The Video Face Book

Nipun Pande, Mayank Jain, Dhawal Kapil, and Prithwijit Guha

TCS Innovation Labs, New Delhi, India
{nipun.pande,mayank10.j,prithwijit.guha}@tcs.com,dhawalkapil@gmail.com

Abstract. Videos are often characterized by the human participants, who in turn, are identified by their faces. We present a completely unsupervised system to index videos through faces. A multiple face detector-tracker combination bound by a reasoning scheme and operational in both forward and backward directions is used to extract face tracks from individual shots of a shot segmented video. These face tracks collectively form a face log which is filtered further to remove outliers or non-face regions. The face instances from the face log are clustered using a GMM variant to capture the facial appearance modes of different people. A face Track-Cluster-Correspondence-Matrix (TCCM) is formed further to identify the equivalent face tracks. The face track equivalences are analyzed to identify the shot presences of a particular person, thereby indexing the video in terms of faces, which we call the "*Video Face Book*".

1 Introduction

Videos are generally identified by actors, sceneries or specific activities. Home videos (e.g. brother's wedding, papa's birthday etc.), movies (e.g. Mel Gibson's "Braveheart") and TV series (e.g. Jennifer Aniston's "Friends") are generally referred to by the human participants. Human face is one of the most important objects in news programs. Identifying such actors from videos become a tough computer vision task if performed in a supervised framework. In such a scenario, one has to undergo a tedious supervised learning procedure to perform the task of face recognition for individual actors or for each friend/relative in a home video. In contrast, an unsupervised approach would detect and track the face regions and cluster them to generate video intervals where a certain face appears. This is also similar to the way humans perform, by associating scene intervals with the occurrence of (previously) unseen faces. The explosive growth of image and video data available both off-line and on-line further stresses the need for unsupervised methods to index, search and manipulate such data in a semantically meaningful manner.

A system for building extremely large face datasets from archival video has been introduced by [7]. The system does shot detection, tracking using color histograms for hair,face and torso followed by grouping the tracks using agglomerative clustering. For handling large number of objects in large number of dimensions a technique using Relevant Set Correlation (RSC) has been proposed by [5]. News videos are decomposed into shots followed by face detection and

K. Schoeffmann et al. (Eds.): MMM 2012, LNCS 7131, pp. 495–506, 2012.

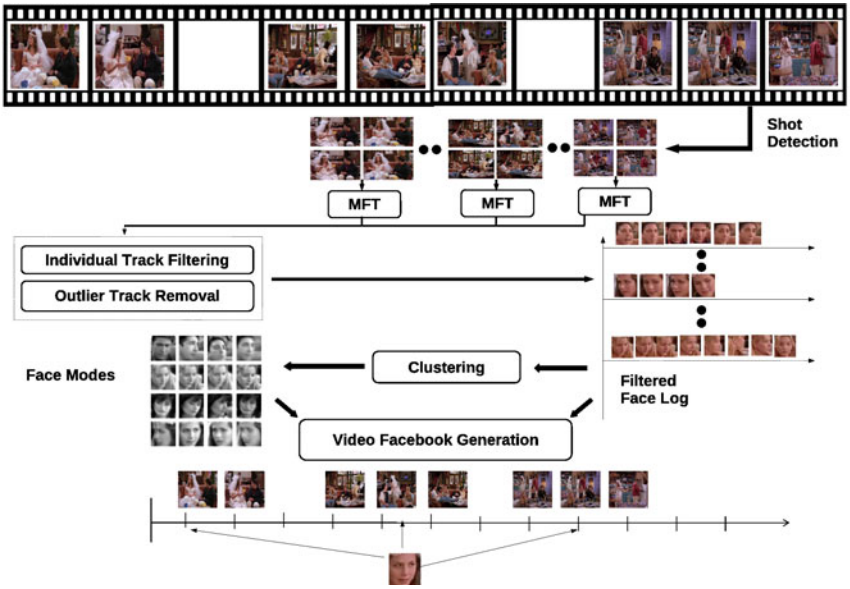


Fig. 1. The shot segmented input video is subjected to multiple face tracking (Section 2) to extract face tracks from individual shots, which are filtered in two stages to remove outliers (Section 3). The resulting face log is clustered using a GMM variant to discover the modes of facial appearances of different people in varying facial poses. The face based video index is generated by analyzing the track and cluster correspondences, which we call the *Video Face Book*.

