



TAMPERE UNIVERSITY OF TECHNOLOGY

UMAR IQBAL

IMPORTANT PERSON DETECTION FROM MULTIPLE VIDEOS

Master's thesis

Examiner(s):

Professor Moncef Gabbouj

Dr. Igor Curcio

Examiner and topic approved by the
Faculty Council of the Faculty of Com-
puting and Electrical Engineering on 3rd
April 2013.

ABSTRACT

TAMPERE UNIVERSITY OF TECHNOLOGY

Master's Degree in Information Technology

UMAR IQBAL: Important Person Detection from Multiple Videos

Master of Science Thesis, 76 pages

Month and year of completion: August 2013 (Examiner and topic were approved in the faculty council meeting on 3rd April 2013)

Major: Signal Processing

Examiner(s):

Prof. Moncef Gabbouj, Department of Signal Processing, Tampere University of Technology

Dr. Igor Curcio, Principal Scientist, Nokia Research Center Tampere

Keywords: important person, unsupervised, re-identification, community contributed, crowd-sourced, videos, face tracks, clustering.

Given a crowd-sourced set of videos of a crowded public event, this thesis addresses the problem of detecting and grouping appearances of every person in the scenes. The persons are ranked according to the amount of their occurrence. The rank of a person is considered as the measure of his/her importance. Grouping appearances of every individual from such videos is a very challenging task. This is due to unavailability of prior information or training data, large changes in illumination, huge variations in camera viewpoints, severe occlusions and videos from different photographers. These problems are made tractable by exploiting a variety of visual and contextual cues – appearance, sensor data and co-occurrence of people. This thesis provides a unified framework that integrates these cues to establish an efficient person matching process across videos of the same event.

The presence of a person is detected based on a multi-view face detector followed by an efficient person tracking that tracks the detected persons in remaining video frames. The performance of person tracker is optimized by utilizing two independent trackers; one for the face and the other for clothes, and the clothes are detected by taking a bounding box below the face region. The person matching is performed using the facial appearance (biometric) and colors of clothes (non-biometric). Unlike traditional matching algorithms that use only low-level facial features for face identification, high-level attribute classifiers (i.e., Gender, ethnicity, hair color, etc.) are also utilized to enhance the identification performance.

Hierarchical Agglomerative Clustering (HAC) is used to group the individuals within a video and also across videos. The performance of HAC is improved by using contextual constraints, such as a person cannot appear twice in the same frame. These constraints are directly enforced by altering the HAC algorithm. Finally the detected individuals are ranked according to the number of videos in which they ap-

pear and ‘N’ top ranked individuals are taken as important persons. The performance of the proposed algorithm is validated on two novel challenging datasets.

The contribution of this thesis is twofold. First, a unified framework is proposed that does not require any prior information or training data about the individuals. The framework is completely automatic and does not require any human interaction. Second, we demonstrate how usage of multiple visual modalities and contextual cues can be exploited to enhance the performance of persons matching under real life problems. Experimental results show the effectiveness of the framework and ensure that the proposed system provides competitive results with the state-of-art algorithms.

PREFACE

This thesis was done in collaboration between Department of Signal Processing, Tampere University of Technology and Nokia Research Center (NRC), Finland.

In this regards, I would like to thank all those who helped me in completing this work. I am so grateful to Allah the almighty for his utmost blessings, divine guidance and the strength he gave me to accomplish this task. I pay my heartiest gratitude to my supervisor Prof. Moncef Gabbouj for his continuous support and guidance. My thanks to my manager Dr. Igor Curcio for believing in me and providing this wonderful opportunity to work on such an exciting topic.

The experience of working at NRC has been worth it, I thank all the laboratory fellows for a motivating research atmosphere. Special thanks to Dr. Esin Guldogan for helping me with the implementation of the clothes tracker. Thanks to Francesco Cricri and Dr. Vinod Kumar for interesting discussions.

I would like to dedicate this thesis to my parents for their endless love, support and encouragement throughout my life. I also dedicate this thesis to my beloved wife Mehru for her unconditional love, patience and support. Thanks a lot to my siblings for making me laugh during hard times of staying abroad.

I thank my friends Ali-bin-Tariq, Farhan, Habib, Faisal, Ejaz, Mukesh, Adnan, Ali Zaib, Faraz, Rehman, Kashif, Shoaib, Shahbaz and Angriassa for their help in collecting and annotating the datasets used in this thesis. Special thanks to Zohaib Hassan without whom my stay in Tampere would not have been as amazing as it is. Thanks to Mubashir Ali for his delicious foods and tips on thesis writing and, Srikanth for being my mate during overnight stays at university and completing numerous projects together.

Finally, I would like to thank the band “Eternal Erection” for allowing me to use their videos in this thesis.

Umar Iqbal
6th August, 2013.

Table of Contents

1.	Introduction	1
1.1.	Literature Review	4
1.2.	Thesis Outline	7
2.	Theoretical Background	8
2.1.	Face Detection.....	8
2.1.1.	Multi-Block Local Binary Pattern.....	8
2.1.2.	Boosting and Cascade Classifiers	10
2.1.3.	Multi-view Face Detection and Pose Estimation.....	11
2.2.	Visual Tracking	12
2.2.1.	Particle Filter Based Visual Tracking	13
2.3.	Feature Extraction	20
2.3.1.	Textural Features.....	20
2.3.2.	Color Features	21
2.4.	Hierarchical Agglomerative Clustering	22
2.5.	Support Vector Machines.....	23
2.6.	Performance Metrics	25
2.6.1.	Quality Measures for Detection Algorithms.....	25
2.6.2.	Quality Measures for Classification Algorithms	26
2.6.3.	Quality Measures for Clustering	26
2.6.4.	Quality Measure for Important Person Detection.....	27
3.	Person Representation Scheme	28
3.1.	Person Detection and Tracking	28
3.1.1.	Detector Assisted Face Tracking	28
3.1.2.	Clothes Detection and Tracking.....	31
3.1.3.	Merging Face and Clothes Tracks	33
3.2.	Representation using Facial Features.....	34
3.2.1.	Facial Landmarks Detection and Face Alignment.....	35
3.2.2.	Feature Extraction	36
3.3.	Representation using High-Level Attributes.....	37
3.3.1.	Selected Attributes	37
3.3.2.	Training of Attribute Classifiers	38
3.3.3.	Accuracies of Attribute Classifiers	44
3.3.4.	Final Features	45
3.4.	Representation using Clothing Colors.....	45
4.	Important Person Detection.....	46
4.1.	Temporal Video Segmentation.....	48
4.1.1.	Camera Panning Detection.....	48
4.2.	Unique Identity Assignment.....	49
4.2.1.	Person Track Clustering.....	49

4.3.	Global Ranking of Individuals and Important Person Detection	54
5.	Experimental Setup and results	55
5.1.	Dataset Details.....	55
5.1.1.	Single-Event Dataset.....	55
5.1.2.	Multi-Event Dataset	57
5.2.	Weights for Each Modality	57
5.3.	Results	58
5.3.1.	Results on Single-Event Dataset.....	58
5.3.2.	Results on Multi-Event Dataset	61
5.4.	Analysis of Computational Complexity	64
6.	Conclusion and Future Works.....	67
6.1.	Conclusion.....	67
6.2.	Possible Future Directions	68
	REFERENCES.....	70

LIST OF FIGURES

Figure 2-1: Illustration of the Multi-Block LBP [44].	9
Figure 2-2: Illustration of the cascade architecture used in face detection	11
Figure 2-3: Example of the face detection. (a) Overlapping regions detected by the face detector. (b) Final detections using the multi-view face detector.	11
Figure 2-4: Example of the temporal relation achieved by using visual tracking.	12
Figure 2-5: Pseudocode for the particle filtering algorithm.	19
Figure 2-6: Example of the dendrogram obtained after HAC.	22
Figure 2-7: A separating hyperplane for the linearly separable classes. [13]	24
Figure 3-1: Example of the particles drawn around the detected face region in the next frame. (a) Detected face region at frame t . (b) Drawn particles around the face region at frame $t+1$.	29
Figure 3-2: Sample face track generated by the detector assisted face tracker.	30
Figure 3-3: Example of the clothes bounding box detected based on the face detector.	31
Figure 3-4: Illustration of the face and cloth track merging.	33
Figure 3-5: Preprocessing steps for each face image.	35
Figure 3-6: Schematic diagram of the block-wise LBP histogram based feature extraction.	36
Figure 3-7: The complete architecture for the automatic selection of the optimal feature, and training of an attribute classifier [59].	38
Figure 3-8: Selected face regions to train the attribute classifiers.	40
Figure 3-9: Example training images with the face regions drawn with different colors. We can see how the regions are overlapped with each other.	40
Figure 3-10: Pseudocode for the automatic feature selection and training of the attribute classifiers.	43
Figure 3-11: Example of the representation using high-level attribute classifiers.	45
Figure 3-12: Example of the representation using clothing color features.	45
Figure 4-1: The schematic diagram for the important person detection framework.	47
Figure 4-2: Example of the temporal video segmentation using compass data analysis.	49
Figure 4-3: Example of a video frame where uniqueness constraint holds.	51
Figure 4-4: Illustration of endorsing a uniqueness constraint.	52
Figure 4-5: Example of the dendrograms before and after the enforcement of the uniqueness constraints (CC-constraint is used between P_{11} and P_{13}) with cutoff = 0.8. (a) Before. (b) After.	52
Figure 5-1: Examples of the video frames from the single-event dataset (each row representing a unique event).	56
Figure 5-2: Examples for the video frames from ND-QO-Flip dataset.	57
Figure 5-3: Example frames from the videos of two different events. The number assigned to each face is the unique identity assigned to every person after within-video clustering. (a) Example frames from event-1. (b) Example frames from event-3.	59

Figure 5-4: Comparison of the SOR values obtained using different methods for the individual events.	61
Figure 5-5: Comparison of the CON obtained using different methods for the individual events.....	61
Figure 5-6: Example frames from two different videos of the multi-event dataset. The number assigned to each face is the unique identity assigned to every person after within-video clustering.....	62
Figure 5-7 Flow diagram of the important person detection framework with different abstraction levels.	64
Figure 5-8 A pie chart illustrating the average processing time taken by different modules of the proposed framework.....	64
Figure 5-9 A pie chart representing the average processing time taken by different feature extractors.....	65

LIST OF TABLES

Table 1: Selected high level attributes for the face representation.	37
Table 2: Count of the labeled training samples after combining images from the FaceTracer and PubFig datasets.	39
Table 3: Feature type options for the training of an attribute classifier.	41
Table 4: 5-fold cross validation accuracies of all the attribute classifiers with selected combinations for feature extractors (in order of their selection).	44
Table 5: Performance of the face detector for the multi-event dataset.	58
Table 6: Experimental results on the single-event dataset, averaged over all events.	60
Table 7: Performance of the face detector over the multi-event dataset.	62
Table 8: Experimental results for the multi-event dataset.	63
Table 9: Comparison of the proposed algorithm with the state of the art.	63

LIST OF ABBREVIATIONS

HAC	Hierarchical Agglomerative Clustering
ROI	Region of Interest
LBP	Local Binary Pattern
ELBP	Extended Local Binary Pattern
DCT	Discrete Cosine Transform
HOG	Histogram of Oriented Gradients
MRF	Markov Random Fields
SVM	Support Vector Machines
RBF	Radial Basis Functions
FFS	Forward Feature Selection
AdaBoost	Adaptive Boosting
KLT	Kanade-Lucas-Tomasi, an optical flow tracker
SIFT	Scale Invariant Feature Transform
SURF	Speed-Up Robust Features
MCT	Modified Census Transform
OpenCV	An Open-source Computer Vision library
LibSVM	A C++ library for Support Vector Machine
SMC	Sequential Monte Carlo
ROI	Region of Interest
FPR	False Positive Rate
FRR	False Rejection Rate
ARP	Auto Regressive Process
MB-LBP	Multi-Block Local Binary Patterns
pdf	Probability Density Function
SOR	Self-Organization Rate
TBB	Intel® Threading Building Blocks, an Open-source C++ library for multi-threading

1. INTRODUCTION

The amount of multimedia data is surging up with the increase in usage of digital cameras. Every day, numerous images and videos are captured and uploaded over social and media sharing websites i.e., Facebook, YouTube, Vimeo, etc. This type of multimedia data is often referred as a crowd-sourced or community-contributed media and it is normally labelled by textual information such as tags or titles. The text assigned to a multimedia data provides very limited information about it and can vary significantly, regardless of the actual contents. Content analysis of such multimedia data has a vast variety of applications. However, it is humanly impossible to manually analyze every single element of such a huge amount of data. This gives rise to the necessity of an automated analysis of multimedia data which in turn has brought forth the emerging field of Automatic Multimedia Content Analysis.

An important category of crowd sourced data contains a set of images and videos captured at a unique public or crowded event. Examples of such events include concerts, weddings, graduation ceremonies, parties and other public events that are captured by multiple photographers simultaneously. For examples, videos captured at an indoor public concert are shown in Figure-1. This type of data, as a whole, contains more useful information than a single video or image of the respective event. By looking thoroughly at the videos, we can find occurrences of the same person in multiple videos despite the fact that these videos were recorded by different people. Similarly, all videos share several other characteristics such as similar/same sound, background similarity, time stamps, etc. Hence, this data can be utilized more efficiently to extract valuable information about the event.

The objective of this thesis is to analyze the crowd-sourced videos of a single event to detect important/mainstream persons appearing in that event. The importance of an individual is a subjective judgment and it can vary, considerably, from one person to another. However, in some cases, it can also be generalized based on some fair assumptions such as in public events people usually capture notable individuals, for instance, singers/performers in concerts, bride/groom during the wedding ceremony, etc. Hence, these persons happen to appear relatively often in the data, and it can fairly be considered that the person who is captured by most of the people has high importance among the majority. Automatic detection of such individuals from multiple videos has a variety of applications that can easily be realized in identity-specific multimedia content retrieval, automatic video remixing, and also in surveillance applications.



Figure 1-1: Examples of the videos captured at the same event but by different photographers. We can see that many individuals are appearing in multiple videos even though the videos are taken by different people.

In this thesis, such individuals are referred as **Important Persons** and all others are called **Casual Persons**. The problem posed here is similar to unsupervised person re-identification where appearance of each person, appearing in multiple videos, is detected and identified. Subsequently, a method to rank these persons according to their amount of occurrence is needed to finally detect important persons.

For person re-identification, the most distinct feature that can be used for detecting and identifying a person is his/her face. Face detection and identification are very well studied research topics in the field of computer vision. However, both areas are still considered unsolved for unconstrained environments. Current work toward this direction focuses on multi-view face detection for detecting faces under severe pose variation [10, 47]. For identification, most of the research has been carried to cope with illumination variations, appearance/pose variations and other real world challenges [30, 84]. However, in most cases, these problems are tackled by restricting face pose in frontal position and keeping the lightening condition homogeneous.

Considering the aforementioned challenges, re-identification of every individual from the crowd sourced video is even a more challenging task as videos are captured by unknown users and the capturing conditions cannot be restricted to help the detection and identification modules. For example, the videos can be captured from different view angles, using different types of cameras and at varying time intervals. Similarly, the environment of the event is also totally unconstrained and can vary from indoor to outdoor, night to day time, etc. Hence, pose, facial expressions, scale, illumination conditions, image quality and occluded regions of a person vary significantly from one video to another, and it increases the complexity of detection and identification algorithms.

Recently, detection of high-level attributes (i.e., gender, ethnicity, eye wears, beard/non-beard, hair colors, etc.) has gained a lot of attention for the task of the person identification, and they are proven to be robust against face pose and expres-

sion variations [41, 57, 86]. In case of videos from a single event, these attributes are even more useful as most of such attributes will not change in all videos.

Other than face related features, clothing information of a person can also be utilized and has been proven helpful for the identification process [1, 6, 29]. Similar to high level facial attributes, one can strongly assume that the clothes of a person will not change in a single event. Though, most of the color features suffer from the variations in illumination conditions, change in image quality and capturing devices but color information can still provide useful information as it is considered to be robust against pose variations.

In addition to face and clothing features, human gait and biometric information i.e., iris, fingerprints, voice are also being utilized in the literature for person recognition. But none of these features are available in case of crowd-sourced videos as for the gait recognition; the whole body of the person should be visible in the frame which is unsure in the current case. Whole body could be available in videos captured from mid or wide-angle shots, whereas it is the other way around in the close-ups. Similarly, iris and fingerprint information is also not obtainable from such videos. Speaker identification needs a correlation between image and audio data which also cannot be ensured in the domain of the current problem.

Furthermore, unlike a single image, videos contain more enriched information that can be extracted by exploiting the temporal relation between consecutive frames. This task can be accomplished by employing a tracking algorithm (e.g. Kalman filters, Particle filters, etc.) to track persons in successive frames which will result in a more diverse amount of data. The idea is to represent the appearance of a person based on the information extracted from all these frames. This will help, for example, in cases where the person is moving or changing his/her body pose. In such scenarios the information will be available from different views and the identification algorithm can exploit the best possible information to improve the identification performance.

It is clear from the discussion above that the person specific properties that are readily available in crowd-sourced videos, and will not vary significantly, are the faces, clothing color and high-level attributes. In this thesis, all these modalities are exploited to benefit the overall performance. First, clothing information along with face is used to enhance the tracking of every individual. Later, these modalities are utilized for better re-identification, as they provide complementary information to each other and can result in a better identification.

The divide and conquer rule is recognized to be useful in applications where a huge amount of data is needed to be processed. In the current problem, the data consist of videos where the video can vary from a small to a very large length. For longer videos, it is better to divide them in smaller clips, commonly referred to as shots, based on a criterion such as camera panning, variation in camera angle, etc. This provides better boundaries for the tracking algorithm, and also provides enhanced possibilities of parallelism to improve the computation cost of the algorithm.

In this regards, any traditional shot boundary detection algorithm can be used to divide a larger video based on scene change. However, in this thesis, a very simple yet effective, sensor based algorithm is used to accomplish this task.

Moreover, as the data is crowd-sourced, no prior information about a person's identity or training data is available to aid the identification algorithm. Therefore, this problem requires an unsupervised classification technique that can group appearances of every individual from all videos and assign a unique identity to each one of them. For this purpose, any unsupervised learning method can be used such as K-means clustering, Hierarchical Agglomerative Clustering, etc. Although there is no labelled data available, one can still extract some useful information by just using common sense constraints. For example, in case of videos, multiple persons who appear in overlapping frames cannot belong to the same identity. This is very useful information as we have some clues about different identities and these evidences can be effectively used to help clustering/classification algorithms such as for K-means clustering in [39], HAC in [14] and for MRF in [48].

Subsequently, ranking of persons can be done based on a simple criterion. For example, as described earlier, important person is the one who appears the most in the whole event. Therefore, one can simply rank every individual based on the total number of video frames where the respective person appears. Another approach could be to count the number of video sequences in which a person appears and important persons are the one, who appear in a number of videos, greater than a certain threshold.

Built upon these ideas, this thesis provides a unified and robust framework to deal with aforementioned difficulties and to detect important persons without any prior information or training data. The whole framework is completely automatic and does not require any human intervention. For performance evaluation, the proposed framework is evaluated on videos captured in five different events, recorded by different cameramen/people. To compare the performance with state of the art methods, the proposed framework has also been tested on a multi-event dataset where videos are captured at different events and the goal is to detect important persons together from videos of all events. In this case, no clothing information is used for identification between different videos as the assumption of homogenous clothing color is false for multi-event scenarios.

1.1. Literature Review

Crowd-sourced video analysis is a very recent research area and there is not much work available in this direction, particularly for single-event case. However, literature can be found on topics related to person re-identification and clustering of individuals in videos or images. In the rest of this section related works, already proposed in the literature, are discussed.

Recently, an increased amount of attention has been gained by the algorithms for unsupervised/semi-supervised person re-identification from videos. In this direction, the work that is closest to ours is of Barr et al. [32] where the most appearing persons, referred as “questionable observer” in their work, are detected from videos using an unsupervised learning method. Individuals are detected by running a commercial face detector at every possible frame of the videos. Later, temporal correspondence between the faces of distinct persons is tracked by using Kanade-Lucas-Tomasi optical flow tracker [33] to form the face tracks. As a post-processing step, outliers are eliminated from extracted face tracks to avoid noisy patterns followed by feature extraction and clustering of face tracks using HAC. In result, clusters originated from a number of videos greater than a certain threshold are considered as most appearing persons.

The problem of important person detection shares many common properties with identity specific video indexing. Many video indexing algorithms count on the repeated appearance of an individual to divide a larger video into more logical subsequences. Recently, P. Hao et al. [65] have proposed an automatic people organization algorithm for individual retrieval from videos. After tracking of individuals based on faces, scene tracks are formed by merging faces, appearing in a particular time span. Temporal information about scene tracks is then used to automatically gather training data, following a rule that the tracks occurring in overlapping frames surely represent different identities. Later, an improved distance metric is learnt, using the automatically collected training data, to find the distances between scene tracks and identify unique individuals. Other related work has been proposed by Cinbis et al. [69] where a self-supervised similarity metric is learnt from face tracks of the characters appearing in unconstrained TV-videos. Other works focusing on the same problem of face track clustering include; constraint propagation based unsupervised by Tao et al. [31, 34], a divide and conquer based strategy by G. Gou et al. [19] and a video diarization algorithm by E. Khoury et al. [17]. However, most of these works are targeted for TV-videos captured by professional photographers. Unlike crowd sourced videos, TV-videos are more structured and normally contain more close-up scenes. Moreover, very less variability in video quality can be found from one episode to another. Hence, face detection and tracking is easier in these scenarios.

Identity based organization of photo/video albums is another very interesting topic that have much in common with important person detection. Although most of the work in this direction has been done for photo-albums, the ideas proposed in these works can directly be used for videos, as a video is nothing but a sequence of images. Most of the album organization methods rely on face and cloth information to find similarities in people [6, 43]. One of the earliest works in this direction is of B. Suh et al. [6]. In their work, a semi-automatic image annotation tool to cluster images corresponding to same event is proposed. First, timestamp information is used to group images of a same event. These images are then clustered again to

group appearances of each individual using visual feature extracted from the torso of a person where the torso is taken as a bounding box below the face region.

