Key-frame Detection in Raw Video Streams

Stanislava Budvytytė
stabudv@stud.ktu.lt

Armantas Ostreika
armantas.ostreika@ktu.lt

Kaunas University of Technology, Department of Multimedia Engineering
Studentų St. 50, LT-51368 Kaunas, Lithuania

Abstract. The first, in this paper a review on condensed video representation techniques, especially, keyframe detection is given. The second, a histogram comparison method used for recognizing the environment is described. And the third, a modification of the histogram comparison method to use it for keyframe detection in raw video streams is proposed. The algorithm is tested using video records, that were created as a car routes while driving through a city.

Keywords: Condensed video representation, key frame detection, shot detection, video analysis, raw video streams.

1 Introduction

Video is a manifold media and it requires a lot of storing space in both, digital or analog formats. In resent decade a digital storage formats and devises have been developing firmly and storing space price have decreased vastly. At the same time various digital devises having built-in video cameras have been created. As a result an amount of video information is increasing enormously. A demand on video information analysis and processing techniques is increasing at the same time.

In this paper we focus only on analysis of visual content of the video — we analyse methods used for frame features extraction and comparison of the frames. The methods can be grouped into two groups: shot boundary detection algorithms and key frame extraction methods. An overview of these methods is given in Section 2.

Even taking only a visual component of video stream, one do not get a continuous information flow. Based on the storing formats, visual information is composed of a huge amount of still images — key frames and other supportive data describing their flow from one key frame to another. The data storage key frames and key frames that are interesting for a user as a compact way to review video content is not the same. These differences are described in Section 3.1.

In Section 3.2 we introduce a method for raw video representation as a set of key-frames. Our method is mainly based on the technique described in [10]. In the paper the authors described a method for histogram comparison. They have tested it indoors for recognizing environmental objects and their changes. We found it as an interesting approach and applied it for key frame extraction in raw video streams filmed by route tracking equipment and body-mounted cameras.

Our experiments, results and other observations are described in Section 4. In Section 5 we conclude with discusion and future work assumption.

2 Related Work

An amount of information stored in various video formats takes terabites of space in hard disks and other storage devices. And it has a tendency to increase further. Problems of condensed video representation and video storage and analysis has been researched from different perspectives for many years now. The problem of condensed video representation is closely related to other video and image analysis problems — shot detection and pattern matching. In the solution of the problem an important role is played by heuristic methodology of human perception.

In [1] a review of different video shot detection and condensed representation methods is given. This paper is a good choice to begin a research in the area. Shot boundary detection and condensed representation algorithms use different features and metrics extracted form video frames or parts of frames called, regions of interest (ROI) [1]. The authors name the possible choices for these video components to analyze:

- Features such as luminance / color, luminance / color histogram, image edges,

motion, coefficients of different transformations, e.g. discrete Fourier transformation or discrete cosine transformation.

- Spatial feature domain — single pixel, rectangular blocks, arbitrary shaped blocks, whole frame.
- Feature similarity metric is a mathematical expression of similarity or dissimilarity of the frames or ROIs.
- Temporal domain of continuity metrics such as the simplest way is to compare two neighboring frames, n-frame window — uses all frames in a window, this method is commonly used, or interval since last shot change computing statistics of changes from the last shot or one of the last detected shots.
- Shot change detection methods, such as static thresholding, adaptive thresholding, probabilistic detection, trained classifiers, are also used.

The next important thing about shot detection is performance evaluation [1]. For this purpose standard technical characteristics are used: recall, precision, and accuracy.

The subproblem of condensed video representation has several specific techniques for video analysis. Firstly, it is highlighting — extraction of frames that contain the most relevant information of a whole video or of a single shot. It is sometimes called key frame extraction, too. Hierarchical highlighting tries to extract "*a hierarchy or a tree of highlighted frames, so that user can then interactively request more detail on the part on the video that interests him.*"[1]. Skimming technique tries to extract appropriate features from video segments instead of images.

