Video Understanding and Content-Based Retrieval

Yun Zhai[1], *Jingen Liu*[1], *Xiaochun Cao*[1]
Arslan Basharat[1], *Asaad Hakeem*[1], *Saad Ali*[1], *Mubarak Shah*[1]
[1]School of Computer Science
University of Central Florida

Costantino Grana[2], *Rita Cucchiara*[2]
[2]Dipartimento di Ingegneria dell'Informazione
Universita' degli Studi di Modena e Reggio Emilia

1 Introduction

This year, the joint team of UCF and the University of Modena has participated in the following tasks: (1) shot boundary detection, (2) low-level feature extraction, (3) high-level feature extraction, (4) topic search and (5) BBC rushes management. The shot boundary detection was contributed by the Image Lab at the University of Modena. The other tasks were performed by the Computer Vision Team at UCF.

1.1 Shot Boundary Determination

This is done by the Image Lab at the University of Modena. The results submitted to NIST have been obtained using frame based comparisons to detect if the difference behavior over time resembled that of a linear transition. Using this approach we can detect with the same algorithm cuts, linear dissolves and wipes, and part of many graphical effects. The distance between frames employed is based mixed pixel and histogram based. For the pixel based, we used the sum of squared distances on the Y channel, while for histograms the χ^2 distance was used. Two parameters need usually to be tuned: $Peek_{\frac{n}{w}}$ and $err_{\frac{n}{w}}$. The second one was not used in the TRECVID2005 experiments, since it didn't add much to the precision, sometimes reducing recall rate, in contrast with what observed on Formula1 video sequences which are much more regular. In fact it reports how well the measure shape fits the linear model, so usually causes problems on the non linear transitions. To get the different runs, we simply changed the peek coefficient value in the equation:

$$Peek_{\frac{n}{w}} \cdot C_{Peek} > 1. \quad (1)$$

The formula is written this way, because also the error part was contributing to the total. Of course in this way, we are just thresholding the peek value.

runid	C_{Peek}
TRECVID2005_UNIMORE_26	0.00026
TRECVID2005_UNIMORE_27	0.00027
TRECVID2005_UNIMORE_28	0.00028
TRECVID2005_UNIMORE_29	0.00029
TRECVID2005_UNIMORE_30	0.00030
TRECVID2005_UNIMORE_31	0.00031
TRECVID2005_UNIMORE_32	0.00032
TRECVID2005_UNIMORE_33	0.00033
TRECVID2005_UNIMORE_34	0.00034
TRECVID2005_UNIMORE_35	0.00035

Changing the threshold value allowed us to move along the recall/precision curve, but no significant difference was observed between runs. Probably the value should have been changed by a larger amount.

Based on this results, it is clear that the initial hypothesis of linear transition is a bit too strong for general video material, because the obtained results are lower than expected on the transition. Looking at the videos, we observed that many "special effects" were missed. On the contrary searching for the best transition position and length allowed to fine tune around cuts, proving particularly effective.

Lessons learned: a flash detector is needed for news videos; our estimated transition length was too conservative and obtained extremely high precision, at the cost of recall. This is not a matter of thresholds, since it is the output of an optimization procedure and is not influenced by anything else but the data. We need to speed up the detection, probably by including some early stop conditions in the optimization. The detailed description of the proposed method is in Section 2.

1.2 Low-Level Feature Extraction

The detection of the motions is based on the analysis of the homography transformation and the fundamental matrix between two consecutive frames. We have submitted four runs:

1. VISION1: ranked results using the global displacement, motion continuity and the period of the motion.
2. VISION2: ranked results using the global displacement only.
3. VISION3: ranked results using the global displacement and the motion continuity.
4. VISION4: ranked results using the motion continuity and the period of the motion.

The major contribution of our system is that, the system is able to distinguish the 2D motions caused by 3D camera rotations (pan/tilt) from the ones caused by 3D camera translations (track/boom). The detections of pans and tilts are based on the analysis of the homography transformations of the images, while tracks and booms are detected by analyzing the epipolar geometry of the images. The detailed description of the proposed method is in Section 3.

1.3 High-Level Feature Extraction

The task of high level feature extraction has been implemented using three approaches. First method was based on global features that were subdivided into fixed sized patches. . This technique exploits consistent appearance of similar patches in a specific part of the image. This technique was useful for detecting features that occupy most of the image and had a lot of in-class variation, such as mountain, building, waterscape, and sports. This technique used the spatial constraint of the local features for high level feature detection but it has certain limitations. This includes the problems due to the change in camera orientation and zoom, partial occlusion of features, etc. Second approach was based on local features of image segments. This technique was useful for detecting features that had a certain structure but still had in-class variation, such as explosion/fire, map, US flag and car. The effectiveness of this technique was dependent on the quality of image segmentation. Over-segmentation (too many segments) was preferred over under-segmentation (very few segments) as two different regions with very similar features should have very close feature points. On the other hand under-segmentation might fuse two actually different regions together into one segment with a common feature vector. This problem had been observed in some tests and might have a bad influence on the results. Third approach used feature points and patch appearance similarity. This method was useful for the features that have a consistent appearance signature and with small in-class variation, such as US flag. The rest of the features had diverse appearance; hence can not be detected by using patch similarity. Using these methods, results were submitted for all features except people walking/running. Please refer to Section 4 for the detailed description of the proposed method.

1.4 Topic Search

The search system, PEGASUS, is implemented with four components: ASR, OCR, global histograms of the keyframes and the high-level semantic lexicons.

1. UCFVISION1: run based only on ASR.
2. UCFVISION2: run based on ASR, OCR, global histogram and high-level features.

Based on the evaluation results, the interactive run using multi-modalities performs better than the one using only the ASR information. For the second run, we have achieved median performance among all the runs. The search baseline was provided by the search on the ASR. The other features expanded the baseline with more positive returns, and the interactive mechanism allows the user to refine the query and improve the performance. The detailed description of the proposed method is in Section 5.

2 Shot Boundary Determination

Automatic tools for video segmentation and annotation are the chimera of all digital video library management systems. The goal is to find automatic and general procedures to segment videos into blocks and to annotate them with textual data or with metric information that could be useful for further indexing, querying, summarization, fast browsing and so on.

In this paper, we propose a new two steps iterative algorithm, which relies on a linear transition model, able to identify transition center and length. Our approach is strictly focused on gradual transitions with a linear behavior, including abrupt transitions. A precise model is exploited allowing achieving more discriminative power than general techniques. We developed an iterative algorithm that, given a frame of possible transition, alternatively tries to find to best center position for the transition and the best length, by minimizing an error function, which measures the fitness of data to the linear model.

Before describing our algorithm in detail, it is useful to define the ideal model of linear transition and to underline

its important properties. These will be exploited by the algorithm to cope with non idealities and to measure the confidence of the detection.

2.1 The Transition Model

Let's consider two consecutive shots in a video sequence, the first one ending at frame e, and the second one starting at frame s, with $e < s$. If $s = e + 1$ we have an abrupt cut, otherwise there are some frames of gradual transitions between e and s.

