NTTLAB AT TRECVID 2008

BBC Rushes Summarization Task

Uwe Kowalik[1] *Hidenobu Nagata*[2] *Yousuke Torii*[1] *Yongqin Sun*[1]
Kota Hidaka[1] *Go Irie*[1] *Mitsuhiro Wagatsuma*[1] *Yukinobu Taniguchi*[1]
Akira Kojima[1]

[1]NTT Cyber Solutions Laboratories, NTT Corporation
1-1 Hikarinooka Yokosuka-shi
Kanagawa 239-0847 Japan
+81 (0) 46 859 2258
E-mail: kowalik.uwe@lab.ntt.com

[2]NTT Electronics Corporation
1841-1 Tsuruma, Machida
Tokyo 194-0004 Japan
+81 (0) 42 796 2496
nagata_hidenobu@lsig.nel.co.jp

ABSTRACT

This paper presents our approach on the BBC rushes summarization task in the context of TRECVID2008. We combined cut detection and color histogram based features with DP matching for 'junk' frame removal and performed audio-visual event detection for calculating an importance measure in order to select video material subjected to inclusion into the final summary. Furthermore we present the novel **C**lipping **V**ariable **S**peed **F**ast **F**orward (C*VSFF*) algorithm that allows for effective video browsing of the summaries by a human editor.

1. INTRODUCTION

The 'nttlab' research group participated this year the first time in the TRECVID competition. This paper describes the approach taken by our group on the BBC rushes summarization task for the TRECVID 2008 summarization track. In contrast to typical high quality broadcast contents rushes are characterized by consisting of unedited raw material containing many 'junk' portions such as color bar frames, black and/or white frames, clap boards and poor audio quality. Moreover repeated shots are included due to multiple camera takes of the same scene under occasionally different view angles and distances. A human professional editor faces the problem of manually identifying 'junk' material that must not be included into the final broadcast contents as well as selecting 'good' takes for inclusion according to the story board. Automatic detection of (in) appropriate parts inside the raw material is required in order to decrease the human effort in browsing and selecting video material from rushes during the production process of broadcast contents and is therefore of huge benefit. The goal of the 2008 TRECVID rushes summarization task was to create summaries from 38 rushes with a maximum length of 2% of the original video length that should contain as much as possible of the given ground truth material while having as less as possible 'junk' and repeated material included. Moreover the submitted summary was assessed regarding its usability by a professional human editor. Fig. 1 shows the general flow chart of our approach.

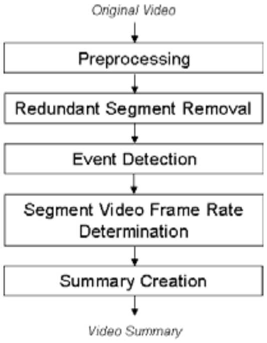


Fig. 1 General Workflow

In the *preprocessing* step visual features such as color histogram, scene cut points and shot segment information is calculated. Next 'junk' shots and repeated video segments are detected and removed in the *redundant segment removal* step. In the following *in-shot event detection* step events such as emphasized voice, close-up appearance and face presence are detected and an event score is calculated as the base for shot inclusion/exclusion decision. Each shot is then analyzed in order to determine an individual segment video frame rate in the *segment video frame rate determination* step with respect to the target summary length of 2%. The final summary is constructed in the *summary creation* step.

2. SUMMARY CREATION APPROACH

This section provides a detailed description of our approach on the rushes summarization task.

2.1. Preprocessing

The goal of the preprocessing is decoding, performing video segmentation into shots and feature extraction. As for the shot segmentation we derive shot boundaries from detected cut points. Cut point detection is performed by the approach described in [1].

