Shot Boundary Detection Method Based on a New Extensive Dataset and Mixed Features

Alexander Gushchin[1], Anastasia Antsiferova[1] and Dmitriy Vatolin[1]

[1]Lomonosov Moscow State University, Leninskiye Gory, 1, Moscow, 119991, Russia

Abstract

Shot boundary detection in video is one of the key stages of video data processing. A new method for shot boundary detection based on several video features, such as color histograms and object boundaries, has been proposed. The developed algorithm was tested on the open BBC Planet Earth [1] and RAI [2] datasets, and the MSU CC datasets, based on videos used in the video codec comparison conducted at MSU, as well as videos from the IBM set, were also plotted. The total dataset for algorithm development and testing exceeded the known TRECVID datasets. Based on the test results, the proposed algorithm for scene change detection outperformed its counterparts with a final F-score of 0.9794.

Keywords

shot boundary detection, machine learning, color histograms, boundary gradients, abrupt/gradual scene changes,

1. Introduction

One of the basic steps in video processing is video scene splitting. For example, scene cutting is a necessary step in video annotation and indexing [3], keyframe searching [4], and automatic video format changing [5]. Existing algorithms have achieved high accuracy in detecting transitions between scenes in general cases, but still make mistakes in detecting complex transitions (Fig. 1).

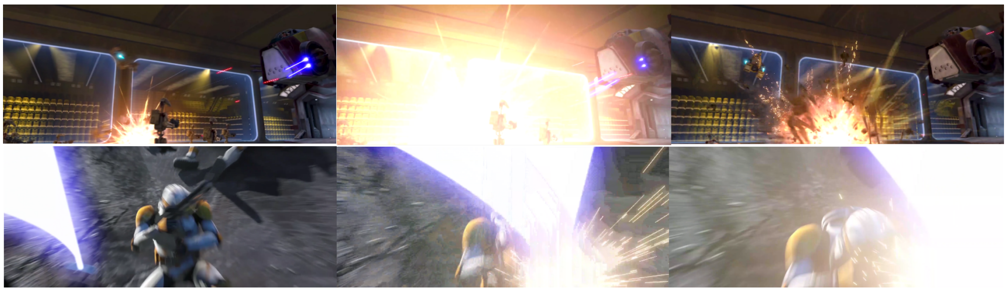

Figure 1: Examples of flashes which cause errors in shot boundary detection.

GraphiCon 2021: 31st International Conference on Computer Graphics and Vision, September 27–30, 2021, Nizhny Novgorod, Russia

alexander.gushchin@graphics.cs.msu.ru (A. Gushchin); aantsiferova.graphics.cs.msu.ru (A. Antsiferova); dmitriy@graphics.cs.msu.ru (D. Vatolin)

0000-0002-4055-7394 (A. Gushchin); 0000-0002-1272-5135 (A. Antsiferova); 0000-0002-8893-9340 (D. Vatolin)

Also, existing algorithms have been developed based on open data sets that may contain errors. For example, when analyzing one of the most popular BBC [1] datasets, several frame inaccuracies were found in the markup of transitions: for example, in the From Pole to Pole video, the first scene ends at 632 frames, but the second begins at 650 frames. Since the algorithm must specify the frame number with the scene change, such errors have been corrected to work more accurately. Thus, the challenges of creating a method for quickly and accurately partitioning video into scenes, as well as creating a volumetric data set with accurate partitioning, are relevant.

Since different definitions for scene transitions are found in the literature (for example, the definition of a scene varies – it can be defined as a gluing of two perspectives or a semantic part of a movie), the following are the basic definitions that will be used in this work. In this paper, we have relied on the definitions given in the formal statement of the problem formulated by the authors in [6]. The basic element of video is a frame, frames are combined into scenes (shots), and scenes are combined into semantic scenes. A scene—is a continuous stretch of video, shot with a single camera, without stitches or interruptions. A semantic scene is a sequence of scenes with the same semantics. The task of the shot boundary detector is to indicate all the frames in which a scene change has occurred. In most cases, the content within a scene changes gradually, and at the boundaries there is montage gluing, so this task is trivial for humans. Scene changes themselves are divided into two types – abrupt and gradual. Abrupt changes in scenes - the momentary transition from a frame of one scene to a frame of the next. This can be dissolve (the gradual appearance of a new scene on top of the previous one), fade (a gradual transition to a black frame and back) or wipe. Examples of such transitions are shown in Fig. 2.