Similarly, J. Sivic et al. [29] have used facial features along with clothing and hair color information to find people in repeated shots of the same scene. Single-linkage HAC is then employed to group occurrences of the same individuals. Color based pictorial structure model is used to complement face detector and to detect people that were missed by the face detector. Most of the aforementioned work relies on clustering techniques that require some prior knowledge about the number of unique clusters or a cutoff value to stop the clustering process. To tackle the problem of prior knowledge, recently, another very interesting work for the automatic face association, in photo albums, has been proposed by Presti and Cascia [42]. For this purpose, an online-learning method is utilized and a collection of classifiers is updated at every iteration. First, images are ranked according to the number of faces in it and individual classifiers are trained for all the faces appearing in the image with the maximum number of faces. Afterward, faces detected at every image are classified by the classifiers trained for the already discovered identities. Confidence values received from all classifiers are used in a probabilistic framework to either assign a new face to already known identities or a new identity classifier is trained if the classification accuracy is below a certain threshold (that means a new person is discovered). In case a face was assigned to already known identities, respective classifier is updated by adding a new face as training data. Unlike traditional clustering algorithms, this approach neither requires any prior knowledge about the number of clusters nor a cutoff threshold.

During the past few years an extensive amount of research has been carried out for supervised person re-identification from videos. Although these works require prior knowledge about the person's identity or the training data but the modules of person detection/tracking, feature extraction, fusion of multiple modalities remain the same. Hence, several research motivations can be found from these works. Few of the interesting works worth referring are mentioned next. M. Tapaswi et al. in [48] propose a framework for re-identification of characters in videos (TV series/movies). Instead of face tracks, the attention is shifted toward person tracks to also identify those people who are missed by the face detector. For this purpose, a separate full-body tracker is employed in their work and the result of the face and body tracker is merged to get optimal person tracks. Supervised learning methods are adopted for face and speaker recognition. Results of face, clothing color and speaker recognition along with other contextual cues are merged together using Markov Random Field to enhance the performance of identification. For the same objective, the work of M. Everingham et al. [49] for recognition of characters in TV-videos and an algorithm for automatic labelling of faces appearing in videos by J. Sivic [35], are also few to be named.

1.2. Thesis Outline

The rest of this thesis is organized as follows. Chapter 2 explains all the required knowledge about individual modules of the important person detection scheme. The face detection algorithm is briefly explained followed by the illustration of visual tracking using particle filters. The description of feature extraction, clustering algorithm, classification technique and performance metrics used in this thesis are also depicted in this chapter. Based on the theoretical foundations formed in Chapter 2, Chapter 3 develops the basis of individual modules of the important person detection scheme and describes the implementation details of the face and body tracker, followed by an explanation of their merging strategy. It also contains the description of person representation schemes using low-level facial features, clothing color and high-level attributes. The whole important person scheme is explained in Chapter 4 followed by the evaluation of the whole framework on two different datasets in Chapter 5. Finally, Chapter-6 concludes the whole thesis along with a detailed sketch of possibilities for further enhancements and possible future directions.

2. THEORETICAL BACKGROUND

This chapter develops the foundations for the key concepts used in this thesis. It discusses the theoretical details of these concepts. However, the ways in which they are used in the important person detection scheme are explained in Chapter 3. The chapter starts with the explanation of face detection algorithms followed by an in-depth description of visual tracking using particle filters. Later, other key concepts, such as unsupervised classification using HAC, SVMs for supervised classification and the performance metrics used in this thesis are described.

2.1. Face Detection

Face detection is an essential part of any face recognition/identification problem. Numerous face detection algorithms have been proposed in the literature. One of the most famous algorithms for face detection is proposed by Viola and Jones [66] which utilizes the boosting algorithm [88] for learning a strong classifier. The cascade architecture of classifiers is utilized for a computationally efficient implementation. The original algorithm proposed in [66] is based on Haar-like features. However, using almost the same algorithm, several other features have been used in the literature such as an extended version of Haar-like by Lienhart et al. [71], MCT by B. Froba et al. [7], LBP features based face detector by S. Liao et al. [73], and numerous others.

In this thesis an already available implementation of the face detector, available in OpenCV [20], is used. The OpenCV implementation of the face detector utilizes two different features; extended Haar-like features [71] and MB-LBP [73]. In this work, MB-LBP features based face detection is opted due to its computational efficiency compared to Haar features, and its robustness against illumination variations. The rest of this section briefly explains the key ideas behind the face detection algorithm using MB-LBP features.

2.1.1. Multi-Block Local Binary Pattern

Local Binary Patterns, first proposed by Ojala et al. [77], have been proved to be robust against illumination changes and efficient for capturing the underlying textural information of an image [73, 77]. Since the development of LBP, its many variants have been proposed in the literature such as Extended-LBP [77, 85], Improved-LBP [23], MB-LBP [73], etc.

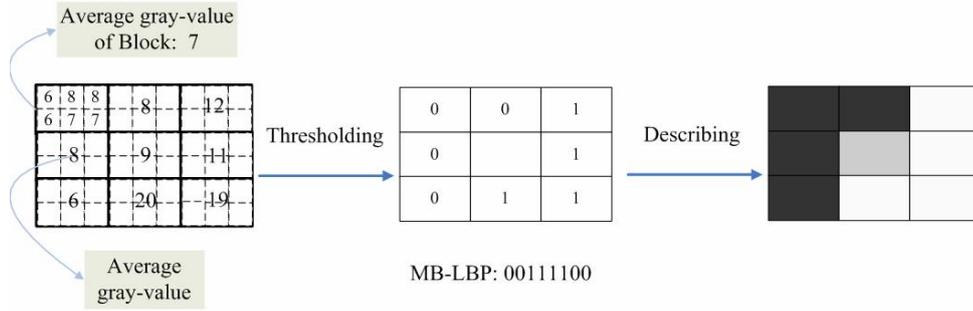


Figure 2-1: Illustration of the Multi-Block LBP [44].

The original LBP utilizes a 3x3 kernel to encode the local spatial structure of images by comparing pixel intensity of the center pixel with its eight neighbors. The values of neighboring pixels are thresholded to 0 or 1 and in result an ordered binary pattern is generated whose decimal form gives the final LBP-code [77]. Based on the same idea, instead of each pixel, MB-LBP encodes rectangular regions by comparing their average intensity value g_c with average intensities of eight neighboring rectangles as illustrated in Figure 2-1. The final MB-LBP code is defined as;

$$\text{MB-LBP} = \sum_{i=1}^8 f(g_i - g_c) 2^n, \quad (2.1)$$

where g_i is the average intensity value of neighboring pixels ($i = 1, \dots, 8$) and $f(x)$ is defined as follows;

$$f(x) = \begin{cases} 1, & \text{if } x > 0 \\ 0, & \text{if } x < 0 \end{cases} \quad (2.2)$$

In total 256 unique MB-LBP codes are obtained and these codes are directly used as features in later stages of face detection. MB-LBP is computed over the whole image and the resultant image is processed in patches of size $N \times N$. For a patch of size 20×20 , 2049 MB-LBP features can be computed that can directly be used for classification of a patch as either face or non-face [44]. However, most of these 2049 features are redundant and hence, a boosting approach is utilized for selecting the most discriminating features for classification of a patch either as face or non-face. The boosting approach for feature selection and efficient classification is briefly explained next.

2.1.2. Boosting and Cascade Classifiers

Boosting for Weak Classifiers

Viola & Jones [66] utilize boosting [90] for first learning weak classifiers $h_i(X)$ and then forming a strong classifier, $H(X)$, by taking the linear combination of weak classifiers as follows;

$$H(X) = \sum_{i=0}^n w_i h_i(X), \quad (2.3)$$

where w_i are the weights assigned to each weak classifier and, are learnt using the boosting approach. At the first iteration, uniform weights are assigned to each training sample followed by the selection of weak classifiers such that the one that minimizes the weighted error-rate over the entire training sample is selected first. In every iteration, the weights for misclassified samples are increased, and the error rates for remaining classifiers are evaluated again over the entire training set. Hence, the boosting algorithm emphasizes more on samples that are more difficult to classify. Afterward, a strong classifier is formed as given in equation (2.3). Several boosting techniques have been proposed in the literature e.g., AdaBoost, Gentle AdaBoost, Real Adaboost, Discrete Adaboost, etc. [71]. Any of these techniques can be used for this purpose, as most of them only differ in the weighting strategies.

Cascade Architecture

To detect every possible face from images, a sliding window approach is needed, which evaluates every patch of size $N \times N$ for classification. In addition to detecting faces of different sizes, the process is repeated at different scales. This procedure in its simplest form requires a huge amount of work which is not favorable for real-time applications. For this reason, Viola and Jones [66] also proposed an efficient algorithm consisting of a cascade of classifiers which, not only give better detection accuracy, but also drastically decrease the computation time.

The cascade architecture consists of M stages and at each stage a boosted classifier is trained. The idea is to first use simple classifiers to reject most of the background regions in earlier stages, and use more complicated classifiers in later stages. In this way only strong candidates of face will go in later stages and more complex classifiers will be used only for these candidates. A simple illustration of the cascade architecture can be seen in Figure 2.2. For more detail on face detection, the readers can refer to [23, 44, 66, 85].

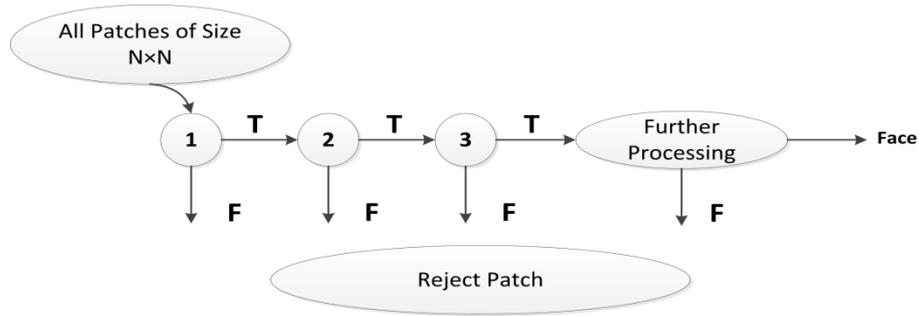


Figure 2-2: Illustration of the cascade architecture used in face detection

2.1.3. Multi-view Face Detection and Pose Estimation

In most of the face verification problems, normally frontal or near-frontal face pose is assumed. However, it is not the case in crowd sourced video due to the very little or no cooperation between the photographer and the person on the scene. Hence, a pose invariant face detector capable of detecting faces from variant pose angles is needed. One way of getting such detector is to train a single detector with training samples from variant yaw angles. Training a detector with this method will put too much burden on the classifier and, as a result, it will not be able to generalize any face angle. The rest of this section explains the opted approach in detail.

Several multi-view face detector approaches have been proposed in the literature [24, 50, 87]. However, a simple yet effective approach is adopted in this thesis which utilizes several face detectors trained at dedicated face angles. Several MB-LBP based face detectors are trained for the following yaw-angles;

$$\theta = \{0^\circ, \pm 15^\circ, \pm 30^\circ, \pm 45^\circ, \pm 60^\circ\}.$$

Five out of nine detectors ($0^\circ, \pm 30^\circ, \pm 60^\circ$) are run in parallel over the whole frame to detect faces from these angles. Later a voting strategy is used to merge the output of each detector for final face localization. As the classifiers used for face detector are insensitive to small localization errors, each detector gives multiple overlapping detections around face regions. This normally happens because of running detectors at every possible image-patch and also at different scales. An example of such overlapping detections can be seen in Figure 2-3.

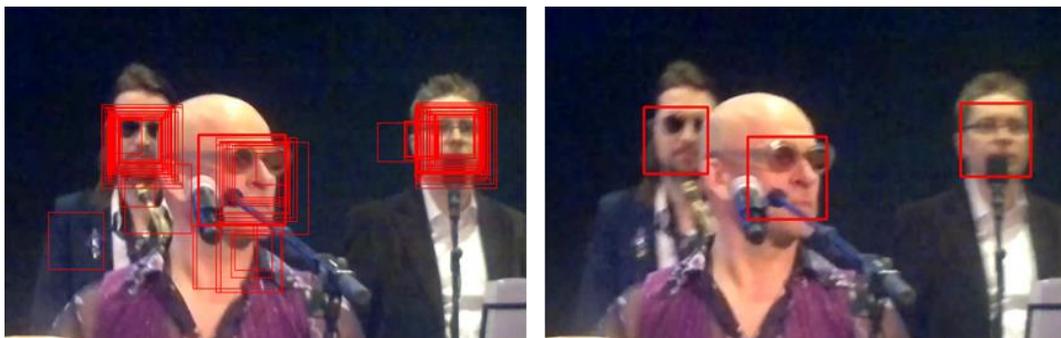


Figure 2-3: Example of the face detection. (a) Overlapping regions detected by the face detector. (b) Final detections using the multi-view face detector.

Such overlapping detections usually appear with fewer consistencies for less confident face regions or background patches [89]. We use this assumption and consider the number of such overlapping detections as a confidence value for a detected face. Detection is considered as true if at least three detectors give a confidence value greater than a certain threshold. Following, the winner takes all strategy; the x-y coordinates for the detected face are chosen as the one given by the detector with the highest confidence value. Similarly, face pose is estimated by assigning the angle value of the detector with the maximum confidence value. Estimated pose value will be used in several preprocessing steps at later stages.

For training of face detectors, training images are gathered from various datasets that also provide information about the face pose. These datasets include FERET [67], PIE [78], Prima Head Pose [58], BioId [64], and AR face dataset [2]. To further increase the training set, more training samples are generated by applying simple geometrical transformation to the existing images such as changes in scale, position, slight rotation, shear, translation, etc. [37]. Moreover, under the assumption of facial symmetry, training samples for a particular pose angle are also used for its negative pair by rotating them in horizontal direction.

2.2. Visual Tracking

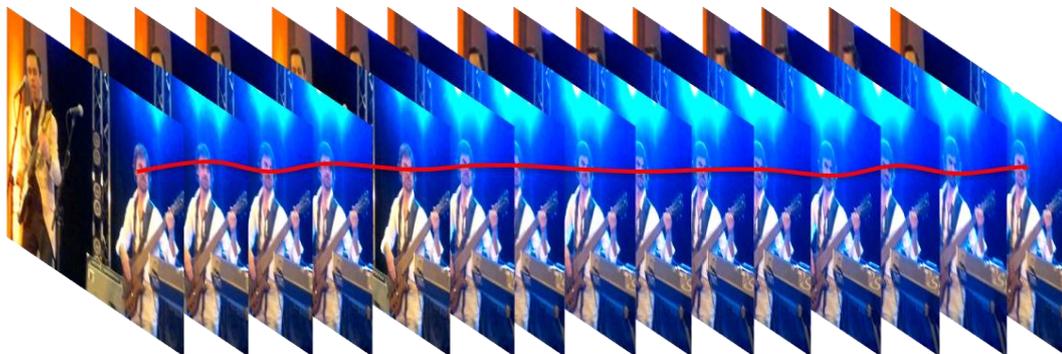


Figure 2-4: Example of the temporal relation achieved by using visual tracking.

Face detection is computationally very expensive task and running several detectors on each frame of a good quality video is far from real time, even with modern computers. To cope with this problem a visual tracking algorithm is needed. The idea is to detect faces on only the first frame and track them in the rest of the videos. In addition to computational improvement, visual tracking can also provide information about the temporal correspondence between detected faces of the same person in consecutive frames as shown in Figure 2.3. In this section the basic concepts of visual tracking algorithm, used in this thesis, are explained and its implementation details are provided in Chapter 3.

Visual tracking is a process of estimating the state of a system based on the noisy measurements evaluated on the respective system. Different techniques are

available in the literature in this regard. Kalman Filter [72], Particle filters [40, 51], KLT for feature tracking [33] are a few of the most famous tracking techniques. Among these, particle filters based visual tracking is the most reliable as the Kalman filter normally does not work well in real life scenarios due to its assumption of linear or Gaussian probability distribution. For KLT based tracking, sub-regions to track features in consecutive frames are needed to be known and tracking performance is considerably dependent on the size and location of these regions. Therefore, particle filters based tracking is adopted in this thesis as it can cope with multi-modal probability distributions including linear and Gaussian distributions [40, 51].

2.2.1. Particle Filter Based Visual Tracking

Particle filters, also known as Sequential Monte Carlo (SMC) method, is based on a time (t) dependent system of dynamical model and measurements. It requires two models to analyze the system; a system model and a measurement model [51]. The system model, also known as transition model, represents the evolution of the system in time and it is represented as follows:

$$\mathbf{x}_t = \mathbf{f}_t(\mathbf{x}_{t-1}) + \mathbf{v}_{t-1}, \quad (2.4)$$

where \mathbf{x}_t is the state of the system and \mathbf{v}_t represents the stochastic error (noise) in the state update. Noise \mathbf{v}_{t-1} in the state update is a random variable of known statistics and according to the property of Markov process of order-one, the state \mathbf{x}_t depends only on the previous state \mathbf{x}_{t-1} . Hence, equation (2.4) describes a probability density function (pdf) $p(\mathbf{x}_t|\mathbf{x}_{t-1})$, called as prior distribution function.

The measurement model represents the noisy measurements (observations) \mathbf{z}_t taken on the system. It depends on the state of the system \mathbf{x}_t and an error term \mathbf{n}_t that is caused due to the uncertainty while measuring the state. This measurement model is expressed as follows:

$$\mathbf{z}_t = \mathbf{h}_t(\mathbf{x}_t) + \mathbf{n}_t. \quad (2.5)$$

Measurement model also represents a pdf $p(\mathbf{z}_t|\mathbf{x}_t)$ as the error term \mathbf{n}_t is also a stochastic variable. This pdf is normally termed as likelihood function. In object tracking, our goal is to estimate the state of the system \mathbf{x}_t by calculating the posterior density $p(\mathbf{x}_t|\mathbf{z}_t)$ using the observation density and posterior density at time instance t-1. In particle filtering, this is done in two steps: prediction and the update. During prediction step, the state of the object is modified according to a transition/dynamics model $p(\mathbf{x}_t|\mathbf{x}_{t-1})$. In the later stage, the state of the object is estimated using the likelihood density $p(\mathbf{z}_t|\mathbf{x}_t)$. Both prediction and update stages are briefly explained next.

2.2.1.1 Transition Model

In the first step of particle filtering, the evolution of the object over time is predicted using a state transition model $p(\mathbf{x}_t|\mathbf{x}_{t-1})$. We consider a non-linear and non-Gaussian motion of the object. Hence, a model to effectively characterize the behavior of the object is needed. Auto Regressive Process (ARP) is the most commonly used model that describes the behavior of a system in this scenario based on its previous states [36]. ARP efficiently represents the object's behavior and also helps to avoid an exhaustive search. The order of the ARP depends on the application such that how well it represents the motion of the object in a particular application. In the 1st order ARP [3], the state of the object is estimated by using the previous state and additive Gaussian noise. The 2nd order ARP [91] utilizes the linear combination of two previous states with additive Gaussian noise. The number of previous states increases with the order of ARP. In order to effectively capture the movement of the object, a 2nd order ARP is used in this thesis as follows;

$$\mathbf{x}_t = A\mathbf{x}_{t-1} + B\mathbf{x}_{t-2} + C\mathcal{N}(0, \sigma), \quad (2.6)$$

where A, B and C are the auto-regression coefficient and $\mathcal{N}(0, \sigma)$ is Gaussian noise with zero mean and unity standard deviation. In this work these coefficients are selected empirically to get the optimal results.

Representation of the state \mathbf{x}_t also varies with the application. However, for visual tracking, its most basic form includes the x-y coordinates of the spatial location, height and width of the object as given in equation (2.7). Adding more parameters will result in more efficient tracking. However, the number of parameters also determines the trade-off between accuracy and computation cost of the algorithm.

$$\mathbf{x}_t = [x_t \ y_t \ w_t \ h_t], \quad (2.7)$$

where x_t and y_t represent the spatial location of the object and w_t and h_t are its width and height respectively. Two different representations of the object's state are used in this work; one for face tracking and other for the clothes tracking.

2.2.1.2 Observation Model

Observation (measurement) model $p(\mathbf{z}_t|\mathbf{x}_t)$, also known as likelihood, is another important model in visual tracking. It gives us the likelihood value such that how confident it is about the particular state of the object. These observations are used to estimate the position of the object. Selection of observation model also critically depends on the characteristics of the object to be tracked. Various observation models have been adopted in the literature such as color features [15, 68], gradients [74], detector's confidence [52, 53], multimodal features [8], etc. In this the-

sis, the color and detector confidence based observation models are used for clothes and face tracking respectively. Both observation models are briefly explained next. However, the implementation details for both are given in Chapter 3.

Detector's confidence for face tracking

For face tracking, we have directly used our multi-view face detector as observation model to take the measurements \mathbf{z}_t . Face detector is run at a given region defined by \mathbf{x}_t and the confidence value (Section 2.1.3) given by the detector is considered as likelihood. Further implementation details are given in Chapter 3.

HSV color histogram for clothes tracking

The most intuitive property of clothes is the color and unlike texture of the clothing, the color information will not vary much with the change in human pose. Moreover, color based representation has been proven to be robust against many practical problems such as scale variations, rotation and partial occlusion [15, 68]. Hence, in this work, an HSV color histogram is utilized to form an observation model for clothes tracking. The motivation behind the usage of HSV color space is that it is less sensitive to variations in lightening conditions as it decouples the intensity and color information [68]. The simplest way is to compute N-bins histogram from the region where the object of interest appeared at state \mathbf{x}_{t-1} (Target region) and also from the candidate region where the object is expected to appear at state \mathbf{x}_t . Once the histograms of both, the target $hist_{target}$ and candidate $hist_{candidate}$, are composed, the distance between them is taken as hood $p(\mathbf{z}_t|\mathbf{x}_t)$. Various similarity measures exist in the literature such as Euclidean distance, histogram intersection, Bhattacharya distance etc. [92]. In this thesis, the Bhattacharya coefficient measure is used to find the similarity between both histograms as follows:

$$\rho[hist_{candidate}, hist_{target}] = \sum_{n=1}^N \sqrt{hist_{candidate}^n \cdot hist_{target}^n}, \quad (2.8)$$

ρ is the Bhattacharya similarity coefficient and gives the information about similarities between both histograms; a higher value of ρ represents more similar objects and $\rho = 1$ a perfect match. The Bhattacharya distance is considered as likelihood and can be computed as follows:

$$d = \sqrt{1 - \rho[hist_{candidate}, hist_{target}]}, \quad (2.9)$$

Histogram directly computed from the rectangular clothes region is not very promising as it is very likely that the region contains pixels from the background. Hence in this thesis, a more sophisticated technique to evaluate likelihood is used and its implementation details are given in Chapter 3.