To design a shot segmentation algorithm, two assumptions must be done: the first one is that a feature $F(t)$ is computable for each frame at time t, with the characteristic of being discriminating and almost constant within the shot; ideally

$$\begin{array}{l} F(t) = F(e), \forall t \leq e \\ F(t) = F(s), \forall t \geq s \\ F(e) \neq F(s) \end{array} \quad (2)$$

The second assumption is that a distance function exists in the feature space Φ: $d : \Phi \times \Phi \rightarrow \mathbb{R}$, which shows a constant behavior during the transition. Ideally:

$$d(F(t), F(t-1)) = c \quad e < t \leq s \quad (3)$$

Sometimes there is confusion on the definition of length of a transition, because one may include in the count the first frame of the new shot after the transition (e.g. [1]), or the last one of the previous one. In our model, the length is the number of frames in which the transition is visible, that is $L = s - e - 1$. Note that this model includes in the definition of transition abrupt cuts too, as transitions with length $L = 0$. The transition center is defined as $\overline{n} = (e + s)/2$ and may correspond to a non-integer value, that is an inter-frame position. This is always an inter-frame position in case of cuts.

Differently from other difference metric formulations, instead of computing the difference between the frames $F(i)$ and $F(i + w)$, with w being the *frame-step*, we calculate a metric M_w^n centered on frame or half-frame n, with $2n \in \mathbb{N}$, and with frame-step $2w \in \mathbb{N}$. It is defined as:

$$M_w^n = \begin{cases} d[F(n-w), F(n+w)] & n + w \in \mathbb{N} \\ \frac{1}{2}\left[M_w^{n-\frac{1}{2}} + M_w^{n+\frac{1}{2}}\right] & otherwise \end{cases} \quad (4)$$

The second term of the expression is a linear interpolation adopted for inter-frame positions. This is necessary because the feature F is relative to a single frame and cannot be computed at half-frames. The reason for expressing the metric as $d[F(n-w), F(n+w)]$ instead of $d[F(n), F(n+2w)]$ will be explained in section 2.2.2.

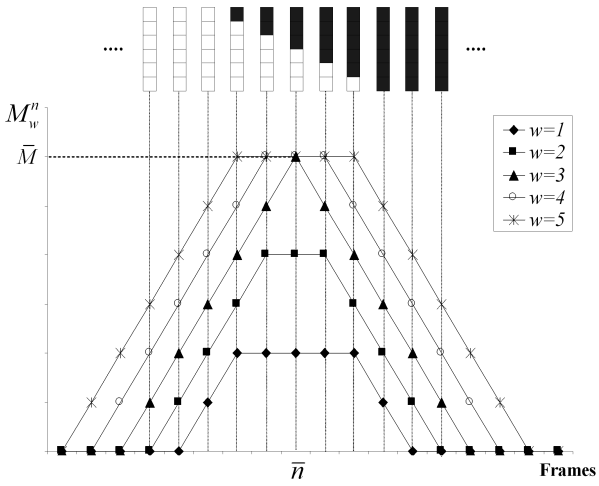


Figure 1: Values of M_w^n for an ideal linear transition with $L = 5$ at varying w.

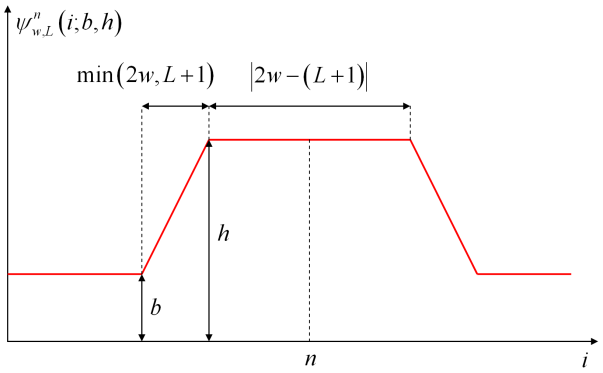


Figure 2: Trapezoidal shaped function $\psi_{w,L}^n(i; b, h)$

In Fig. 1 we see an example of an ideal linear transition with $L = 5$, from a shot with white pixels to one with black pixels. If the transition is perfectly linear according with the hypothesis of Eq. 2 and Eq. 3, the shape of function M_w^n is an isosceles trapezoid centered in $\overline{n}$, for each w, that degenerates into a triangle when $2w = L + 1$.

We can verify that in this ideal case, given the model and Eq. 4, both the up and down slopes last for $\min(2w, L + 1)$ frames, and that the plateau of absolute maximum is $|2w - (L + 1)|$ long. It's also straightforward to verify that:

$$\begin{array}{ll} M_w^{\overline{n}} < \overline{M}, & \text{if } 2w < L + 1 \\ M_w^{\overline{n}} = \overline{M}, & \text{if } 2w \geq L + 1 \end{array} \quad (5)$$

where $\overline{M} = \max_{w,n} M_w^n$ (see Fig. 1). We define $\psi_{w,L}^n(i, b)$ the generic trapezoidal function, centered in n, whose value is M_w^n at the center (the absolute height of the minor base) and b is the value outside the trapezoid.

The function is plotted in Fig. 2. We define $\psi^n_{w,L}(i) = \psi^n_{w,L}(i,0)$, the function which corresponds to the ideal transition case.

In the real case, camera and objects motion, color and luminance variation and so on cause the feature F to be non constant on the shot, thus making Eq. 2 and Eq. 3 not satisfied. The consequence is that the shapes of both the slopes and the plateau are usually disturbed.

2.2 Two-Step Algorithm

Due to lack of ideality in most of the shot transitions, instead of relying only on correlation between data and the ideal $\psi^n_{w,L}(i)$ function, we employ an algorithm constructed of two steps: the first one searches for the transition center position n, assuming a fixed frame step $2w$, and the second searches for the transition length L, by trying different values of w, but keeping the transition center fixed. While in the ideal case even the first step would be sufficient, in real cases an error in locating the center position would also lead to a wrong estimate of the length. For this reason a second step is introduced to provide a different view of the function behavior, a possible confirmation on the first step outcome and a new estimate for the window size. Iteratively repeating the two steps allows progressively decreasing the error. In this section we explain in details our transition detection algorithm. We perform the following analysis on overlapped windows of 30 frames, distant 15 frames each other, since we suppose that transitions are much shorter and farther than that.

2.2.1 First step

In the first step the values of M^n_w are calculated using the frame-step $\overline{w}$, which is found in the previous iteration of the algorithm, or it's arbitrary chosen for the first iteration. The best trapezoid $\psi^n_{w,L}(i)$ is searched by moving the center n, and trying different values for L, but keeping $\overline{w}$ fixed. The trapezoid extends over $\delta = \min(2w, L+1) + |w - (L+1)/2|$ frames on the left and on the right of the center frame. For each couple of n and L the following matching measure is computed:

$$\Lambda^n_{\overline{w},L} = \sum_{i=n-\delta}^{n+\delta} \min\left(M^i_{\overline{w}}, \psi^n_{\overline{w},L}(i)\right) - \sum_{i=n-\delta}^{n+\delta} \left|M^i_{\overline{w}} - \psi^n_{\overline{w},L}(i)\right| \quad (6)$$

The value of n is searched within the 30 frames window, and also L must be selected such that $n+\delta$ and $n-\delta$ don't exceed the window.