Most shot boundary detection algorithms work in 2 steps:

- Calculating the value of the frame difference metric or metrics
- Setting the threshold for frame classification. Also at this stage, machine learning is often used for automatic classification.
- An additional step can be filtering frames for false positive detections.

The purpose of this work was to create a new method for shot boundary detection and compare it with existing methods on a new large volume of data. The paper is further structured as follows:

- In section 2, an overview of algorithms from the field is given, as well as datasets to compare them
- Section 3 gives a detailed overview of the proposed approach to solve the problem at hand
- Section 4 contains the results of testing the open algorithms and comparing them with the proposed method
- Section 5 contains conclusions and further plans.

2. Related work

In one of the most detailed works devoted to the analysis of [7] shot boundary detection algorithms, the authors considered their drawbacks as ways to improve them. The main

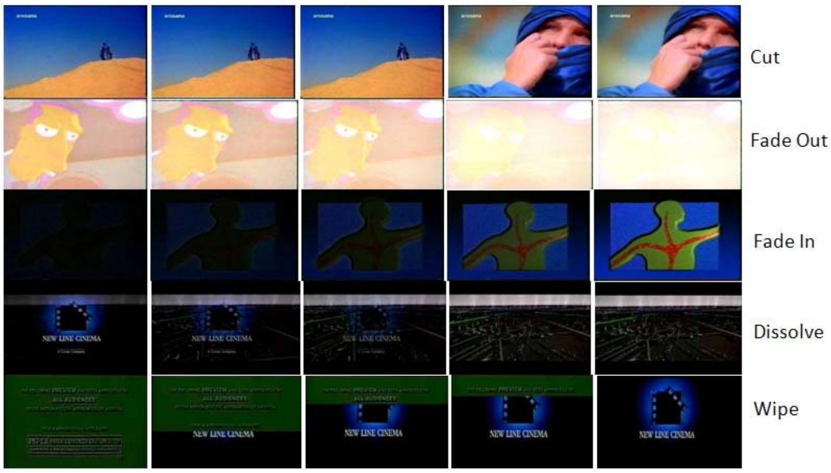

Figure 2: Examples of scene changes.

drawbacks include the slow speed of operation, as well as errors in cases of flashes, fast camera movement, etc.

In most existing methods, the first step is the calculation of features for frames. One of the frequently used is the frame similarity metric for finding the degree of difference between frames. As the scene changes, the value of this metric will increase, while inside the scene it is close to zero. The most popular techniques are: calculation of color histograms [7], [8], boundary gradients [9] [10] [11], geometric transformations [12] [13], motion vectors [14] [15]. One of the simplest methods of constructing this metric is a pixel-by-pixel comparison of frames [7]. Other difference metrics are also calculated between frames — for example, [16] uses L*a*b* space and the formula for the distance between colors in it.

The construction of color histograms was used, for example, in [7], [8]. Histograms can be computed both for RGB and other color spaces (HSV, YCbCr, L*a*b*). With this approach, the algorithm is less sensitive to motion within the frame, but may produce many false positives for scenes with flashes and rapid light changes. The use of boundary gradients partially solves the problem of false positives when the camera or objects move within the frame, allowing you to use frame boundary matching without relying on lighting. Such a technique was used in [9] and [10]. In [11], the authors used object boundaries within frames to construct a histogram of directional gradients. The histograms for different levels of the resolution pyramid are concatenated. This approach allowed the authors to obtain the characteristics of object boundaries in the frame at different levels.