2.2.1.3 Particle Filtering

Until now we have built the concepts required for particle filtering. In this section these concepts are utilized and a description is given for estimating the posterior density $p(\mathbf{x}_t|\mathbf{z}_{1:t})$, where the vector \mathbf{x}_t represents the state of an object at time t and $\mathbf{z}_{1:t}$ defines the noisy measurements (observations) and its history from the start. As discussed earlier, in visual tracking the goal is to estimate the state \mathbf{x}_t of the object by utilizing the observations $\mathbf{z}_{1:t}$. Under the assumption of \mathbf{x}_t satisfying the property of Markov process and observations being conditionally independent from the state of the object, the posterior density can be computed using the Bayes' rule as

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) = \frac{p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{z}_{1:t-1})}{p(\mathbf{z}_t|\mathbf{z}_{1:t-1})}, \quad (2.10)$$

where the term $p(\mathbf{z}_t|\mathbf{x}_t)$ is the likelihood density described in Section 2.2.1.2, $p(\mathbf{x}_t|\mathbf{z}_{1:t-1})$ is called prediction term and is responsible for prediction stage of particle filtering and the term $p(\mathbf{z}_t|\mathbf{z}_{1:t-1})$ is referred as observation prior or evidence.

The prediction term $p(\mathbf{x}_t|\mathbf{z}_{1:t-1})$ is computed via transition prior $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ at time t and the posterior density of the previous state of the system $p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})$. For this purpose the Chapman-Kolmogorov equation is used as follows:

$$p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})d\mathbf{x}_{t-1}. \quad (2.11)$$

The normalizing term (observation prior) is used to ensure that posterior density is legitimate and is calculated as

$$p(\mathbf{z}_t|\mathbf{z}_{1:t-1}) = \int p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) d\mathbf{x}_t. \quad (2.12)$$

It can be seen that all the terms required in the formulation of the posterior density given in equation (2.10) are available. Substituting the terms in equation (2.11) and equation (2.12) into equation (2.10), will provide us the final formulation of posterior density as follows:

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) = \frac{p(\mathbf{z}_t|\mathbf{x}_t) \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})d\mathbf{x}_{t-1}}{\int p(\mathbf{z}_t|\mathbf{x}_t) \left(\int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})d\mathbf{x}_{t-1} \right) d\mathbf{x}_t}. \quad (2.13)$$

For real world applications, such as visual tracking, the exact solution to the equation (2.13) is intractable and can only be obtained for linear or Gaussian distributions. To cope with this problem, Monte Carlo sampling is used to estimate the solution. Monte Carlo sampling is the most important step of particle filtering where the goal is to estimate the posterior density using a collection of random

samples $\{\mathbf{x}_t^i, \omega_t^i\}_{i=1}^{N_p}$, called particles, with weights ω_t^i associated with each one of them and N_p being the total number of particles. Estimation of posterior density approaches the optimal Bayesian solution with a number of particles equal to infinity. The sum of N_p Dirac functions, centered at each sample, can be used to estimate the posterior density at time instance t and is formulated as follows:

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) = \sum_{i=1}^{N_p} \omega_t^i \delta(\mathbf{x}_t - \mathbf{x}_t^i), \quad (2.14)$$

where the weights ω_t^i are normalized such that they sum unity and, are expressed as

$$\omega_t^i \propto \frac{p(\mathbf{x}_t^i|\mathbf{z}_{1:t})}{q(\mathbf{x}_t^i|\mathbf{z}_{1:t})}, \forall i = 1, 2, 3 \dots, N_p. \quad (2.15)$$

In equation (2.15), $p(\mathbf{x}_t^i|\mathbf{z}_{1:t})$ is the target distribution that we want to estimate and $q(\mathbf{x}_t^i|\mathbf{z}_{1:t})$ is the importance density function used to approximate the current set of particles, \mathbf{x}_t^i . Let, $\{\mathbf{x}_{t-1}^i\}_{i=1}^{N_p}$ are the particles at time $t-1$ and posterior density $p(\mathbf{x}_{t-1}^i|\mathbf{z}_{1:t-1})$ is the density approximated over these particles. Then by using the representation of $p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})$ as in equation (2.14), substituting it in equation (2.11) and using equation (2.13), the following formulation of $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ is obtained,

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) \propto p(\mathbf{z}_t|\mathbf{x}_t) \sum_{i=1}^{N_p} \omega_{t-1}^i p(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i). \quad (2.16)$$

At time instant $t-1$, the predicted posterior density $p(\mathbf{x}_t|\mathbf{z}_{1:t-1})$ can be approximated using transition probabilities weighted by weights ω_{t-1} . Hence, using equation (2.14) and equation (2.16), a modified set of weights $\{\omega_t^i, i = 1, 2, \dots, N_p\}$ are generated as

$$\omega_t^i \propto \omega_{t-1}^i \frac{p(\mathbf{z}_t|\mathbf{x}_t) p(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i)}{q(\mathbf{x}_t^i|\mathbf{z}_{1:t})}. \quad (2.17)$$

In Bayesian tracking, it is suitable to use prior knowledge as importance density,

$$q(\mathbf{x}_t^i|\mathbf{z}_{1:t}) = p(\mathbf{x}_t|\mathbf{x}_{t-1}^i). \quad (2.18)$$

Hence, substituting equation (2.17) into equation (2.16) produces,

$$\omega_t^i \propto \omega_{t-1}^i p(\mathbf{z}_t|\mathbf{x}_t). \quad (2.19)$$

A very common problem associated with SMC particle filter is the degeneracy of particles, where some of the particles after a few iterations, have almost zero weight and do not contribute in the posterior density $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ approximation. To avoid this problem, Resampling algorithm is introduced to overcome the degeneracy phenomenon as explained next.

2.2.1.4 Re-sampling of Particles

Re-sampling in particle filtering framework allows reducing the effect of degeneracy such that the variance of the importance weights increases with the increase in time t and some particles tend to have higher weights while some of the particles have almost zero weight. Therefore, the basic idea of re-sampling is to eliminate the particles that have small weights and generate new particles by re-sampling (with replacement) from particles that have higher weights. A measure of degeneracy N_{eff} is defined as

$$N_{eff} = \frac{1}{\sum_{i=1}^{N_p} (\omega_t^i)^2} \quad (2.20)$$

Re-sampling is done only when N_{eff} falls below a certain threshold such that $N_{eff} \leq N_T$ and new particles are assigned with the new weights $\omega_t^i = 1/N_p$.

2.2.1.5 State Estimation

Finally, after approximating a new weighted set of particles $\{\mathbf{x}_t^i, \omega_t^i\}_{i=1}^{N_p}$ at time t , the state of the target is estimated as a linear combination of the samples \mathbf{x}_t^i and weights ω_t^i ,

$$\varepsilon(x_t) = \sum_{i=1}^{N_p} \mathbf{x}_t^i \omega_t^i \quad (2.21)$$

A pseudocode for the particle filtering algorithm is given in Figure 2-5.

Particle Filter Algorithm

$$\left[\{\mathbf{x}_t^i, \omega_t^i\}_{i=1}^{N_p} \right] = \text{PARTICLE_FILTER} \left[\{\mathbf{x}_{t-1}^i, \omega_{t-1}^i\}_{i=1}^{N_p} \right]$$

- 1 Sampling with Replacement
- 2 FOR $i = 1: N_p$
- 3 Draw $\mathbf{x}_t^i \sim p(\mathbf{x}_t | \mathbf{x}_{t-1}^i)$
- 4 Evaluate weights for the particle, $\omega_t^i \propto \omega_{t-1}^i p(\mathbf{z}_t | \mathbf{x}_t)$
- 5 END FOR
- 6 Calculate total weight : $weight_sum = \text{SUM} \left[\{\omega_t^i\}_{i=1}^{N_p} \right]$
- 7 FOR $i = 1: N_p$
- 8 Normalize the weights : $\omega_t^i = \omega_t^i / weight_sum$
- 9 END FOR
- 10 Calculate $N_{eff} = \frac{1}{\sum_{i=1}^{N_p} (\omega_t^i)^2}$
- 11 IF $N_{eff} \leq N_T$
- 12 $\left[\{\mathbf{x}_t^i, \omega_t^i\}_{i=1}^{N_p} \right] = \text{RESAMPLE} \left[\{\mathbf{x}_t^i, \omega_t^i\}_{i=1}^{N_p} \right]$
- 13 END IF

Figure 2-5: Pseudocode for the particle filtering algorithm.

2.3. Feature Extraction

Feature extraction is one of the most intrinsic steps of any recognition/identification application. It is the process of extracting discriminating information in a useful and compact form about an object i.e., face or clothes in the current problem. Good feature extraction techniques that give less intra-class variance and large inter-class variance regardless of variations in pose, illumination and other real life problems, are the most crucial element in the performance of any identification algorithm.

Extracted features are often divided into two different types according to their abstraction level; the low-level features and high-level features [12]. Low level features are the most commonly used in any multimedia analysis application due to ease of computation and their objective nature. They are normally used to find the similarities and dissimilarities between data objects. However, they lack the sense of the underlying semantics of the multimedia content they describe. High Level features, on the other hand, represent the meaning and purpose of the data objects. They normally cannot be extracted automatically and requires some training data or prior information. Utilizing the provided prior information, high-level features are normally detected based on smart combinations of low-level features. Due to the nature of high level features, they are completely application specific and are normally detected as a binary classification between different semantic concepts [57, 76].

In this thesis, both the low-level features (e.g., texture, color) and high level features (e.g., gender, race, hair color and such properties of a person's appearance) are used to represent the appearance of the person. In the rest of this chapter, low-level features used in this thesis are briefly described. As mentioned earlier, the high-level features are also detected based on the information extracted by the low-the level features. Therefore, the implementation detail of high-level feature extraction is given later in Chapter 3.

2.3.1. Textural Features

Textural features are normally the first choice whenever it comes to differentiate between objects based on the spatial arrangements of colors or intensities in an image. The textural features opted in this thesis includes Extended LBP (ELBP) [77] Edge Magnitudes, Edge Directions along with their different variants obtained by applying different normalization and aggregation types. The idea about a similar version of LBP has already been explained in Section 2.1.1. Therefore ELBP is not explained in this section. Readers interested in knowing more about Extended LBP can refer to [77, 79].

Edge detection is the most common technique to determine the presence and complexity of texture in an image or an ROI. The goal of edge detection is to find the most rapid changes in image intensities that are normally computed by the

Gradient operator,

$$\nabla I = \left[\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y} \right], \quad (2.22)$$

where I is our input image, $\frac{\partial I}{\partial x}$ and $\frac{\partial I}{\partial y}$ are its gradient toward x and y directions respectively. The direction of the edges can be computed as follows;

$$\theta = \tan^{-1} \left(\frac{\frac{\partial I}{\partial y}}{\frac{\partial I}{\partial x}} \right), \quad (2.23)$$

and the edge strength is given by the gradient magnitudes as

$$\|\nabla I\| = \sqrt{\left(\frac{\partial I}{\partial x} \right)^2 + \left(\frac{\partial I}{\partial y} \right)^2}. \quad (2.24)$$

Both the edge magnitudes and edge directions can be used as different features for the analysis of texture. The gradient of the image is normally approximated by using kernel operators such as Canny, Sobel, Prewitt, etc. In this thesis, the Sobel operator is used for this purpose. The block wise histograms of Extended-LBP is used to extract features of the face region to be used in the identification process and all other textural features are used to train the classifiers for detecting high-level attributes.

2.3.2. Color Features

Color features are used to represent the color related properties of an image. Color features used in this thesis include RGB, HSV and grayscale, pixel values or histograms. Histogram is the most commonly used method to represent the color distribution of an image. Each bin of the histogram represents the number of pixels from a particular range in the image. Normally 1-dimensional histograms are computed from the grayscale images. However, from color images, histograms can be calculated in two different ways.

The first and simple approach is to get a histogram of each channel of the image and concatenate them one after another. However, it does not give the actual representation of color as the inter channel association is lost. The other method is to compute a combined histogram directly from the 3D image, and in result a 3D histogram is obtained. Although the computation of 3D histogram is more complex than the simple histogram, it provides an accurate representation of colors and is highly quantized. For example, consider a bin size of $4 \times 4 \times 4$, it will quantize $64 \times 64 \times 64$ pixel combinations to 1 number (combination of $256/4 = 64$ values from each channel will go into a single bin).

HSV color histograms are used for cloth tracking (Section 2.2.1.2) and to represent the clothing color to be used for person identification. All other color features are used to train classifiers for high-level attribute classifiers.

2.4. Hierarchical Agglomerative Clustering

Given a collection of objects, clustering is an unsupervised technique to group objects such that the objects in the same group are closer to each other than the objects in other groups. Each group is called a cluster which has its own unique label/identity. Clustering has numerous applications in image processing, machine learning, pattern recognition, information retrieval, etc. In this thesis, HAC is used to group all occurrences of every individual based on the features extracted from their face and clothing regions.

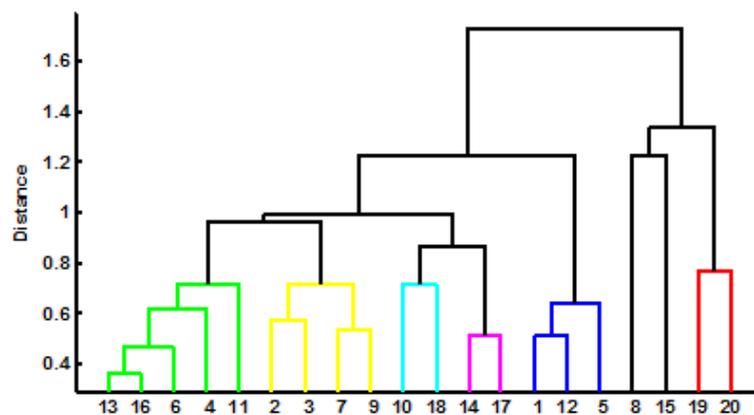


Figure 2-6: Example of the dendrogram obtained after HAC.

Hierarchical Agglomerative Clustering [80] is one of the most commonly used clustering algorithms in the literature. Unlike other algorithms such as K-means [75], it does not require any prior information about the number of clusters. Moreover, K-means critically depends on the initializations of centroids and, is prone to converge at local optima. On the other hand, HAC does not require initialization of centroids and always gives the globally optimum clusters. HAC follows a bottom-up approach and merges the closest pairs of clusters at each iteration. This process is often referred as linkage. The linkage process continues until the distance between two pairs is greater than a certain threshold. The result of linkage process can be seen as a binary tree normally termed as ‘dendrogram’. An example of dendrogram is given in Figure 2-6 where different colors represent unique clusters obtained after the linkage process. The X-axis in Figure 2-6 represents the total number of clusters and the y-axis is the distance between pairs of cluster at a particular level.

The result of HAC completely depends on the linkage process. Many linkage criteria have been proposed in this direction. The one used in this thesis along with

a few basic techniques are shortly explained next. Let C_1 and C_2 are two different clusters where N_1 and N_2 are the number of elements in each cluster respectively. Let us define $d(C_1, C_2)$ as the distance between two clusters and d_{ij} as the distance between the i^{th} element of C_1 and the j^{th} of C_2 , according to some distance metric.

Single Linkage

In single linkage, inter-cluster similarity is considered as the distance between the closest elements of both clusters. It gives more diverse and chain-like clusters. It is useful in the situations where very compact clustering is not needed. The distance $d(C_1, C_2)$ is defined as

$$d(C_1, C_2) = \min_{i \in C_1, j \in C_2} d_{ij}. \quad (2.25)$$

Complete Linkage

As opposed to single linkage, in complete linkage the inter-cluster similarity is considered as the distance between the farthest elements of both clusters. Unlike, single-linkage, it provides more compact clusters. The distance $d(C_1, C_2)$ in this case is represented as

$$d(C_1, C_2) = \max_{i \in C_1, j \in C_2} d_{ij}. \quad (2.26)$$

Average Linkage

As depicted from its name, average-linkage utilizes the average of distances between all pairs of elements in both clusters and the distance is denoted as follows:

$$d(C_1, C_2) = \frac{1}{N_1 N_2} \sum_{i \in C_1} \sum_{j \in C_2} d_{ij}. \quad (2.27)$$

Other linkage techniques include Centroid Linkage, Ward Linkage, V-Linkage, etc.

2.5. Support Vector Machines

Support vector machines (SVM), originally proposed by Boser et al. [9], is the most commonly used supervised learning methods in pattern recognition. SVMs are often referred as large margin classifiers due to their capability of learning the hyperplanes that separate the nearest training samples (support vectors) with the largest possible margins in higher-dimensional features space. The distance between the support vectors and separating hyperplane is called the margin of the classifier. The goal of SVMs is to learn the parameters of a mapping function that can map all the training samples to some real valued functions which separates them efficiently.

SVMs can be linear or non-linear. In the cases when data points are not linearly separable, SVMs can also utilize kernel functions, the approach named as kernel tricking, to transform the features into a higher-dimensional space. The goal of kernel tricking is to make the features linearly separable. In the case the samples are not linearly separable, even in higher-dimensional feature space, cost functions are used to penalize the function for allowing data samples to exist on the wrong side of the hyperplane.

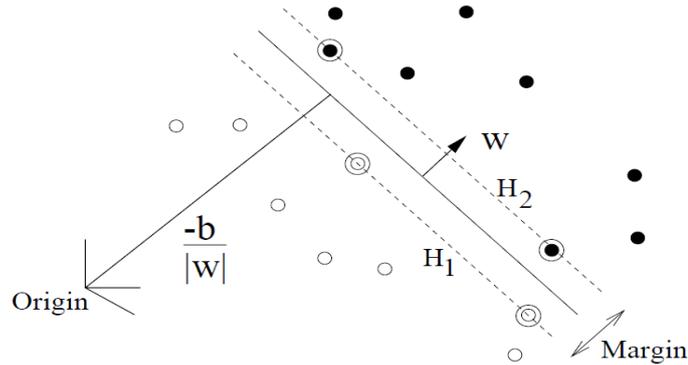


Figure 2-7: A separating hyperplane for the linearly separable classes. [13]

Formally, consider the labelled training samples $\{x_i, y_i\}_{i=1 \dots m}$ where $x \in \mathbb{R}^d$ and $y \in \{-1, +1\}$, the goal of SVMs is to learn a decision function $f(x, \alpha)$ that maps any arbitrary input x_b , with function parameters α , to a real value closer to its original label. If the training samples are linearly separable, the hyperplane that separates both classes is defined by the points satisfying $x_i \cdot w + b = 0$. Consider all training samples satisfy the following constraints

$$y_i(w \cdot x_i + b) \geq 1, \forall i = 1 \dots m, \quad (2.28)$$

where w is the normal to hyperplane and the parameter b is offset of the hyperplane from the origin. The margin of such classifier is defined by two separate decision planes, H_1 and H_2 in Figure 2-7, and is equal to $\frac{2}{\|w\|}$. For the same problem several set of the parameters w and b can be found that correctly classify the training samples. However, the classifier with lower margin will tend to have a higher expected error. Hence, to maximize the margin, the objective can be formed to minimize the Euclidean norm $\|w\|$ with the constraints in equation (2.28). For computational ease and to generalize the same formulation for nonlinear case this problem can also be expressed in form of Lagrange function [13] as follows;

$$L_p = \frac{1}{2} \|w\|^2 - \sum_{i=1}^m \alpha_i y_i (x_i \cdot w + b) + \sum_{i=1}^m \alpha_i, \quad (2.29)$$

where α_i are the Lagrange multipliers. This problem requires minimization of a convex objective function and is solved by convex quadratic programming [13].

The problem can be reduced even more by representing the normal vector as $w = \sum_{i=1}^m \alpha_i y_i x_i$ and under the constraints $\alpha_i \geq 0$ and $\sum_{i=0}^m \alpha_i y_i = 0$. Substituting these constraints and w , in equation (2.29) will provide a new dual form of Lagrange function as,

$$L_D = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i^T x_j. \quad (2.30)$$

To generalize the function for nonlinear cases the dot product in equation (2.30) can be replaced by the kernel function k as follows:

$$L_D = \sum_{i=1}^m \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j k(x_i, x_j). \quad (2.31)$$

As mentioned earlier, the kernel function is used to transform the feature vectors in higher dimensional feature space to make the problem linearly separable. Among the most famous kernel function used in SVMs framework are Radial Basis Function (RBF), Sigmoid and Polynomial.

2.6. Performance Metrics

Until now we have discussed several individual elements that are used in the proposed framework of important person detection. The overall performance of the whole system critically depends on the performance of every module. Hence to individually evaluate the performance of these modules and also to assess the final results of the whole framework, several quality metrics are used in this thesis. The rest of this section briefly explains these quality metrics.

2.6.1. Quality Measures for Detection Algorithms

The precision and recall are the most commonly used metrics to evaluate the performance of detection algorithms and are calculated using true positives T_P , false positives F_P and false negatives F_N . T_P corresponds to the count of correctly detected objects. False positives F_P , on the other hand, represents the number of mistakenly detected objects. Finally false negatives, as clear for the name, give the number of missed detections (i.e., the object is not detected when in reality it exists).

Using these counts, precisions and recall are described as follows;

$$Precision = \frac{T_P}{T_P + F_P}, \quad (2.32)$$

$$Recall = \frac{T_p}{T_p + F_N}, \quad (2.33)$$

Precision gives us the ratio of correctly detected objects from detection results. *Recall* provides the information about correctly detected objects from the whole collection of data. To calculate the *Recall*, ground truth / labelled data is needed.