simple tracking based on estimating the sizes and locations of faces in consecutive frames. Principal Component Analysis (PCA) is used for reducing the number of dimensions of the feature vector for face representation. For finding repeated faces, a clustering method based on RSC is used. For automatic labeling of faces of characters in TV or movie material with their names, using only weak supervision from automatically aligned subtitle and script-text [8] follow an approach where frontal/profile detections of the same face are merged using agglomerative clustering based on the overlap of the detections. Kanade-Lucas-Tomasi (KLT) feature tracker is used for feature point tracking. An approach which uses face features extracted using Discrete Cosine Transform (DCT) is proposed by [2]. Nearest neighbor classification is used to merge the tracks with distances less than a threshold. On similar lines a technique for efficient face retrieval from large video datasets using Local Binary Patterns (LBP) is proposed in [6]. A novel face indexing system that takes advantage of the internet connection of a Set Top Box (STB) to construct a Face Recognition (FR) engine has been proposed by [3]. Faces are clustered and the clustered images are combined using a weighted feature fusion scheme.

The proposed approach (Figure 1) – The video is first segmented into shots using hue-saturation histograms computed from images. The component frames of each shot interval are subjected to frontal/profile face detection and shots without any detection success are rejected as they are irrelevant to our purpose of face extraction which is dependent on such detection results. The individual shots are subjected to multiple face tracking using a detector-tracker reasoning scheme operational in both backward and forward directions (Section 2). The extracted face tracks are filtered through a two stage process where non-face regions are first removed track-wise followed by outlier track removal (Section 3). All the faces from the filtered face log are clustered to capture the facial appearance modes of different persons (Section 4). We further compute a face Track-Cluster-Correspondence-Matrix ($TCCM$) to identify the equivalent tracks and hence acquire the different shot presences of the same person. This results in the generation of the face based video index, which we call the "*Video Face Book*".

2 Multiple Face Tracking

We have used the Haar feature based face detectors [9] to segment the regions of left/right profile or frontal faces in the image sequence. However, these detectors are extremely sensitive to the facial pose. Thus, although they are very accurate in detecting faces in left/right profile or frontal faces, they fail when the facial pose changes. It is also not practical to use a lot of detectors, each tuned to different face orientations as that would lead to both high memory and processor usage. Thus, a detection reduced to a local neighborhood search guided by face features is advantageous to satisfy real-time constraints. Such a necessity is achieved by the procedure of tracking. We initialize the tracker with a face detection success, continue tracking where detection fails (due to facial pose variations) and update the target face features at times when the detectors succeed during the frame presence of the face.

Existing works in multiple face tracking have generally focused on methodologies for face detection and tracking using (skin) color distributions and/or motion cues [7,8]. These satisfy the tracking algorithm necessities of "*target representation*" and "*inter-frame target region correspondence*". However, in cases involving multiple targets, a "*reasoning*" method is required for handling various situations like tracking failure, new target acquisition, entry/exit etc. We next describe the proposed face region representation/localization schemes (Sub-section 2.1) and the adopted methodology of reasoning for tracking multiple faces (Sub-section 2.3).

2.1 Face Representation and Localization

The location of the face F in the image is identified by the face bounding rectangle $\mathbf{BR}(F)$ with sides parallel to image axes. We use a second order motion model (constant jerk), continuously updated from the 3 consecutive centroid positions of $\mathbf{BR}(F)$. Using this model, The centroidal position $\hat{\mathbf{C}}_t(F)$ at the t^{th} instant is predicted as $\hat{\mathbf{C}}_t(F) = 2.5\mathbf{C}_{t-1}(F) - 2\mathbf{C}_{t-2}(F) + 0.5\mathbf{C}_{t-3}(F)$. The color distribution $\mathbf{H}(F)$ of the face F is computed as a normalized color histogram, position weighted by the Epanechnikov kernel supported over the maximal elliptical region $\mathbf{BE}(F)$ (centered at $\mathbf{C}(F)$) inscribed in $\mathbf{BR}(F)$ [4]. Mean-shift iterations initialized from the motion model predicted position converge to localize the target face region in the current image. The mean-shift tracking algorithm maximizes the Bhattacharya co-efficient between the target color distribution $\mathbf{H}(F)$ and the color distribution computed from the localized region at each step of the iterations. The maximum Bhattacharya co-efficient obtained after the mean-shift tracker convergence is used as the tracking confidence $tc(F)$ of the face F [4]. We combine this color based representation with an appearance model to encode the structural information of the face. The RGB image region within $\mathbf{BR}(F)$ is first resized and then converted to a $q \times q$ monochrome image which is further normalized by its brightest pixel intensity to form the normalized face image nF of the face F. The normalization is performed to make the face image independent of illumination variations.