A fuzzy video content representation technique is presented in [2]. The focus is taken on traditional video sequences made of a collection of different shots. So, firstly a video shot cut detection is applied to detect a sequences of frames with similar visual content. Another step is a color/motion segmentation for each frame in a single shot. It is performed using "*the recursive shortest spanning tree (RSST) algorithm, called M-RSST*" [2] which is adopted for both color and motion segmentation. The algorithm uses two parameters — an initial image resolution level to detect the boundary blocks of an image and an adaptive threshold for terminating the algorithm. "*[. . .]At each iteration, the Euclidean distance of color or motion intensities between two neighboring segments, weighted by the harmonic mean of their areas, is first calculated and then the distance histogram is created. The half of maximum histogram value is considered as the appropriate threshold for the given iteration. The segmentation is terminated if no segments are merged form one step to an other.*" [2] As a result, for each frame color properties such as size, location, and average color of color components of the frame are calculated. In parallel, motion properties such as size, location, and average motion vectors of motion components of the frame are determined. All color/motion properties are classified into pre-determined classes so that each feature vector corresponds to a specific class. In each class a degree of membership is allocated. The authors have called this allocation a fuzzy classification. It leads to that every segment can belong to any number of classes but with different degree of membership. Depending on this classification for each frame a fuzzy multidimensional histogram is created. Finally, frames consisting similar content are discarded. It is done by content-based sampling algorithm which calculates correlations between different frames.

A number of video analysis parameter that are useful both in shot cut and in key-frame detection are described in [9]. The paper describes an algorithm for low bit-rate video coding. A key role is played by two thresholds in the algorithm. The first, a high value fixed threshold is used as a security measure forcing an algorithm to record a frame if the number of frames passed since the last encoded frame detection. And the second, an adaptive threshold which depends on an average frame comparison value since the last detected frame for encoding. If this value increases an average — a frame is encoded. An extra memory parameter ensures that an adaptive threshold would not force to encode frame too fast. Moreover, a limit parameter do not allows to skip a frame from encoding when the frame is almost encoded[1]. The last parameter is a time span measured in milliseconds for avoidance of frame encoding too close in time.

In [6] a histogram-based fuzzy c-means clustering algorithm for video segmentation is presented. For each frame RGB color space histograms are created with respect to YCbCr color space and compared to the histogram of the previous frame. The results of comparison are sorted into three groups using a fuzzy c-means clustering algorithm: shot change (SC), suspected shot

[1]The algorithm is developed for real-time low bit-rate video encoding.

change (SSC), and no shot change (NSC). All frames in SC are recognized as shot change key frames. Other shot change frames are selected using heuristic methods from SSC group.

A two-stage hierarchical video summary extraction method is described in [3]. In the first stage alpha-trimmed average histograms for whole sequence of frames are created. An alpha-trimmed histograms are created for each frame in a sequence, too. Each frame histogram is compared to the average histogram and their comparison values are grouped according to fuzzy clusterization into a number of classes. Based on star selection algorithm the best key frames are selected. In the second stage of the method, based on user defined parameters hierarchical structure of key frames is created.

3 Methods

3.1 Background of Video Analysis

In the beginning we specify the terms of video streams elements and analysis stages. From the analysis point of view a structure of a video can be analysed from two different perspectives — logical and physical.

From the physical point of view video unit can be a shot, a scene or a sequence. *Shots, which are the smallest physical video units, consist of one or more consecutively generated and recorded frames, representing continues action in time and space.* [12]. The end of one shot and the beginning of an other one is detected because of a break of continuous view. The breaks are called shot boundaries and are formed because of a camera breaks and editing points. In time and space semantically related shots are ussally grouped together to form a *scene*. In some cases relating scenes are grouped to make a sequence.

Moreover, physically because of video compression methods frames are not the same in video sequences. For example, in MPEG standart there are three types of frames: I-frame, P-frame, and B-frame. I-frame (**i**ntra-**frame**), sometimes called key-frame, is encoded as a single image, with no reference to any past or future frames. A P-frame (**p**redicted **frame**) is encoded relative to the past reference frame. And a B-frame (**bi**-directional predicted **frames**) is encoded relative to the past reference frame, the future reference frame, or both frames [7].

Logically, starting form the lowest level, one can assume that video stream consist of a sequence of a still images — *frames*. At this point we do not analyse a content of a frame. Single frames that are retrieved from video sequences in some cases can be analysed by image analysis techniques. We do not take it into account and claim that a frame is the smallest possible part of video sequence. In our case a shot is a continuous sequence of relative frames. A shot boundary is an editing point or a camera break. As in physical video structure, a scene is a sequence of related shots.

