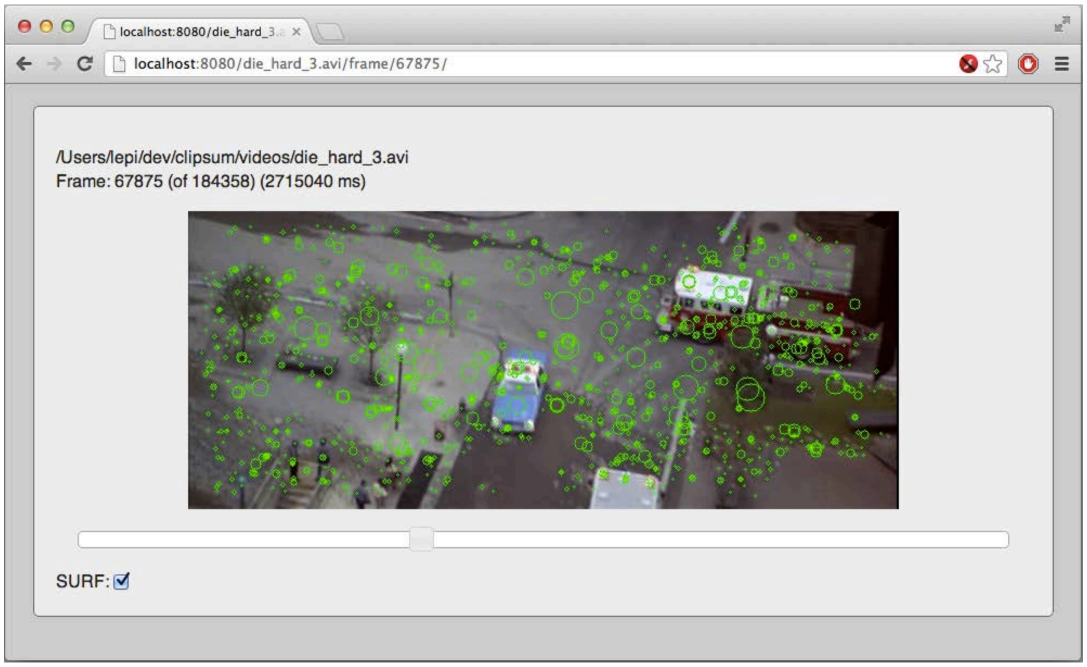
IV. Implementation and Results

A solution has been implemented in a general-purpose, high-level programming language: Python [28], whose design philosophy emphasizes code readability. Using third-party tools, Python code can be packaged into standalone executable programs. Python is often used as a scripting language. The Python has been selected for this work, because interpreters are available for many operating systems, syntax is clear and expressive, furthermore free and open source modules (e.g. module for image handling [27]) can be used in implementation.

One of the open source modules is OpenCV (Open Source Computer Vision Library) [24] which a library of programming functions mainly aimed at real-time computer vision. OpenCV is written in C++ and its primary interface is in C++, but it still retains a less comprehensive though extensive older C interface, e.g. there is now full interface in Python [25]. So OpenCV third-party library (and extension of Python, so called NumPy) has been used in the solution described in this paper.

NumPy [31] is an extension to the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large library of high-level mathematical functions to operate on these arrays. NumPy is open source (licensed under the BSD license, enabling reuse with few restrictions) and has many contributors. NumPy is a prerequisite of OpenCV, because mathematical operations are required for many image functions.

A web-application has been also elaborated for providing an easy-to-use graphical interface for this system. The Fig. 6 shows the results of the implemented SURF algorithm in a sampled picture in the video.


localhost:8080/die_hard_3.a
localhost:8080/die_hard_3.avi/frame/67875/
SURF: ☑

/Users/lepi/dev/clipsum/videos/die_hard_3.avi
Frame: 67875 (of 184358) (2715040 ms)