In Eq. 6, two components are evident: the first one is needed to maximize the area under the trapezoid, while the second component describes the similarity of our linear hypothesis with the data. It is very important to include both components, since we expect the distance measure to give a trapezoidal shape (the second term in Eq. 6), but we also request its *strength*, i.e. the amount of difference between the first and the second scene, to be significant. The first term in Eq. 6 in fact describes how much the value of M^n_w surpasses the ideal trapezoid. After finding the trapezoid which maximizes $\Lambda^n_{\overline{w},L}$, we consider $\overline{n} = \arg\max_n \Lambda^n_{\overline{w},L}$ the candidate transition center. In Fig. 3 we show an example of trapezoid fitting with real data.

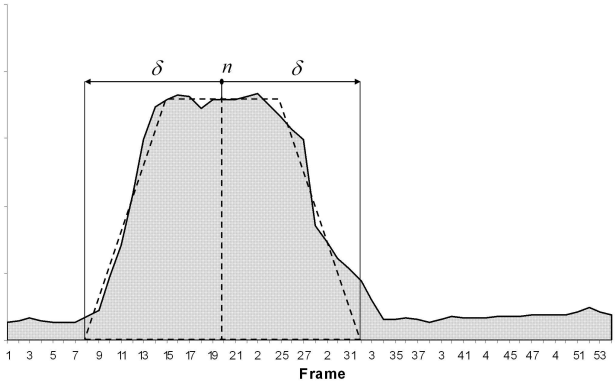


Figure 3: Example of real M^n_w values and the best trapezoid fitted.

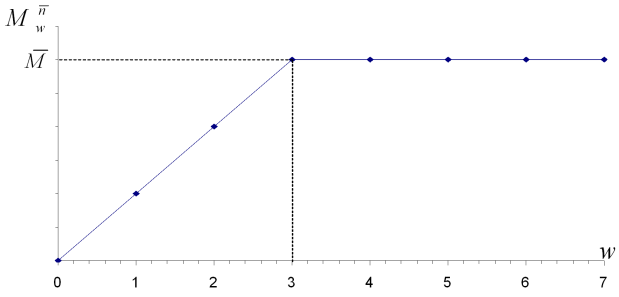


Figure 4: Values of the distance metric M^n_w, with respect to different w values. This corresponds to the transition of Fig. 1.

2.2.2 Second Step

Thanks to the definition of M^n_w as a distance function centered in n, as in Eq. 4, increasing the frame-step w makes the value of M^n_w to grow up to an absolute maximum when $w = (L+1)/2$ and then to be stable. It is easy to demonstrate that, in the ideal case, this growth is linear. Thus the growing function can be plotted as shown in Fig. 4, with a linear slope followed by a horizontal line, when the value of M^n_w is stable. The second step of the algorithm uses

this propriety to give an estimate of the transition length, by finding the smallest w which maximizesM_w^n. To provide a technique able to deal with noise, the tilt change of the chart is searched by minimizing the function:

$$Z_w^{\overline{n}} = \sum_{i=0}^{w} \left| M_i^{\overline{n}} - \frac{M_w^{\overline{n}}}{w} i \right| + \sum_{i=w+1}^{W} \left| M_i^{\overline{n}} - M_w^{\overline{n}} \right| \quad (7)$$

where W is the maximum size that a transition can assume. The w value that minimizes $Z_w^{\overline{n}}$ becomes our current frame step for the next iteration of the algorithm.

In simple cases the algorithm progressively narrows the trapezoid minor base leading to the expected triangular shape. Convergence is not guaranteed in non ideal conditions, and, for this reason, we add a convergence constraint: at each iteration the minor base of $\psi_{w,L}^n(i)$ is forced to become smaller. In Fig. 5 the M_w^n values are shown for 4 successive iterations of the algorithm in a real gradual transition case. At each iteration, we achieve a more precise estimate of the transition center and length, and thus a shape more similar to a triangle.

2.2.3 Decision Space

Given the transition length $L = 2w - 1$ and its center $\overline{n}$, as detected by the algorithm, the function $\psi_{w,L}^n(i)$ becomes triangular shaped. We must now verify the significance of the transition and how much the real data fit to the linear transition model. We introduce the following measure:

$$Peak_{\overline{w}}^{\overline{n}} = M_{\overline{w}}^{\overline{n}} - \min\left(M_{\overline{w}}^{\overline{n}-2\overline{w}}, M_{\overline{w}}^{\overline{n}+2\overline{w}}\right). \quad (8)$$

The Peak value measures the height of the center value with respect to the lower of the two values of M in correspondence to the extremes of the triangle, and provides information on the transition significance. In fact, while in the model $M_w^{n\pm 2w} = 0$, in real cases this is not true, because of object and camera motion that causes the feature F to be not constant before and after the transition. To cope with this we have to get rid of the hypothesis of having an isosceles triangle and define the fitting error measure as:

$$\begin{aligned} err_{\overline{w}}^{\overline{n}} = \tfrac{1}{4\overline{w}} \sum_{i=1}^{2\overline{w}} & \left| M_{\overline{w}}^{\overline{n}-i} - \psi_{\overline{w},L}^{\overline{n}}(\overline{n}-i, M_{\overline{w}}^{\overline{n}-2\overline{w}}) \right| \\ & + \left| M_{\overline{w}}^{\overline{n}+i} - \psi_{\overline{w},L}^{\overline{n}}(\overline{n}+i, M_{\overline{w}}^{\overline{n}+2\overline{w}}) \right| \end{aligned} \quad (9)$$

The error sum is divided by the triangle's base $4\overline{w}$ to obtain a measure which is independent from the transition length. A minimum threshold on the Peak value, $T_P$and a maximum threshold on error, T_E, are employed to discriminate real shot changes from false ones. The final decision space is then based on two parameters only which are the same for cuts and transitions.

2.3 Conclusions

We presented an algorithm for shot detection able to detect both abrupt cuts and gradual transitions in sport videos. Experimental results and comparisons show that our algorithm performs better than other techniques at the state of the art, and that it requires only two parameters, thus making the learning process easy.

3 Low-Level Feature Extraction

The low level features are the global motion models: (1) pan/track: horizontal motion of the camera; (2) tilt/boom: vertical motion of the camera; (3) zoom: camera focal length change. In this task we have employed the analysis of the transformations between images, including the homography and the fundamental matrix. Instead of classifying the pan and track as a single feature, we separate their detections based on different methods. Pan and tilt motions are caused by camera 3D rotation, while track and boom are caused by 3D camera translation. Image motions caused by 3D camera rotations can be formulated by the homography transformation, which is insufficient to model the motion that are caused by 3D translation, when depth information exists. Therefore, we detect the pan, tilt and zoom using the homography analysis, and detect the track and boom based on the fundamental matrix analysis. Finally, the results of the pan and the track detections are merged to produce the results of feature 1, and the results of the tilt and the boom detections are merged for the results of feature 2. In the proposed method, we used the SIFT operators [5] as the sparse optical flow to estimate the frame transformations.