3.2 Keyframe Extraction in Raw Video Streams

As noted before, a concern of our research is a continues non-edited video streams filmed by body-mounted camera or filmed by route tracing equipment. Theoretically, such records consist of a sequence of frames. The view in the frame changes continuesly in time and space, without editing points or camera breaks. It can be considered as a single shot. Shot boundary techniques, e.g. [11], used on such streams do not give a sufficient result for condensed representation. As we are interested in logical video analysis, the semantic of keyframes is our main concern. There are several keyframe extraction methods developed, too [3, 13], and they all start from shot boundaries detection and only then the key frames are extracted.

Definition 1 *We define a keyframe is a frame in a shot which image is different in comparison to previous frames.*

Most of the algorithms, developed for shot cut or key frame detection, are tested on clean video streams, where shot boundaries and key frames are easy to determine. In this paper a method for raw video streams is developed. The challenge is that such streams usually are shaky and have a lot of noise in the view. The scene changes constantly or stays still underfined period of time and have no real shot boundaries. All changes are natural environmental changes caused by movement of a camera or movement in the environment.

Our method was develop on the bases of open source shot boundary detection program *Shotdetect* [11]. The original method of frame comparison was changed and user interface was addopted to the needs of our algorithm.

3.2.1 An Idea of the Algorithm

Video records that we analyse share the folowing characteristics: they have none editing points (any type, e.g.. cut, fade, dissolve, wipe, or computer generated transformations [1]), they are filmed by body mounted or other kind of wearable camera. Such records are shaky and sometimes noisy, they might have unfocused or low contrast view at some points. Moreover, the change of a scene is either constant (e.g. user is driving) or none (e.g. user is standing because of red traffic light).

We take a video record which satisfies these conditions and, starting form the first view frame, for every N^{th} frame we calculate measurement for frame comparison. If a frame is recognized as different enough compared to previous one, it is recorded as an image.

Each frame is divided into rectangular segments and for every segment hue histogram is extracted. Further, a difference of two histograms from respective segments of two frames is calculated. Histograms are considered different if the distance between them is greater than the selected percent of maximum possible distance. Two frames are considered different if a selected number of segments is recognized as different.

As a parameter for histogram, hue from the HSV color space was selected because this color space is much closer to how human eye really sees colors [8]. Althought, RGB color space is much easier to use in calculations, real understanding of an image is more important in image analysis applications (including video).

3.2.2 Parameters

The following parameters are used in the algorithm.

- *Step* — taking into account the fact that the view in a video stream has no cut change, a step of skiping frames in the process of the comparison is taken into account. In [5] similar parameter was applied. It is claimed that the parameter can be selected as a number of frames per second, and experiments were performed with a value equal to 22.

- *Dencity of pixels* — in order to make calculations faster the parameter of pixels dencity was added. If it is equal to 1, every pixel is taken into account, otherwise the number of pixels are skipped in every row and column.

- *Sectors* — number of rectangular sectors for dividing frame into regions. Possible value: 1 (whole frame), 4, 9, 16. Theoretically any number in power of 2.

- *Different sectors* — a maximum number of different rectangular sectors allowed for frame to be recognized as similar. The parameter has to be smaller than a number of sectors in a frame.

- *Histogram bins* — a number of histogram bins for dividing hue values into them.

- *Similarity percent* — two histograms are considered not similar if the distance between them is greater or equal to the percent of maximum possible distance between histograms.

It is important to note that the authors of the histogram comparison have tested their method on indoor scenes for visible pattern matching [10]. In case when we take *sectors* parameter equal to 1 and *different sectors* parameter equal to 0, we actually use their method for pattern matching. Unfortunately, they do not give a parameter when they consider two images similar.

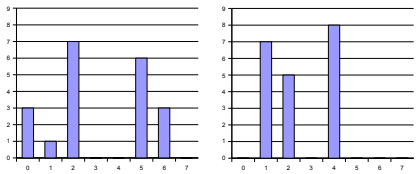
3.2.3 An Algorithm

A pseudocode of a video stream processing algorithm is given in Algorithm 1 and a pseudocode of comparing two frames is given in Algorithm 2. Some of the pseudocode methods are used in both of the algorithms and share the same name. A pseudomethod $F := ExtractVideoFrame(V)$ replaces a code that is necessary to extract a frame from a video stream. (We use methods that are described in FFmpeg [4] multimedia processing library.) A pseudomethod $CreateHistograms(F, dencity, sectors, bins)$ creates hue histograms for every rectangular sector in the frame F. It takes into account every $dencity^{th}$ pixel of a frame rows and columns and puts its hue value into one of $bins$. An auxiliary pseudomethod $val := PercentValue(bins, dencity, sectors, percent)$ is for calculating a value which later is used for comparing it with a distance between histograms to identify whether histograms are similar or not.