Fig. 6 SURF points in a sample picture of the examined video

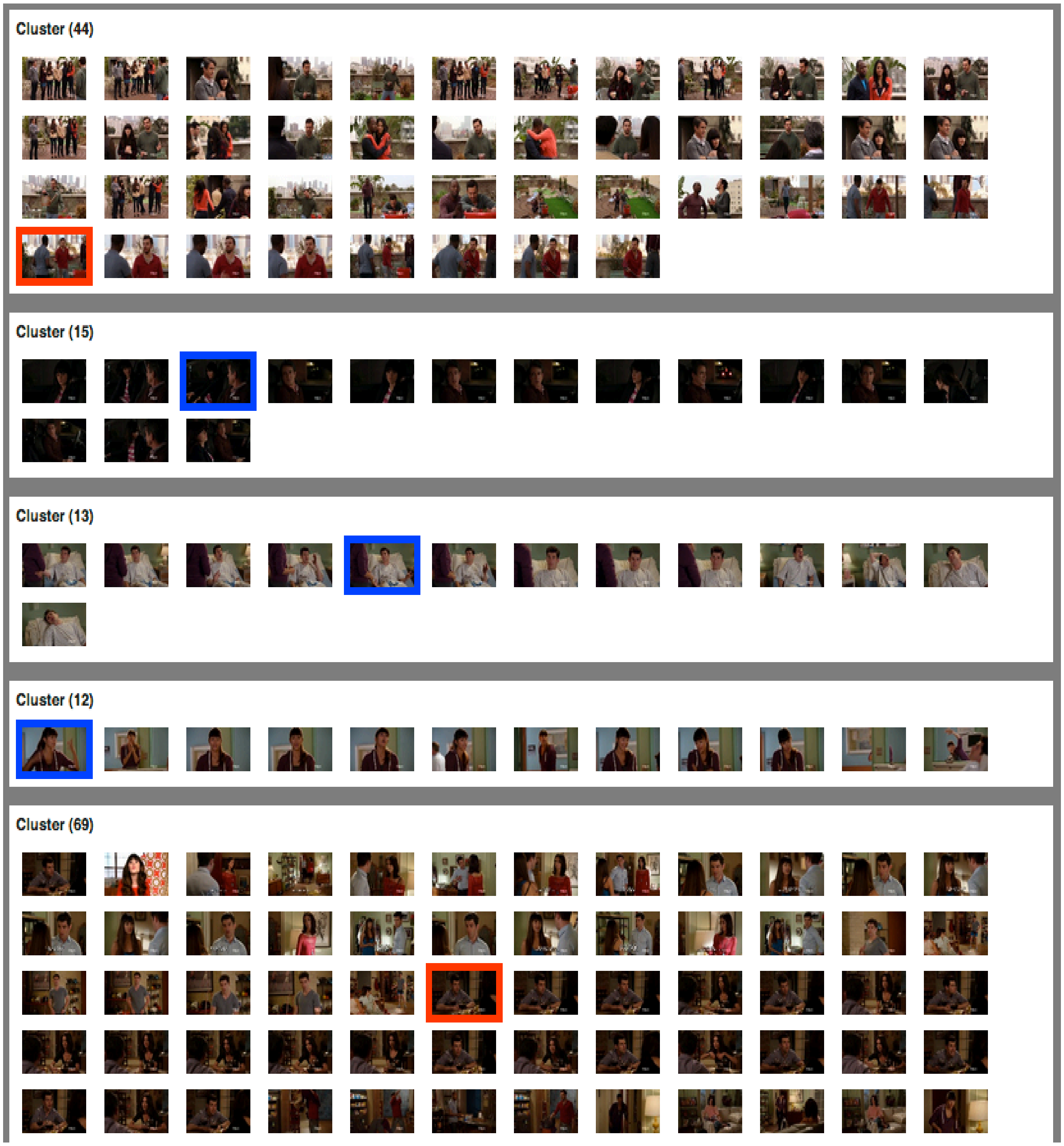


Fig. 7 Segments of a video and index pictures

Further standalone helpful software, FFmpeg has been used for handling the videos. FFmpeg [11] is a complete, cross-platform solution to record, convert and stream audio and video.

Segments of an example video and index pictures can be seen in Fig.7, where the frames of index pictures in the large clusters are red, and frames corresponding to little clusters are blue.

The evaluation method is a little bit similar to procedure used in reranking of relevant images [36], where the minimizing the number of irrelevant images was more important because of real-world usage scenarios.

Three human evaluators have selected the index picture (as most representative picture), then second ones, etc., so they have ranked the images in a segment. The implemented solution has also selected the index picture. The machine results have been compared with the aggregated order of three human decisions (the aggregation is based on Borda method).

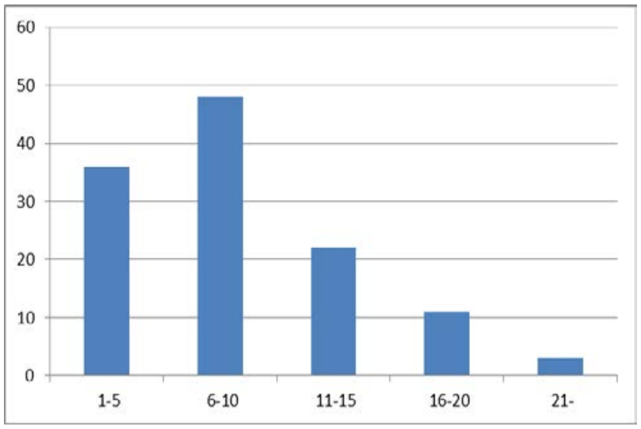


Fig. 8 Segments of a video and index pictures

The results of 120 segments can be seen in the Fig. 8., where the range of positions are 1-5, 6-10, etc. and the histogram shows how many index pictures of segments can be found in the range of human' positions. The machine results (index picture) have been not always the same as the first in the humans' order, but we have counted 70% of segments, where the machine results are in the best 10 representative pictures.