We use two types of features: color histogram and frame difference. Each video frame is first divided into $k \cdot k = A$ rectangular regions of equal size (*k=3*). Next each region is filtered with a 3x3 averaging filter in order to remove noise. A label 1…A is assigned to each region in z-order. As for the color histogram feature an average color vector $\boldsymbol{d}_n^a$ is calculated were n is the frame index and a equals to the region index with $a \in [1 \ldots A]$. The color feature vector for describing a video frame consists of 27 elements. The frame difference vectors between consecutive frames are then calculated on top of the color feature vectors thus the difference vectors are of length 27 elements as well. Finally feature vectors are normalized.

2.2. Redundant Segment Removal

This step aims to get rid of unwanted video material (junk) such as black and/or white frames, frames containing color bars or a clapper board as well as shot repetitions and very short shots. This time we did not consider a specific technique for clapper board detection/removal.

2.2.1. Junk Shot Removal

a) Short segment removal

Based on the result of the cut point detection we identify *short segments* by applying a threshold to the duration. We define shots with a duration *T<1000ms* as 'junk shots' subjected to removal.

b) Color bar removal

As for the detection of color bar frames two *SAD* measures HCB_n and VCB_n describing the distance between histogram vectors of adjacent regions in horizontal and vertical direction are calculated by equation (1) and equation (2) respectively as illustrated in Fig. 2.

$$\mathrm{HCB}_n = \sum_{a=1}^{A-k} \left| \mathbf{d}_n^{a+1} - \mathbf{d}_n^a \right| \qquad (1)$$

with $a \neq Ck$ where C is an integer constant

$$\mathrm{VCB}_n = \sum_{a=1}^{A-k} \left| \mathbf{d}_n^{a+k} - \mathbf{d}_n^a \right| \qquad (2)$$

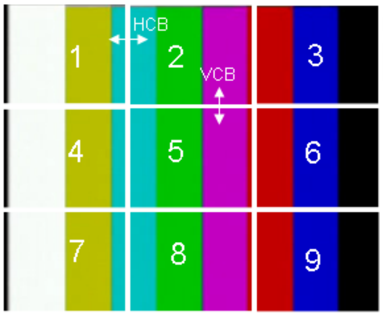


Fig. 2 *HCB* and *VCB* measures calculated between adjacent regions

c) Black/white frame removal

As for the detection of black or white frames we calculate an intra region measure BH_n which is the sum of vector norms as given in equation (3).

$$\mathrm{BH}_n = \sum_{a=1}^{A} \left| \mathbf{d}_n^a \right| \qquad (3)$$

We introduce a 'junk'-measure J_n which is calculated for each video frame n according to the rules given in below.

$$\mathrm{VCB}_n * \mathrm{HCB}_n < \varepsilon \rightarrow \mathrm{J}_n = \mathrm{J}_n + \mathrm{j}_{\mathrm{vcb/hcb}}$$

$$\mathrm{BH}_n > \mathrm{th}_{\mathrm{white}} \rightarrow \mathrm{J}_n = \mathrm{J}_n + \mathrm{j}_{\mathrm{BHwhite}}$$

$$\mathrm{BH}_n < \mathrm{th}_{\mathrm{black}} \rightarrow \mathrm{J}_n = \mathrm{J}_n + \mathrm{j}_{\mathrm{BHblack}}$$

Where n is again the frame index, $\mathrm{j}_{\mathrm{vcb/hcb}}$, $\mathrm{j}_{\mathrm{BHwhite}}$ and $\mathrm{j}_{\mathrm{BHblack}}$ are fixed increments. ε, $\mathrm{th}_{\mathrm{white}}$ and $\mathrm{th}_{\mathrm{black}}$ are arbitrary threshold values. A certain video frame n is labeled as 'junk' when J_n exceed a threshold *th* or labeled as 'good' otherwise. Shots containing many 'junk' frames are later excluded from the final summary.