As mentioned above an Algorithm 2 describes how two frames are compared in order to decide whether they are similar or not. Firstly, we take histograms of a previous frame

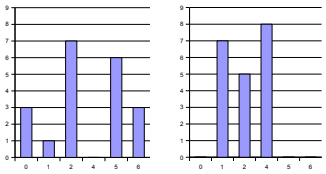
as follows: $H\_prev := GetHistograms()$, create hue histograms for a new frame — $CreateHistograms(F, dencity, sectors, bins)$ and take them in the same way: $H := GetHistograms()$. For every two corresponding histograms from current and previous frames an extended signatures are created.

The comparison of the histograms is done according the method described in [10]. Based on the data of two histograms the extended signatures are generated $\{S\_prev[i], S[i], length\} := CreateExtendedSignatures(H\_prev[i], H[i])$. According to the definitions of signature and extended signature, we take two histograms and analyse every pair of their elements. If any element of a pair is not equal to zero, the elements and the number of the histograms bin are recorded into respective extended signatures. *[...]Given a pair of signatures to be compared, the number of bins is the same. Moreover, each bin in both signatures represents the same bin in the histograms.* [10]. In Figure 2 the extended signatures of two histograms A and B in Figure 1 are shown. Together with each pair of extended signatures, a length of the original histogram is stored. It is necessary to know in further calculations.



1.1. H(A) 1.2. H(B)

Figure 1: Example of two histograms



2.1. ExS(A) 2.2. ExS(B)

Figure 2: An extended signatures of histograms in Figure 1

A comparison of the generated extended signatures is preformed in method, called $Dmod[i] := ModuloDistance(S\_prev[i], S[i], length)$. It is an implementation of *Modulo distance* algorithm, described in the same paper [10]. As our histograms are hue histograms they are modulo type histograms:

> *[...]In a modulo type histogram or signature [...] the first bin and the last bin are considered to be adjacent to each other, and hence, it forms a closed circle, due to the nature of the data type.* [10]

If modulo distance between two signatures (and histograms at the same time) is greater than the *val* parameter the histograms are considered not similar. Finally, if more sectors of the frames are recognized different than the $diff$ value, the new key frame is recorded. The current frame becomes a previous frame.

The described process compares frames till it reaches the end of the record.

3.2.4 Variations of the Algorithm

In Section 3.2.2 we have described a set of parameters we used in the algorithm. In this section the exact values combinations, used in the algorithm, is named.

- *Step* — a step between the frames to be compared. In most of the experiments it was 24. Some testing was also performed with a step which was equal to a frame rate per second of a selected records (in different records it might be different).

- *Dencity of pixels* — the final experiments were performed taking the value equal to 4 (every forth pixel in every forth line).

- *Sectors* — one of key parameters in our algorithm. The algorithm was tested when taking whole the frame (1 sector), 4, 9, or 16 rectangular sectors.

- *Different sectors* — a maximum number of different sectors allowed for similar frames. For whole frame the parameter is equal to zero, for others it must be smaller then the number of sectors. In variantions that were tested the different sectors parameter was smaller that a half of s sectors number.

- The algorithm is designed so that the number of *histogram bins* can be 2^n, where n theoretically can be any integer number. The algorithm was tested with $2^8 = 256$ bins.

Data: V – a video stream; *step* – frame step; *dencity* – dencity of pixels; *sectors* – number of frame sectors; $diff$ – number of different sectors; *bins* – number of histogram bins; *percent* – percent of distance between two histograms.
Result: KS – a set of selected keyframes.
begin
    $F := ExtractVideoFrame(V)$;
    $i := 0$;
    $j := 0$;
    $val := 0$;
    while $F \neq NULL$ **do**
        $j++$;
        if $i > 0$ **then**
            if $j = step$ **then**
                $CompareFrames(F, F\_prev,\ dencity, sectors, bins, val,\ diff)$;
                $i++$;
                $F\_prev := F$;
                $j := 0$;
        else
            $CreateHistograms(F, dencity,\ sectors, bins)$;
            $val := PercentValue(bins, dencity,\ sectors, percent)$;
            $SaveFrame(F)$;
            $i++$;
            $F\_prev := F$;
        $F := ExtractVideoFrame(V)$;
end