2.2 Normalized Face Cluster Set

During the course of tracking, a person appears with various facial poses. We propose to cluster the normalized faces obtained from the different facial poses to learn the modes of his/her appearances thereby forming a *Normalized Face Cluster Set* ($\mathbf{NFCS}(F)$, henceforth). The normalized face image nF is re-arranged in a row-major format to generate the $d = q \times q$ dimensional feature vector $\mathbf{X}(nF)$. To achieve computational gain, we assume that the individual dimensions of the feature vector are un-correlated and hence, a diagonal co-variance matrix is sufficient to approximate the spread of the component Gaussians. A distribution over these feature vectors is approximated by learning a variant of the Gaussian mixture models where we construct a set of normalized face clusters.

The **NFCS** with K clusters is given by the set $\mathbf{NFCS} = \{(\mu_r, \sigma_r, \pi_r); r = 1, \dots K\}$, where μ_r, σ_r are the respective mean and standard deviation vectors of the r^{th} cluster and the weighing parameter π_r is the fraction of the total number of normalized face vectors belonging to the r^{th} cluster. The **NFCS** initializes with $\mu_1 = \mathbf{X}(nF_1)$ and an initial standard deviation vector $\sigma_1 = \sigma_{init}$ and $\pi_1 = 1.0$.

Let there be K_{l-1} clusters in the **NFCS** until the processing of the vector $\mathbf{X}(nF_{l-1})$. We define the belongingness function $B_r(u)$ for the u^{th} dimension of the r^{th} cluster which is set to 1.0 if $|\mathbf{X}(nF_l)[u] - \mu_r[u]| \leq \lambda\sigma_r[u]$ and to 0.0, otherwise. Here λ is the *cluster membership threshold* and is generally chosen between $1.0 - 5.0$ (Chebyshev's inequality). The vector $\mathbf{X}(nF_l)$ is considered to belong to the r^{th} cluster if $\sum_{u=1}^{d} B_r(u) \geq (1 - \eta_{mv})d$, where $\eta_{mv} \in (0, 1)$ is the *cluster membership violation tolerance threshold* such that $\eta_{mv} \times d$ denotes the upper limit of tolerance on the number of membership violations in the normalized face vector. If $\mathbf{X}(nF_l)$ belongs to the r^{th} cluster, then its parameters are updated as,

$$\pi_r \leftarrow (1-\alpha_l)\pi_r + \alpha_l \tag{1}$$

$$\sigma_r^2[u] \leftarrow (1-\beta_r(l,u))[\sigma_r^2[u] + \beta_r(l,u)D_{lr}^2[u]] \tag{2}$$

$$\mu_r[u] \leftarrow \mu_r[u] + \beta_r(l,u)D_{lr}[u] \tag{3}$$

where $\alpha_l = \frac{1}{l}$, $\beta_r(l,u) = \frac{\alpha_l B_r(u)}{\pi_r}$ and $D_{lr}[u] = \mathbf{X}(nF_l)[u] - \mu_r[u]$. For all other clusters $r' \neq r$, the mean and standard deviation vectors remain unchanged while the cluster weight $\pi_{r'}$ is penalized as $\pi_{r'} \leftarrow (1-\alpha_l)\pi_{r'}$. However, if $\mathbf{X}(nF_l)$ is not found to belong to any existing cluster, a new cluster is formed ($K_l = K_{l-1} + 1$) with its mean vector as $\mathbf{X}(nF_l)$, standard deviation vector as σ_{init} and weight $\frac{1}{l}$; the weights of the existing clusters are penalized as mentioned before.

The parameter updates in equation 3 match the traditional Gaussian Mixture Model (GMM) learning. In GMMs, all the dimensions of the mean vector are updated with the incoming data vector. However, here we update the mean and standard deviation vector dimensions selectively with membership checking to resist the fading out of the mean images. Hence, we call the **NFCS** as a variant of the mixture of Gaussians. Figure 2(a) shows a few mean images of the normalized face clusters learned from the tracked face sequences of the subject.

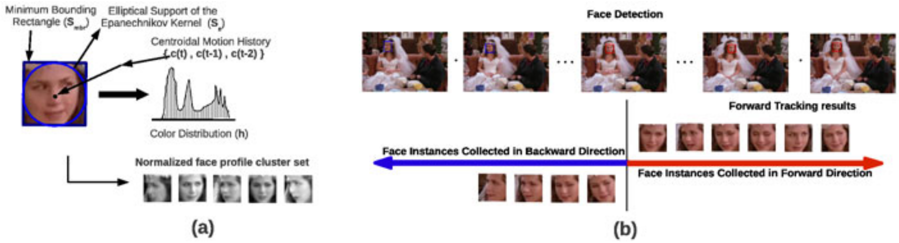

Minimum Bounding Rectangle (S_{mbr})
Elliptical Support of the Epanechnikov Kernel (S_e)
Centroidal Motion History { c(t) , c(t-1) , c(t-2) }
Color Distribution (h)
Normalized face profile cluster set
(a)
Face Detection
Forward Tracking results
Face Instances Collected in Backward Direction
Face Instances Collected in Forward Direction
(b)