2.6.2. Quality Measures for Classification Algorithms

Classification accuracy is the simplest and widely used measure to assess the quality of the classification / recognition algorithms. It is simply calculated by taking the ratio of correct classification with the total number of tests.

$$Classification\ Accuracy = \frac{Correct\ Classifications}{Total\ Number\ of\ Tests} = \frac{T_p}{T_p + F_p}. \quad (2.34)$$

Then the error rate is simply given as;

$$error\ rate = 1 - Classification\ Accuracy. \quad (2.35)$$

2.6.3. Quality Measures for Clustering

As mentioned earlier, HAC is used to group the appearances of every individual. After performing HAC an N number of clusters are obtained where each cluster represents a unique person. To check how well HAC has performed the grouping task, we have adopted the same quality metrics as used in [32]. This includes Self-Organization Rate (SOR) and Cluster Conciseness (CON).

Self-Organization Rate

SOR gives the information about the homogeneity of the clusters such that the amount of data samples that are grouped in to the correct classes (unique individuals in our case). SOR is described as follows;

$$SOR = \left(1 - \frac{\sum N_{AB} + N_e}{N} \right), \quad (2.36)$$

where N_{AB} indicates the number of data samples representing class-A that are grouped into a cluster dominated by the samples from class-B. N_e denotes the number of data samples that are assigned to a cluster in which no class dominates with more than half, and N represents the total of samples available for the clustering. Higher value of SOR represents more homogeneous clusters and higher accuracy of clustering.

Cluster Conciseness

CON provides the information about redundant clusters such that if more than one cluster represent the same class (person) all clusters except one are redundant. If data samples from a class-A represent majority in C_A clusters, then $C_A - 1$ of those clusters are redundant. The total number of redundant clusters C_r is given by

$$C_r = \sum_A C_A - 1. \quad (2.37)$$

The amount of non-redundant clusters is obtained as

$$CON = \left(1 - \frac{C_r}{C}\right), \quad (2.38)$$

where C is the total number of clusters obtained after the clustering algorithm. Similar to the SOR, higher value of CON represents good clustering efficiency.

2.6.4. Quality Measure for Important Person Detection

Since our goal is to detect important persons, it is possible that an important person is not detected as important or vice versa. Detection accuracy of important person detection is captured by False Positive Rate (FPR) and False Negative Rate (FNR). FPR represents the amount of casual persons classified as important. FNR denotes the amount of important persons that are missed by the system (not classified as important). Both FPR and FNR are described as follows:

False Positive rate =

$$\frac{\text{Number of casual persons classified as important persons}}{\text{Total number of known casual persons}}, \quad (2.39)$$

False Negative rate =

$$\frac{\text{Total number of important persons, not classified as important}}{\text{Total number of known important persons}}. \quad (2.40)$$

Both FPR and FNR range from 0 to 1 and the lower value of both measures shows better detection accuracy.

3. PERSON REPRESENTATION SCHEME

To re-identify a person across multiple videos, one needs to extract the useful information to efficiently represent the person's appearance. Most of the work toward this direction has been focused on extracting low-level features from the detected face regions. In low-level features, the main emphasis has been put for extracting textural features and then training a classifier, based on labelled training samples. Trained classifiers are used to recognize a person in other videos [48, 53, 55]. In case of supervised learning, the classifier automatically learns the intra-class similarities and avoids dissimilarities. However, for the current problem, no training data is available. Therefore, we need to directly use the coarse features for the purpose of re-identification in an unsupervised way.

Therefore, every individual is represented using three different modalities to get the best possible representation of a person's appearance. This includes low level facial features, high-level attributes and the clothing colors. All three modalities can provide complementary information under pose, illumination and appearance variations. Of course, before extracting such features, one needs to localize appearing individuals. The rest of this chapter explains our implementation of person detection and tracking followed by the details about person representation schemes, using all three modalities (face, clothes, and high-level attributes).

3.1. Person Detection and Tracking

Face is the most intuitive region to detect a person in images/videos and hence, in this thesis, face detectors are used as the core module for person detection. The face detection algorithm used in this thesis has already been explained in Section 2.1. The rest of this section gives the implementation details of face tracking, cloth detection and tracking along with a discussion on how outcomes of both trackers are merged to get better results for person tracking.

3.1.1. Detector Assisted Face Tracking

As discussed in Section 2.2, face detection is a very expensive task including the implementation of multi-view face detection (Section 2.1.3), where there is need to run several face detectors, trained at various yaw-angles, over a whole frame. To reduce the computational complexity and get temporal correspondence between consecutive frames (Figure 2-4), a particle filters (Section 2.2.1) based tracking

algorithm [53] for face tracking is adopted. Its implementation details are given next.

Similar to multi-view face detector, the face tracker used in this work is also invariant to face pose and partial occlusions. We start by running the multi-view face detector over the first frame to get the initial detections. Afterward, a unique tracker is initiated for every detected face to track it in the rest of the frames. The state \mathbf{x}_t of the face consists of x-y coordinates of the bounding box, the scale s that corresponds to the size of face, and an angle α which represents the pose of the detected face. Hence, the transition model for face tracking is represented as follows:

$$\mathbf{x}_t = [x_t \ y_t \ s_t \ \alpha_t]. \quad (3.1)$$

We propagate N particles around every detected face region at frame t , following the same approach as described in Section 2.2.1. An example of propagated particles can be seen in Figure 3-1.



Figure 3-1: Example of the particles drawn around the detected face region in the next frame. (a) Detected face region at frame t . (b) Drawn particles around the face region at frame $t+1$.

For each particle, \mathbf{x}_t^i , the weight ω_t^i is computed by running a face detector, trained at the angle nearest to the particle's pose angle α_t^i , at the region defined the particle coordinates and scale. The confidence value provided by the corresponding detector is directly taken as the weight value. Unlike multi-view face detector, in this case we just run a single face detector as it can fairly be assumed that the pose of the face does not vary drastically and only running the detector with the closest pose angle is sufficient. Moreover, we run detector only at the region defined by particle's location and size. This helps us to avoid an exhaustive search and to limit the use of face detector to a very small portion of the possible sub-regions.

As the particles are propagated by independently drawn Gaussian noise (Section 2.2.1.1), there exist particles with a diverse range of pose angles. By selecting the detector with the closest yaw angle, the particles that best describe the current face pose are assigned the highest weights as it can be seen in Figure 3-1 (b) where the particles in the central region of the face have green weights (bright green cor-

respond to higher weights) and other particles are black (zero or near to zero weights as the green get dims toward black).

The state \mathbf{x}_t of face is estimated by having the linear combination of particle's location and weights assigned to each particle as given in equation (2.21). The particles are resampled once the effective number of particles becomes less than a certain threshold (Section 2.2.1.4) which also helps the tracker to adopt with the variations in face pose. To initiate trackers for new appearances, the multi-view face detector is run after every k^{th} frame. A tracker is terminated when no detection is found during the evaluation of particles' weights such that no face is found in the regions defined by all particles, for more than 5 frames.

Furthermore, care has to be taken to avoid identity switching of overlapping face tracks such that particles from different trackers do not get scored on the same face or get replaced with each other. This is ensured by keeping the trackers away from each other such that if two trackers come in the vicinity very close to each other we terminate one of these trackers and re-initialize it once they again get far enough. For this purpose a very simple technique is adopted such that if the distance between the centers \bar{X}_1 and \bar{X}_2 , of two trackers becomes less than a certain threshold τ , then one of these trackers is terminated:

$$\|\bar{X}_1 - \bar{X}_2\| < \tau. \quad (3.2)$$

In this thesis, the choice of τ is made empirically to get the optimal results.

False positives can also occur due to misclassification by the face detectors which in turn will initiate a face tracker for it. However, for false positives the particles in tracker die out very rapidly and the tracker does not last very long. Hence, to avoid false positives, we eliminate all face tracks with the length less than a threshold ($l = 5$).

An example face track can be seen in Figure 3-2. We can see the variations in appearance due to changes in facial pose, noise due to motion, and illumination changes.



Figure 3-2: Sample face track generated by the detector assisted face tracker.

Now the idea is to represent a person with a whole track rather than just a single face image. Extracting useful features from the complete face track helps us to recognize/cluster same individuals across different pose angles.

In our experiments we have used the number of particles, N_p , equal to 1000 to efficiently capture the motion of face and to have less computational complexity. An increase in the number of particles will of course increase the tracking accuracy. The main bottleneck in the tracker's computational cost is the initialization of new detections (when 5 detectors are run over the whole frame to form multi-view face detector). The gap, k , in running multi-view face detector also decides the trade-off between tracking efficiency and the computational cost. We have chosen $k = 10$ and found that it works well in real life scenarios. Our face tracker runs near real-time with a processing time of almost 15 frames per second, over 2.53 GHz Intel Xeon processor with 92GB of RAM, for a high quality video with frame size 720x576. The implementation of multi-view face detector and also the weight evaluation of all particles is parallelized with Intel's TBB [93], an open source C++ library for multi-threading.

3.1.2. Clothes Detection and Tracking

Clothes Detection

Clothing color information, extracted from the upper-body part of an individual provides very important clues about the appearance of a person, particularly within a single video, or videos captured at the same event. Clothing information has been used successfully to aid face recognition/clustering in many algorithms [1, 6, 29, 35, 48, 55]. As compared to face, body detection is a much difficult task due to the huge amount of appearance variation and also due to non-homogeneous body structure. The simplest way for body detection is to take a bounding box below the face region. Several independent full/upper body detectors have also been proposed in the literature [45, 61, 82, 83]. Due to the very high computational complexity of independent body detectors, the simpler approach is adopted to meet the real-time computational needs. Example body regions can be seen in Figure 3-3.



Figure 3-3: Example of the clothes bounding box detected based on the face detector.

Although, body detection in this case is completely dependent on face detector and we are not able to detect persons whose faces are not visible (as in [29, 48]), we can still use clothing information to merge the broken face tracks, due to the large amount of variations in face pose, and hence can get better tracking results. For this purpose, after the body detection, an independent clothes tracker is also initiated as described next.

Clothes Tracking

For clothes' tracking we follow the same procedure as for face tracking but with a different observation model as described in Section 2.2.1.2. There is no pose information available for the detected clothes region. Hence the transition model in this case is as follows:

$$\mathbf{x}_t = [x_t \ y_t \ w_t \ h_t], \quad (3.3)$$

where x_t and y_t correspond to the x-y coordinates of the clothes bounding box, w_t and h_t are the width and the height of the bounding box respectively.

We propagate N ($N = 1000$) particles around the detected clothes region and compute a 3-dimensional color histogram from all the regions defined by the particles. We use $N_H \times N_S \times N_V$ bins for the histogram which gives us an m -dimensional feature vector for each particle. In this thesis we use $8 \times 8 \times 8$ bins that, in turn, give us 512-bins histogram. As discussed in Section 2.2.1.2, computing a histogram from a rectangular region is not optimum. Hence to deal with partial occlusion and avoid background regions, a weighting technique is utilized.

The weighting algorithm works such that the pixels that are near the center of the clothes region will get higher weight as compared to the farther pixels, exploiting the fact the farther pixels are more likely to get occluded. For this purpose, the weighted kernel profile proposed in [15] is used to estimate the color distribution as follows;

Let z_i be the location (x, y and z coordinates) of a pixel value in clothes region and z_c be the location of the center of this region. The weighting kernel k , to assign smaller weights to the pixels farther from the center, is defined by

$$k(r) = \begin{cases} 1 - r^2 & \text{if } r < 1 \\ 0, & \text{otherwise} \end{cases} \quad (3.4)$$

where r is the normalized distance between the pixel location z_i and the region centre z_c . Then the color distribution $\mathbf{p}_{z_c}^u, u = 1 \dots m$ of clothes the cloth region centered at z_c would be

$$\mathbf{p}_{z_c}^u = C \sum_{i=1}^n k \left(\left\| \frac{z_c - z_i}{h \times w} \right\|^2 \right) \delta[b(z_i) - u]. \quad (3.5)$$

In equation (3.5), $b : \mathbb{R}^3 \rightarrow \{1 \dots, m\}$ is a function that gives the index, for the pixel at location z_i , of the bin in the histogram. The term n gives the total number of pixels in the clothes' region, δ is the Kronecker delta, h and w are the height and width of the region respectively. C is the normalization factor to ensure that distribution is legitimate, i.e., $\sum_{u=1}^n p_{z_c}^u = 1$ and represented as

$$C = \frac{1}{\sum_{i=1}^n k \left(\left\| \frac{z_c - z_i}{h \times w} \right\|^2 \right)}. \quad (3.6)$$

We use the same method to compute the clothing color distribution of the target region (cloth region at previous frame) and candidate regions (regions defined by all particles). Afterward, the Bhattacharya distances (Section 2.2.1.2) between distributions of target and particles are converted to weights and used to estimate the position of clothes as explained in Section 2.2.1.5. Similar to face trackers we also terminate one of the trackers if they come close to each other.

Clothes tracks obtained from every tracker are used in the next section to complement face tracks and to achieve better tracking results.

3.1.3. Merging Face and Clothes Tracks

It is easily understandable that face and clothes track can provide complementary information to each other. Information from one tracker can be leveraged to assist the other, e.g., in the situation where a face tracker gets terminated due to large pose or illumination variation, but the cloth tracker is still able to track the body or vice versa. Exploiting this observation, a track merging strategy is adopted to take advantage of both trackers and in result obtain better tracks that are referred as **person tracks** (face and body combined) in this thesis. Figure 3-4 depicts the aforementioned motivation in pictorial form where a broken face track and a fragmented clothes track are merged to obtain an improved person track.

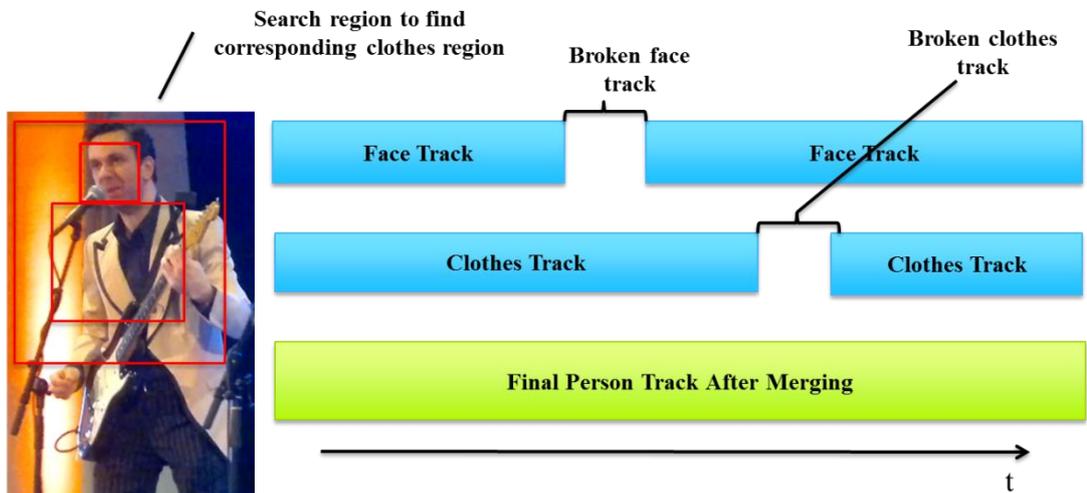


Figure 3-4: Illustration of the face and cloth track merging.

As mentioned earlier (Section 3.1.2), for every detected face, independent face and clothes tracker are introduced. We keep the information of jointly created tracks and utilize this information in the merging strategy to check the status of their pair at any stage. For every face tracker, we check whether its corresponding clothes track is still alive or has been terminated. If the clothes tracker is active, its spatial location is checked whether it is tracking the correct body region or not. For this purpose, we select a fixed bounding box around the face region, as shown in the Figure 3-4, and search for the clothes tracker in this region. If the clothes tracker happens to be within the selected bounding box it means both trackers are working fine and no updating process is needed. However, if the clothes tracker is not present in the selected region, we re-initialize the values of particles, according to the current location of the face. This is done to make the clothes tracker within the correct range. In case the clothes tracker has already been terminated, we simply start a new clothes tracker and assign the same identity as of the previous tracker. this process after every k ($k = 10$) frames and keep making up for misplaced/expired clothes trackers in an online manner.

To merge the face tracks that are torn apart, clothes track information is utilized. If there is a clothes tracker which does not have its active face pair, we search for a new face tracker (with a different identity) by considering a ROI around the current state of clothes (Figure 3-4). If there is a new face tracker present in the region of interest (it means an old face track was terminated and a new tracker is initialized for the same person), we change its identity with the identity of the previous face tracker to merge them together. In case there is no face tracker available in the region of interest, we keep checking for it for k frames ($k = 50$, for a video with 25 fps) unless a face tracker is found. Otherwise, we forcibly terminate the clothes tracker as well. While initiating a new face tracker, it is likely that an older clothes tracker will still be present for the same person. To ensure that two separate clothes tracker are not initialized for the same person, we always check before initiating a clothes tracker if there is already one in the vicinity of face. If this is the case, we simply pair the new face tracker and the older clothes tracker and do not initiate a new tracker for the clothes.

Following this approach we can merge most of the face tracks that were broken due to sudden variations (pose, illumination, motion, etc.) and hence in result obtain the combined person tracks that are used in the later stages for feature extraction. Each element of a person track contains three kinds of features: Low-level facial features, High-level attribute and clothing color features. Details on feature extraction are given in the next section.

3.2. Representation using Facial Features

The face is one of the most crucial biometric information for person recognition/identification. To extract robust features of the face, most of the work has

been done toward extracting low-level features to capture the texture of the face. In this context LBP, the use of local appearance descriptors such LBP [77], HOG [61] and DCT [26], Gabor jets [38], SIFT [16] and SURF [25] have become increasingly common due to their robustness against occlusion, expression variations, pose and scale variations, etc.

In this thesis, we focus on descriptor based LBP (Section 2.1.1) as it is computationally very effective and has been proved to be very useful for face recognition [77]. In crowd sourced videos, face pose is completely unrestricted, and hence the tracked faces can contain all kinds of pose angles. Extracting features directly from these images will cause a huge amount of intra-track variation and can affect the identification process. We have overcome this difficulty by converting all faces to fixed size and aligning them to a near frontal view. The rest of this section explains the face alignment technique chosen in this thesis followed by a brief discussion on feature extraction.

3.2.1. Facial Landmarks Detection and Face Alignment

Before feature extraction, we align all faces such that the eyes and mouth of all faces appear at nearly the same spatial location. For this purpose, the eyes and mouth are detected using a Deformable Part Models based facial landmark detector by [54] and its already available implementation is used. As facial landmark detection is not the topic of this thesis, the in-depth details of the topic are skipped. Readers who are interested to know more may refer to [54].

The facial landmark detector provides spatial coordinates for the seven facial landmarks including lateral and medial canthus of the eyes, nose and both corners of mouth. The centers of both eyes and mouth are then estimated from these coordinates that are used as key points for the face alignment. Alignment is done by warping the image using a 2D transformation matrix. The transformation matrix is obtained using affine transformation to map the detected key-points to a set of fixed key points in the aligned image where the fixed key points in the aligned image are selected heuristically. A pictorial representation of the face alignment process is given in Figure 3-5.

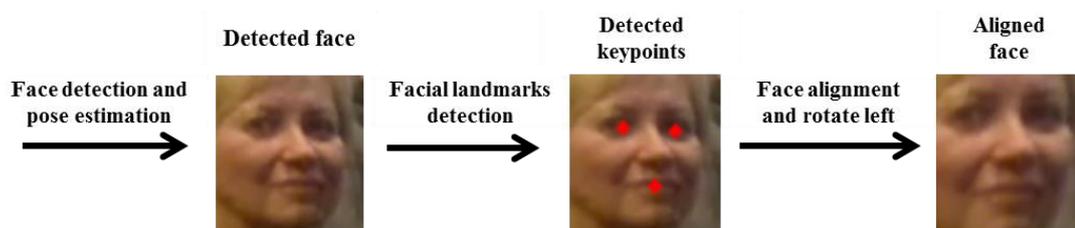


Figure 3-5: Preprocessing steps for each face image.

Under the assumption of facial symmetry, every face is rotated to a negative yaw angle, utilizing the estimated pose information (α in face transition model, Section 3.1.1). This little tweak helps in the identification stage as the features are always extracted from the good side of the face.

3.2.2. Feature Extraction

The aligned face image is converted to grayscale and an ELBP image is extracted. Unlike the basic LBP operator (Section 2.1.1), the ELBP operator utilizes circular neighborhoods and bilinear interpolation to allow any radius and number of pixels in the neighborhood and is often represented as $LBP_{P,R}$, where P denotes a neighborhood of P equally spaced sampling points on a circle defined by the radius R . ELBP gives 2^P unique binary patterns; however, not all of them are important to represent an image. It is found that the uniform patterns (patterns that contain at most two transitions from 0 – 1 or vice versa e.g., 001110000, 11100001, etc) contain 90% of the important information [77]. Hence, only the uniform patterns are utilized and the ELBP histogram is extracted such that all uniform patterns are accumulated into unique bins, whereas all non-uniform patterns are accumulated into a single bin of the histogram. For $P = 8$ and $R = 1$, this results in a 59 bin histogram.

A global histogram for a whole face image can be formed to represent the face. However, computing a global histogram causes the loss of the spatial information. Hence, the ELBP image is divided into 4×4 blocks and an ELBP histogram is extracted from each block. Final feature vector of length 944 ($4 \times 4 \times 59$) is formed by concatenating all block-wise histograms as depicted in Figure 3-6.

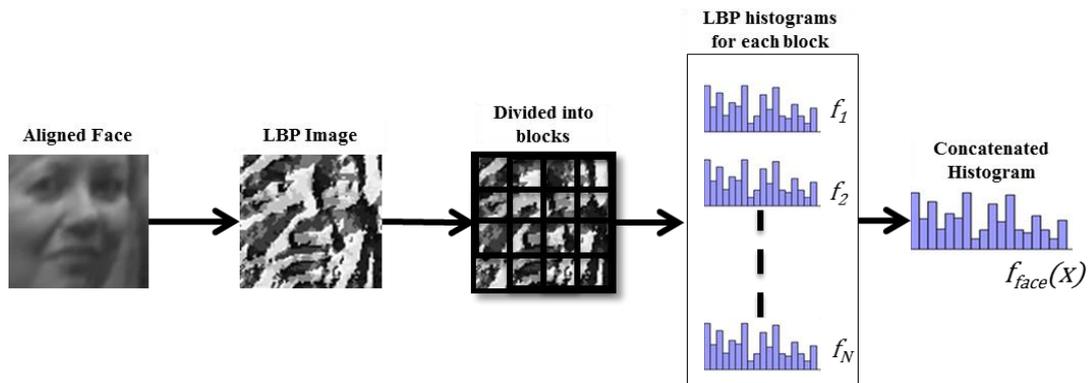


Figure 3-6: Schematic diagram of the block-wise LBP histogram based feature extraction.