The motion vectors were used in the following works: [17] [14] [15]. By using them, the scene change detection algorithm can be adapted to the movement within the scene, the camera movement or the appearance of a large object in the frame. Vectors take longer to compute than approaches based on color or border histograms, but they can be used together with fast computable metrics and achieve high accuracy (for example, if we consider motion vectors before comparing frame boundaries). Also one of the popular techniques is geometric transformations of frames – Contourlet, Fourier transform [12], Walsh Hadamard [13]. These methods are sensitive to frame motion and resolution, which can seriously increase the running time of the algorithm. The authors from [Contourlet] used an improved contourlet, which is not sensitive to the problems mentioned above. Rarer ways of constructing metrics include SIFT, SURF, entropy [18] [19]. They can give comparable accuracy, but require more computational resources.

Many of the methods described above can be applied not to the whole image, but to a part of it. In this approach frame is divided into blocks (overlapping or not) and metric is calculated for each block. Vector of such metrics can be concatenated, histograms (including cumulative ones) [10]) or use statistics (e.g., expectation and variance). Partitioning into blocks allows the algorithm to be less susceptible to changes in certain parts of the scene (e.g., rapid movement of objects or flashes). Thus, using a combination of features based on different characteristics of video and individual frames allows to achieve higher detection accuracy, reduce the number of false positives, but increases the runtime of the algorithm.

After calculation of frame similarity metrics, each frame is classified into one of three categories: abrupt scene change (cut), gradual scene change (dissolve, vipe, fade in or fade out, no scene change. Since the algorithm needs to analyze all frames, which is a resource-intensive task, some authors use video preprocessing: they select, using additional fast algorithms, the segments where scene changes are supposedly present and further process only those segments. This approach assumes that no scene changes occur on frames that are not in these segments. There are three main approaches to classification: classification by threshold, adaptive threshold, and machine learning. Threshold (or a set of thresholds) is the simplest way to classify. The values of a metric or metrics are compared to a predetermined threshold and a decision is made as to whether a class belongs to a certain class. This approach is rarely used, as it is more advantageous to select thresholds for each individual video based on its features. The adaptive threshold does not have this disadvantage and can not only change depending on video [10], but also depending on metrics values in some neighborhood of the frame [20]. Thus, the threshold is adjusted not for the whole video, but for a particular scene. Recently, due to the development of machine learning algorithms, they are increasingly used for classification: SVM[21], bagged tree classifier[11], k nearest neighbors, neural networks. The authors of [22] have analyzed the techniques used— according to their research, SVM showed the best results.

Certain video artifacts significantly complicate the detection algorithms–for example, flashes and camera/object motion. Some of them can be eliminated at preprocessing stage — for example, separate metrics for flashes [23] are introduced.

2.1. Existing datasets and benchmarks

In most papers on scene-shift detection methods, the authors compare the performance of algorithms on the dataset used in the TRECVID competition. This is one of the most famous and extensive comparisons of shot boundary detection methods, which has been conducted annually for 7 years. It tested 57 algorithms using different sets of marked videos. After the end of each competition, articles were released analyzing the participants' solutions and their results (e.g., [22]). The dataset included mostly documentaries and television shows. There are also a number of articles comparing shot boundary detection methods (e.g., [7]).

We requested access to the TRECVID dataset, but unfortunately, due to Covid-19, the authors were unable to provide it. (The vendor agreement requires sending the dataset on DVDs, and the authors can’t do shipments at this time.) An alternative to this dataset is the use of public datasets. There are several datasets at the moment – their comparison is given in the Table 1. The BBC Planet Earth dataset consists of documentaries, RAI — from 10 randomly selected segments of the broadcast of the television channel RAI scuola. These are mostly talk shows and documentaries. Clipshots includes 20 categories of videos (sports, animals, amateur ...) collected from youtube and weibo. The TRECVid dataset most commonly used for method testing is a collection of 7 datasets that have been selected from different video categories. Unfortunately, the size of the available datasets is inferior to the TRECVID set. In this paper, a new dataset was created to train and test a new method for scene change detection (Table 1).