Fig. 2. (a) Color distribution $\mathbf{H}(F)$, second order motion model and the *normalized face cluster set* ($\mathbf{NFCS}(F)$) are used for *face representation and tracking.* (b) *Backward-Forward tracking* – Jennifer Aniston's face gets detected somewhere at the middle of the shot interval; multiple face tracker detects a new face region and starts tracking (marked with red bounding box) in forward direction. Mean-shift tracker initialized from the first detection is used to localize the face in backward direction (marked with blue bounding box).

2.3 Handling Multiple Faces

Tracking multiple faces is not merely the implementation of multiple trackers but a reasoning scheme that binds the individual face trackers to act according to problem case based decisions. For example, consider the case of tracking a face which gets occluded by another object. A straight through tracking approach will try to establish correspondences even when the target face disappears in the image due to complete occlusion by some scene object leading to tracking failure. A reasoning scheme, on the other hand, will identify the problem situation of the disappearance due to the occlusion of the face and will accordingly wait for the face to reappear by freezing the concerned tracker. Our approach to multiple face tracking proposes a reasoning scheme to identify the cases of face grouping/isolation along with the scene entry/exit of new/existing faces.

The process of reasoning is performed over three sets, viz. the sets of *active*, *passive* and *detected* faces. The active set $\mathcal{F}_a(t)$ consists of the faces that are well tracked until the t^{th} instant. On the other hand, the passive set $\mathcal{F}_p(t)$ contains the objects for which either the system has lost track or are not visible in the scene. The set of detected faces $\mathcal{F}_d(t)$ contains the faces detected in the t^{th} frame. The system initializes itself with empty active/passive/detected face sets and the objects are added or removed accordingly as they enter or leave the field of view. During the process of reasoning, the objects are often switched between the active and passive sets as the track is lost or restored. We start the process of reasoning at the t^{th} frame based on the active/passive face sets available from the $(t-1)^{th}$ instant. The faces in the active set are first localized with motion prediction initialized mean-shift trackers (Sub-section 2.1. We compute the extent of overlap between the tracked face regions from the active set and the detected face regions to identify the isolation/grouping state of the faces. The reasoning scheme based on the tracked-detected region overlaps is described next.

Consider the case where m faces are detected ($\mathcal{F}_d = \{dF_j; j = 1 \ldots m\}$) while n faces were actively tracked till the last frame ($\mathcal{F}_a = \{aF_i; i = 1 \ldots n\}$). We define the fractional overlap between the faces F_1 and F_2 as $\gamma(F_1, F_2) = \frac{|\mathbf{BR}(F_1) \cap \mathbf{BR}(F_2|)}{\mathbf{BR}(F_1)}$ to analyze the correspondence between F_1 and F_2. We consider aF_i and dF_j to have significant overlap with respect to a certain threshold η_{ad}, if the predicate $\textsc{Overlaps}(aF_i, dF_j) \Rightarrow [\gamma(aF_i, dF_j) \geq \eta_{ad}] \vee [\gamma(dF_j, aF_i) \geq \eta_{ad}]$ is satisfied.

Let $\mathbf{S_{df}}(i) = \{dF_k : [dF_k \in \mathcal{F}_d] \wedge \textsc{Overlaps}(aF_i, dF_k)$ denote the set of detected faces which has significant overlap with the face aF_i in the active set and $\mathbf{S_{af}}(j) = \{aF_r : [aF_r \in \mathcal{F}_a] \wedge \textsc{Overlaps}(aF_r, dF_j)$ represent the set of faces in the active set which has significant overlap with the detected face dF_j. Based on the cardinalities of these sets associated with either of aF_i/dF_j and the tracking confidence $tc(aF_i)$, we identify the following situations during the process of tracking.

Isolation and Feature Update – The face aF_i is considered to be isolated if it does not overlap with any other face in the active set – $\forall r \neq i \neg \textsc{Overlaps}(aF_i, aF_r)$; $aF_i, aF_r \in \mathcal{F}_a$. Under this condition of isolation of the tracked face, we update its color distribution and motion features from the associated detected face if there exists a pair (aF_i, dF_k) which significantly overlap only with each other and none else – $\exists k \textsc{Overlaps}(aF_i, dF_k) \wedge |\mathcal{S}_{df}(i) = 1| \wedge |\mathcal{S}_{af}(k) = 1|$.