2.2.2. Repeated Shot Removal

We detect repeated shots by employing a DP matching method. Our approach is illustrated in Fig. 3. A query vector **v** is constructed from the first N frames after the first cut point CP_t where all frames $n \in N$ lay inside the query interval $\mathrm{T}_{\mathrm{query}}$. We set $\mathrm{T}_{\mathrm{query}}$=2000ms during summary creation for our submission. The elements of ***v*** consist of the absolute differences $D = \left| \boldsymbol{d}_{t+\delta} - \boldsymbol{d}_t \right|$ between the frame color vectors $\boldsymbol{d}_n$ of adjacent frames. We add only those values D with $D < th_{max}$, i.e. considering only relatively static video portions for the matching process. As a result the length of $\boldsymbol{v}$ is dynamic. The vector $\boldsymbol{v}$ is then matched against the first frames inside an interval T_{query} after the following cut point CP_{t+T} in a DP matching step.

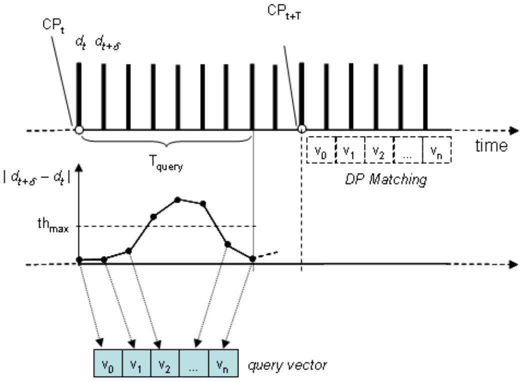
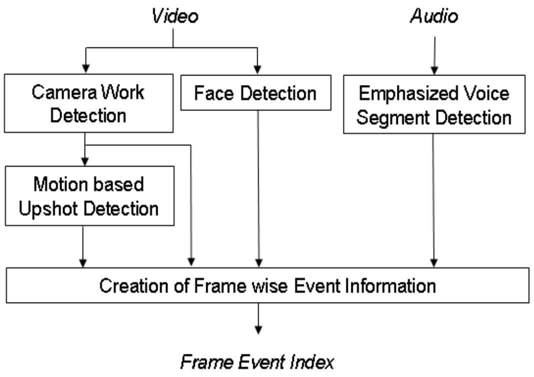


Fig. 3 DP matching

In the case that the matching result shows high similarity (above a certain threshold) we consider the following shot as a retake, label the previous shot (starting at CP_t) as junk and continue matching with the shot starting at the next cut point after CP_{t+T}. This methodology keeps the last shot of a sequence of similar shots and is based on the assumption that the last shot can be considered as the 'best' shot of a scene and should therefore be preserved.

2.3. Event Detection

After removing 'junk' frames from the original contents, prospective frame candidates for inclusion into the final summary are selected by multimodal event detection performed on the remaining material. As for the choice of detectors we have roughly analyzed the provided ground truth textual information regarding descriptive phrases and decided the architecture depicted in Fig. 4 for the event detection process. We detect four different events inside the video and audio track of the original video. An occurrence score $f_{Event} \in [0..1]$ for each event is calculated on a per frame basis.



Video
Audio
Camera Work Detection
Face Detection
Emphasized Voice Segment Detection
Motion based Upshot Detection
Creation of Frame wise Event Information
Frame Event Index

Fig. 4 Workflow of the in-shot event detection

We briefly introduce the employed event detection methods in the following subsections. We refer to existing publications in most cases and give a detailed description only when necessary. The main focus lies on the derivation of the normalized event scores used in *CVSFF* algorithm explained later.

2.3.1. Camera Work Detection

We describe the camera motion with 4 parameters using the 2D-Helmert transform, defined as given in (4).

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \frac{1}{z} \begin{pmatrix} 1 & -\theta & dx \\ \theta & 1 & dy \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \qquad (4)$$

where (dx, dy) is the translation along the x and y axis, z is the scaling parameter, and θ determines the rotation around the image center. We denote $p = (x, y)$ and $p' = (x', y')$ a point pair and $C = \{z, \theta, dx, dy\}$ denotes the parameter set describing the camera motion.