V. Summary

In this paper a new construction is described for two purposes, as two phases of larger aim: video segmentation and index picture selection from segments. The first phase is working without semantic information, but in the second phase, the index picture selection is based on semantic information. In the future we are planning to construct a solution, which takes semantic information in both phases into consideration.

In the automatic video segmentation sampled pictures are analyzed, the difference between consecutive ones is measured, and compared it to a predefined threshold. The contribution of the paper is the 1-dimensional clustering in segmentation, which is able to help to refine the inner results. The goal of this clustering is the filtering of non-adequate candidates in the segmentation.

The second phase, the index picture selection procedure consists of feature extraction, finding common key points, common similarity calculation from two factors of similarity indicators, clustering using k-means++ method, and final selection. The contribution of this phase is the area and density factors of similarity among pictures, and the common similarity indicator taking the semantic information into consideration. The last contribution is the adaptation of k-means++ clustering for the similarity-based picture set, where the distance (Euclidean distance) between the images is not defined, only the similarity values are available.

In the extended version of the solution the cross-modality will be included, because this seems promising way [22] for further improvement, where not only the visual information will be used, but audio as well.

Acknowledgments

The publication was supported by the TÁMOP-4.2.2.C-11/1/KONV-2012-0001 project. The project has been supported by the European Union, co-financed by the European Social Fund.

References

[1] A. Hampapur, "Semantic Video Indexing: Approach and Issues", *SIGMOD Record*, Vol. 28, No. 1, March 1999, pp. 32–39.

[2] A. Wichert, "Content-based image retrieval by hierarchical linear subspace method," *Journal of Intelligent Information Systems*, August 2008, Volume 31, Issue 1, pp. 85–107.

[3] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos, "Fast subsequence matching in time-series databases," In *Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data*, Minneapolis, Minnesota, USA, pp. 419-429.

[4] C. G. M. Snoek and M. Worring "Multimodal Video Indexing: A Review of the State-of-the-art," *Multimedia Tools and Applications*, 2005, Volume 25, Issue 1, pp. 5–35.

[5] C. G. M. Snoek and M. Worring, "Multimedia event-based video indexing using time intervals," *IEEE Transactions on Multimedia*, vol.7, no.4, Aug. 2005, pp. 638- 647.

[6] D. Arthur, S. Vassilvitskii, "k-means++: the advantages of careful seeding," in Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms (SODA '07), 2007, pp. 1027-1035.

[7] D. G. Lowe, "Object recognition from local scale-invariant features," *Proceedings of the 7th International Conference on Computer Vision (ICCV'99)* (Corfu, Greece), 1999, pp. 1150–1157.

[8] D. G. Lowe, "Distinctive image features from scale-invariant key points," *International Journal of Computer Vision*, Vol. 60., Num. 2., 2004, pp. 91-110.

[9] E. Potapova, M. Egorova, and I. Safonov, "Automatic Photo Selection for Media and Entertainment Applications," *GraphiCon'2009, Proceedings of The 19th International Conference on Computer Graphics and Vision*, October 5-9, 2009, Moscow, Russia, pp. 117-124.

[10] F. DuFaux, "Key frame selection to represent a video," *International Conference Image Processing (ICIP)*, 2000, pp. 275-278.

[11] FFmpeg – Official Website, http://www.ffmpeg.org/

[12] G. Bradski and A. Kaehler, *Learning OpenCV: Computer vision with the OpenCV library*, O'Reilly Media, Incorporated, 2008.

[13] G. Szűcs, T. Leposa, S. Turbucz, „Representative Picture Selection from Albums," In: *The Third International Conferences on Advances in Multimedia (MMEDIA 2011),* Budapest, Hungary, 17-22 April, 2011, pp. 114-117.

[14] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, "SURF: Speeded Up Robust Features," *Computer Vision and Image Understanding (CVIU)*, Vol. 110, No. 3, 2008, pp. 346-359.

[15] H. Bay, T. Tuytelaars, and L. V. Gool, "SURF: Speeded Up Robust Features," *Computer Vision, Lecture Notes in Computer Science* (ECCV 2006), Volume 3951, 2006, pp. 404-417.

[16] K. Vaiapury and M. S. Kankanhalli, "Finding Interesting Images in Albums using Attention," *Journal of Multimedia* 3(4), pp. 2-13, 2008.

[17] L. Chunxi, H. Qingming, and J. Shuqiang, "Query Sensitive Dynamic Web Video Thumbnail Generation," *18th IEEE International Conference Image Processing (ICIP),* Brussels, 11-14 Sept. 2011, pp. 2449-2452.