Features are extracted from all the face patterns in a face tracks, and feature for a unique face sample will be represented as f_{face} , hereafter. These features are utilized to find similarities between two different face tracks as described in Chapter 4.

3.3. Representation using High-Level Attributes

It is a well-known fact that the images of different identities in the same pose are more similar compared to the images of the same identity in different poses. Although a face track comprises more than one image of a person's face, it is still unsure whether all possible face poses and expressions are available. Moreover, faces in a face track are temporally related and contain less illumination variation as compared to face tracks of the same person but extracted from a different video (particularly for the videos captured by different photographer or from different camera sensors).

As mentioned earlier, occluding region of a person face can also vary with the camera angle and also the distance from the object. In such situations, low-level facial features are not enough to effectively represent an identity, especially the textural features. Hence, a better representation that is robust to these challenges is needed. High level attributes are proven to be robust against such variations [41, 57, 86] and are opted in this thesis. Particularly for person identification in a unique event, these attributes provide very strong clues about the appearance of a person. This can help in differentiating individuals under large pose variations.

3.3.1. Selected Attributes

We have selected 15 most evident attributes that cannot vary considerably in an event. However, more attributes can also be considered for further representation such as in [57], 73 attributes are used for person identification. A list of selected attributes is given in Table 1.

Table 1: Selected high level attributes for the face representation.

Type	Attributes
Race	Asian, White, Black, Indian
Hair related characteristics	Brown Hair, Gray Hair, Black Hair, Blond Hair, Wearing Hat
Age Group	Youth (Baby, Child, Young), Not-Youth (Middle-Aged, Senior)
Gender	Male, Female
Facial Appearance	Beard/Non-Beard
Eyes related characteristics	No Eyewear, Wearing Sunglasses, Wearing Spectacles

Our aim is to label each face, in a face track, with a confidence value about the presence of a particular attribute. For this purpose, binary classifiers for each attribute are trained and the confidence value provided by the classifier is used as a feature. Nevertheless, training such a huge amount of different classifiers is a time consuming task as to manually train each attribute classifier; one needs to select

effective features and good face regions by hand. Different face regions can be advantageous for classification of a specific attribute. Similarly, different pixel value types can play a vital role for efficient classification. Training of tens or hundreds of attribute classifiers requires a huge human effort, which leads to the necessity of an automatic way of training such classifiers. For this reason, a forward feature selection algorithm proposed by N. Kumar et al. [57] is adopted. The complete architecture for automatic training of attribute classifiers is explained in the next section.

3.3.2. Training of Attribute Classifiers

The overall architecture for training an attribute classifier can be seen in Figure 3-7, where optimal face regions and pixel value types are automatically selected and each classifier is automatically trained in a supervised way.

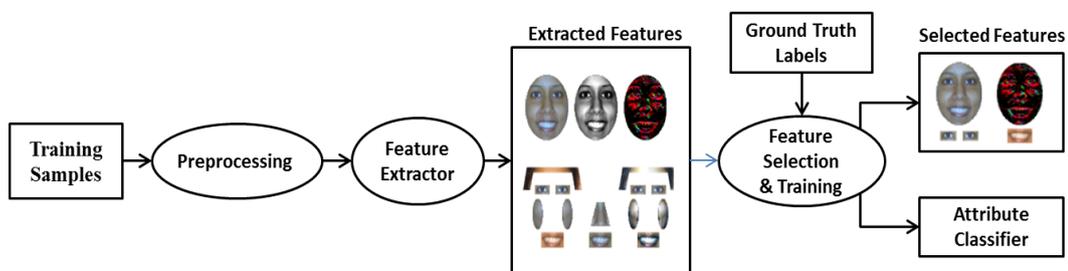


Figure 3-7: The complete architecture for the automatic selection of the optimal feature, and training of an attribute classifier [59].

For any supervised learning method, the first step to be performed is the collection of training samples. A note on training samples used in the thesis is given next.

3.3.2.1 Training Samples

For the purpose of image retrieval based on facial attributes, a database named as “FaceTracer” has been proposed in [60], that consist of 15,000 images downloaded from the internet using a search engine. The database contains professional and amateur photos of various celebrities and a subset of images is labelled for several attributes such as gender (male/female), race (Asian, white, black), age (baby, child, youth, middle aged, senior), hair color (blond/not blond), eyewear (none, eyeglasses, sunglasses) etc. However, the number of labelled samples is quite little and on average contains 200 labelled images per attribute. On the other hand, for our purpose, 500 to 2000 training samples are needed for the training of each attribute classifier. To collect more training data, we have labelled the remaining images in FaceTracer database. In addition to FaceTracer, another public dataset “PubFig” has been proposed in [57] that contains a large collection of images of public celebrities. We have combined images from both datasets and, have collected the required number of training samples for each attribute. Images are labelled by multiple persons and have been cross checked once. Both datasets provide bounding boxes for detected faces in each image and we have directly used this

information instead of running our own face detectors. The numbers of collected training samples for each attribute are given in Table 2.

Table 2: Count of the labeled training samples after combining images from the FaceTracer and PubFig datasets.

Attribute	Labelled Images #	Attribute	Labelled Images #
East Asian	272	Not-Youth	1200
White	2517	Male	3091
Black	938	Female	3172
South Asian	2356	Beard	818
Brown Hair	1459	Non-Beard	1465
Gray Hair	800	No Eyewear	2000
Black Hair	1875	Wearing Sunglasses	690
Blond Hair	1725	Wearing Spectacles	1115
Youth	833	Not Wearing Hat	1342
Wearing Hat	574		

3.3.2.2 Automatic Training

Given the labelled training samples for a particular attribute, our aim is to learn a function $h(x)$ that can map a query face image to a real value a_i . For instance, in case of attribute “Wearing Sunglasses”, we need to learn a function that will map all images with sunglasses to positive values and without sunglasses to negative values. Moreover, we also want this function to measure the degree of confidence of a particular attribute. In the rest of this section, the complete architecture (Figure 3-7) for automatic classifier training is described.

Preprocessing and Feature Extraction

As a preprocessing step, first, all training samples are aligned using affine transform following the same technique as described in Section 3.2.1. However, in this case we select a slightly larger face region to keep the hairs and chin visible. Different feature and face regions can be crucial to efficiently learn a classifier for a particular attribute such as for the attribute “Eyeglasses” regions around the eyes are more crucial than other regions like cheeks, hairs, etc. Similarly for hair color classification, hairs and forehead region are more important than a person’s chin. Therefore, before feature extraction, the face is divided into several functional parts such as the eyes, nose, forehead, etc.

These regions are manually defined in affined aligned coordinates and are fixed for every aligned face. The face regions used in this thesis are shown in Figure 3-8. Regions are kept large enough to contain the corresponding face part under pose variations. Moreover, most of the regions are slightly overlapping (Figure 3-9) that provides robustness to small errors in alignment.

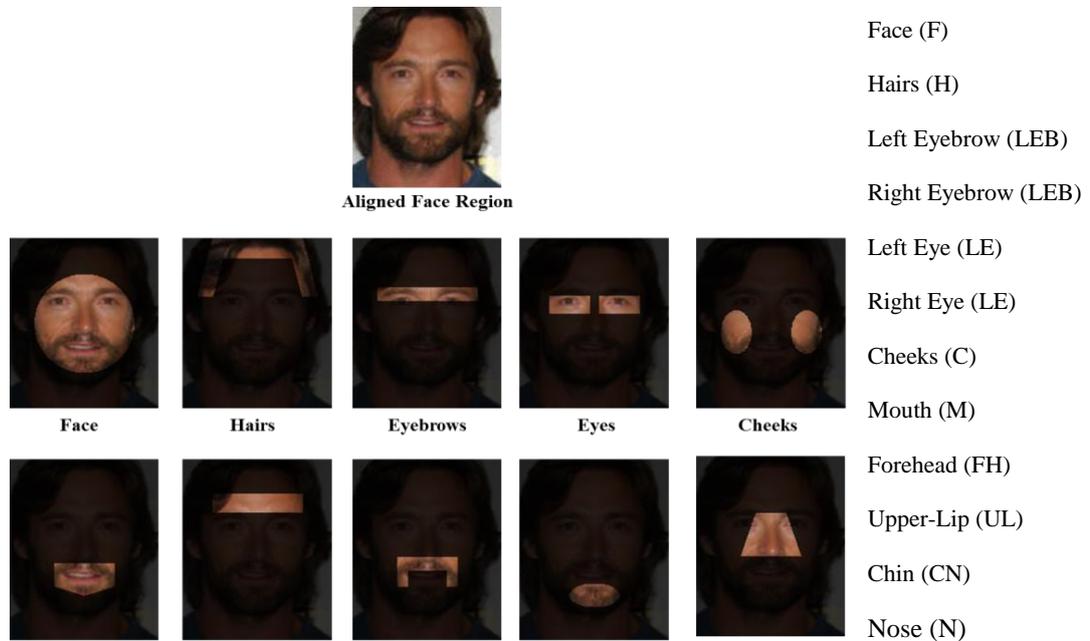


Figure 3-8: Selected face regions to train the attribute classifiers.

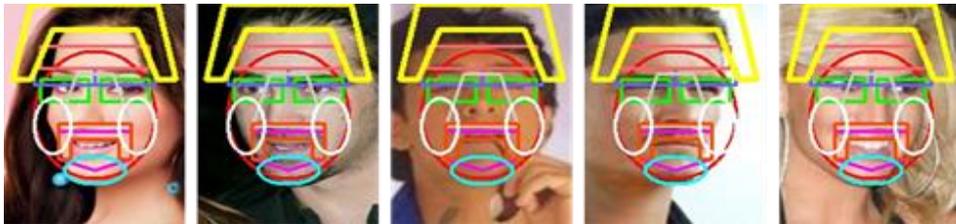


Figure 3-9: Example training images with the face regions drawn with different colors. We can see how the regions are overlapped with each other.

Subsequently, each face region is converted to various pixel types including different color spaces (RGB, HSV, and Image Intensity), edge magnitudes and orientations. The importance of a pixel value type also varies with the attribute e.g., for age and gender classification edge orientation/ magnitude are more significant than pixel intensities. On the other hand for race classification both intensity values and edges are important. Pixel value types used in this work can be seen in Table 3.

Table 3: Feature type options for the training of an attribute classifier.

Pixel Value Types	Normalizations	Aggregation
RGB (R)	None (N)	None (N)
HSV (H)	Mean Normalization (MN)	Histogram (H)
Image Intensity (I)	Energy Normalization (EN)	Mean/ Variance (MV)
Edge Magnitude (EM)		
Edge Orientation (EO)		

Afterwards, the extracted pixel values can be further normalized for better generalization and robustness to illumination variations. We have used the same normalization techniques as used in [57] that are mean normalization and energy normalization. Mean normalization, $\hat{x} = \frac{x}{\mu}$, helps in eliminating illumination gain and the second, $\hat{x} = \frac{x-\mu}{\sigma}$, removes both the gains and offsets. x is the original pixel values, and μ and σ are their mean and variance respectively. These pixel values, normalized/not-normalized, can directly (by concatenating one after another) be used to train classifiers. However, using such high dimensional feature vectors may lead to over-fitting for some cases. Hence, for better generalization, pixel values can be aggregated using various methods such as histogram of pixel values in corresponding region, the mean and variance, etc. A combination of face region from Figure 3-8, feature value type, normalization and aggregation method from Table 3 form a unique feature extractor, f_i , and the complete set of feature extractors, $F(I)$, is represented as follows:

$$F(I) = \{f_1(I), \dots, f_k(I)\}, \forall k = 1, \dots, N.$$

Each feature extractors takes an aligned face image as input and extract a feature vector from a specific region, according to a combination from Table 3. It is to note that all possible combinations are not usable, such as taking the mean and variance as aggregation type, for energy normalized pixel values, will give zero mean and unity variance for every sample, and therefore it is not a discriminating feature at all.

Feature Selection and Attribute Classifiers

To create a classifier for a specific attribute, we apply all feature extractors on an image which in turn gives us N different features. The simplest way to train a classifier is to concatenate all extracted features one after another and let the classifier decide which features are useful and which are not. However, providing a huge number of non-discriminative features will put too great burden on the classifier and will not be able to generalize well. To cope with this problem, N. Kumar in [57] has adopted a forward feature selection algorithm to pick only the useful features and used only the selected feature to train a classifier.

FFS is a greedy algorithm that iteratively selects the optimal features. In the first iteration, classifiers are trained for every feature extractor in the set $F(I)$ and their performance is evaluated using cross validation. The feature extractor with the minimum error rate is selected as a good feature and is added to the output set. In the rest of the iterations, individual classifiers are trained with the features already selected in the output set, concatenated with the feature-region combinations other than those in the output set. The combination that drops the error rate the most is added to the output set. By following this approach we can select any number of features that are good for learning the training set. The whole approach is independent of the attribute type. This can select good features for any high-level attribute, provided that the training samples are efficiently representing both classes. We keep on adding features into the output set until the classification error rate does not drop anymore or the number of selected features exceeds a certain threshold, which in our case is six. To reduce the training time, 70% of the features with high error-rate are dropped in every iteration but at least 10 features are always kept.

Support Vectors Machines (Section 2.5) are used as classifiers and the publicly available implementation of libSVM [11] is used in this thesis. Positive and negative samples for training are gathered from the images summarized in Table 2. To make the extracted features better separable, kernel tricking is used in SVMs. Different kernels including Polynomial and RBF kernels were tried and it was found empirically that RBF kernel works better in most cases. The regularization parameter C and γ of RBF kernel are selected using the grid search method. For the evaluation of the classifiers, 5-fold cross validation accuracies are used as a measure of efficiency. A complete pseudocode for training an attribute classifier is given in Figure 3-10.

The training time for single classifiers depends on the number of training samples and on the type of attribute. In our case it varied from 1 day to a week over a 2.53 GHz Intel Xeon processor with 92GB of RAM and a parallelized implementation using TBB [93].

Feature Selection and Training of Attribute Classifiers

```

 $F(I) = \{f_1(x), \dots, f_k(x)\}, \forall k = 1, \dots, N$ 
1   Output_Set = EMPTY
2    $x_i = \text{Training\_Samples}, i = 1:M$ 
3    $y_i = \text{Labels}, i = 1:M$ 
4   WHILE SIZE_OF(Output_Set) == 6 OR Error rate is not decreasing
5       FOR i = 1:k
6           IF( $f_k \notin \text{Output\_Set}$ )
7               FOR i = 1 : M
8                    $Features_i = f_i(x_i),$ 
9               END FOR
10              Final_Features = CONCATENATE(Output_Set, Features)
11               $[h_k, error_k] = \text{TRAIN\_SVM}(\text{Final\_Features}, y)$ 
12          END IF
13      END FOR
14       $k_{min} = \underset{j}{\text{ARGMIN}}(error)$ 
15      Output_Set.APPEND( $f_{k_{min}}$ )
16       $h_{final} = h_{k_{min}}$ 
17  END WHILE

```

Figure 3-10: Pseudocode for the automatic feature selection and training of the attribute classifiers.

3.3.3. Accuracies of Attribute Classifiers

To measure the efficiency of classifiers, we have used 5-cross folds accuracies that are summarized in Table 4. We can strongly argue that these accuracies reflect the performance of our classifiers in real world scenarios as the images used for training are gathered from real world images [60]. Selected feature-region combinations for each attribute are also listed in the last column of Table 4. From the selected combinations, it can be seen that the chosen combinations are appropriate for the majority of the attributes.

Table 4: 5-fold cross validation accuracies of all the attribute classifiers with selected combinations for feature extractors (in order of their selection).

Attribute	Accuracy (%)	Selected Combinations (Region: Pixel Value: Normalization : Aggregation)
Asian	81.79	F:R:EN:N, REB:R:EN:N, FH:R:N:N, F:H:N:N, LE:I:EN:N, H:I:EN:H
White	86.01	F:R:MN:N, F:R:EN:H, F:R:N:MV
Black	94.22	F:R:N:N, F:EM:MN:N, N:R:N:H, M:R:N:MV, F:I:N:H
Indian	88.34	F:R:N:N, FH:I:MN:N, N:R:MN:N
Wearing Sunglasses	98.05	F:R:N:N, F:EM:MN:N, F:R:EN:N, LE:H:N:H, LE:R:MN:H
Wearing Spectacles	94.84	N:R:EN:N, N:EM:MN:N, LE:EM:MN:N, LE:I:EN:N, LE:R:MN:N
No Eyewear	94.72	N:R:EN:N, N:EM:EN:N, LE:EM:MN:N, LE:R:MN:N
Wearing Hat	97.96	FH:R:N:N, H:I:N:N, FH:R:EN:N, FH:EM:N:N, H:EM:N:H
Black Hair	94.45	H:H:N:N, N:R:N:N, FH:R:N:N, F:R:EN:N, F:R:N:N,
Brown Hair	94.04	H:R:N:H, FH:R:N:MV, H:H:N:MV, H:H:N:H, H:I:MN:H, F:I:MN:N
Blond Hair	94.42	H:R:N:N, F:H:EN:N, RE:R:N:N, N:EM:N:N, UL:H:EN:N, RE:H:N:N
Gray Hair	96.80	F:R:EN:N, H:R:N:N, LE:R:N:N, FH:R:N:N, LEB:R:N:N
Youth / Not-Youth	92.82	F:R:EN:N, F:EM:MN:N, N:EM:N:N, N:I:N:N, LEB:EO:MN:N, F:EM:N:N
Beard / Non-Beard	94.10	F:R:EN:N, UL:R:EN:N, UL:EM:N:N, F:H:N:N, LE:R:MN:N
Gender (Male / Female)	95.84	F:R:N:N, LEB:I:MN:N, REB:I:MN:N, F:R:EN:N, F:EM:N:N

Due to our coarse division of the face region into several sub parts, attribute classifiers are also invariant to partial occlusions, provided that the selected face regions are visible.

3.3.4. Final Features

Finally each face pattern in face track is represented by the confidence values provided by the attribute classifiers. These confidence values are concatenated into a single 15-dimensional feature vector, f_{attr} , as shown in Figure 3-11.

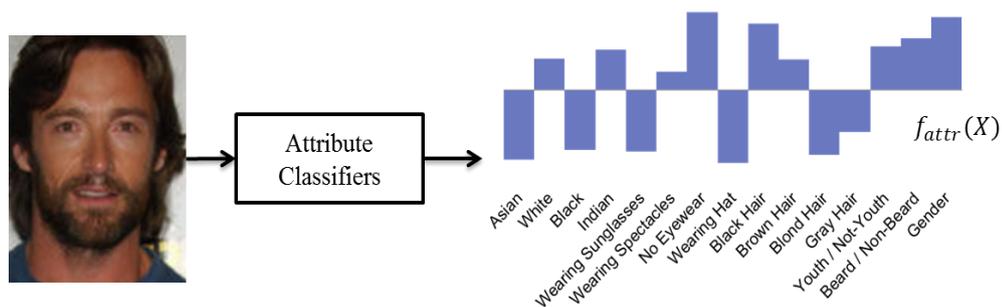


Figure 3-11: Example of the representation using high-level attribute classifiers.

The extracted feature vectors from each face pattern of face track are used in the later stages for person identification.

3.4. Representation using Clothing Colors

For representation using clothes, we simply take the clothing region and extract a weighted YUV color histogram as used for clothes tracking (Section 3.1.2). The feature vector extracted from the clothes will be referred as $f_{clothes}$ from now onward.

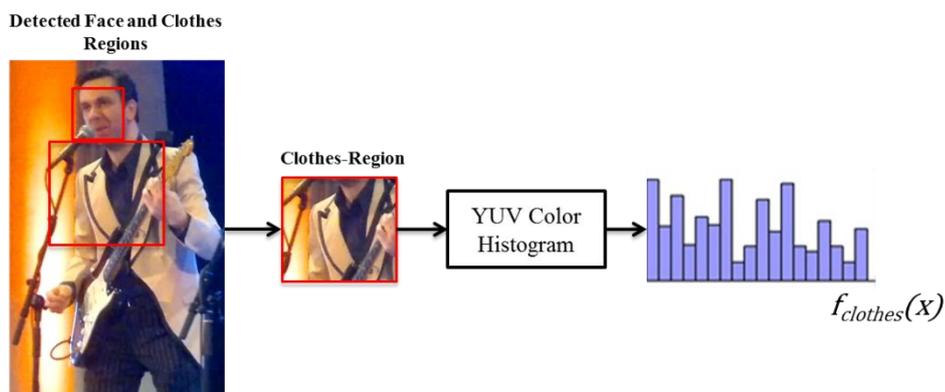


Figure 3-12: Example of the representation using clothing color features.

Until now we have represented every detected person using all three modalities. These modalities are used in the clustering algorithm to form the distance matrix to group some individuals and assign them a unique identity. The complete framework for important person detection is elaborated in the upcoming chapter.

4. IMPORTANT PERSON DETECTION

In the previous sections we have developed the foundations (Chapter 2) of concepts used in this thesis along with the discussion on implementation details of individual modules (Chapter 3). In this chapter the proposed framework for important person detection is formed and a detailed discussion is given on how the aforementioned modules are integrated to form an automatic and efficient framework. Figure 4-1 gives a complete picture of the proposed design that is discussed in the rest of this chapter.

The framework requires videos and, if available, sensor data as input. The sensor data is utilized for temporal segmentation of videos that in turn gives us coherently divided sub segments that are easy to process and manipulate. Afterwards, person tracks, containing face and clothing information of individuals, are obtained by running the person tracker (Section 3.1) over each video segment. Every person track is represented using three modalities (face, high-level attributes and clothing features) as described in Chapter 3. Once the features for all person tracks are obtained, we start by grouping the appearances of every individual within a single video, first intra-segment (within the sub-segments obtained after temporal video segmentation) and then inter-segments or intra-video (within a single video or between sub-segments originated from the same video). Subsequently, global clustering is performed between the available videos of a particular event and unique identities are assigned to every individual. Individuals are ranked according to the number of their presence in the whole event. This finally gives us the information about important persons. The rest of this section explains all the steps in detail.