In the camera motion model, there are two assumptions: a) the salient object is positioned relatively near the camera; and b) the region that is relatively far from the camera can be described by a parallel motion model and is considered to be background.

Fig. 5 shows the flowchart of the camera motion parameter estimation.

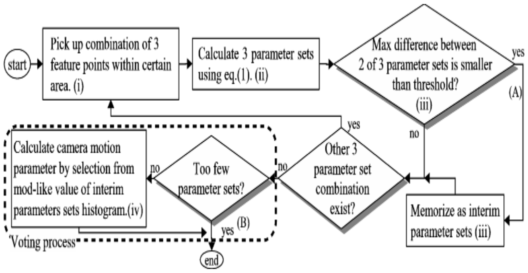

start
Pick up combination of 3 feature points within certain area. (i)
Calculate 3 parameter sets using eq.(1). (ii)
Max difference between 2 of 3 parameter sets is smaller than threshold? (iii)
yes
(A)
no
Memorize as interim parameter sets (iii)
Other 3 parameter set combination exist?
yes
no
Too few parameter sets?
no
yes
(B)
Calculate camera motion parameter by selection from mod-like value of interim parameters sets histogram.(iv)
'Voting process'
end

Fig. 5 Camera motion parameter estimation

(i) Steps (ii) and (iii) for all pairs of three points within a certain area are repeated.

(ii) Three parameter sets *C_1, C_2, C_3* are calculated from each pair of three points using equation (4).

(iii) Distances $C'_k = |C_i - C_j|$ between two of three parameter sets C_i where *(i,j,k = 1,2,3)*, are calculated. If two of the three C'_k are smaller than a threshold, the two parameter sets are saved as interim parameter sets C^*_n.

(iv) Mode-like value of each parameter in C^*_n is calculated as estimated parameter set C^*. Mode-like value means that the neighborhood value, which was selected in previous frame process, is selected from a certain number of greater peaks on the histogram of each parameter C^*_n.

The camera motion estimation is stabilized by filtering (iii). Interim parameter sets selected in step (A) of Fig. 5 are calculated for pairs of three points located in the background. When interim parameter sets C'_k are calculated with points that include a point on moving foreground objects, C'_k tends to be large since such objects often have stereoscopic motion and do not always follow the camera motion model. The reason only two parameters are used in step (iii) is to avoid overfiltering the parameter set calculated from points in the background because the employed camera motion model is an approximate model.

The estimated motion parameter vector $C = \{z, \theta, dx, dy\}$ is then classified into one of four camera motion patterns i.e. *zoom*, *pan*, *tilt* and *none*. As for the derivation of the frame wise camera work event score f_{camera} we use only the binary information that is *camera work present* or *not present*.

2.3.2. Close-Up Shot Detection

We consider a frame as containing a close-up shot when the following conditions are met:

(I) Camera motion estimation fails.

Camera motion estimation is considered to be a fail if the rest of the parameter sets are too few on parameter sets filtered by the combination of feature-point motions (in case (B) of Fig. 2). In such a case, a large portion of the video frame is occupied with moving objects and few feature points are detected in the background.

(II) Moving area rate, which approximates the area occupied by a moving object to the whole frame, is larger than a threshold. The moving-area rate is derived from *background points* (BPs) and *moving points* (MPs) as follows.

First, a feature point *p* is classified as either a BP, if the distance $\Delta' = |p' - q|$ is below a threshold; otherwise it is classified as an MP. Here, *q* denotes a feature point that corresponds to point *p* by feature tracking [2] and *p'* denotes a point estimated by the method described in subsection 2.3.1. A point classification example is shown in Fig. 6 where "X" and "O" indicate BPs and MPs, respectively.