Table 1
Sizes of datasets used to analyze metrics and create a new SBD method. MSU CC and OS VSD datasets were manually marked up.

Dataset name	Length (minutes)	Scene changes
Rai [2]	93	985
BBC Planet Earth [1]	539	4844
MSU CC	39	274
OS VSD	954	6871
TRECVID 2007	360	2317

3. Proposed method

3.1. Marking up a new dataset

To create an algorithm for detecting scene shifts, a set of OS VSD [24] data was collected using Yandex.Toloka [25]. The creation of a dataset is divided into several steps:

- A few algorithms configured in a way to maximize the completeness of the results was running on all videos
- A list of potential scene changes was created by combining the results of all algorithms
- Each potential scene change was cut from the original video as a short video sequence of 40 frames long
- Yandex.toloka was used to show peoples all these sequences for markup
    - For each video segment, observers indicated whether there was a scene change in it
    - Each video segment was shown at least 5 different people, if the results were not unambiguous the number of observers increased until an agreement between observers was reached

This resulted in an additional 19 videos with a total duration of 965 minutes surpassing the existing TRECVID. The table above shows other comparative characteristics of this set.

3.2. Base features

Methods that have shown high accuracy in existing comparisons were used as base features for the new algorithm. To analyze them, a newly created OS VSD dataset was used, on which these methods were compared.

In the first step, the proposed algorithm uses several metrics to describe frame differences. These metrics are built on a boundary gradient, a frame color histogram to describe frame differences. This approach allows to take into account several factors possible when changing scenes and to get more information about the frames being compared. On the second stage, lgbm algorithm is applied to these metrics for classification. It was chosen as a result of experiments with different machine learning techniques.

First, let us describe the features that our algorithm relies on.

- Metrics proposed in [26]
    - These metrics use the average value and standard deviation of the brightnesses of the pixels in the block of frames.
- Cumulative color histogram metric
    - It is based on [10]. First, the Sobel operator is applied to the frames to find the edges, the trapezoidal smoothing function is applied, and the cumulative histogram of frame blocks is calculated.
- Metric proposed in the Max Remain repository[27]
    - It calculates the difference between color histograms of two consecutive frames and builds the difference between them. The output is a vector of length $3 * n$, $n-$ number of columns in histogram.
- Histogram of edges of objects in the frame
    - At the beginning we apply Sobel operator to find the edges, divide the frame into 100 non-overlapping blocks, build a histogram of edges and compare it with neighboring frames. A threshold is applied to cut off blocks which are different in neighboring frames.
- Metric proposed in the aysebilgegunduz[28] repository
    - metric is the distance bhattacharyya between histograms of consecutive frames.
- Metric proposed in the PyScene algorithm[29]

The [30] features were also tested, but were discarded during the experiments due to their low accuracy compared to the other metrics.

3.3. Training

As a training dataset were selected 19 videos on video hosting youtube.com total duration of 26 minutes (38917 frames) . Also 2 videos from BBC Planet Earth set were added with total duration of 96 minutes (144700 frames). Thus, the training dataset consisted of 21 videos of 122 minutes duration (183617 frames). There were 917 abrupt scene changes and 54 gradual scene changes. The test dataset consisted of 9 videos taken from the BBC Planet Earth dataset and 10 videos and the RAI dataset. The total test dataset consisted of 563 minutes of video (804883 frames), with 4510 sharp and 348 gradual scene changes. Linear and logistic regression, SVM, K-means, LGBM, and random forest were tried as a learning algorithm. The LGBM algorithm showed the best results, and its parameters were chosen using crossvalidation. The graph of the contribution of the features in the final model can be seen in Fig. 3.