Face Grouping – The face is considered to be in a group (e.g. multiple persons with overlapping face regions) if the bounding rectangles of the tracked faces overlap. In this case, even if a single detected face dF_k is associated to aF_i, we only update the motion model of aF_i as we are not confident about the correspondence on account of multiple overlaps.

Detection and/or Tracking Failure – This is the case where face detection fails due to facial pose variations. However, if the face aF_i is tracked well ($tc(aF_i) \geq \eta_{tc}$), we update only the motion model of aF_i and do not update the color distribution. However, in case of both detection and tracking failure, aF_i is not associated with any detected face and the tracking confidence also drops below the threshold (η_{tc}). In this case, we consider aF_i to disappear from the scene and transfer it from $\mathcal{F}_a$ to $\mathcal{F}_p$ i.e. $\textsc{Disappears}(aF_i) \Rightarrow |\mathcal{S}_{df}(i) = 0| \wedge [tc(aF_i) < \eta_{tc}]$.
New Face Identification – A new face in the scene does not overlap with any of the the bounding rectangles of the existing (tracked) faces. Thus, dF_j is considered a new face if $\mathbf{S_{af}}(j)$ is a null set i.e. $\textsc{NewFace}(dF_j) \Rightarrow |\mathbf{S_{af}}(j)| = 0$. Note that, the system might lose track of an existing face whose re-appearance is also detected as the occurrence of a new one. Hence, the newly detected face region is normalized first and checked against the $NFCS$ of the faces in $\mathcal{F}_p$. If a match is found, the track of the corresponding face is restored by moving it from $\mathcal{F}_p$ to $\mathcal{F}_a$ and its color and motion features are re-initialized from the newly detected face region. However, if no matches are found, a new face is added to $\mathcal{F}_a$ whose color and motion features are learned from the newly detected face region.

During the course of multiple object tracking, the faces in the active set are identified in one of the above situations and the feature update or active to passive set transfer decisions are taken accordingly. By reasoning with these conditions, we initialize new trackers as new faces enter the scene and destroy them as the faces disappear.

2.4 Backward-Forward Tracking

Our work assumes that a certain person will be detected in either front/profile face at some time in a shot (of duration $[t_s, t_e]$, say). However, it may well happen that the person gets detected only at the t^{th} instant ($t_s < t < t_e$), although he/she was present from the very beginning (t_s) with a facial pose different from either frontal or left/right profile. In such cases, tracking in only forward direction will not provide us with all the face instances of the person. To avoid this, we also run a backward tracker initialized with the first detection to provide us with all the facial pose variations of the tracked person. The tracker is terminated when the tracking confidence dips below the threshold η_{tc}. Figure 2(b) illustrates the combined scheme for tracking in both backward and forward direction for acquiring the face instances in varying poses; including the ones prior to first detection.

2.5 Results: Multiple Face Tracking

We present results from 3 shots from the movies "*300*" (624 images) and "*Sherlock Holmes*" (840 images); and the TV Series "*Friends*, an episode from Season 1 (143 images). The results of multiple face tracking in these videos are shown in figure 3. The proposed approach for multiple face tracking is implemented on a single core 1.6 GHz Intel Pentium-4 PC with semi-optimized coding and operates at 13.33 FPS (face detection stage included).

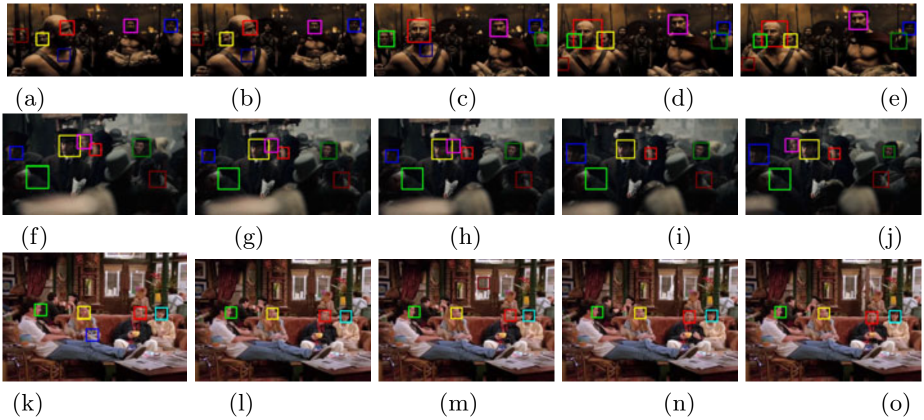


Fig. 3. Results of multiple face tracking under occlusions. (a)-(e) Movie *300* - faces are relatively unoccluded; (f)-(j) Movie *Sherlock Holmes* – The face marked with the pink bounding rectangle undergoes partial and full occlusion and the track is successfully restored as it reappears. (k)-(o) TV series *Friends*, Season 1. Note that apart from faces, trackers are also initialized on non-face regions in (f)-(o) due to false detections which are filtered later. (Section 3)