3.1 Geometric Analysis of Camera Motions

In this section, we model the camera motion between two image frames by the following motion parameters (see Fig. 5): (1) pan angle A_x (left or right): rotation angle around the Y-axis, (2)tilt angle A_y (up or down): rotation angle around the X-axis, (3) zoom factor s, the ratio of the camera focal lengths between two image frames, and (4) translation vector $\mathbf{v} = (v_x,\ v_y,\ 1)^T$. The first two cases can be considered as "pure" rotations when there is no change in the camera's intrinsic parameters. The last case deals with a pure translation without any rotation or any change of the intrinsic parameters. Horizontal to/from camera movement is called "tracking" and is usually executed with a movable "dolly". Vertical translation movements are called "booming" or "craning" and are accomplished with a "crane" or "jib".

Before the analysis of different camera motion, we first introduce the pin-hole camera model that models a real world camera. Then, for each different camera motion, we

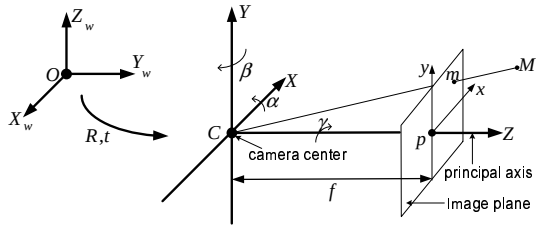


Figure 5: The extrinsic parameters of a pin-hole camera represent the rigid body transformation between the world coordinate system (centered at $\mathbf{O}$) and the camera coordinate system (centered at $\mathbf{C}$). The intrinsic parameters, e.g. the focal length f, stand for the camera internal geometry.

give geometric explanation of their properties and present the method to compute the motion parameters, e.g., the rotation angles and the zoom factors.

3.2 Pin-hole Camera Model

A pin-hole camera (see Fig. 5), based on the principle of collinearity, projects a region of $\mathbb{R}^3$ lying in front of the camera into a region of the image plane $\mathbb{R}^2$. As is well known, a 3D point $\mathbf{U} = [X\ Y\ Z\ 1]^T$ and its corresponding 2D projection $\mathbf{u} = [u\ v\ 1]^T$ in the image plane are related via a 3×4 projection matrix $\mathbf{P}$ as,

$$\mathbf{u} \sim \mathbf{PU} = \mathbf{K}[\mathbf{R} \mid \mathbf{v}]\mathbf{U}, \quad \mathbf{K} = \begin{bmatrix} f & \gamma & u_0 \\ 0 & \lambda f & v_0 \\ 0 & 0 & 1 \end{bmatrix}, \quad (10)$$

where $\sim$ indicates equality up to a multiplication by a non-zero scale factor, $\mathbf{R}$ is the 3×3 orthonormal rotation matrix , $\mathbf{v} = -\mathbf{RC}$ is the translation vector, where $\mathbf{C} = [C_x\ C_y\ C_z]^T$ represents the coordinates of the camera center in the world coordinate system, and $\mathbf{K}$ is a non-singular 3×3 upper triangular matrix known as the camera calibration matrix, which has five parameters: the focal length f, the aspect ratio λ, the skew factor γ accounting for non rectangular pixels, and the principal point $(u_0,\ v_0)$. The principal point is the intersection between the optical axis and the image plane. The intrinsic parameters in $\mathbf{K}$ define the internal imaging geometry of the camera, while the extrinsic parameters ($\mathbf{R}$ and $\mathbf{v}$) relate the world coordinate system (centered at $\mathbf{O}$) with the camera coordinate system (centered at $\mathbf{C}$).

3.3 Pure Rotation - Pan / Tilt

Algebraically, if $\mathbf{u}$ and $\mathbf{u}'$ are the 2D projections of a 3D scene point P before and after the pure rotation. For simplicity purpose, the world coordinate system is chosen to coincide with the camera's, so that the camera projection matrix $\mathbf{P} = \mathbf{K}[\mathbf{I} \mid \mathbf{0}]$, and,

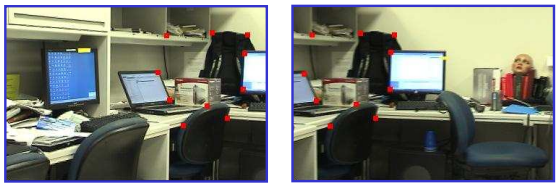


Image 1 Image 2

Figure 6: Two images from a pan sequences. The red points represent the correspondences in the images. The rotation axis of this sequence is $[0.0007, 1.0000, 0.0000]$, and the magnitude of the rotation angle is $16.75°$.

$$\begin{aligned} \mathbf{u} &= \mathbf{K}[\mathbf{I} \mid \mathbf{0}]\mathbf{U}, \\ \mathbf{u}' &= \mathbf{K}[\mathbf{R} \mid \mathbf{0}]\mathbf{U} = \mathbf{KRK}^{-1}\mathbf{K}[\mathbf{I} \mid \mathbf{0}]\mathbf{U} = \mathbf{KRK}^{-1}\mathbf{u}, \end{aligned} \quad (11)$$

so that $\mathbf{u}' = \mathbf{Hu}$ with $\mathbf{H} = \mathbf{KRK}^{-1}$. This 2D homography $\mathbf{H}$ is a conjugate rotation that has the same eigenvalues (up to scales) as the rotation matrix $\mathbf{R}$, namely $\{\mu, \mu e^{i\theta}, \mu e^{-i\theta}\}$, where μ is an unknown scale factor (if $\mathbf{H}$ is scaled such that $\det(\mathbf{H}) = 1$, then $\mu = 1$).

Consequently, the eigenvector of $\mathbf{H}$ corresponding to the real eigenvalue is the vanishing point of the rotation axis, the location of which can be used as the criteria to decide the camera rotation. Ideally, panning motion has the rotation axis exactly parallel to the image y-axis, and with noise present in the real data, the rotation axis should be almost parallel to the image y-axis, and the axis vanishing point should be located virtually at infinity in the y direction. Similarly, when the camera is tilting, the axis vanishing point should be located virtually at infinity in the x direction. Therefore, the angle of rotation between views can be computed directly from the phase of the complex eigenvalues of $\mathbf{H}$.

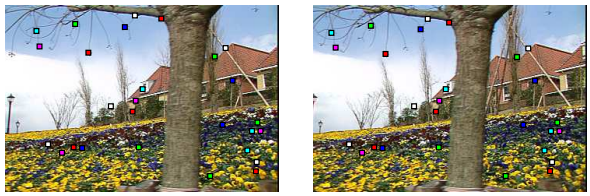
For example, between images shown in Fig.6, there is a pure horizontal rotation (pan) of the camera. The corresponding points are automatically found by the SIFT operator [5]. Based on the correspondences, the computed homography $\mathbf{H}$ is,

$$\mathbf{H} = \begin{pmatrix} 1.1938 & 0.0108 & -235.7323 \\ 0.0289 & 1.1420 & -8.3286 \\ 0.0005 & 0.0000 & 1.0000 \end{pmatrix}.$$

From $\mathbf{H}$, the angle of the horizontal rotation is estimated as $16.75°$, and the rotation axis vanishing point as $[0.0007, 1.0000, 0.0000]$, i.e., it is the y-axis. Therefore, we classify this shot as a panning shot based on the analysis we have made above.