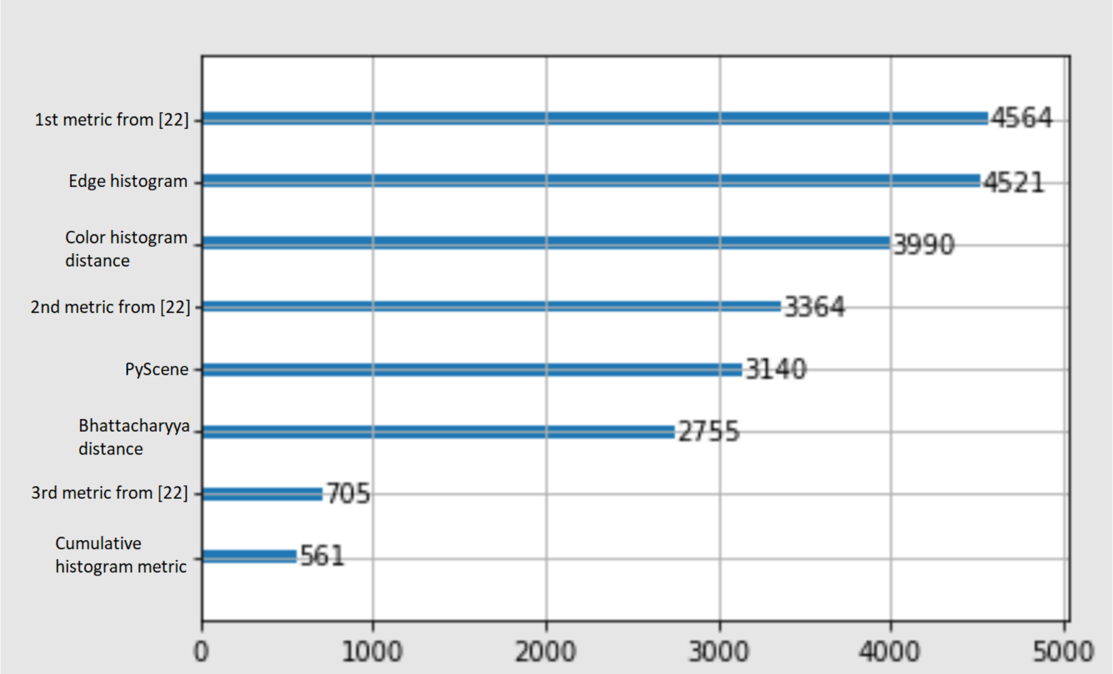

Figure 3: Feature importance.

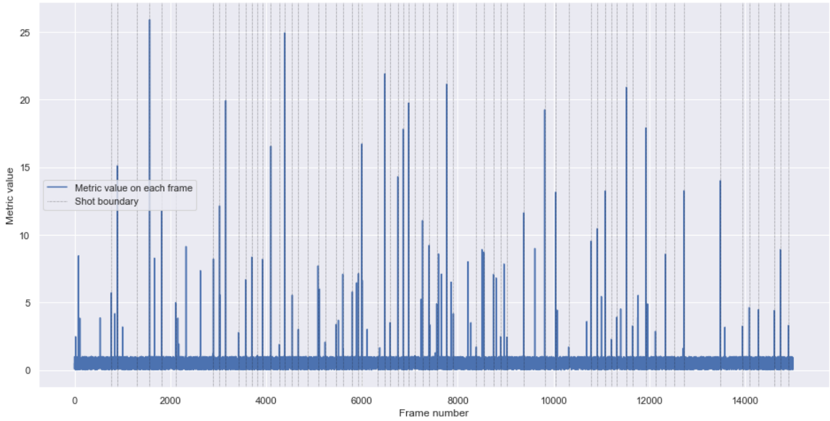

Figure 4: The metric value from the article [26] on a random segment of the video from the test dataset. Blue line is metric value, vertical lines are frames with scene change.

4. Results

The accuracy of the algorithm was measured on a test dataset, and a comparison with counterparts was made. The F1 score metric was used to measure the accuracy and recall of the found scene changes. Table 2 shows the scores obtained: the proposed algorithm outperformed popular methods in terms of accuracy.