Performance Analysis – We present an object centric performance analysis by manually inspecting the surveillance log for computing the average rates of tracking precision and track switches. Consider the case of a tracker with a life span of T frames, of which for the first T_{trk} frames, the tracker successfully tracks the same face over which it is initialized and then successively switches track to N_{switch} number of (different) faces(s) during the remaining $T - T_{trk}$ frames. The *tracking precision* of an individual object is then defined as $\frac{T_{trk}}{T}$ and the average tracking precision computed over the entire set of extracted faces is called the **Tracking Success Rate** for the entire video. In the same line, the **Tracker Switch Rate** is evaluated as the average number of track switches over the entire set of extracted objects. After a track switch from the $T_{trk} + 1$ frame onwards, a different tracker may pick up the trail of this object through a track switch from some other face or through the initialization of a new tracker – let there be N_{reinit} number of tracker re-initializations on some face region. The **Tracker Re-initialization Rate** is defined as the average number of tracker re-initializations per face computed over the entire set of extracted faces. Refer to Figure 4.

3 Face Log Processing

The cropped face regions acquired by tracking are stored in a face log. However, the face log also contain non-face regions (outliers) on account of detection/tracking failure. We note that such outliers are of two types – first, trackers initialized on proper face regions which occasionally drift to non-face regions

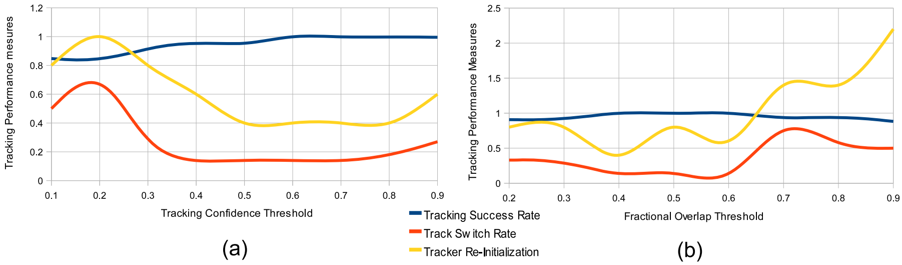


Fig. 4. Multiple face tracking performance analysis. The rates of tracking success, track switches and tracker re-initialization are plotted with respect to (a) tracking confidence threshold (η_{tc}) and (b) fractional overlap threshold (η_{fo}) varied in the interval of $[0.1, 0.9]$ in steps of 0.1. We choose $\eta_{fo} = 0.4$ and $\eta_{tc} = 0.6$ for optimal performance by referring to these graphs.

due to motion-model failure or pre-mature mean-shift convergence; and second, trackers initialized from non-face regions (false detections) continuously tracking these outlier regions during the entire shot. We propose a two-stage filtering scheme to remove such outliers based on three assumptions – first, hue-saturations histograms computed from face regions will have similar distributions for the skin pixels while non-face regions will have completely different distribution profiles; second, in each track the face regions are in the majority and hence the average color distribution will be considerably different from the color distributions of non-face regions; and third, in face-tracks initialized on false detections, there will be hardly any face region and thus the average hue-saturation distribution of that track will be significantly different from an average distribution computed from only face regions.

Consider the case where N face tracks ($T_i; i = 1, \dots N$) are extracted where the i^{th} track contains n_i faces ($T_i = \{F_{ij}; j = 1, \dots n_i\}$). Let $H_{hs}(i, j)$ denote the hue-saturation distribution computed from F_{ij} and we compute the average $\bar{H}_{hs}(i) = \frac{1}{n_i} \sum_{j=1}^{n_i} H_{hs}(i, j)$ from all the faces in T_i. Based on our assumptions, we declare the q^{th} face as an outlier if $\mathbf{B}_c(H_{hs}(i, q), \bar{H}_{hs}(i)) < \eta_{cm}$ where η_{cm} is a color distribution match threshold. The outliers, if present are removed from each track and leaves us with $T_i = \{F_{ij}; j = 1, \dots n'_i\}; i = 1, \dots N$. Note that this process only removes outliers from each track but can not filter the ones where the trackers were initialized on non-face regions due to erroneous face detections (Figure 5(a)).