Algorithm 1: ProcessVideo

Data: F – frame; F_prev – previous frame; *dencity* – dencity of pixels; *sectors* – number of frame sectors; *bins* – number of histogram bins; val – comparison value; $diff$ – number of different sectors.
Result: If frames are not similar — record new keyframe.
begin
    $H\_prev := GetHistograms()$;
    $CreateHistograms(F, dencity, sectors,\ bins)$;
    $H := GetHistograms()$;
    $param := 0$;
    for $i \leftarrow 0$ **to** *sectors* **do**
        $\{S\_prev[i], S[i], length\} := CreateExtendedSignatures(H\_prev[i],\ H[i])$;
        $Dmod[i] := ModuloDistance(S\_prev[i], S[i],\ length)$;
        if $Dmod[i] >= val$ **then**
            $param++$;
    if $param > diff$ **then**
        $SaveFrame(F)$;
    $F\_prev := F$;
end

Algorithm 2: CompareFrames

- *Similarity percent* is probably the most important parameter in the algorithm, because it enables to charge whether two histograms are similar or not. Our selected values varied between 25 to 5 percents.

4 Experiments and Results

As mentioned shortly earlier, the described algorithm was implemented on the basis of open source program, called *Shotdetect* [11]. It is written using C++ programming language. Processing of video stream is based on methods from *FFmpeg* library, released on LGPL license [4].

Two groups of videos we experimented — video records filmed my body-mounted camera (slow but shaky movement, passing though objects, etc.) and videos recording route that were filmed form a car while driving through a city (stable enough, smooth movement and smoothly appearing and disappearing objects). For illustrating our results we use video records made while driving through a city.

Firstly, we would like to note that the parameter of the *step* is one of critical parameters in our application. For example, taking a very small step close or equal to 1, analysis time increases drastically and the method becomes very sensitive for such view noises as shadows of the trees. Because of this reason, a step value was empirically selected equal to 24. A small step has an other disadvantage too. In case, some object in a view is appearing and disappearing very slowly, small step is not able to detect environmental changes and keyframes showing this object is not created.

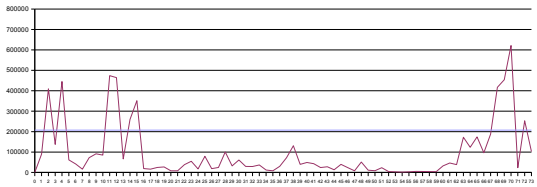
An other problem solved by using step value is recognition of zooms and pans. They are much easier to detect and keyframes are extracted in more meaningfull stages of zooming or panning.

In Figure 4 keyframe selection using histogram comparison method described in [10] is shown. Here hue histograms are extracted from the whole image for comparing image paterns. The allowed maximum difference of the histograms to be recognized as similar is 25% of maximum possible modulo distance between two histograms. Value of similarity was also selected empirically, because the authors of the histogram comparison method do not give information about the level of difference allowed in their experiments. The selected step value is equal to 24. It is used in order to be able to compare the results of this method and the results of the one we propose.

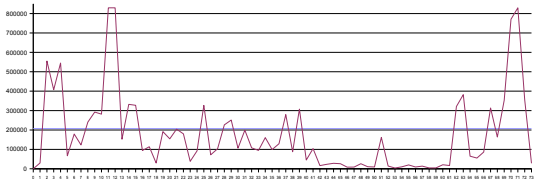
This method of analysis has a big disadvantage — it is very sensitive for small noisy motion and shadows (even using the selected step value). A number of key frames having differences only in shadows are recorded. On the other hand, a number of smoothly appearing objects is mising in the representation.

In the last page in Figure 6 an example of the same video sequence frames is shown. Here one can see every 24^{th} frame of 70 second length record. And in Figure 5 an extracted key frames of the same sequence using out proposed algorithm is given.

For the comparison of the parameters of the methods we give a graphical example in Figure 3. In Figure 3.1 a normalizes selection criteria of the whole frame is shown. And in Figure 3.2 a criteria for the same sequence using our proposed method dividing frame into 4 sectors is given. It is very obvious that when sectors are used selection criteria gets emphasised and more environmental highlights are selected into the sequence of the keyframes. In both charts the strait bold line marks the similarity percent value.