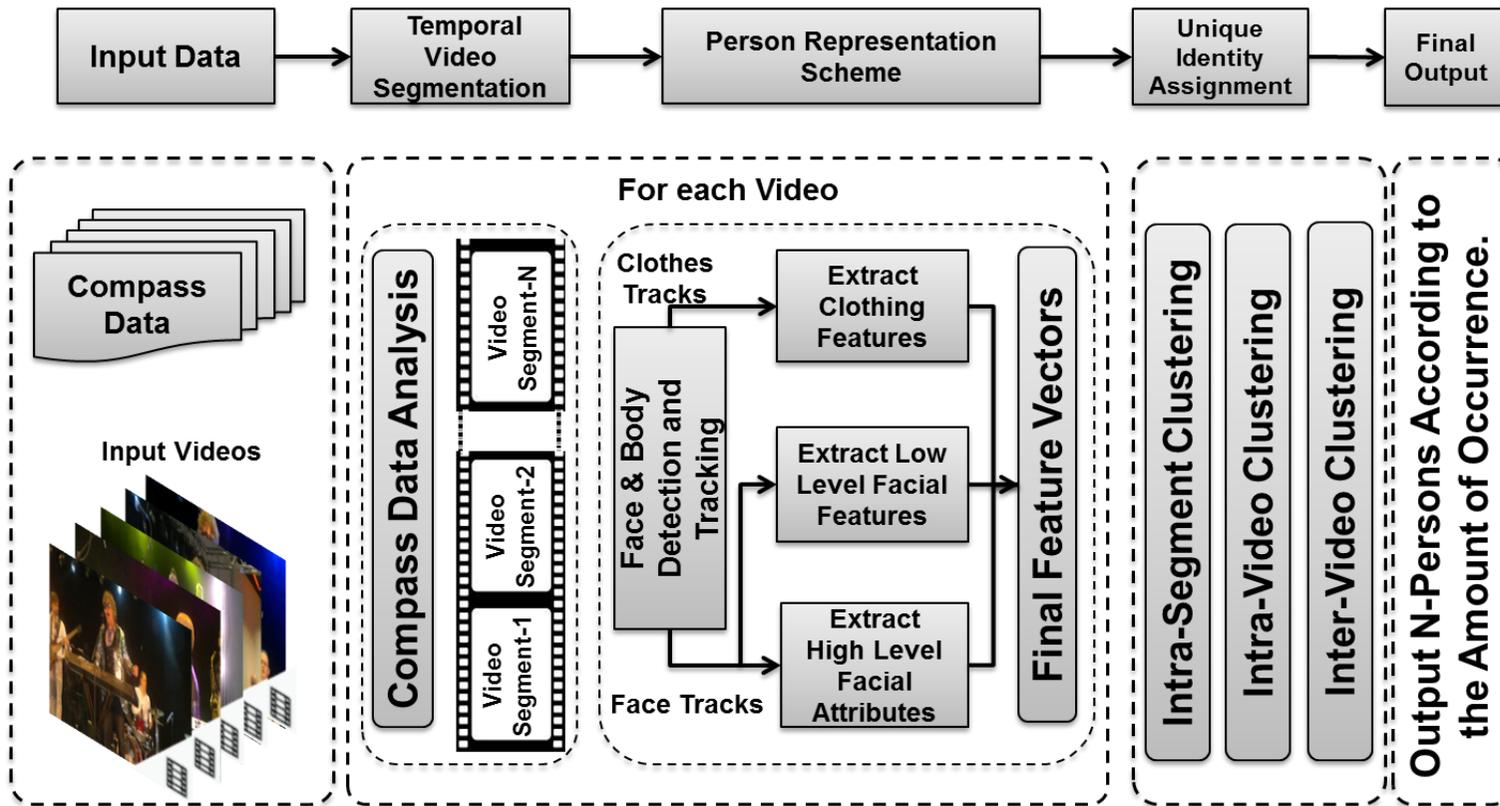


Figure 4-1: The schematic diagram for the important person detection framework.

4.1. Temporal Video Segmentation

The goal of temporal video segmentation is to divide larger videos into more manageable sub-sequences that are easy to manipulate and contain less appearance variations. It also ensures that the clustering within a sub-sequence is accurate and wrong clusters are not propagated into the next clustering stages. Moreover, it also helps to increase the possibilities of parallelism for processing multiple videos at the same time.

Temporal segmentation of videos is considered as a solved problem in computer vision research. Most of the techniques toward this direction look for cuts (abrupt transitions from one video to another), fades and dissolves (gradual transition from one scene to another with a fading effect) [28] and camera motion analysis [22, 62] for finding the appropriate points for segmentation. As the focus of this work is on videos recorded from handheld cameras, the videos are unedited and unstructured. Therefore, the videos are single shot and do not contain transition effects. Consequently, the camera motion analysis is the most intuitive thing we can do to understand camera panning (horizontal movement of the camera) and scene changes. The points where fast camera panning (fast camera panning gives indication of scene change) is detected can be utilized to divide a video. The camera panning detection algorithm used in this thesis is concisely explained next.

4.1.1. Camera Panning Detection

Most of the techniques for the camera motion analysis are based on content based analysis of videos [22, 62]. Hence, they are computationally expensive. For this reason we have adopted the sensor based camera panning detection and classification algorithm proposed by F. Cricri et al. [18]. Their technique utilizes the compass orientations, provided by the built-in electronic compass available in the majority of modern mobiles or digital cameras, to analyze the horizontal movement of hand and detect the camera panning. Later, camera panning is classified into two classes: fast panning and slow panning. The algorithm is computationally very efficient and does not require content analysis. Therefore, it is robust to object movement in the videos. The already available implementation in the lab is used, and is summarized below.

Given the compass orientations (temporally aligned with content of videos) with respect to magnetic North, the following steps are performed to detect and classify camera panning:

- Apply low-pass filter to raw compass data to eliminate peaks due to shaky camera movements.
- Compute the first-order derivative of the filtered compass data to find a measure of how the camera movement is changing.

- Select the peaks from derivative that are greater than a certain threshold. Those peaks are considered as the points where the camera panning occurred.
- Camera panning is classified as slow or fast based on its speed. Gradual change in compass orientation represents slow panning, whereas rapid change corresponds to fast camera panning.

Slow panning represents that the photographer is following any object of interest or trying to cover a panoramic scene, whereas fast panning represents that the photographer intended to change the whole scene. We exploit this observation and divide a video from all the points where fast panning is detected. An example of temporal video segmentation can be seen in Figure 4-2, where the sub-segments extracted from a single video are shown. Readers interested in knowing more about sensor based temporal segmentation may refer to [18].

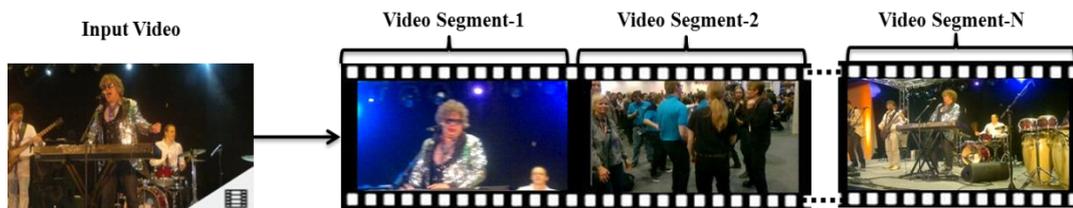


Figure 4-2: Example of the temporal video segmentation using compass data analysis.

4.2. Unique Identity Assignment

After temporal segmentation of videos, for which the sensor data were available, person trackers are run over all video segments and person tracks are formed as explained in Section 3.1. From the person tracks, we extract features for all three modalities face, clothing color, and high-level attributes as described in Chapter 3. These features are jointly used for person track clustering as explained in the next section.

4.2.1. Person Track Clustering

As described earlier, that HAC (Section 2.4) is used to group the person tracks of the same individuals. HAC takes a distance matrix and forms a dendrogram (tree), where a pair of clusters (person tracks in this case) is merged at each level. We begin from the leaf node (each leaf node represents a singleton cluster/person track) and start merging the closest pairs at each level. Finally, the last level (root node) represents a single cluster that contains all the person tracks. The levels of the dendrogram represent the distance between clusters such that the clusters that are near the leaf nodes are more similar.

We perform HAC at three different steps:

- In the first stage, all person tracks within sub-segment, achieved after the temporal video segmentation, are grouped.
- Afterwards, clustering within a video is performed. In case the video was divided into sub-segments, the clusters obtained by applying HAC for all sub-segments are clustered. If the video was not segmented, clustering is done on the person tracks from the complete video.
- In the third stage, clusters originated from all videos are clustered to globally group the same individuals.

The distance d between two clusters, C_i and C_j , is formed as follows:

As a cluster can contain more than one person tracks P_k , we decompose all the person tracks, within a cluster, and aggregate their elements into a single set $S = \{p_1, p_2, \dots, p_n\}$, $n = \sum_k |P_k|$, $\forall P_k \in C_i$, where each element of the set consists of a face, attribute and clothing color feature vector. Once the sets S_i and S_j are formed for each cluster, we find N closest pairs of elements from both sets and form a new set Q of these pairs as,

$$Q = \{\{a_1, b_1\}, \{a_2, b_2\}, \{a_3, b_3\} \dots \{a_N, b_N\}\}, \quad a \in S_i, b \in S_j. \quad (4.1)$$

The mean of the distances between these pairs is taken as the distance between two clusters as given below:

$$d(C_i, C_j) = \frac{1}{N} \sum_{n=1}^N \gamma(a_n, b_n), \quad (4.2)$$

where $N = \min(|S_i|, |S_j|)$ to ensure that the distance is not biased toward the set with higher length. The distance $\gamma(a, b)$ is defined as the weighted sum of distances according to each modality as follows;

$$\gamma(a, b) = d_{face} * w_{face} + d_{attr} * w_{attr} + d_{clothes} * w_{cloth}, \quad (4.3)$$

where d_{face} and d_{attr} are the distances between facial and high-level attributes respectively and are taken as the Euclidean distance between their feature vectors. $d_{clothes}$ is the Bhattacharya distance between clothes patterns of both elements a_n and b_n . The weights w_{face} , w_{attr} and $w_{clothes}$ are the weights assigned to each modality. The discussion on weight selection is given in the next chapter. To ensure that no modality takes higher weight than the one assigned to it, all feature vectors are normalized such that their distances range between 0 and 1.

Once the symmetric distance matrix, $D(i, j) = d(C_i, C_j)$, from the distances between all clusters is computed, we can perform HAC with any Linkage Criterion (Section 2.4) to develop a dendrogram. However, by utilizing some common sense

constraints, there are possibilities to further enhance the quality of clustering as described below.

4.2.1.1 Incorporation of Uniqueness Constraints

Although no prior information about a person’s identity is available, we can still extract some useful information such that the persons who are appearing in the overlapping video frames should not clustered together as shown in Figure 4-3. This extra information can be extracted directly from the frame numbers of person tracks, and the distance matrix for the HAC is slightly changed to enforce these constraints (such constraints are usually called as Cannot-Link Constraints in literature).



Figure 4-3: Example of a video frame where uniqueness constraint holds.

The distance matrix can be seen as a fully connected graph where each node C_i (Singleton cluster / person track) is connected to others C_j with a weight $d(C_i, C_j)$. Our aim is to update the graph such that the distance between all the persons that appear in overlapping time instance is increased to infinity. For this goal we create a matrix $CC_{constraints}$ as

$$CC_{constraints}(i, j) = \begin{cases} 1, & \text{if } C_i \text{ and } C_j \text{ contain} \\ & \text{cooccurring persons,} \\ 0, & \text{otherwise} \end{cases} \quad (4.4)$$

and obtain a new distance matrix D' as follows;

$$D'(i, j) = \begin{cases} \max(D) + 1, & \text{if } CC_{constraints}(i, j) \text{ is } 1 \\ D(i, j) & \text{if } CC_{constraints}(i, j) \text{ is } 0 \end{cases} \quad (4.5)$$

In D' the distance between all overlapping person-tracks is increased. However, note that updating the distance matrix in this way results in the loss of its metricity. Moreover, the triangular inequality is also violated. Consider three clusters C_1, C_2 and C_3 that form a triangle in the graph as shown in Figure 4-4. Let the cannot-link constraint for C_1 and C_2 be 1 and the distance between them be increased

using the aforementioned technique. Then, the triangular inequality $d(C_1, C_2) \leq d(C_1, C_3) + d(C_2, C_3)$ is not valid, and therefore chances are there that C_1 and C_2 will get connected due to C_3 .

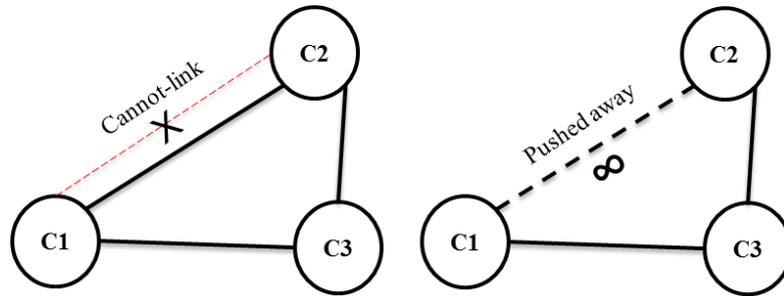


Figure 4-4: Illustration of endorsing a uniqueness constraint.

Therefore, we perform HAC with Complete Linkage (Section 2.4) to ensure that C_1 and C_2 will not get merged. Complete linkage always considers the maximum distance between the elements of two clusters, and therefore, even if C_1 and C_3 are clustered together, C_2 cannot get connected with them due to maximized distance between C_1 and C_2 .

Complete linkage also gives more compact cluster that decreases the chances that a cluster will contain person tracks representing different individuals. Finally, the resulting dendrogram is more optimized and does not cluster co-occurring persons. An example of enforcing the cannot-link constraint can be seen in Figure 4-5.

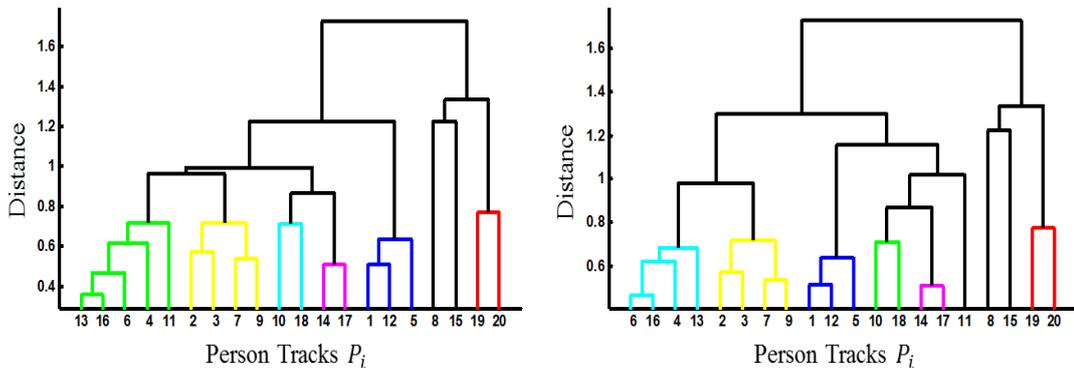


Figure 4-5: Example of the dendrograms before and after the enforcement of the uniqueness constraints (CC-constraint is used between P_{11} and P_{13}) with cutoff = 0.8. (a) Before. (b) After.

We use uniqueness constraints in all three stages of HAC. First they are used for all the person tracks which appear in the overlapping frames in a sub-segment. After the first stage of clustering we use cannot-link constraints for all the clusters originating from the same subsegment. It ensures that clusters from first clustering level are not getting merged in the later stages. Subsequently, these constraints are also used at the last clustering stage such that the clusters originating from the same videos are not merged again.

4.2.1.2 Cutoff Selection

Once the dendrogram is formed, we need to select a cutoff level where the optimal clustering is achieved. The cutoff level is the only parameter that HAC requires as input. Selecting a very small cutoff level will result in a large amount of redundant clusters such that a person will be represented in multiple clusters. It can affect the performance of important person detection if none of these clusters contains person tracks from more than one video. On the other hand, selecting a large cutoff will merge different identities into the same clusters. It can increase the false acceptance rate for important persons. Of course we want to select the optimal level where the clusters are homogeneous with respect to identity and are less redundant. For this reason, we select the cutoff level at which the ratio between intra-cluster distances and inter-cluster distances is minimized. This ensures that the person tracks within a cluster are closer to each other and are very far from person-tracks in other clusters.

To compute the intra-cluster distance for cluster C_i we form the set $S_i = \{p_1, p_2, \dots, p_n\}$, as done before. The intra-cluster distance is then defined as

$$dist_{intra}(C_i) = \frac{1}{\sum_{i=1}^n i} \sum_{i=1}^n \sum_{j=i+1}^n \gamma(p_i, p_j). \quad (4.6)$$

Inter-cluster distance, $dist_{inter}$, is calculated by taking the mean of pairwise distances between all clusters at that level. The distance between a pair of clusters C_i and C_j is computed in the same way as done before in equation (4.2). Finally we select the cutoff level that minimizes the following cost function:

$$f = \beta \frac{\sum_{i=1}^c dist_{intra}(C_i)}{dist_{inter}} + (1 - \beta) * c, \quad (4.7)$$

where c is the total number of clusters formed at a given level and the constant β defines the trade-off between clustering accuracy and redundancy. The larger value of β will result in very compact and accurate clusters but with larger redundancy. On the other hand, smaller β will contain less redundant clusters but with less homogeneity. As we have used uniqueness constraints at all stages of clustering, the value of β should be selected carefully such that no errors are propagated to the next stages of clustering. For example, if two person tracks belonging to the same person, within a subsegment, are not clustered during the first stage, then they will not be merged in later stages due to the uniqueness constraint (remember that we use the uniqueness constraints for all the clusters originating from the same video/sub-segment). We empirically selected β to give the best results, as explained below.

Our first and second stage of clustering leverages more benefits from usage of multiple modalities as the clothing color and attribute features would be

homogenous. Therefore, we can perform clustering with more confidence in these stages. Hence, for the first stage and second stage, we selected a higher value of β and a slightly lesser for the third stage.

4.3. Global Ranking of Individuals and Important Person Detection

Once all clustering stages are complete, each of the final clusters represents a unique individual. To detect important persons, we rank every individual based on the amount of their occurrence. There are several possibilities for this such as:

- Rank every individual according to the number of unique videos from which the person tracks in a cluster originate.
- Rank according to the count of sub-segments, obtained after temporal video segmentation, from which the person tracks in a cluster originate.
- To make the ranking dependent on the amount of time, we can also scale the ranking obtained from above two criteria with the lengths of all person tracks within the corresponding cluster.

In our implementation, the second principle is adopted based on the assumption that if a person is appearing multiple times in a video at different time instances, he/she is more important than a person who just appeared once. The third method will make the ranking strategy dependent on the person tracker such that the person who appears in a single video (but is tracked for a long time) will get more preference than a person who is captured by two photographers but not tracked for very long by the person tracker.

Finally, persons represented by the first N clusters are taken as important persons and all others are classified as casual persons. The experimental results of the complete framework are discussed in detail in the next chapter.

5. EXPERIMENTAL SETUP AND RESULTS

In this chapter, we evaluate the performance of our framework for important person detection using two video datasets – a single-event and a multi-event dataset. In the rest of this chapter the details of both datasets are given followed by a detailed discussion on experimental results. A brief note on the ceiling analysis for computational complexity of the proposed framework is also provided.

5.1. Dataset Details

As discussed in the first chapter, there has not been much work done in the direction of facial appearance based person re-identification in a unique crowded public event. Therefore, we have collected the single-event dataset where multiple photographers recorded the same scene using different cameras and view angles. Moreover, the proposed algorithm is also assessed on a novel multi-event dataset, named as ND-QO-Flip, proposed by J. Barr et al. in [32]. Both datasets exhibit unique properties and complexities as described next.

5.1.1. Single-Event Dataset

This dataset corresponds to the actual goal of the thesis; to detect the most occurring persons in a public event. We have recorded five unique events where the number of videos in a single event varies from three to seven. Videos are recorded by different photographers from different scales, view angles and using different mobile cameras (e.g., Nokia Lumia 900, Nokia Lumia 800, Nokia Pureview 808, Nokia N8, etc.). The length of the videos varies from 1 minute up to 3 minutes. Similarly, the number of persons appearing in these videos also differs with the event.

Among the five events, one is recorded in a real indoor public concert. However, all others are simulated by us. Videos in the public concert are completely unconstrained and are captured from different viewpoints around the stage. In this event, the five band members appear in almost all videos, whereas the audience varies from video to video. Hence, our aim is to detect band members as important persons. In simulated events, three are recorded outdoor in different weather conditions (i.e., sunny, overcast, etc.) and one is recorded indoor. Few (one to three) of the persons in these videos are considered as important and videos are recorded such that important persons appear in more videos than others. The appearance of the persons varies across the videos as all photographers were situated at different

locations and view angles. Furthermore, all crowd members were allowed to make any kind of facial expressions and vary their facial and body pose as they want. Photographers were also allowed to move, zoom & pan the camera but under the constraint that the important persons appear in more videos. The resolution of the videos is either Full-HD 1920×1080 or HD 1080×720 .

The experimental results on the real concert will provide us the idea about the results of the proposed framework in real world scenarios. On the other hand, results on remaining events validate the proposed algorithm in relatively simple situations. Example of videos from single event dataset can be seen in Figure 5-1 where we can see the variations in appearance of crowd members such as face and body pose, expressions and illumination conditions. From now on, the row number for each event in Figure 5-1 will be considered as event-id.