Table 2
Average F1 score for the proposed methods and widely used and popular shot boundary detection methods on BBC Planet Earth and RAI datasets.

Method	Speed (FPS)	F score	Precision	Recall
Proposed	31	0.9794	0.9784	0.9803
VQMT[31]	308	0.9476	0.9352	0.9604
FFmpeg[32]	165	0.9448	0.9307	0.9594
PyScene[29]	321	0.9526	0.9467	0.9586

Table 3
F1 score for the proposed methods and widely used and popular shot boundary detection methods on OS VSD dataset.

Method	Speed (FPS)	F score	Precision	Recall
Proposed	31	0.8316	0.8212	0.8423
VQMT[31]	203	0.7379	0.7523	0.7241
FFmpeg[32]	104	0.7464	0.7571	0.7361
PyScene[29]	214	0.7560	0.7782	0.7351

The efficiency of the used metrics was also analyzed. Fig. 5 shows the values of metrics pairs on each frame of training dataset. Different colors indicate the presence and absence of scene changes. From the illustration we can see that many pairs of metrics make a clear classification, for example, a pair of metrics from [26].

Fig. 4 shows an example of the [26] metric, which has the largest contribution to the accuracy of the model, on a segment of video from the test set. It can be seen that on the frames with scene changes the metric takes large values, easily separable from the rest.

5. Conclusion and future work

In this paper, we proposed a new method for determining scene changes based on different metrics. The algorithm was tested on BBC Planet Earth and RAI datasets; its accuracy was 0.9784 and completeness was 0.9803. The proposed method outperformed its counterparts in