3.1.



3.2.

Figure 3: Selection criteria of the example video stream: Fig. 3.1 a normalized criteria using whole frame, and Fig. 3.2 a criteria of the frame of 4 sectors

5 Discusion and Future Work

The whole problem of key frame selection in a video sequence is hardly to define unambiguously.

4.1. 4.2. 4.3. 4.4. 4.5. 4.6.

4.7. 4.8. 4.9. 4.10. 4.11. 4.12.

Figure 4: An example output of testing the method described in [10] (Figure 4.12 is the last frame of the video sequence)

5.1. 5.2. 5.3. 5.4. 5.5. 5.6.

5.7. 5.8. 5.9. 5.10. 5.11. 5.12.

5.13. 5.14. 5.15. 5.16. 5.17. 5.18.

5.19. 5.20. 5.21. 5.22. 5.23. 5.24.

Figure 5: An example output of testing our proposed method (Figure 5.24 is the last frame of the video sequence)

Manual selection would probably give slightly different sequence of the keyframes. In some cases different users might select quite different frames depending on the content of the sequence and on their own attitude.

Our proposed method seems to work fine with a route tracking filmed video records as it is able to select those frames that represent such changes in the route as turns, new visual environmental objects. At the same time, such objects as other cars or clouds is a big disturbance in this kind of data and it forces new unmeaningfull keyframes to be extracted. One of possible ways for further research could be an attempt to avoid particular objects (differant cars) in selection process.

An other disturbance, for the method is cant changes in the view. As the method uses precise rectangular blocks such changes are detected as keyframes, although the view is very similar to the previous frames. A tolerance for view changes of cant manner should be implemented for the method.

Acknowledgement

We would like to acknowledge Siemens Master Program which allowed Stanislava Budvytytė to study with a status of visiting student in Digital Media master program in Bremen University. Also, we thank the Automemento project group — Lutz Dickmann, Tobias Lensing, supervisor Stéphane Beauregard and the rest members of the project group "AutoMemento meets Persuasive Technology" for support and collaboration in the initial stage of this research.

References

[1] C. Cotsaces, N. Nikolaidis, and I. Pitas. Video Shot Detection and Condensed Representation. A review. *Signal Processing Magazine, IEEE*, 23(2):28–37, 2006.

[2] A. D. Doulamis, N. D. Doulamis, and S. D. Kollias. A fuzzy video content representation for video summarization and content-based retrieval. *Signal Process.*, 80(6):1049–1067, 2000.

[3] A. M. Ferman and A. M. Tekalp. Two-stage hierarchical video summary extraction to match low-level user browsing preferences. *IEEE Transactions on Multimedia*, 5(2):244–256, 2003.

[4] FFmpeg. Last visited 2007.08.08, available at http://code.google.com/soc/2007/ffmpeg/about.html.

[5] A. Hanjalic. Shot-boundary detection: unraveled and resolved? *IEEE Trans. Circuits Syst. Video Techn.*, 12(2):90–105, 2002.

[6] C.-C. Lo and S.-J. Wang. Video segmentation using a histogram-based fuzzy c-means clustering algorithm. *Comput. Stand. Interfaces*, 23(5):429–438, 2001.

[7] MPEG.ORG. Last visited 2008.04.16, available at http://www.mpeg.org.

[8] M. J. Pickering and S. Rüger. Evaluation of key frame-based retrieval techniques for video. *Comput. Vis. Image Underst.*, 92(2-3):217–235, 2003.

[9] J. Sastre, P. Usach, A. Moya, V. Naranjo, and J. López. Shot Detection Method For Low Bit-Rate H.264 Video Coding. In *Proceedings of the 14th European Signal Processing Conference, Eusipco 2006*, Florence, Italy, September 2006.

[10] F. Serratosa and A. Sanfeliu. Signatures versus histograms: Definitions, distances and algorithms. *Pattern Recogn.*, 39(5):921–934, 2006.

[11] Shotdetect. Last visited 2008.04.16, available at http://shotdetect.nonutc.fr/shotdetect/.

[12] R. Tusch, H. Kosch, and L. Böszörményi. *VIDEX*: an integrated generic video indexing approach. In *ACM Multimedia*, pages 448–451, 2000.