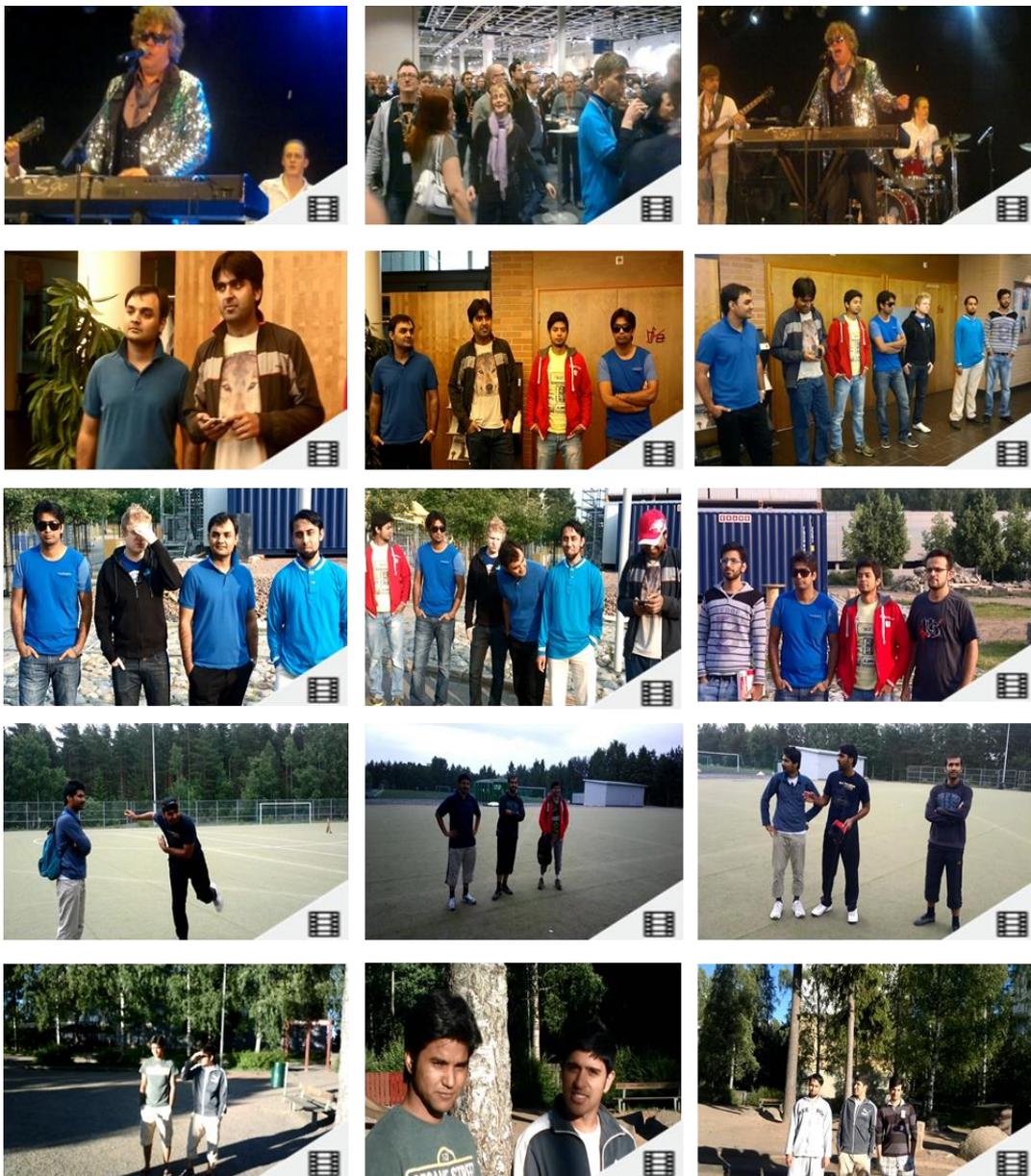


Figure 5-1: Examples of the video frames from the single-event dataset (each row representing a unique event).

5.1.2. Multi-Event Dataset

A multi-event dataset is used to evaluate the proposed algorithm in cases when the videos are not recorded in the same event. In this case, the situation normally becomes even worse as the clothing color, and facial appearance (i.e. beard, glasses, etc.) can vary across the videos. The dataset consists of fourteen 25-59 second video clips recorded by Cisco Flip handheld camcorder at different locations of a University of Notre Dame Campus. Videos were taken in a period of seven months under different environments and weather conditions that in turn enriches the appearance variations. 12 out of 14 videos were captured outdoor including 6 recorded on sunny weather, three in snow cover and one during the snowfall. The other two videos were recorded indoor.

Unlike our dataset, creators of the ND-QO-Flip dataset have restricted the facial pose to near frontal, which reduces the complexity of the dataset. However, crowd members were allowed to vary the facial expression. The dataset contains 90 subjects overall, five of them appeared in more than one video and all others appeared in a single video. Hence, for this dataset, the goal is to detect five persons who appear in multiple videos.



Figure 5-2: Examples for the video frames from ND-QO-Flip dataset.

5.2. Weights for Each Modality

The weights, w_{face} , w_{attr} and $w_{clothes}$, for each modality in equation (4.3) should be assigned differently for each dataset due to their different nature and complexities. For the single-event dataset fixed weights are assigned at all stages of identity clustering (Section 4.2). It is due to our assumption of same clothing of individuals across different videos. We have empirically selected w_{face} , w_{attr} and $w_{clothes}$ to get the best performance. The weight selection is based on intuition that biometrics such as low-level facial features are the most discriminating features among all

modalities. Therefore the higher value is taken for w_{face} . On the other hand high-level attributes and clothing colors can be similar for two different individuals. Hence, we assign a relatively less weight to high-level attributes and minimum weight to clothing features as clothing colors are more likely to be similar than high-level attributes. By closely examining the example images of the single - event dataset in Figure 5-1 we can ensure that these intuitions generalize well for real life scenarios.

For multi-event datasets, the weights are assigned differently for each stage. It is due to the fact that the videos in this dataset are not captured at the same event and clothing color information cannot be used to perform clustering between different videos. Therefore, for the first and second stage HAC we use the same weights as for the single event dataset. However, for the last stage, we eliminate clothing color information and increase w_{face} and w_{attr} . For comparison of different combinations of modalities, we have also used the combination of face and clothing color information. In this case we again assign higher weight to face and relatively lower to clothing colors.

5.3. Results

This section discusses the experimental results on both datasets. To check that the usage of multiple modalities and uniqueness constraints really aid person-track clustering, we start with a baseline method. The baseline method utilizes only the facial features for person representation. Afterwards, new features are added to the baseline one by one until we reached to the proposed framework that utilizes all three modalities and uniqueness constraints. Furthermore, the performance results of the person tracker used in this work are also given in this section.

5.3.1. Results on Single-Event Dataset

For the single event dataset, we have evaluated experimental results on each event separately. Moreover, averages of all quality measures are also calculated to have a holistic picture of the performance of the framework. The results of person detector for all events are summarized in Table 5.

Table 5: Performance of the face detector for the multi-event dataset.

Event Id #	Number of person tracks	True Positives (T_P)	False Alarms (F_P)	Precision
1	102	84	18	0.823
2	108	101	7	0.935
3	126	122	4	0.968
4	104	102	2	0.980
5	66	45	19	0.712
Average	101	91	10	0.883

Table 5 provides precision of face detection over videos of all events. We can see that the average precision is 88% that shows very good detection results such that 88% of the detected persons are true 12% are false detections. However, precision alone does not represent the actual performance of detection and recall is needed to check how many persons are not detected. As we don't have ground truths for person tracks, we are unable to quantitatively calculate the recall. Nonetheless, by visually inspecting the detection results we found that recall is also sufficiently good in most of the videos. However, the face detector missed several faces in event-1 and event-5. It is often due to full profile faces, very harsh lightening conditions, severe occlusion, etc.

For assessment of person-track clustering and important person detection, the quality measures discussed in the Section 2.6 are used. The examples of person detection can be seen in Figure 5-3 where the number at the top of each face region indicates the unique identity assigned to every person after intra-video clustering. Frame numbers can be seen at the upper-left corner of each frame. It can be seen how the same identities are assigned to disjoint person-tracks of the same person.



Figure 5-3: Example frames from the videos of two different events. The number assigned to each face is the unique identity assigned to every person after within-video clustering. (a) Example frames from event-1. (b) Example frames from event-3.

Table 6 presents the average results over complete single-event dataset. Results are evaluated for all methods used in this thesis, using the quality metrics explained in Section 2.6.

Table 6: Experimental results on the single-event dataset, averaged over all events.

Method	SOR	CON	FPR	FNR
Only Face Features (Baseline)	0.663	0.551	0.420	0.320
Face + Clothes	0.702	0.600	0.480	0.291
Face + Attributes	0.666	0.575	0.406	0.320
Face + Clothes + Attributes	0.708	0.599	0.386	0.280
Face + Clothes +Attributes + Uniqueness constraint (Proposed)	0.738	0.651	0.286	0.180

We can see how the final results depict the effectiveness of all intuitions discussed earlier. The increase in SOR and CON with additions of new modalities into the baseline is clearly evident. The success of using clothing color can also be seen by the increase in SOR and CON and also decrease in the FNR for important person detection, from 0.32 to 0.29. However, use of clothing color slightly increases the FPR. It is likely due to merging of persons with similar clothing color. Despite the usage of high-level attributes with facial features, overall results do not show a significant increase in performance. However, the capability of high-level attributes can be seen on the results of individual events as shown in Figure 5-4 and in Figure 5-5. Figure 5-4 shows the comparison between SOR obtained from all methods on individual events, while Figure 5-5 gives a comparison of CON obtained from all the methods.

The usage of high-level attributes increases the performance of clustering for most of the events. Similarly, a combination of all three modalities also enhances the clustering performance and also the detection of important person as compared to other methods. It increases the SOR from 0.66 (baseline) to 0.73. Also the FPR and FNR are decreased to 0.28 and 0.180 as compared to 0.42 and 0.32, achieved using the baseline method. The usage of the uniqueness constraint also shows promising results and appears to be very effective in all events except event-4 where it slightly decreases the CON. For all other quality measures it exhibits convincing increment for all events.

All these results clearly demonstrate the potency of using multiple modalities and common sense constraints in unconstrained environments where there is no restriction on human pose, lightening conditions, movements, etc.

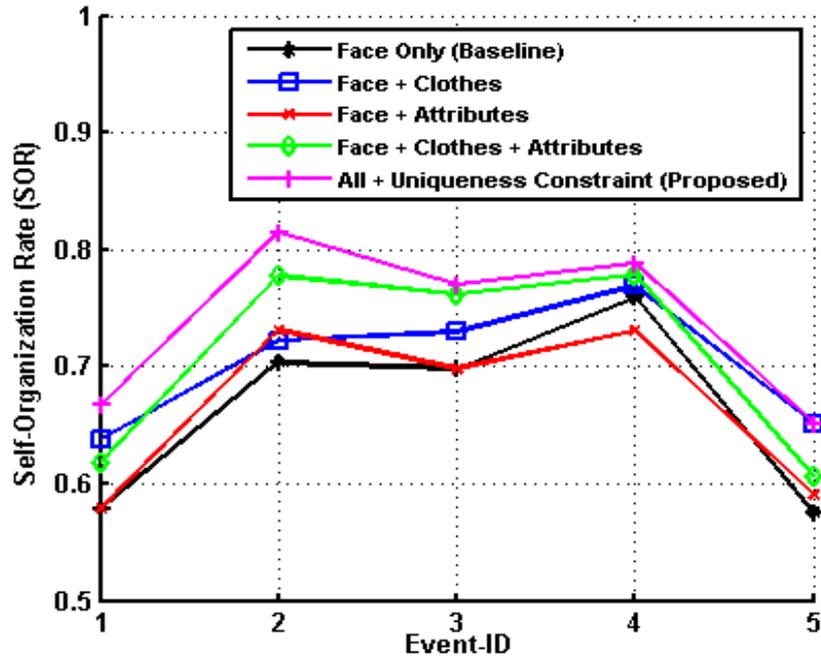


Figure 5-4: Comparison of the SOR values obtained using different methods for the individual events.

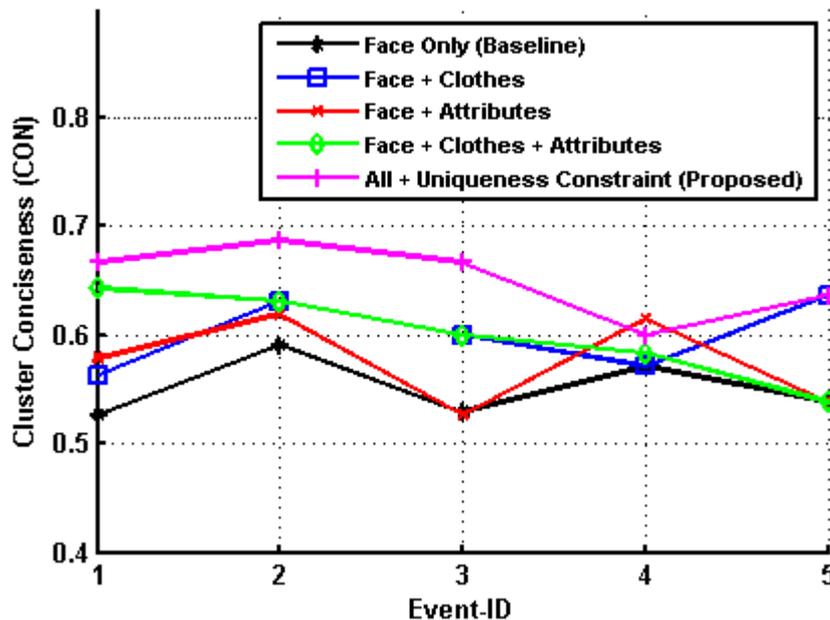


Figure 5-5: Comparison of the CON obtained using different methods for the individual events.

5.3.2. Results on Multi-Event Dataset

For the multi-event data set, the faces are detected with the precision of 0.90 and, due to non-availability of labelled person tracks, the recall is not calculated. However, we again visually inspected the recall and it appears to be better than single-event dataset. This is due to the restriction of near frontal face pose. By manually examining all videos it is found that all persons, except two, were detected at least

once by the face detector. The person detection results are given in Table 7 and examples of face detection can be seen in Figure 5-6. Results of unique identity assignment after intra-video clustering (stage one and two of HAC) can also be seen in the same figure.

Table 7: Performance of the face detector over the multi-event dataset.

Number of person tracks	True Positives (T_p)	False Alarms (F_p)	Precision
292	264	28	0.904



Figure 5-6: Example frames from two different videos of the multi-event dataset. The number assigned to each face is the unique identity assigned to every person after within-video clustering.

Table 8 summarizes the average results obtained over the multi-event dataset using all methods. We can see that usage of multiple modalities again works well and improves the performance of person-track clustering and important person detection. A clear increment of all quality measures can be seen from the baseline to the proposed algorithm. We are able to detect all important persons of the multi-event dataset with SOR, CON and FPR equal to 0.95, 0.69 and 0.035 respectively. These results demonstrate the capacity of the proposed algorithm in videos captured in different illumination, weather conditions, occlusions and severe appearance variations. Moreover, in this case no clothing color information is used for inter-videos clustering.

Table 8: Experimental results for the multi-event dataset.

Method	SOR	CON	FPR	FNR
Only Face Features (Baseline)	0.889	0.689	0.047	0.400
Face + Clothes	0.916	0.725	0.059	0.600
Face + Attributes	0.912	0.706	0.047	0.400
Face + Clothes + Attributes	0.938	0.733	0.035	0.200
Face + Clothes +Attributes + Uniqueness constraint (Proposed)	0.958	0.697	0.035	0.00

These results also show that the proposed algorithm is comparable to the state-of-the-art algorithms. Table 9 gives a comparison between the results obtained by our framework and the one stated by J. Barr et al. in [32]. We can see that our approach achieved almost equal SOR and higher value of CON. Moreover, FPR is also lower than achieved by their method. However, it is to note that this comparison is not completely legitimate due to the differences in face-tracks caused by different face detection and tracking algorithms.

Table 9: Comparison of the proposed algorithm with the state of the art.

Method	SOR	CON	FPR	FNR
Proposed	0.958	0.697	0.035	0.00
J. Barr et al [32]	0.960	0.664	0.056	0.00

Despite the results obtained for multi-event dataset are relatively better than the one obtained on single-event dataset, we should remember that the facial pose in this case is restricted to near frontal. This shows the increase in complexity of the problem due to pose variation and differences in view angles. It also validates the well known observation that the images of two different persons in the same face pose are more similar than the images of same persons but in different poses.

Therefore, the importance of face pose variation should not be neglected in such systems as it is obvious in real life scenarios.

5.4. Analysis of Computational Complexity

Multimedia applications such as the one posed in this thesis are computationally very expensive and require sophisticated hardwares. Therefore, it is really important to have a ceiling analysis of the framework to know the computation power required by different modules. This can help in analyzing its applicability in real time systems and potential of enhancements in complexity. For this purpose, the proposed framework is divided into four abstract modules and analyzed the average time taken by each one of them. A simplified version of the proposed framework can be seen in the flow chart given in Figure 5-7. The first module includes the complete implementation of the person tracker. The second represents feature extraction using all three modalities (face, clothes, high-level attributes). Person track clustering comes into the third module and finally all post processing steps (i.e., ranking of individuals, detection of important person, etc.) are clutched into the fourth.

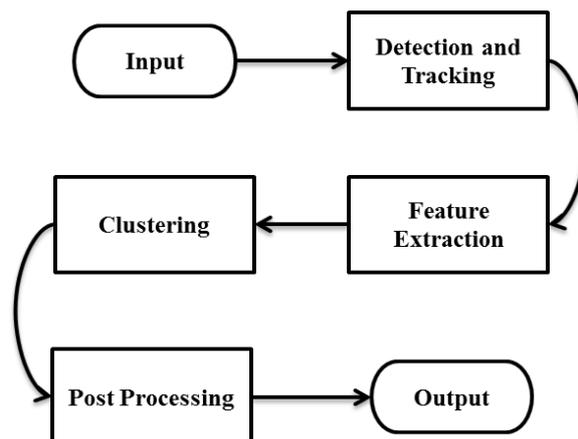


Figure 5-7 Flow diagram of the important person detection framework with different abstraction levels.

The average times taken by each of these modules while evaluating the experimental results of the proposed framework are given in Figure 5-8. From the figure

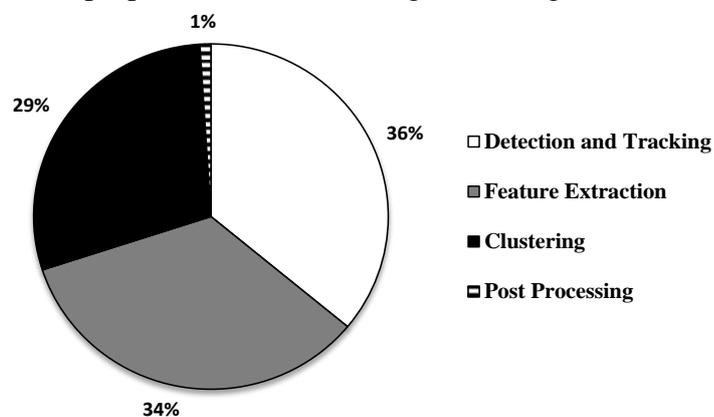


Figure 5-8 A pie chart illustrating the average processing time taken by different modules of the proposed framework.

it can be seen that the detection and tracking modules takes the highest (36% of the total) amount of time. The main computational cost of this module is due to the multi-view face detector. As face tracking is also dependent on the face detector, the performance of this module can be improved by having a computationally effective face detector. Furthermore, the number of particles used for face and clothes trackers also crucial for computational complexity.

Feature extraction also takes a significant amount of time with 34% of the overall computation cost. For a better understanding of the time taken by this module, the average times taken by feature extractors for different modalities are also calculated as illustrated in a pie chart given in Figure 5-9. The longest time is taken by high-level attributes with 54% of the overall time followed by low-level facial features with 45% of the total time. The time consumed for clothing color features is almost negligible. Note that both face related features consume notably more time as compared to clothing color features. This is mainly due to preprocessing (i.e., facial landmark detection, face alignment, etc.) steps to align all faces to the canonical pose. Furthermore, in case of high-level attributes, several different combinations of features are extracted and 15 SVMs are run in parallel to get the confidence scores. Therefore, it takes much more time as compared to others (calculation of confidence scores for each attribute is independent from each other and TBB [93] is used to evaluate them in parallel).

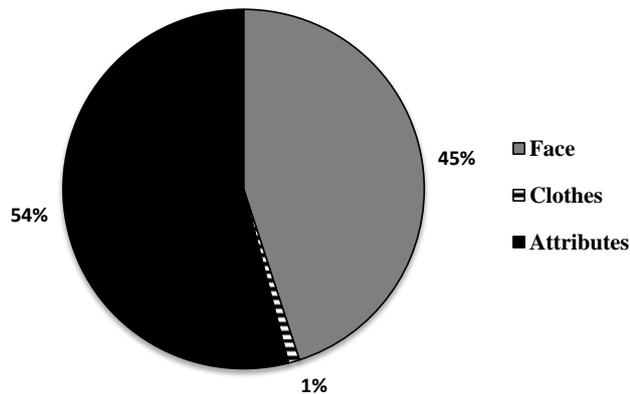


Figure 5-9 A pie chart representing the average processing time taken by different feature extractors.

Person track clustering takes 29% of the overall time. Remember that the clustering is performed in three stages. Our implementation of Hierarchical Agglomerative is an $O(N^2 \log N)$ task. Moreover, to find the cutoff level of the dendrogram, calculation of the inter-cluster and intra-cluster distances is needed. Both of these tasks are $O(N^2)$ with respect to the number of person tracks and number of elements in a person track respectively. This means, the complexity of this module will exponentially increase with the number of face tracks. The complexity of this module can be decreased with a better implementation of HAC (there are some implementations with $O(N^2)$ complexity) and finding another effective solution for the cutoff level.

All other post processing tasks take very minor amount of time as shown in Figure 5-9. In the current implementation of the framework, HAC and the selection of the cutoff level have quite less possibilities for optimization due to their nature. However,

there is a plenty of room for enhancements in the detection and tracking module and, also for the feature extraction.

6. CONCLUSION AND FUTURE WORKS

In this chapter, this thesis is concluded with brief concluding remarks. A short discussion on possibilities for further improvements and some new ideas are also given. Some reference papers are also provided for the readers who wish to carry on this work.

6.1. Conclusion

In this thesis the problem of unsupervised person re-identification, with application to important person detection was addressed. A standalone framework was proposed that utilizes several visual modalities and contextual constraints to group the occurrences of every individual across different videos. Low-level facial features and high-level attributes were utilized for face identification, while clothes identification was performed using clothing color information extracted from the upper-body region of the person. All modalities were combined in a sophisticated manner for efficient identification process. Sensor data was also used as a preprocessing step to divide larger videos into more coherent sub-segments based on camera movements. The final grouping was performed using Hierarchical Agglomerative Clustering. The HAC algorithm was smartly altered to incorporate the uniqueness constraints to further enhance the grouping task.