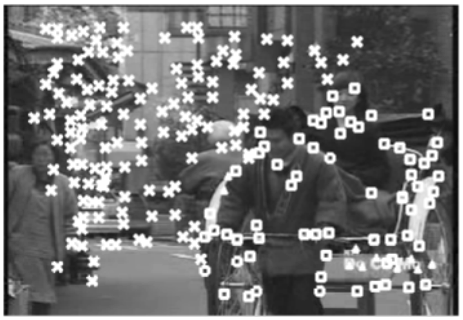


Fig. 6 Example of point classification into BPs and MPs

Next, P_n is defined as a collection of maximum n_a feature points p_i, taken in the order of distance from p_j that satisfy the conditions:

a) $|v_i - v_j| < V$ with *V=const.*

b) $|p_i - p_j| < D$ with *D=const.*

where $v_i = q_i - p'_i$, $v_j = q_j - p'_j$ denotes a point tracked from MP p_i by feature tracking

Condition a) is intended to extract points with similar motion as those on the same moving objects. Condition b) is introduced in order to avoid using points too far from point p_i points when there are too few neighborhood points. The average distance w_i of point p_i is calculated using the expression (5).

$$w_i = \begin{cases} 0 & n'_a = 0 \\ \dfrac{1}{n'_a} \sum_{j|p_j \in P_n} \left| p_i - p_j \right|^2 & n'_a \neq 0 \end{cases} \tag{5}$$

where n'_a is the number of points in P_n. The average distance w_i can be considered to be a rectangular approximation of the area represented by the point. Finally, the moving area E rate is determined. E approximates the size of a moving object and is calculated as given in (6).

$$E = \frac{1}{S_p} \sum_i w_i \tag{6}$$

where S_p denotes the area of a video frame. In case that E exceeds a threshold, the frame is detected as a close-up. Detected frames within a short time interval or in between the same cut points are merged in order to form a close-up frame sequence.

As for the frame wise upshot event score f_{up} we use the normalized E.

2.3.3. Face Detection

Frame wise face detection is performed by using the 'Joint Probabilistic Increment Sign Correlation' (JPrISC) approach proposed in [3]. This is an improved version of the traditional 'Probabilistic Incremental Sign Correlation' (PrISC) method introduced in [4]. An English introduction to a similar algorithm can be found in [5].

We derive a normalized face event score f_{face} by integrating the detection result over time with a summation filter of length T_{filter} = 2000ms. The number of detected faces per frame is ignored i.e. we use only the binary information whether faces were detected or not. Fig. 7 illustrates the face score derivation process.

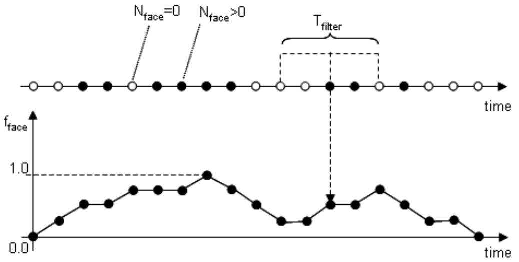


Fig. 7 Derivation of face event score f_{face}

2.3.4. Emphasized Voice Segment Detection

We consider video segments containing emphasized voice as important and therefore such segments should be considered as candidates for inclusion into video summary. We use the approach described in [6] for the detection of emphasized portions. We calculate the degree of emphasis D_E for each audio segment (T_{audio}=50ms) inside the audio track and assign the normalized value f_{emp} with $0 \leq f_{emp} \leq 1$ as the event score to each video frame according to the frame time.

2.3.5. Event Score Fusion

Event score fusion is performed by calculating the average event score for each video frame by formula (7).

$$f^{event} = \frac{1}{3} \sum_i \lambda \cdot f_i \tag{7}$$

with $i \in [up, face, emp]$ and

$$\lambda = \begin{cases} s & (f_{camera} = True) \\ 1.0 & (f_{camera} = False) \end{cases}$$

The camera work score f_{camera} is used for amplifying the individual event scores by a factor s in case that camera work is present. We used s=2.0 for the submitted summaries.