[18] M. Bastan, H. Cam, U. Gudukbay, and O. Ulusoy, "An MPEG-7 Compatible Video Retrieval System with Integrated Support for Complex Multimodal Queries," (Accepted for publication in 2009), *MultiMedia, IEEE*, to be published.

[19] M. Bastan, H. Cam, U. Gudukbay, and O. Ulusoy, "Bilvideo-7: an MPEG-7- compatible video indexing and retrieval system," *MultiMedia, IEEE*, vol.17, no.3, July-September 2010, pp. 62-73.

[20] M. Christel, "Evaluation and user studies with respect to video summarization and browsing," *SPIE Conference on Multimedia Content Analysis, Management, and Retrieval*, January 2006.

[21] M. Yang, X. Mu, and G. Marchionini, “VIVO - a Video Indexing and Visualization Organizer,” *Proceedings of Joint Conference on Digital Libraries*, 27-31 May 2003, pp. 403.

[22] N. Rasiwasia, J. C. Pereira, E. Coviello, G. Doyle, G.R.G. Lanckriet, R. Levy, and N. Vasconcelos, "A new approach to cross-modal multimedia retrieval," In *proceedings of the international conference on Multimedia (MM '10)*, ACM New York, NY, USA, 2010, pp. 251-260.

[23] Open Video Digital Library, www.open-video.org

[24] OpenCV – Official Website, http://opencv.org/

[25] OpenCV 2.1 python reference http://opencv.willowgarage.com/documentation/python/index.html

[26] P. Yu, K. J. Chen, C. Y. Ma, F. Seide, “Vocabulary-Independent Indexing of Spontaneous Speech,” *IEEE transaction on Speech and Audio Processing*, *Special Issue on Data Mining of Speech, Audio and Dialog*, Vol.13, No.5, 2005, pp. 635-643.

[27] Python Imaging Library (PIL) http://www.pythonware.com/products/pil/

[28] Python v2.7.3 documentation, last updated Feb 28, 2013, Python Programming Language – Official Website, http://www.python.org/

[29] R. Datta, D. Joshi, J. Li, and J. Z. Wang, ”Image retrieval: Ideas, influences, and trends of the new age,” *ACM Comput. Surv*. 40, 2, Article 5, April 2008, 60 pages, DOI = 10.1145/1348246.1348248 Available: http://doi.acm.org/10.1145/1348246.1348248

[30] S. Ardizzoni, I. Bartolini, and M. Patella, "Windsurf: region-based image retrieval using wavelets," in *Proc. of Tenth International Workshop on Database and Expert Systems Applications*, 1999, pp. 167-173.

[31] Scientific Computing Tools For Python – Numpy http://www.numpy.org/

[32] W. Chu and C. Lin, “Automatic selection of representative photo and smart thumbnailing using near-duplicate detection,” in *Proc. 16th ACM Int. Conf. Multimedia*, 2008, pp. 829–832.

[33] X. Hua, S. Li, and H. Zhang, “Video booklet,” *IEEE International Conference on Multimedia and Expo (ICME)*, 6-8 July 2005.

[34] Y. C. P. Beevi and S. Natarajan, “An efficient Video Segmentation Algorithm with Real time Adaptive Threshold Technique,” *International Journal of Signal Processing, Image Processing and Pattern Recognition*, Vol. 2, No.4, December 2009, pp. 13-28.

[35] Y. Gong and X. Liu, “Generating video summaries,” *International Conference Image Processing (ICIP)*, 2000.

[36] Y. Jing and S. Baluja, “PageRank for product image search,” *Proceedings of the 17th international conference on World Wide Web, (WWW '08)*, 2008, pp. 307-316.

[37] Z.-Y. Zhou, P. Yu, C. Chelba, and F Seide, "Towards spoken-document retrieval for the internet: lattice indexing for large-scale web-search architectures," in *Proceeding of HLT-NAACL '06 conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, Association for Computational Linguistics Stroudsburg, PA, USA 2006, pp. 415–422.

Gábor Szűcs was born in Hungary in 1970. He has received MSc in Electrical Engineering from Budapest University of Technology and Economics (BME) in Budapest in Hungary in 1994.

He is experienced in modeling and simulation, railway systems, traffic systems; he has received PhD degree in this field from BME in 2002. His further and currently research areas are data mining, multimedia mining, content based image retrieval, semantic search. He is associate professor at Department of Telecommunications and Media Informatics of BME. The number of his publications is more than 80.

Dr. Szűcs is vice president of the Hungarian Simulation Society (EUROSIM), deputy director of the McLeod Institute of Simulation Sciences Hungarian Center. He has earned János Bolyai Research Scholarship of the Hungarian Academy of Science in 2008.