Image 1 Image 2

Figure 7: Two images from a zooming sequences. The red points represent the correspondences in the images. The scaling factor $\lambda \approx 1.55$, and the principle point is $[177.9,\ 116.0]$, which is very close to the image center (image size is $[360] \times [240]$).



(a) (b)

Figure 8: A pair of frames from a horizontal translation shot, (tracking right). The epipole (direction of the translation) locates at $[0.9742,\ -0.2258,\ -0.0012]$.

3.4 Zooming

In the case of zooming, the motion between images can be approximated as a simple magnification, where we assume the zooming perturbs neither the principal point nor the effective camera center. Algebraically, if $\mathbf{u}$ and $\mathbf{u}'$ are the images of a point U before and after zooming, respectively, then,

$$\begin{aligned} \mathbf{u} &= \mathbf{K}[\mathbf{I} \mid \mathbf{0}]\mathbf{U}, \\ \mathbf{u}' &= \mathbf{K}'[\mathbf{I} \mid \mathbf{0}]\mathbf{U} = \mathbf{K}'\mathbf{K}^{-1}\mathbf{K}[\mathbf{I} \mid \mathbf{0}]\mathbf{U} = \mathbf{K}'\mathbf{K}^{-1}\mathbf{u}, \end{aligned} \tag{12}$$

so that $\mathbf{u}' = \mathbf{H}\mathbf{u}$ with $\mathbf{H} = \mathbf{K}'\mathbf{K}^{-1}$. If only the focal lengths differ between $\mathbf{K}'$ and $\mathbf{K}$, then it is straight-forward to show that,

$$\mathbf{K}'\mathbf{K}^{-1} = \begin{pmatrix} \lambda\mathbf{I} & (1-\lambda)\mathbf{u}_0 \\ \mathbf{0}^T & 1 \end{pmatrix}, \tag{13}$$

where $\mathbf{u}_0$ is the inhomogeneous principal point, and $\lambda = f'/f$ is the magnitude/scaling factor. Therefore, the special form of $\mathbf{H}$ can be used as the criteria to decide the zooming motions. In particular, the lower triangular elements of $\mathbf{H}$ are all zeros (or close to zeros), and the first two diagonal elements are equal. For the example shown in Fig.7, the estimated homography $\mathbf{H}$ is,

$$\mathbf{H} = \begin{pmatrix} 1.5632 & 0.0036 & -98.3642 \\ 0.0208 & 1.5426 & -64.1500 \\ 0.0002 & 0.0001 & 1.0000 \end{pmatrix}.$$

In other words, the zoom factor is ~ 1.55, and the principal point locates at $[177.9,\ 166.0]$, which is very close to the center of the image, whose dimension is $[360] \times [240]$.

3.5 Pure Translation - Track / Boom

In the situation where the motion of the camera is a pure translation, e.g. side-way tracking, without any rotation or change of the camera's intrinsic parameters, points in $\mathbb{R}^3$ move on the straight lines parallel to $\mathbf{p}_0$, and the projected intersection of these parallel lines is the vanishing point $\mathbf{p}$ in the direction of $\mathbf{p}_0$. It is evident that $\mathbf{p}$ is the epipole for both views, and the imaged parallel lines are the epipolar lines.

In the pure translation case, one can assume that the two cameras are $\mathbf{P} = \mathbf{K}[\mathbf{I} \mid \mathbf{0}]$ and $\mathbf{P}' = \mathbf{K}[\mathbf{I} \mid \mathbf{p}_0]$. Therefore,

$$\mathbf{F} = [\mathbf{e}']_\times \mathbf{P}'\mathbf{P}^+ = [\mathbf{e}']_\times, \tag{14}$$

where $\mathbf{P}^+$ is the pseudo-inverse of $\mathbf{P}$, i.e. $\mathbf{P}\mathbf{P}^+ = \mathbf{I}$.

For example, between the images shown in of Fig.8 (a) and (b), there is a pure translation (track or boom) of the camera. The computed fundamental matrix $\mathbf{F}$ is,

$$\mathbf{F} = \begin{pmatrix} 0.0000 & 0.0012 & -0.2258 \\ -0.0012 & 0.0000 & -0.9742 \\ 0.2258 & 0.9742 & 0.0000 \end{pmatrix}. \tag{15}$$

That is, the epipole locates at (0.9742, -0.2258, -0.0012), and thus the shot is classified as having a track motion (horizontal translation).

3.6 Results and Discussion

In the submission, we combined the results of pan and track as the results for feature 1 (pan/track) and combined the results of tilt and boom as the results for feature 2 (tilt/boom). In the news programs, many times the videos are noisy, and even it is difficult for human eyes to differentiate the motions. There is an issue that how strong the feature occurs in a particular shot. Thus, we have submitted four runs with different ranking methods. These ranking methods consider different combinations of three factors,

1. Global Displacement: This refers to the global motion of the entire shot. For instance, in pan/track motion, this represents the absolute displacement of the frames along the major moving direction.

2. Continuity: This refers to the smoothness of the motion. The purpose of this factor is to test if a given shot

is a true motion shot or is a shot with camera shaking motion.

3. Period of Motion: This refers to the temporal length of the motion. Longer periods indicate stronger motion.

Based on the results of the training dataset, we have selected the thresholds for the testing submission. The evaluation results showed that the proposed system has very high precision, but low recall. One reason for this is that we have set a fixed threshold and only returned the shots with high confidence. We have loosen the selection criteria and recall has increased with little sacrifice in precision.

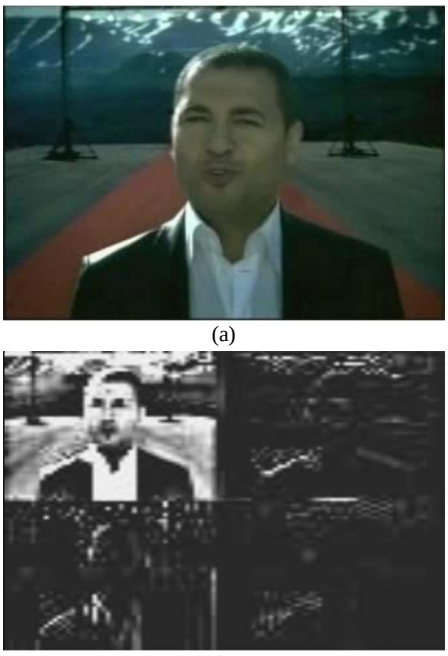
4 High-Level Feature Extraction

The task of high level feature extraction has been implemented using three approaches. First method was based on global features that were subdivided into fixed sized patches. This technique was useful for detecting features that occupy most of the image and had a lot of in-class variation, such as mountain, building, waterscape, and sports. Second approach was based on local features of image segments. This technique was useful for detecting features that had a certain structure but still had in-class variation, such as explosion/fire, map, US flag and car. Third approach used feature points and patch appearance similarity. This method was useful for the features that have a consistent appearance signature and with small in-class variation, such as US flag. Using these methods, results were submitted for all features except people walking/running.

4.1 Fixed Sized Region-based Method

This technique detects high level features based on image features obtained from fixed sized image regions. It exploits consistent appearance of similar patches in a specific part of the image. For example, sky usually appears in top part of the image. The features detected using this method were explosion/fire, map, US flag, building, waterscape, mountain, prisoner, sports and car. These features were detected using Support Vector Machines applied on the low level wavelet feature vectors.