Experimental results on two challenging datasets illustrate the effectiveness of usage of multiple modalities. The usage of clothing colors demonstrates encouraging results by increasing the Self-Organization Rate of clustering from 0.66 to 0.70 in the single-event dataset and from 0.88 to 0.91 in case of multi-event dataset. High-level attributes also proved to be beneficial and helped in increasing the efficiency of clustering and also for important person detection. Similarly, the combination of all modalities (face, high-level attributes and clothing colors) showed promising results. Further enhancement has been achieved by enforcing the uniqueness constraints into the clustering algorithm. The final approach that utilizes all modalities and uniqueness constraints exhibits a sufficient increase in SOR from 0.66 and 0.88 (baseline) to 0.73 and 0.95 for both single and multi-event datasets respectively. A prominent increase in efficiency of important person detection has also been shown for both datasets.

Finally, experiments on two different datasets validate the system on various challenging situations, emphasize on the importance of face pose variations in real life scenarios and encourage us to strive for better face representation techniques.

6.2. Possible Future Directions

Lastly, this thesis is just an initial effort toward this great field of multimedia analysis for unsupervised person identification in public events and there is still a penalty of room for enhancements. There are many possibilities of improvement in the current implementation of the proposed framework. Moreover, there are many potential ideas that were not realizable due to time and resource limitations. The rest of this section briefly discusses these possibilities and ideas.

It is worth mentioning that individual modules used in the proposed framework are independent from each other and play critical roles in the overall performance. Improvement of any of these modules would lead to overall performance enhancement. In this direction, one obvious possibility of improvement is to strive for a better face detector. Better face detection will certainly increase the performance of the person tracker used in this thesis. There has been a significant amount of research done for efficient person detection and tracking in unconstrained environments. An efficient and computationally optimal person detector can also overcome the dependency of person detection over face detector. It shall help to detect persons in the situations where the person's face is not visible. The work of [5] is worth consulting for this purpose.

Other possibilities of improvement are to utilize better feature extraction techniques for the representation of clothing and facial features. Utilizing 3D face models can also aid the performance and it is very intuitive in the current case, as every face track contains more than one face images. Multiple face images can be leveraged to generate efficient face models that are robust to pose and appearance variations. For this purpose, one can start from the work proposed in [81].

For clothing colors based person representation one can investigate other good features such as different color spaces, textural features and fusion of both textural and color features. Facial landmark detection plays a vital role in face alignment, and therefore affects the performance of face identification. In this regards, the implementation used in this work often fails for full profile faces. Therefore, having good and robust facial landmark detection can also improve the overall performance.

Training of attribute classifiers with more realistic training images will also help in better representation of faces. Although the training samples used in this thesis are gathered from real world datasets and contain a wide variety, but most of the images are taken by professional photographers and the style of the images is quite similar such that the subject is often facing the camera and smiling. Therefore, there are no strong shadows and lightening variations. Toward this direction, the work of [63] is a good point to start and gather good training data.

Image sets based face recognition and identification is also an emerging topic. The face tracks used in this thesis are similar to the idea of face-sets where multiple face images of a person are available for the same person. Face recognition is

performed exploiting face-set for efficient performance. The HAC algorithm used in this thesis can directly be replaced with face-set based face recognition algorithms. Intuitively, it appears to be a very nice idea and can cause good enhancement in identity clustering. In this direction, works proposed in [4, 27] can be referred as good initial points.

Common sense constraints are our friends, and can help us in a better identification process as done in [31, 34, 56]. Hence, one should look for as many constraints as possible. Also there are not much datasets available in this domain to extensively evaluate the performance of such systems. Collecting a new dataset of events captured at different events with their annotation will be a huge contribution to this field.

REFERENCES

- [1] A. C. Gallagher, T. Chen, "Clothing Cosegmentation for Recognizing People", IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Alaska, USA, 2008.
- [2] A. Martinez, R. Benavente, "The AR Face Database", Technical Report, Computer Vision Center, Autonomus University of Barcelona, 1998. <http://www2.ece.ohio-state.edu/~aleix/ARdatabase.html> (Last accessed: 18-08-2013).
- [3] A. DelBimbo and F. Dini, "Particle filter-based Visual Tracking with a First Order Dynamic Model and Uncertainty Adaptation", Computer Vision and Image Understanding (CVIU), vol. 115, no. 6, pp. 771-786, 2011.
- [4] A. Mahmood, A. Mian, "Hierarchical Sparse Spectral Clustering for Image Set Classification", British Machine Vision Conference (BMVC), 2012.
- [5] A. Schumann, M. Bauml, R. Stiefenhagen, "Person Tracking by Detection with Efficient Selection of Part-Detectors", International Conference on Advanced Video and Signal-based Surveillance (AVSS), 2013.
- [6] B. Suh, B. B. Bederson, "Semi-automatic Image Annotation using Event and Torso Identification", Technical Report, Computer Science Department, University of Maryland, 2004. <http://hcil2.cs.umd.edu/trs/2004-15/2004-15.pdf> (Last accessed: 18-08-2013)
- [7] B. Froba, A. Ernst. "Face Detection with the Modified Census Transform", In IEEE International Conference on Automatic Face and Gesture Recognition, 2004.
- [8] B. Stenger, T. Woodley, and R. Cipolla, "Learning to Track with Multiple Observers", In Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2009.
- [9] B. E. Boser, I. M. Guyon, V. N. Vapnik, "A Training Algorithm for Optimal Margin Classifiers", In Proc. of 5th Annual ACM Workshop on Computational Learning Theory, pp. 144-152, 1992.
- [10] C. Huang, H. Ai, Y. Li, S. Lao, "High-Performance Rotation Invariant Multiview Face Detection", IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 2007.
- [11] C.-C. Chang and C.-J Lin, "LIBSVM : A Library for Support Vector Machines", ACM Transactions on Intelligent Systems and Technology, 2:27:1-27:27, 2011.
- [12] C. Djeraba, "Content-based Multimedia Indexing and Retrieval", IEEE Transanction on Multimedia, vol. 9, no.2, pp. 18-22, 2002.
- [13] C. J. C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition", Data Mining and Knowledge Discover, 1998.

- [14] D. Klein, S. D. Kamvar, C. D. Manning, "From Instance-level Constraints to Space-level Constraints: Making the Most of Prior Knowledge in Data Clustering", In. Proc. 19th International Conference on Machine Learning (ICML), 2002.
- [15] D. Comaniciu, V. Ramesh and P. Meer, "Kernel-based Object Tracking", IEEE Transaction on Pattern Analysis Machine Intelligence (TPAMI), pp. 564–577, 2003.
- [16] D. G. Lowe, "Distinctive image features from Scale-Invariant keypoints", International Journal on Computer Vision (IJCV), vol. 60, no. 2, pp. 91–110, 2004.
- [17] E. Khoury, P. Gay, J. –M. Odobez, "Fusing matching and biometric similarity measures for face diarization in video", In Proc. of the 3rd ACM Conference on International Conference on Multimedia Retrieval, Dallas, Texas, USA, 2013, pp. 97-104.
- [18] F. Cricri, I.D.D Curcio, S. Mate, K. Dabov, M. Gabbouj, "Sensor-based Analysis of User Generated Video for Multi-Camera Video Remixing", In Proc. of the 18th International Conference on Advances in Multimedia Modeling, pp. 255-265, 2012.
- [19] G. Guo, D. Huang, Y. Wang, "A Novel Video Face Clustering Algorithm Based on Divide and Conquer Strategy", Trends in Artificial Intelligence (PRICAI), Lecture Notes in Computer Science, Vol. 7458, 2012, pp. 53-63.
- [20] G. Bradski, "The OpenCV Library", Dr. Dobb's Journal of Software Tools, 2000.
- [21] G. Kitagawa, "Monte Carlo filter and smoother for non-Gaussian nonlinear state space models", Journal of Computational and Graphical Statistics, vol. 5, no. 1, pp.1–25, 1996.
- [22] G. Abdollahian, C.M. Taskiran, , Z. Pizlo, E.J. Delp, "Camera Motion-Based Analysis of User Generated Video", IEEE Transactions on Multimedia, pp. 28–41, 2010.
- [23] H. Jin, Q. Liu, H. Lu, and X. Tong, "Face detection using improved LBP under Bayesian framework", In Proc. Third International Conference on Image and Graphics (ICIG), pages 306–309, Hong Kong, China, 2004.
- [24] H Rowley, S. Baluja, T. Kanade, "Rotation Invariant Neural Network-Based Face Detection", In Proc of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 1998, pp. 38–44.
- [25] H. Bay, T. Tuytelaars, and L. V. Gool, "SURF: Speeded up robust features", Lecture notes in computer science, vol. 3951, pp. 404, 2006.
- [26] H.K.Ekenel, R. Stiefelhagen, "Local Appearance based Face Recognition Using Discrete Cosine Transform", In Proc. of 13th European Signal Processing Conference (EUSIPCO), Antalya, Turkey, 2005.

- [27] H. Cevikalp, B. Triggs, "Face Recognition Based on Image Sets", In Proc. of IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2567–2573, 2010.
- [28] I. Koprinska, S. Carrato, "Temporal Video Segmentation: A Survey", *Signal Processing: Image Communication*, vol. 10, no. 5, pp. 477-500, 2001.
- [29] J. Sivic, C. L. Zitnick, R. Szeliski, "Finding people in repeated shots of the same scene", *British Machine Vision Conference (BMVC)*, 2006.
- [30] J. R. Barr, K. W. Bowyer, P. J. Flynn, S. Biswas, "Face Recognition from video: A Review", *International Journal of Pattern Recognition and Artificial Intelligence*, 2012.
- [31] J. Tao, Y. P., "Efficient clustering of face sequences with application character-based movie browsing", *IEEE International Conference on Image Processing (ICIP)*, 2008.
- [32] J. R. Barr, K. W. Bowyer, P. J. Flynn, "Detecting Questionable Observers using Face Track Clustering", *Proc. 2011 IEEE Workshop on Applications of Computer Vision*, 2011, pp. 182-189.
- [33] J. Shi, C. Tomasi, "Good features to track", *Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 593–600, 1994.
- [34] J. Tao, Y. –P. Tan, "Face clustering in videos using constraint propagation", *IEEE International Symposium on Circuits and Systems (ISCAS, Seattle, WA, 2008)*, pp. 3246-3249.
- [35] J. Sivic, M. Everingham, A. Zisserman, "Who are you? – Learning person specific classifier from video", *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.
- [36] J. MacCormick, "Probabilistic models and stochastic algorithms of visual tracking", *Ph.D. Dissertation, University of Oxford*, 2000.
- [37] J. Kubinek, "Extending training dataset for face detector learning", *Central European Seminar on Computer Graphics*, 2009.
- [38] J. Zou, Q. Ji, and G. Nagy, "A comparative study of local matching approach for face recognition," *IEEE Transactions on Image Processing (TIP)*, vol. 16, no. 10, pp. 2617–2628, 2007
- [39] K. Wagstaff, C. Cardie, S. Rogers, S. Schroedl, "Constrained K-means clustering with background knowledge", In *Proc. 18th International Conference on Machine Learning (ICML)*, 2001, pp. 577-584.
- [40] K. Okuma, A. Taleghani, N. de Freitas, J. J. Little, D. G. Lowe, "A boosted particle filter: Multitarget detection and tracking", *European Conference on Computer Vision (ECCV)*, pp. 28–39, Prague, Czech Republic, 2004.
- [41] L. Zhang, D. V. Kalashnikov, "A Unified Framework for Context Assisted Face Clustering", *ACM International Conference on Multimedia Retrieval (ICMR)*, Dallas, Texas, USA, 2013.

- [42] L. L. Presti, M. L. Cascia, “An on-line learning method for face association in personal photo collection”, *Image and Vision Computing*, Vol. 30, 2012, pp. 306-316.
- [43] L. Zhang, L. Chen, M. Li, H. Zhang, “Automated annotation of human faces in family albums”, *Proc. 11th ACM International Conference on Multimedia*, 2003, pp. 355-358.
- [44] L. Zhang, R. Chu, S. Xiang, S. Liao, S. Li, “Face Detection Based on Multi-Block LBP Representation”, In *Proc. International Conference on Biometrics (ICB)*, 2007, pp. 11-18.
- [45] L. Bourdev, S. Maji, T. Brox and J. Malik, “Detecting People Using Mutually Consistent Poselet Activations”, In *Proc. of European Conference on Computer Vision (ECCV)*, 2010.
- [46] L. Wolf, T. Hassner, and Y. Taigman, “Descriptor based methods in the wild”, in *Proc. of European Conference on Computer Vision (ECCV)*, 2008.
- [47] M. Viola, M. J. Jones, P. Viola, “Fast multi-view face detection”, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2003.
- [48] M. Tapaswi, M. Bauml, R. Stiefelhagen, “Knock! Knock! Who is it? – Probabilistic person identification in TV-Series”, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.
- [49] M. Everingham, J. Sivic, A. Zisserman, “Taking the bite out of automated naming of characters in TV video”, *Image and Vision Computing*, Vol. 27, 2009, pp. 545-559.
- [50] M Jones, P. Viola, “Fast multi-view face detection”, In *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2003.
- [51] M. S. Arulampalam, S. Maskell, N. Gordon, T. Clapp, “A Tutorial on Particle Filters for Online Non-linear/Non-Gaussian Bayesian Tracking”, *IEEE Transactions on Signal Processing*, Vol. 20, No. 2, pp-174-188, 2002.
- [52] M. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. V. Gool, “Robust tracking-by-detection using a detector confidence particle filter”, in *Proc. of IEEE 12th International Conference on Computer Vision (ICCV)*, 2009.
- [53] M. Bäuml, K. Bernardin, M. Fischer, H.K. Ekenel and R. Stiefelhagen, “Multi-Pose Face Recognition for Person Retrieval in Camera Networks”, *7th International. Conference on Advanced Video and Signal-Based Surveillance*, 2010.
- [54] M. Uricar, V. Franc and V. Hlavac, “Detector of Facial Landmarks Learned by the Structured Output SVM,”, In *Proc. of the 7th International Conference on Computer Vision Theory and Applications (VISAPP)*, 2012.
- [55] M. Everingham, J. Sivic, A. Zisserman, “Hello! My name is... Buffy - Automatic Naming of Characters in TV Video”, *British Machine Vision Conference (BMVC)*, 2006.

- [56] M. Bauml, M. Tapaswi, R. Stiefelhagen, "Semi-supervised Learning with Constraints for Person Identification in Multimedia Data", In Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2013.
- [57] N. Kumar, A. C. Berg, P. N. Belhumeur, S. K. Nayar, "Describable visual attributes for face verification and image search", IEEE Transactions on Pattern Analysis And Machine Intelligence (TPAMI), 2011.
- [58] N. Gourier, D. Hall, and J. L. Crowley, "Estimating Face Orientation from Robust Detection of Salient Facial Features", In Proceedings of Pointing 2004, ICPR, International Workshop on Visual Observation of Deictic Gestures, Cambridge, UK, 2004.
- [59] N. Kumar, "Describable Visual Attributes for Face Images", Technical Report CUCS-035-11, PhD Dissertation, Department of Computer Science, Columbia University, 2011. http://homes.cs.washington.edu/~neeraj/base/papers/nk_phd_thesis2011.pdf (Last accessed: 18-08-2013)
- [60] N. Kumar, P. Belhumeur, S. Nayar, "FaceTracer: A Search Engine for Large Collections of Images with Faces", In Proc. of the 10th European Conference on Computer Vision (ECCV), 2008.
- [61] N. Dalal, B. Triggs, "Histogram of Oriented Gradients for Human Detection", In Proc. of International Conference on Computer Vision and Pattern Recognition (CVPR), 2005.
- [62] N. Peyrard, P. Bouthemy, "Motion-based Selection of Relevant Video Segments for Video Summarisation", IEEE International Conference on Multimedia and Expo (ICME), vol. 2, pp. 409–412, 2003.
- [63] N. Cherniavsky, I. Laptev, J. Sivic, and A. Zisserman. "Semisupervised Learning of Facial Attributes in Video", In The first international workshop on parts and attributes (in conjunction with ECCV 2010), 2010.
- [64] O. Jesorsky, K. J. Kirchberg, R. W. Frischholz, "Robust Face Detection using the Hausdorff Distance", In 3rd International Conference on Audio- and Video-Based Biometric Person Authentication, 2001.
- [65] P.Hao, S. -I. Kamata, "Unsupervised People Organization and its Application on Individual Retrieval from Videos", 21st International Conference on Pattern Recognition (ICPR), Tsukuba, Japan, 2012.
- [66] P. Viola, M. Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features", IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2001, pp. 511–518.
- [67] P. Phillips, H. Moon, S. Rizvi, P. Rauss, "The FERET Evaluation Methodology for Face Recognition Algorithms", IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 22(10), 2000.
- [68] P. Perez, C. Hue, J. Vermaak and M. Gangnet. "Color-based Probabilistic Tracking", In Proc. of the European Conference on Computer Vision (ECCV), Vol. 1, pp. 661–675, Copenhagen, Denmark, 2002.

- [69] R. G. Cinbis, J. Verbeek, C. Schmid, “Unsupervised Metric Learning for Face Identification in TV Video”, International Conference on Computer Vision (ICCV), 2011.
- [70] R. Layne, T. M. Hospedales, S. Gong, “Towards Person Identification and Re-identification with Attributes”, European Conference on Computer Vision (ECCV), 2012.
- [71] R. Lienhart, A. Kuranov, V. Pisarevsky, “Empirical Analysis of Detection Cascades of Boosted Classifiers for Rapid Object Detection”, In Proceedings of the 25th DAGM-Symposium, Magdeburg, Germany, 2003, pp. 297–304.
- [72] R. E. Kalman, “A New Approach to Linear Filtering and Predictive Problems”, Transactions of ASME, Journal of Basic Engineering, Vol. 82, 1960, pp. 34–45.
- [73] S. Liao, X. Zhu, Z. Lei, L. Zhang, S. Z. Li, “Learning Multi-scale Block Local Binary Patterns for Face Recognition”, In Proc. of International Conference on Biometrics (ICB), 2007, pp. 828-837.
- [74] S. Birchfield, “Elliptical Head Tracking using Intensity Gradients and Color Histograms”, In Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 232–237, Santa Barbara, USA, 1998.
- [75] S. Lloyd, “Least Squares Quantization”, IEEE Transactions on Information Theory, Vol. 28, No. 2, pp. 128–137, 1982.
- [76] S.-F. Chang, D. Ellis, W. Jiang, K. Lee, A. Yanagawa, A. C. Loui and J. Luo, “Large-scale Multimodal Concept Detection for Consumer Video”, In Proc. of International Workshop on Multimedia Information Retrieval, pp. 255-264, New York, USA, 2004.
- [77] T. Ojala, M. Pietikäinen, and T. Mäenpää, “Multiresolution Gray-scale and Rotation Invariant Texture Classification with Local Binary Patterns”, IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), Vol. 24, 2002, pp. 971–987.
- [78] T. Sim, S. Baker, M. Bsat, “The CMU Pose, Illumination, and Expression Database”, IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 25(12), 2003.
- [79] T. Ahonen, A. Hadid, M. Pietikäinen, “Face Recognition with Local Binary Patterns”, European Conference on Computer Vision (ECCV), 2004.
- [80] T. Hastie, R. Tibshirani, J. Friedman, “The Elements of Statistical Learning (2nd Edition), New York: Springer, pp. 520-528, 2009.
- [81] U. Park, A. K. Jain, “3D Model-Based Face Recognition in Video”, In 2nd International Conference on Biometrics, 2007.
- [82] W. R. Schwartz, A. Kembhavi, D. Harwood, L. S. Davis, “Human Detection Using Partial Least Squares Analysis”, In Proc. of International Conference on Computer Vision (ICCV), 2009.

- [83] W. Wong, D.Q. Huynh, M. Bennamoun, "Upper Body Detection in Unconstrained Still Images", 6th IEEE Conference on Industrial Electronics and Applications (ICIEA), 2011, vol., no., pp.287,292, 21-23 June 2011
- [84] X. Zhang, Y. Gao, "Face Recognition Across Pose, A Review", Elsevier Pattern Recognition, 2009, pp. 2876 – 2896.
- [85] X. Huang, S.Z. Li, and Y.Wang, "Shape Localization Based on Statistical Method using Extended Local Binary Pattern", In Proc. Third International Conference on Image and Graphics (ICIG), pages 184–187, Hong Kong, China, 2004.
- [86] Y. -H. Lei, Y. -Y. Chen, B. -C. Chen, L. Lida, W. H. Hsu, "Where Is Who: Large-Scale Photo Retrieval by Facial Attributes and Canvas Layout", In Proc. of the 35th International ACM SIGR Conference on Research and Development in Information Retrieval, Portland, Oregon, USA, 2012.
- [87] Y. Li, S. Gong, J. Sherrah, H. Liddell, "Support Vector Machine Based Multi-view Face Detection and Recognition", Image and Vision Computing, Vol. 22, pp. 413–427, 2004
- [88] Y. Freund, R.E. Schapire, "Experiments with a New Boosting Algorithm", In Proc. of the IEEE International Conference on Machine Learning (ICML), pp. 148–156, Bari, Italy, 1996.
- [89] Y. Rodriguez, "Face Detection and Verification using Local Binary Patterns", Ph.D. thesis, EPFL, 2006.
- [90] Y. Freund, R. E. Schapire, "A Short Introduction to Boosting", Journal of Japanese Society for Artificial Intelligence, Vol. 14, 1999, pp. 771–780.
- [91] Y. Cai, N. D. Freitas and J. J. Little, "Robust Visual Tracking for Multiple Targets", European Conference on Computer Vision (ECCV), 2006
- [92] Y. Rubner, J. Puzicha, C. Tomasi, and J. M. Buhmann, "Empirical Evaluation of Dissimilarity Measures for Color and Texture", In Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), vol.84, pp. 25–43, 2001.
- [93] <http://www.threadingbuildingblocks.org/> (Last accessed: 08-18-2013).