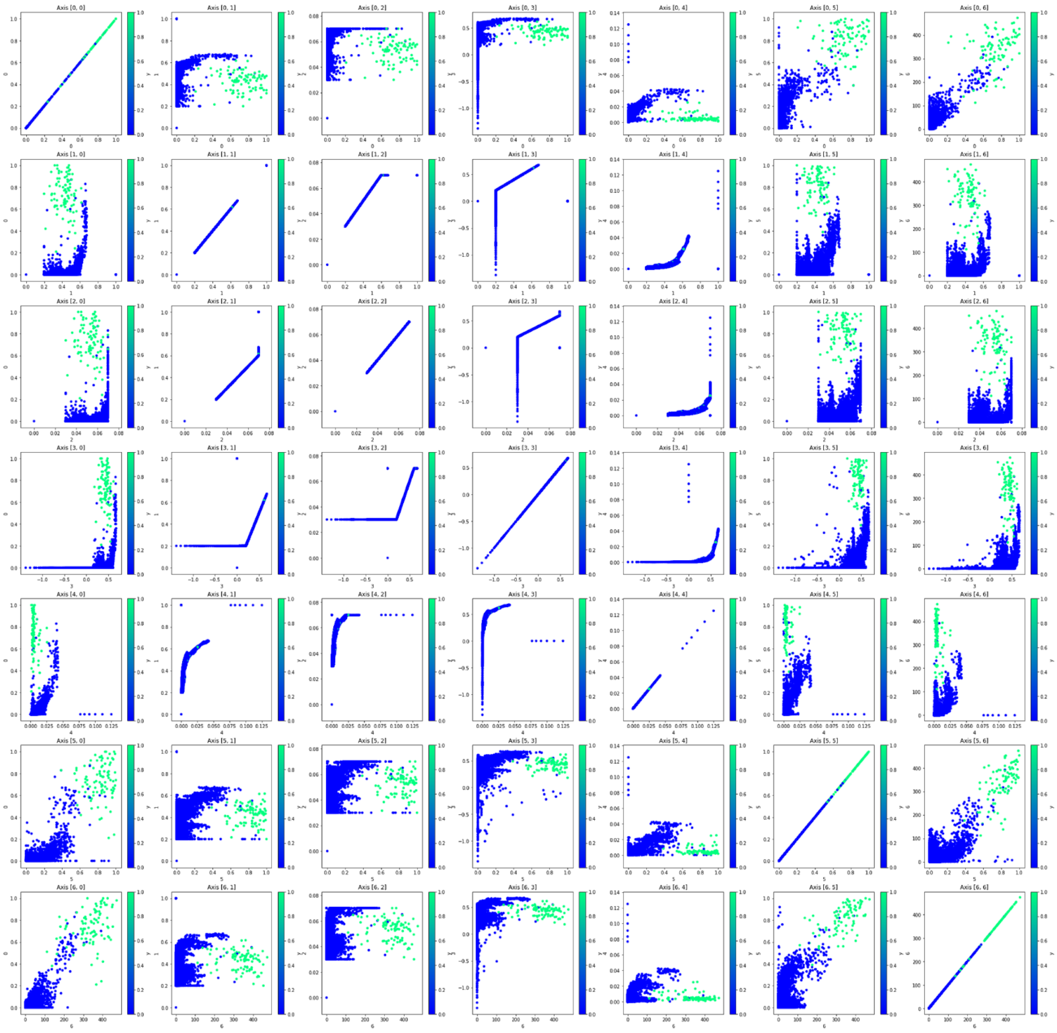

Figure 5: Metrics values on frames from the training dataset. Blue dots–frames without scene changes, green–frames with scene changes

the F1-score metric. At the moment the speed of the algorithm is slower than analogues, as it uses a larger number of features. In the further development of the project it is planned to speed up the proposed method, as well as to analyze the performance of the methods in complex cases for classification.

Acknowledgments

This work was partially supported by the Russian Foundation for Basic Research under Grant 19-01-00785a and the non-commercial fund of science and education development “Intellect”.

References

[1] Bbc planet earth dataset, n.d. URL: https://aimagelab.ing.unimore.it/imagelab/researchActivity.asp?idActivity=19.

[2] Rai dataset, n.d. URL: https://aimagelab.ing.unimore.it/imagelab/researchActivity.asp?idActivity=19.

[3] S.-C. Chen, M.-L. Shyu, C. Zhang, Innovative Shot Boundary Detection for Video Indexing, 2006. doi:10.4018/9781591405719.ch009.

[4] N. H M, H. Chethan, R. B S, Shot based keyframe extraction using edge-lbp approach, J. King Saud Univ.-Comput. Inf. Sci. (2020). doi:10.1016/j.jksuci.2020.10.031.

[5] Autoflip: An open source framework for intelligent video reframing, https://ai.googleblog.com/2020/02/autoflip-open-source-framework-for.html, n.d.

[6] J. Yuan, H. Wang, L. Xiao, W. Zheng, J. Li, F. Lin, B. Zhang, A formal study of shot boundary detection, Circuits and Systems for Video Technology, IEEE Transactions on 17 (2007) 168 – 186. doi:10.1109/TCSVT.2006.888023.

[7] S. H. Abdulhussain, A. R. Ramli, M. I. Saripan, B. Mahmmod, S. A. R. Al-Haddad, W. Jassim, Methods and challenges in shot boundary detection: A review, Entropy 20 (2018). doi:10.3390/E20040214.

[8] J. Boreczky, L. Rowe, Comparison of video shot boundary detection techniques, Proceedings of SPIE - The International Society for Optical Engineering 2670 (1996). doi:10.1117/12.238675.

[9] H.-W. Yoo, H.-J. Ryoo, D.-S. Jang, Gradual shot boundary detection using localized edge blocks, Multimedia Tools Appl. 28 (2006) 283–300. doi:10.1007/s11042-006-7715-8.

[10] R. B S, H. Nagendraswamy, Video shot boundary detection using block based cumulative approach, Multimedia Tools and Applications 80 (2021) 1–24. doi:10.1007/s11042-020-09697-6.

[11] A. Sasithradevi, S. M. M. Roomi, A new pyramidal opponent colorshape model based video shot boundary detection, J. Vis. Commun. Image Represent 67 (2020) 12.

[12] A. Miene, A. Dammeyer, T. Hermes, O. Herzog, Advanced and adaptive shot boundary detection (2002).