2.4. CVSFF and Summary Creation

The goal of the summary creation process is assembling video summaries of duration not longer than 2% of the original video length. In general there are two parameters that can be adjusted in order to achieve this goal: frame rate (speed) and number of frames. Considering the two extreme approaches one could increase the frame rate by 50 times or cut off 98% frames from the original video. Neither approach will lead to satisfying results (un-viewable or low content coverage). We have developed the CVSFF (Clipping Variable Speed Fast Forward) algorithm that aims to balance the costs introduced by speed improvement and cut off.

2.4.1. Algorithm Overview

In the following we give an introduction to the algorithm. Fig. 8 shows the workflow. We use following notation hereafter: j is the shot index, i is the frame index inside a shot and k is the iteration index. First reduction parameters $p_{k,j}^{speed}$ and $p_{k,j}^{cut}$ are calculated from shot length, event scores, frame differences and the target reduction rate.

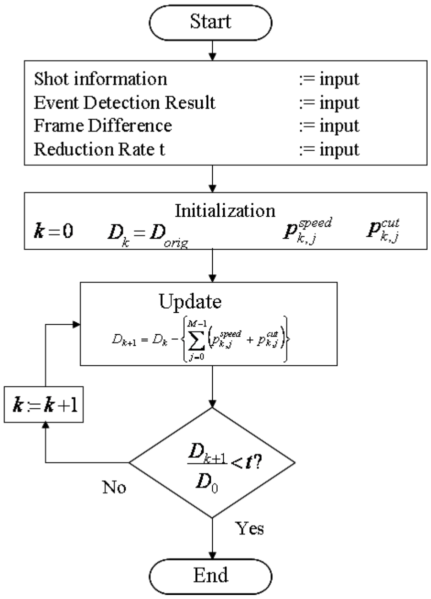

Start
Shot information := input
Event Detection Result := input
Frame Difference := input
Reduction Rate t := input
Initialization
$\boldsymbol{k}=0$ $\boldsymbol{D}_k=\boldsymbol{D}_{orig}$ $\boldsymbol{p}_{k,j}^{speed}$ $\boldsymbol{p}_{k,j}^{cut}$
Update
$D_{k+1}=D_k-\left\{\sum_{j=0}^{M-1}\left(p_{k,j}^{speed}+p_{k,j}^{cut}\right)\right\}$
$\boldsymbol{k}:=\boldsymbol{k}+1$
$\frac{\boldsymbol{D}_{k+1}}{\boldsymbol{D}_0}<\boldsymbol{t}?$
No
Yes
End

Fig. 8 CVSFF Workflow

The parameter $p_{k,j}^{cut}$ controls the shot reduction achieved by frame cut-off and reflects the importance of a shot with regard to event occurrence which is expressed by the average frame event scores of frames inside the shot (8) were f_i^{event} is the frame event score and $N_{k,j}$ is the number of frames inside a shot.

$$c_{k,j}^{event}=\frac{1}{N_{k,j}}\sum_{i=1}^{N_{k,j}}f_i^{event} \tag{8}$$

The parameter $p_{k,j}^{speed}$ controls the reduction of a shot by speed-up via frame rate improvement and considers the in-shot redundancy expressed by the average frame differences between adjacent frames as given in (9) were $|\boldsymbol{d}_{i,i+1,j}|$ is the norm of the frame difference vector between frame *i* and *i+1*.

$$s_{k,j}^{diff}=\frac{1}{N_{k,j}-1}\sum_{i=1}^{N_{k,j}-1}\left|\boldsymbol{d}_{i,i+1,j}\right| \tag{9}$$

$p_{k,j}^{cut}$ and $p_{k,j}^{speed}$ are calculated as given in (10) and (11), were $d_{k,j}$ is the shot duration α is a weight for controlling the contribution of each method and r_k a weight updated iteratively.

$$p_{k,j}^{speed}=\alpha\cdot\frac{d_{k,j}}{s_{k,j}^{diff}}\cdot r_k \tag{10}$$

$$p_{k,j}^{cut}=(1-\alpha)\frac{d_{k,j}}{c_{k,j}^{event}}\cdot r_k \tag{11}$$

The resulting summary duration is compared to the target summary duration (2%) after each iteration step and the algorithm finishes when $D_{k+1}/D_0 < t$.