Wavelet features were extracted using the gradient information of each key frame in the given shot. Moreover, these features were estimated on the entire frame. The wavelet feature vector was composed of four blocks. These blocks were obtained by applying high and low pass wavelet filters in the horizontal and vertical directions. The first block was obtained by applying a low pass filter in the horizontal direction followed by a low pass filter in the vertical direction. The second block was estimated by applying a low pass filter in the horizontal direction followed by a high pass filter in the vertical direction. The third block was derived by applying a high pass filter in the horizontal direction followed by a low pass filter in the vertical direction. And, the fourth block was calculated by applying a high pass filter in the horizontal direction followed by a high pass filter in the vertical direction. An example of the input image and its wavelet feature vector are given in Figure 9.



(a)

(b)

Figure 9: Wavelet feature estimation of images. (a) Mountain image. (b) Wavelet feature vector.

We used the labelled images for training and classified the unlabelled images (testing images) using this method. The wavelet feature vectors were calculated for all the training and testing images. The Support Vector Machine was trained using these estimated wavelet feature vectors. The SVM automatically selects a subset of positive and negative examples (obtained from the labelled data) and these subsets form the support vectors. During testing, the SVM classifies an unknown image as a positive or negative using these support vectors. Each unknown image was given a score by the SVM. This score depended upon how correlated each wavelet feature vector (of the unknown image) was to the positive or negative support vectors. More positive score of the feature vector (of the unknown image) resulted in a higher ranking.

Furthermore, each feature vector was divided into equal sized regions and their support vectors were calculated using the above mentioned method. The final classification was achieved by the voting of the support vectors of each region. Different voting thresholds were set for the differ-

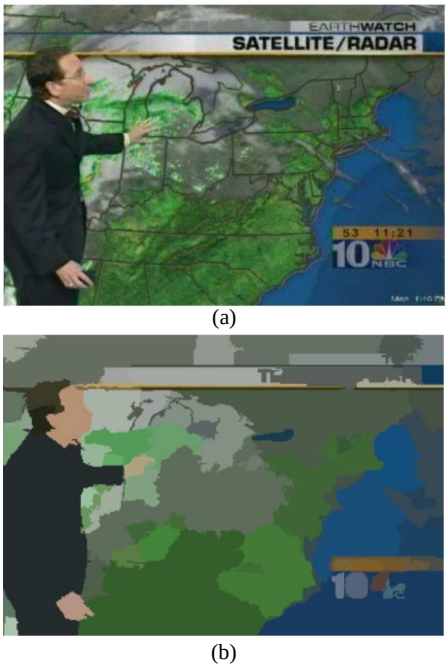
ent runs. Higher threshold resulted in higher precision and lower recall values. This technique used the spatial constraint of the local features for high level feature detection but it had certain limitations. These included mis-detection due to the change in camera orientation and zoom, partial occlusion of features, etc. Following technique was employed to overcome these problems.

4.2 Variable Sized Region-based Method

This technique utilized segmentation of image into regions, and used local features belonging to meaningful image segments for high level feature extraction. The segments could have arbitrary appearance, size and shape but they had a fixed sized feature vector. The relevant segments, belonging to the high-level feature, were recurring and displayed structure. And the irrelevant segments did not have a significant pattern. Thus, using this technique we were able to capture the structure of the recurring segments and distinguish it from the irrelevant segments.

There were five major steps involved in this method for the high level feature detection. First, for a particular high level feature the key frames of development data were segmented using Meanshift image segmentation [2], as shown in Figure 10. The key frame labels were propagated to each image segment. Second, local features corresponding to each image segment were computed. This formed a fifteen dimensional feature vector (for each region) comprising of color mean, color variance, region area, and normalized histogram of gradient directions. These features captured color and texture of the region. Each key frame now had several segments with one feature vector corresponding to each one of them. Third, using Principal Component Analysis (PCA) we determine the principal components of the feature space. The feature points corresponding to the training data were projected into the transformed feature space using the principal components. Fourth, a Support Vector Machine (SVM) was trained on the transformed feature points. A two class SVM, running radial basis function as the kernel was used. Finally, for testing a key frame, first two steps were performed to compute local feature vectors. SVM classified each feature vector in the test image with a confidence value. The decision for individual segments was merged using weighted mean of the confidence of each segment.

The effectiveness of this technique was dependent on the quality of image segmentation. Over-segmentation (too many segments) was preferred over under-segmentation (very few segments) as two different regions with very similar features should have very close feature points. On the other hand under-segmentation might fuse two actually different regions together into one segment with a common feature vector. This problem had been observed in some tests and might have a bad influence on the results.



(a)

(b)

Figure 10: Meanshift segmentation of images. (a) Map image. (b) Meanshift segmented image.

A text analysis algorithm was also used to complement the image based technique. This algorithm built a keyword histogram using the ASR available for the positive shots in development dataset. This module improved the rank of the most relevant shots by measuring the similarity with the keyword histogram.

4.3 Feature Points and Patch Similarity Method

So far we had used image features in fixed or variable sized image regions, but for some features it was important to employ a technique that use direct appearance matching such as template matching. US Flag was an example of such a feature.

This method utilized feature points in key frames and used appearance similarity of image patches around these points. Using the development data, positive examples produce clusters of image patches based on appearance similarity. This data was used to build appearance model of the feature. For each test image, feature points were used to retrieve local image patches, which were then matched to the feature model. The decision was taken based on degree of similarity with the model.

This technique had given better results only in case of features with well-formed structure such as the US Flag. The rest of the features had diverse appearance; hence can

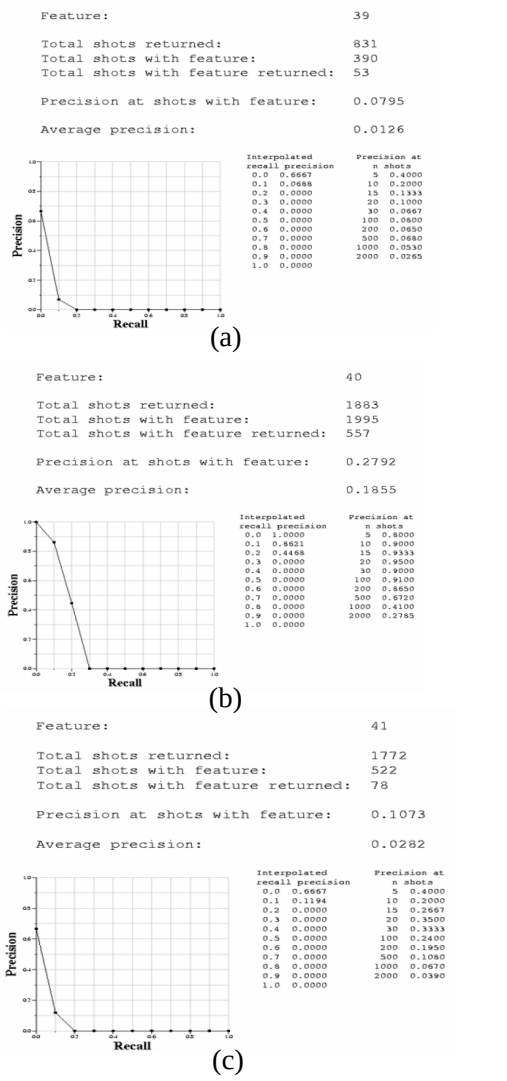


Figure 11: Evaluation analysis for the high-level feature detection. (a) Explosion/Fire (b) Map (c) US flag