The process of individual track filtering leaves us with two kinds of tracks – first, the "pure" ones with only face regions; and second, the ones containing mostly outliers where the tracker was initialized on non-face regions. We compute the average hue-saturation distributions $\bar{H}_{hs}(i)$ from each track and obtain their average as $\tilde{H}_{hs} = \frac{1}{N} \sum_{i=1}^{N} \bar{H}_{hs}(i)$. Proceeding on the same assumptions outlined earlier, we describe the i^{th} track as an outlier, if $\mathbf{B}_c(\tilde{H}_{hs}, \bar{H}_{hs}(i)) < \eta_{cm}$ (Figure 5(b)). The faces belonging to the filtered tracks are clustered further to group the similar faces and are described next (Section 4).

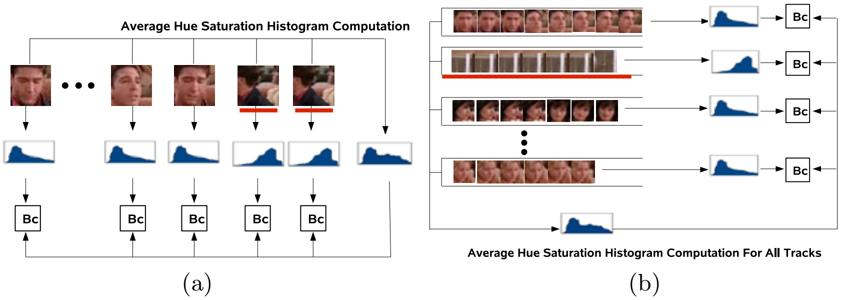
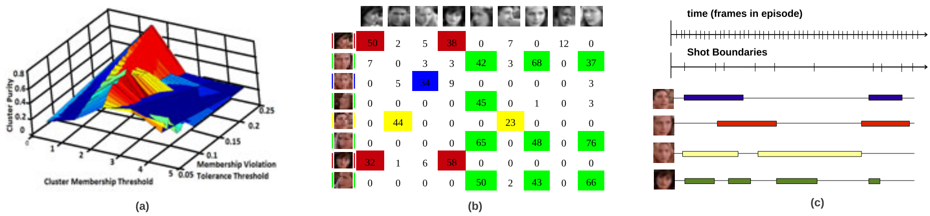

Average Hue Saturation Histogram Computation
Bc
(a)
Bc
Average Hue Saturation Histogram Computation For All Tracks
(b)

Fig. 5. Two stage face log filtering with Bhattacharya coefficient $\eta_{cm} = 0.6$. (a) Non-face instances are removed from individual tracks in first stage. (b) Outlier tracks initialized from non-face regions are filtered next.



50	2	5	38	0	7	0	12	0
7	0	3	3	42	3	68	0	37
0	5	34	9	0	0	0	0	3
0	0	0	0	45	0	1	0	3
0	44	0	0	0	23	0	0	0
0	0	0	0	65	0	48	0	76
32	1	6	58	0	0	0	0	0
0	0	0	0	50	2	43	0	66

Fig. 6. (a) The cluster purity is evaluated by varying the cluster membership threshold (λ) and membership violation tolerance threshold (η_{mv}) for clustering performance analysis. (b) The marked cells of $TCCM$ indicate the face track-cluster linkages which satisfy a thresholded association criterion. A linkage transitivity analysis is performed further to identify the tracks linked through the common cluster(s). (c) A small segment of the *Video Face Book* generated from the TV series "Friends" (episode 1, season 1). Horizontal colored bars indicate the shot presences of different human participants.

4 Face Clustering

The face regions obtained from all tracks of the filtered face log are clustered using the approach outlined in Sub-section 2.2. Ideally, each cluster should contain faces of the same person. However, such a *cluster purity* varies with different values of the *cluster membership threshold* (λ) and *cluster membership violation tolerance threshold* (η_{mv}). Consider the case where K clusters are formed, where the k^{th} cluster contains nC_k faces, of which mC_k number of faces belong to the same person and satisfies the plurality criterion. Then, we define the average cluster purity $cP(\lambda, \eta_{mv})$ for a certain set of chosen thresholds as $cP(\lambda, \eta_{mv}) = \frac{\sum_{k=1}^{K} mC_k}{\sum_{k=1}^{K} nC_k}$. The clustering performance is analyzed by varying λ in $[0.5, 4.5]$ in steps of 0.1 and η_{mv} in $[0.05, 0.25]$ in steps of 0.005. The performance analysis is performed on 3 test data sets (Figure 3) and we have chosen $\lambda = 1.8$ and $\eta_{mv} = 0.215$ by referring to Figure 6(a) for which we achieve the maximum cluster purity of 0.804.