2.4.2. Summary Creation

As one can see the denominators in (10) and (11) must not equal zero since this would result in either infinite frame cut-off or infinite speed. In order to avoid infinite cut-off we remove a shot from the summary in the case its event score is zero. This can be justified when considering those shots as unimportant in terms of contained events. As for shot reduction through speed-up an infinite speed would result in a shot duration of zero and theoretically occurs in case all frames are of similar content. However, such shots still have an importance score $c_{k,j}^{event}>0$ and should therefore not disappear (coverage). We maintain such shots by defining a minimum duration *H* and include H_j in the final summary.

Fig. 9 illustrates how we cut-off frames after each iteration. We equally remove frames right after/before detected cut points moving towards the shot center.

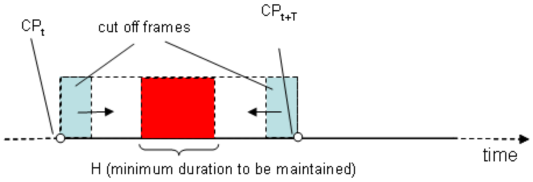


Fig. 9 Cutting of frames

3. EXPERIENCES

The NTTLAB team participated the first time in the TRECVID BBC rushes task. In general it was a very challenging and interesting work and a good opportunity for extending and integrating existing technologies into one framework. This time we did not perform explicit evaluation of contribution and performance of each component and refer to published work instead. However, we would like to give a brief overview of the lessons learned during the task.

Redundant Shot Removal

The used color histogram based features performed well especially for color bar frames and BW frames. As for the repeated shot removal our approach of evaluating only the first two seconds after each cut point by DP matching was a trade-off between performance and computational effort. Evaluating longer portions and using more complex features will probably lead to improvement.

Event Detection

The event detection was the most computational expensive task. Especially the face detection and camera work estimation contributed to the average computation time. Considering the type of provided contents, the face appearance seems to be a very useful feature summarization.

CVSFF

The novel ‘Clipping Variable Speed Fast Forward’ algorithm was developed during the TRECVID work. Although this time diverse parameters were decided on experimental basis, the algorithm has great potential when developing human centered approaches for video summary browsing that determine these parameters automatically in the future.

REFERENCES

[1] Y. Taniguchi, A. Akutsu, T. Tonomura, “PanoramaExcerpts: Extracting and Packing Panoramas for Video Browsing”, Proc. of the 5th ACM Multimedia, pp. 427 – 436, Seattle, USA, 1997

[2] J.-Y. Bouguet, “Pyramidal Implementation of the Lucas Kanade Feature Tracker”, Intel OpenCV document, 1999

[3] S. Ando, A. Suzuki, Y. Takahashi, T. Yasuno, “A Fast Object Detection and Recognition Algorithm Based on Joint Probabilistic ISC” , Proc. of Meeting on Image Recognition and Understanding MIRU2007, Hiroshima, Japan, July 2007 (Japanese)

[4] I. Murase, S. Kaneko, S. Igarashi, “Robust matching by increment sign correlation”, Trans. IEICE 2000, Vol. J83-D-2, No.5, pp.1321-1331 (Japanese)

[5] T. Mita, T. Kaneko, O. Hori, “Probabilistic ISC for Matching Images of Objects Having Individual Difference”, Systems and Computers in Japan, Vol.38, No.3, 2007

[6] K.Hidaka, S.Nakajima, Y.Niihara, “A New Multimedia Content Skimming Method based on Speech Emphasis Extraction and Its Application to Content Variations”, Proc. of the 8th IEEE International Symposium on Multimedia (ISM'06)