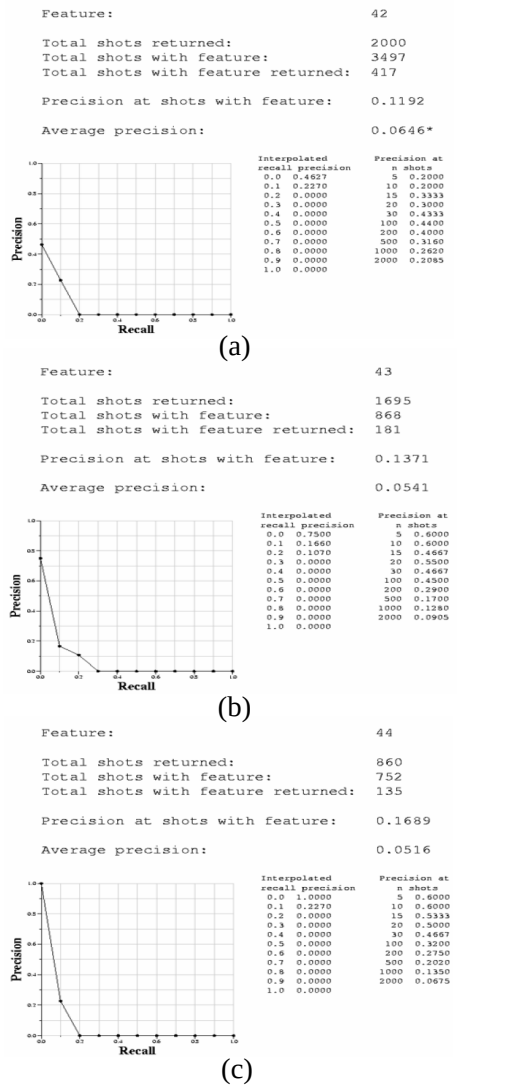


Figure 12: Evaluation analysis for the high-level feature detection. (a) Building (b) Waterscape (c) Mountain

not be detected by using patch similarity.

4.4 Results and Discussion

Using these methods, results were submitted for all features except people walking/running. The performance of the variable sized image segment based approach was better than that of the fixed sized image region based approach. Image segmentation gave regions that were more meaningful and was more robust to the changes in camera orientation, zoom, object position etc. Text analysis algorithm also improved the results, mostly in case of features like maps, where the common keywords in weather news and geographical information made it distinct. This was evident from the results in run1 and run2. Former was with the text analysis and latter was without it. 'Total shots with feature returned' is the same (557 shots) in both cases but 'Average Precision' improves from 0.1648 to 0.1855.

The text analysis algorithm did not significantly improve results in case of other features. This is because the algorithm relied on the fact that the keywords of the correct shots would be common for in-class shots and significantly different for out-of-class shots. This was not the case for any feature except maps. We understand that because of the diversity in the appearance of the high-level features, the appearance based algorithm can only detect them up to a certain extent. To further improve the results we need to employ more sophisticated text processing algorithms in addition to better appearance based detection methods. Evalu-

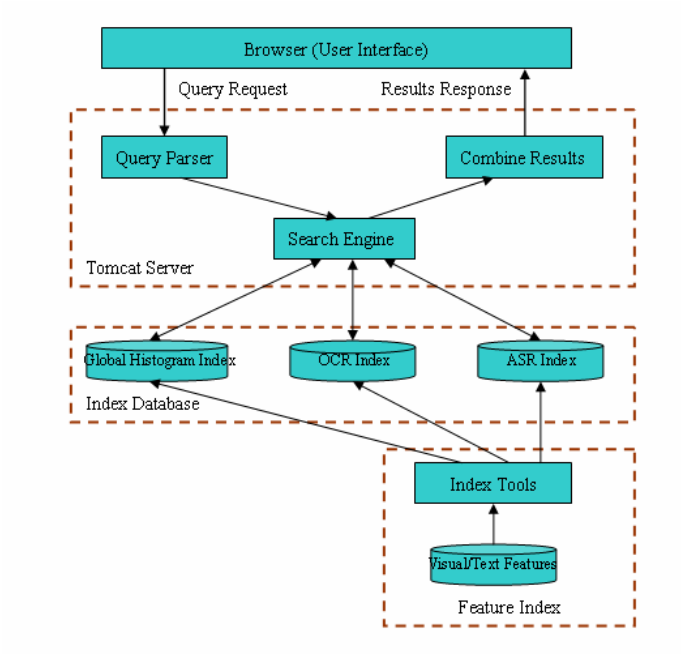


Figure 14: The overview of the PEGASUS search system.

ation analysis for the explosion/fire, map, US flag, building, waterscape, mountain, prisoner, sports and car feature detection for the best run are given in Figures 11, 12 and 13 respectively.

5 Topic Search

This year, it is the first time for UCF Computer Vision Team to participate in the topic search task. We have established the PEGASUS system, a web-based interactive search engine with the graphical user interface. The proposed system has four searching mechanisms: (1) searching by the automatic speech recognition (ASR) transcript, (2) searching by the optical character recognition (OCR) output, (3) searching by matching the global color statistics of the key-frames, and (4) searching by considering K-nearest neighbors. There are several features of the PEGASUS system: (a) ability to combine any number of the four searching mechanisms; (b) ability to evaluate the logical expressions of the search queries. (c) ability to perform the relevance feedback iterations. Figure 14 shows the overview of the PEGASUS searching system.

5.1 Indexing

The system contains two parts, indexing and retrieval. There are four components, based on which the indices were created: ASR, OCR, global histogram and high-level features. Since both the ASR and OCR information are in the text form, we generated separate indices for them using the Lucene full-text index in the Tomcat server.

We have computed the RGB global histograms of the key-frames provided in the ground truth data. The indexing system for the histograms is implemented in the linked-list form. A key-frame is accompanied with a ranked-list of its similar shots. The similarity measure is computed based on the color histogram intersection, and only those whose similarity is above certain threshold are included in the linked-list. We also tried to index the video shots based on the donated high-level semantic features. One problem with this approach discovered in the experiment is that, the ten donated features are either very general or very specific. Therefore, they do not provide much distinctions between the video shots. All three index systems on ASR, OCR and global histograms are constructed individually.

5.2 Retrieval

In the retrieval phase, we have implemented a logical expression parser, which is able to read in the query in up to two levels of logics, including AND, OR and NOT. One example of such query could be ”(bill AND Clinton) OR (president) - bush”. This query refers that the user wants to find out the shots which contain the phrases ”bill Clinton” or ”president”, but not the ones with the word of ”bush”.