[13] X. Zhu, J. Fan, A. K. Elmagarmid, and X. Wu. Hierarchical video content description and summarization using unified semantic and visual similarity. *Multimedia Syst.*, 9(1):31–53, 2003.

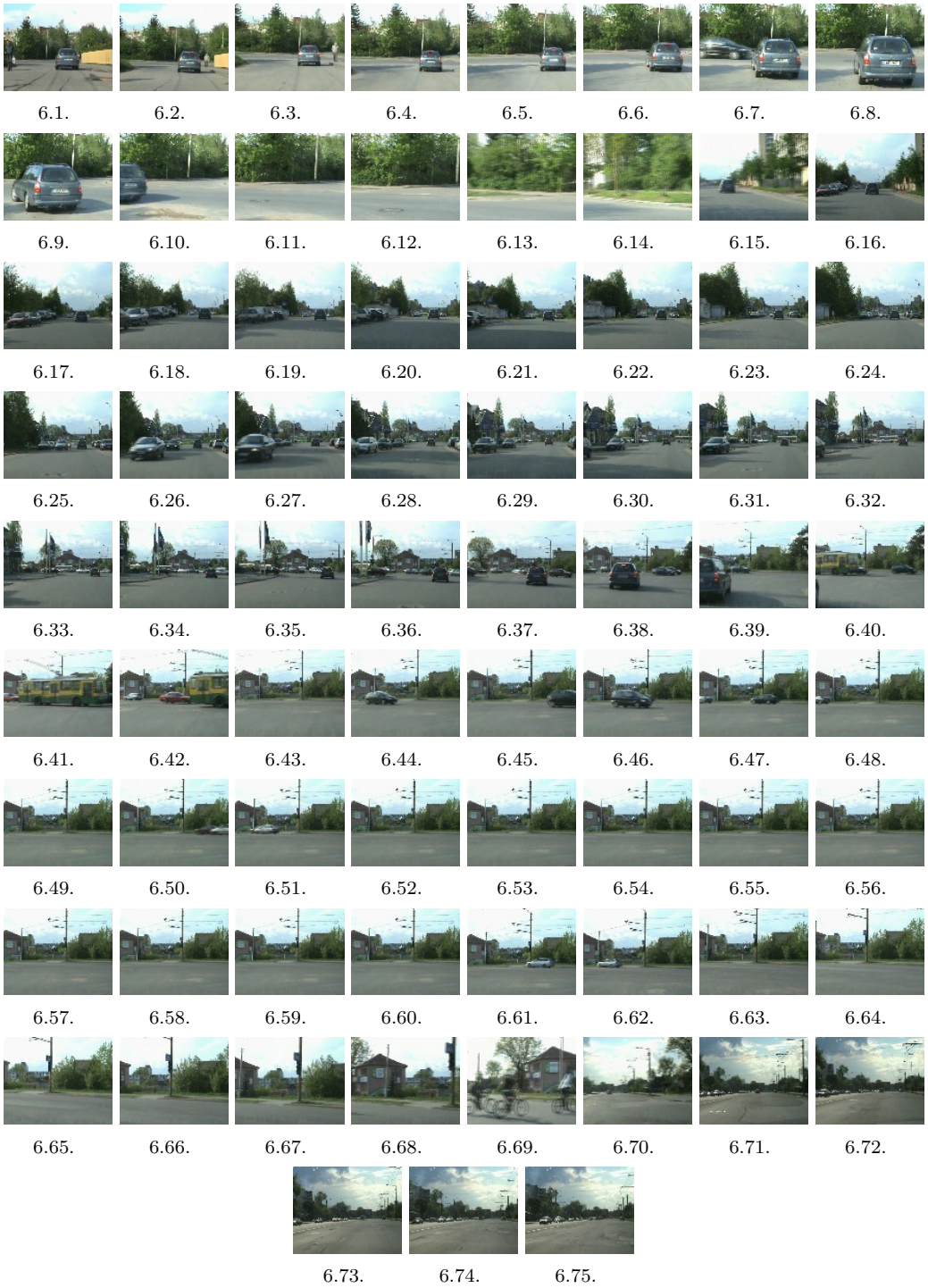


Figure 6: An example output of the test video. An every 24^{th} frame of the sequence is given (Figure 6.75 is the last frame of the video sequence)