[13] L. P. G G, D. S, Walsh–hadamard transform kernel-based feature vector for shot boundary detection, IEEE transactions on image processing : a publication of the IEEE Signal Processing Society 23 (2014). doi:10.1109/TIP.2014.2362652.

[14] P. Panchal, S. Merchant, N. Patel, Scene detection and retrieval of video using motion vector and occurrence rate of shot boundaries, in: 2012 Nirma University International Conference on Engineering (NUiCONE), volume 67, 2012, pp. 1–6.

[15] A. Jacobs, A. Miene, G. Ioannidis, O. Herzog, Automatic shot boundary detection combining color, edge, and motion features of adjacent frames, 2004, pp. 197–206.

[16] S. Chakraborty, A. Singh, D. Thounaojam, A novel bifold-stage shot boundary detection algorithm: invariant to motion and illumination, The Visual Computer (2021) 1–12. doi:10.1007/s00371-020-02027-9.

[17] A. Amel, A. Ben Abdelali, M. Abdellatif, Video shot boundary detection using motion activity descriptor 2 (2010).

[18] J. Baber, N. Afzulpurkar, M. Dailey, M. Bakhtyar, Shot boundary detection from videos

using entropy and local descriptor (2011) 1–6. doi:10.1109/ICDSP.2011.6004918.

[19] E. Apostolidis, V. Mezaris, Fast shot segmentation combining global and local visual descriptors, 2014. doi:10.1109/ICASSP.2014.6854873.

[20] S. Tippaya, S. Sitjongsataporn, M. Khan, K. Chamnongthai, Multi-modal visual features-based video shot boundary detection, IEEE Access PP (2017) 1–1. doi:10.1109/ACCESS.2017.2717998.

[21] J. Mondal, M. Kundu, S. Das, M. Chowdhury, Video shot boundary detection using multiscale geometric analysis of nsct and least squares support vector machine, Multimedia Tools and Applications 77 (2018) 8139–8161. doi:10.1007/s11042-017-4707-9.

[22] A. Smeaton, P. Over, A. Doherty, Video shot boundary detection: Seven years of trecvid activity, Comput. Vis. Image Underst. 114 (2010) 411–418.

[23] X. Qian, G. Liu, R. Su, Effective fades and flashlight detection based on accumulating histogram difference, IEEE Transactions on Circuits and Systems for Video Technology 16 (2006) 1245–1258. doi:10.1109/TCSVT.2006.881858.

[24] Os vsd dataset, n.d. URL: https://www.research.ibm.com/haifa/projects/imt/\video/Video-DataSet.shtml.

[25] Yandex.toloka, n.d. URL: https://toloka.yandex.ru/.

[26] B. J. Bogdanov Alexander, Adaptive moment detector of instantaneous scene changes in a video stream and its training method based on the signs of video stream content: dark / light, calm / dynamic, in: Bulletin of Tomsk State University. Management, computer science and informatics, 4 (37), 2016.

[27] Maxreimann github repository, n.d. URL: https://github.com/MaxReimann/Shot-Boundary-Detection.

[28] aysebilgegunduz github repository, n.d. URL: https://github.com/aysebilgegunduz/ShotBoundary\Detection.

[29] Pyscene shot boundary detection tool, n.d. URL: https://pyscenedetect.readthedocs.io/en/latest/.

[30] O. Murashko, J. Thomson, H. Leather, Predicting and optimizing image compression, in: Proceedings of the 24th ACM International Conference on Multimedia, MM '16, Association for Computing Machinery, New York, NY, USA, 2016, p. 665–669. URL: https://doi.org/10.1145/2964284.2967305. doi:10.1145/2964284.2967305.

[31] Msu vqmt scene change detection tool, n.d. URL: https://www.compression.ru/video/quality-measure/metric-plugins/scd-en.htm.

[32] Ffmpeg shot boundary detection tool, n.d. URL: https://ffmpeg.org/.