The query is submitted to the search system, and the first round of the returned are computed using the ASR indexing

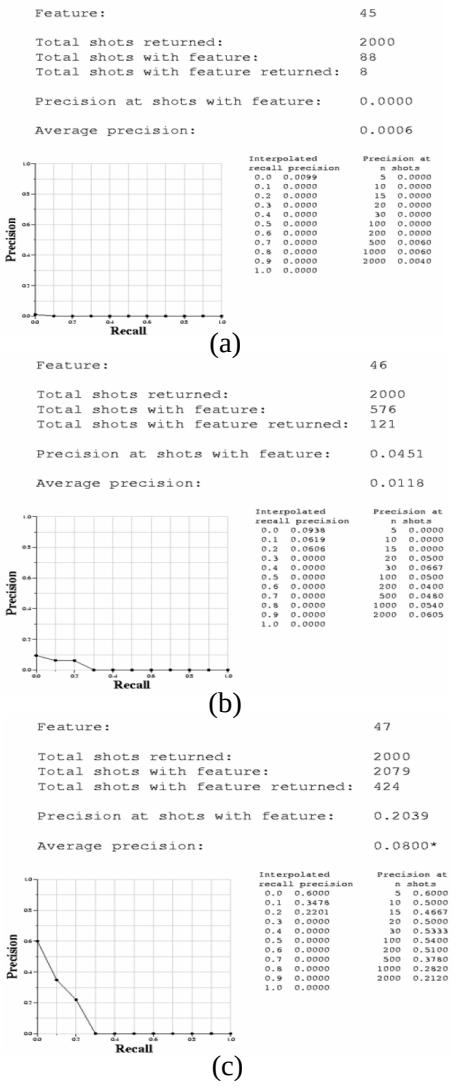


Figure 13: Evaluation analysis for the high-level feature detection. (a) Prisoner (b) Sports (c) Car

database. A set of relevant shots are returned in a ranked list, which is based on the TF-IDF measure between the query and the shot's words. Since neighboring shots often present similar or even the same semantic content, the neighbors of each of the returned relevant shots are also returned for the further refinement. Each shot in the panel is shown by its first key-frame, and it is accompanied with a check-box, which allows user to use this shot for the next round of search or not.

The selected shots in the previous rounds are used for refining the search query. For each of the models, ASR, OCR and global histograms, new query is generated. First, let us consider the ASR case. Based on the selected shots, a word-histogram is generated as obtaining the words with the highest frequencies. Based on the generated word-histogram, user is able to formulate a better query to express the need. For instance, based on our observation, for topic "tennis player on court", user could search by "tennis". A series of the positive shots are selected from the first round of returns. Based on these shots, a word-histogram is generated, in which words "cup" and "tournament" are with the highest frequencies. Therefore, user can refine the query to be "cup OR tournament OR tennis", which provides better search results. Due to the nature of the OCR information, a similar task is performed for the refinement as for the ASR information. For the use of the global histograms of the video key-frames, we have utilized the linked-list built in the indexing phase. For each of the selected shots, its linked-list are returned in the next round. An example is shown in Figure 15, where the left column shows the selected positive shots of President Bush, and they are followed, in each row, by their similar shots.

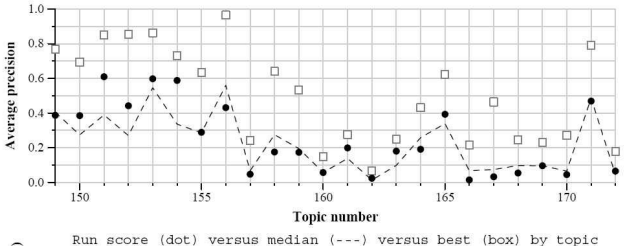
Figure 15: The refined searching results using the global color histograms of the video key-frames.

5.3 Evaluation Results

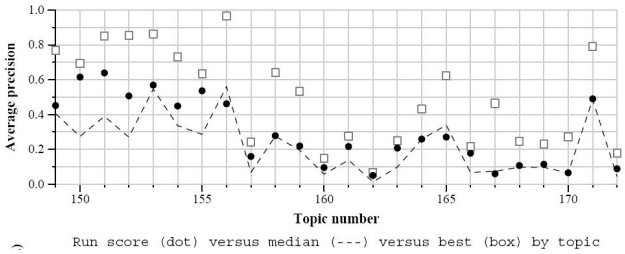
The UCF Computer Vision Team has submitted two runs for the interactive search. Run "VISION1" is purely based on the ASR information, while the second run "VISION2" involves the interactive search using ASR, OCR and global color histograms. In each run, the K-nearest neighbor method is used in the relevant feedback process. As expected, the run using all the information performs better than the one using only the ASR transcript. The average precision plots of both runs are shown in Figure 16. We have achieved the median performance among all the submitted runs.

References

[1] J. Bescos, G. Cisneros, J.M. Martinez, J.M. Menendez and J. Cabrera, "A Unified Model for Techniques on Video-Shot Transition Detection", *IEEE Journal on Multimedia*, Vol.7, No.4, 2005. pp.293-307.



(a) run1



(b) run2

Figure 16: The average precision plots of two submitted runs for the topic search task. (a) Run using only the ASR transcript, and (b) Run using all the information.

[2] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis", *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 24, pp. 603-619, 2002.

[3] L. Duan, M. Xu, Q. Tian and C. Xu, "Nonparametric Motion Model With Applications to Camera Motion Pattern Classification", *Proceeding on ACM Multimedia*, pp.328-331, 2004.

[4] R. Jin, A. Hauptmann and Y. Qi, "A Probabilistic Model for Camera Zoom Motion Detection", *International Conference on Pattern Recognition*, 2002.

[5] D.G. Lowe, "Distinctive Image Features From Scale-Invariant Keypoints", *International Jounral on Computer Vision*, 60, 2, 2004, pp. 91-110.

[6] Y.-F. Ma and H.J. Zhang, "Motion Pttern Based Video Classfication and Retrieval", *EURASIP Journal on Applied Signal Processing*, Vol.2003, No.2, 2002.

[7] N.V. Patel and I.K. Sethi,"Video Shot Detection and Characterization for Video Databases", *Pattern Recognition*, Vol.30, No.4, 1997. pp.583-592.

[8] G. Sudhir and J.C.M. Lee, "Video Annotation by Motion Interpretation Using Optical Flow Streams", *Journal on Visual Communication and Image Representation*, Vol.4, 1996. pp.354-368.

[9] Y.P. Tan, D.D. Saur, S.R. Kulkarni and P.G Ramadge, "Rapid Estimation of Camera Motion from Compressed Video with Application to Video Annotation", *IEEE Transactions on Circuits and Systems for Video Technology*, Vol.10, No.1, 2000. pp.133-146.

[10] H.D. Wactlar, M.G. Christel, Y. Gong and A.G. Hauptmann, "Lessons Learned from Building a Terabyte Digital Video Library", *Computer*, Vol.32, No.2, 1999. pp.66-73.

[11] W. Xiong and J.C.M. Lee, "Efficient Scene Change Detection and Camera Motion Annotation for Video Classification", *Computer Vision and Image Understanding*, Vol.71, No.2, 1998. pp.166-181.

[12] H.J. Zhang, A. Kankanhalli and S.W. Smoliar,"Automatic Partitioning of Full-Motion Video",*Multimedia Systems*, Vol.1, No.1, 1993. pp.10-28.