5 Video Index Generation

Consider the case where the filtered face log contains N' face tracks and K clusters are obtained by face clustering. We form the $N' \times M$ Track-Cluster-Correspondence-Matrix ($TCMM$) to analyze the equivalences of the different tracks present in the face log. Let $cL(i, j)$ denote the cluster index of the j^{th} face in the i^{th} track, i.e. $cL(i, j) \in [1, M]$. The $TCMM$ is thus formed as $TCMM[i][k] = \sum_{j=1}^{n'_i} \delta(cL(i, j) - k)$ where, the i^{th} track contains n'_i faces and $\delta(\bullet)$ is the Kronecker Delta function.

Tracking provides us with various facial poses of the same person while clustering helps us discover the modes of facial appearance. The similar facial appearances are grouped through clustering while the different facial appearances of the same person are linked through tracking. Each row of the $TCCM$ signify the number of occurrences of different facial appearance modes in a certain track and each column of $TCCM$ denote the frequencies of assuming the same facial appearance mode by different tracks. We link a track i to the cluster k if more than 25% faces of the i^{th} track assume the k facial appearance mode i.e. if $TCMM[i][k] \geq 0.25n'_i$. Consider the case where the i^{th} track is linked to the clusters k and p while the r^{th} track is linked with clusters p and q. We perform a linkage transitivity analysis to identify that the tracks i and p have a common link to the p^{th} cluster and use the same to declare the tracks i and j as equivalent. A similar analysis is performed on the entire $TCMM$ to identify the equivalent tracks (Figure 6(b)). Since the face tracks are obtained from indexed shots, analyzing the equivalent tracks reveal the shot presences of the same person. This is illustrated in Figure 6(c) where a part of the *Video Face Book* formed by analyzing the TV series "Friends" (episode 1, season 1) is shown.

6 Conclusion

We present an unsupervised scheme for indexing videos with human participants by using facial information and hence the name *Video Face Book*. The video is initially decomposed into a sequence of shots using the criterion of intra-shot frame hue-saturation distribution consistency. A combination of backward-forward tracking is used to extract the tracks of multiple faces from individual shots. Such tracks obtained from each shot collectively form the crude face log containing outliers along with face instances. Outliers are removed in two stages – first, the non-face regions are filtered from each track and second, the outlier tracks formed due to false detections are removed. All the face instances from all tracks are clustered next to form the face clusters. A person may appear with varying facial poses in the same track and hence traverse the different modes (mean faces of clusters) of facial appearance. Thus people appearing in different shots can be linked through strong correspondences of different tracks with the same cluster. We form a Track-Cluster-Correspondence-Matrix ($TCMM$) to identify such track linkages and hence generate the video index in terms of shot presences of a certain person.

We have demonstrated an unsupervised approach to indexing videos through faces. However, recent research has also proposed unsupervised means of discovering objects from images/videos [1]. These approaches may be used to discover objects from videos first, and the proposed scheme can be used next to detect/track and cluster objects of different categories for indexing videos. However, this will only be the indexing of videos with the *actors*, whose interactions might be discovered and grouped further to index videos in terms of *actions* thereby proceeding a few steps further to achieve the final goal of a cognitive vision system.

References

1. Alexe, B., Deselaers, T., Ferrari, V.: What is an object? In: IEEE Computer Vision and Pattern Recognition (CVPR), San Francisco, pp. 1–8 (June 2010)
2. Bauml, M., Fischer, M., Bernardin, K., Ekenel, H.K., Stiefelhagen, R.: Interactive person-retrieval in tv series and distributed surveillance video. In: MM 2010 Proceedings of the International Conference on Multimedia (2010)
3. Choi, J.Y., Neve, W.D., Ro, Y.M.: Towards an automatic face indexing system for actor-based video services in an iptv environment. IEEE Transactions on Consumer Electronics 56, 147–155 (2010)
4. Comaniciu, D., Ramesh, V., Meer, P.: Real-time tracking of non-rigid objects using mean shift. In: Computer Vision and Pattern Recognition, vol. 2, pp. 142–149 (2000)
5. Le, D.D., Satoh, S., Houle, M.E., Nguyen, D.P.T.: An efficient method for face retrieval from large video datasets. In: Proceedings of the ACM International Conference on Image and Video Retrieval (2010)
6. Nguyen, T.N., Ngo, T.D., Le, D.D., Satoh, S., Le, B.H., Duong, D.A.: An efficient method for face retrieval from large video datasets. In: Proceedings of CIVR 2010, pp. 382–389 (2010)
7. Ramanan, D., Baker, S., Kakade, S.: Leveraging archival video for building face datasets. In: IEEE 11th International Conference on Computer Vision, ICCV 2007, pp. 1–8 (2007)
8. Sivic, J., Everingham, M., Zisserman, A.: Who are you?- learning person specific classifiers from video. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1145–1152 (2009)
9. Viola, P., Jones, M.: Robust real-time face detection. International Journal on Computer Vision 57(2), 137–154 (2004)