Multiple Face Detection and Recognition in Real-Time using Open CV

Pradeep Kumar. G. H
Assistant Professor, Department of CSE, KSIT

Ashwini. M, Divya. G. N, Manjushree. B. N
Students, Department of CSE, KSIT

Abstract:- **With the pervasiveness of monitoring cameras installed in public places, schools, hospitals and homes, video analytics technologies for interpreting the generated video content are becoming more and more relevant to people's lives. Along this context, we develop a human-centric video surveillance system that identifies and tracks people in a given scene. In this paper, Multiple human face detection and recognition in real time application is implemented to detect and recognize the faces in real time using a open source computer vision called Open CV which is a library of programming functions mainly aimed at real time computer vision and with the help of EMGU CV.The application runs efficiently and smoothly, so that multiple people can be simultaneously tracked in real time. Furthermore, significant innovations are involved in this work in making each of the major image analysis modules both fast and robust to variations in pose, occlusions and so on. A software has been implemented that supports finding, tagging, identifying people in live or recorded videos.**

Keywords- Emgu CV open CV face detection, face recognition, principal component analysis

1. INTRODUCTION

As the famous proverb says, "Face is the index of the mind". A face to face interaction between human beings is considered most important and natural way to communicate. Recognition of faces and processing the data is a challenging task with very large database using low-cost desktop embedded system. With the prevalence of surveillance cameras in streets, air-ports, schools, hospitals, data centers, work places and even homes,
fields of homeland security, education, health care, smart city, smart home, among others. This work is aimed at analyzing surveillance videos and building a human- centric system that identifies and tracks people in videos captured at a given scene. While plenty researches have been conducted on object tracking, including human tracking, and many promising approaches have been proposed, there are still significant challenges in recognizing and tracking people in videos with uncontrolled capturing conditions largely due to technologies for video content analytics are playing an increasingly essential role in the
pose variations, as well as occlusions and cluttered background. It is especially complex to detect and identify multiple people simultaneously in real time as the result of the large amount of computations involved. There are numerous technologies and algorithms used in face detection and recognition systems and the most popular among them is viola–jones method ada booster algorithm, harcascade classifiers, principal component analysis with Eigenfaces algorithm which we have implemented in our system. Often real time response is understood to be in order of milliseconds and sometimes microseconds, which is the most crucial criteria in the system design.
The rest of the paper is organized as follows
.In section 2 we glance at a variety of face detection and recognition methods. Section
3 contains an overview of the system, including a description of the Open CV, Emgu CV with and details on the image analysis modules. The methods of two core modules, i.e. Face detection and face recognition, are described in Section 4 and Section 5, respectively. Experimental results are given in Section 6, followed by conclusions and future work in Section 7.

With the goal of detecting and identifying a relatively large number of faces simultaneously in high-resolution video in real time, we studied recent literature on related topics, and in particular, on face detection and face recognition in video.

Video Face Detection

Face detection is a process, which is to analysis the input image and to determine the number, location, size, position and the orientation of face. Face detection is the base for face tracking and face recognition, whose results directly affect the process and accuracy of face recognition. The common face detection methods are: knowledge- based approach, Statistics-based approach and integration approach with different features or methods. The knowledge-based approach can achieve face detection for complex background images to some extent and also obtain high detection speed, but it needs more integration features to further enhance the adaptability. Statistics-based approach detects face by judging all possible areas of images by classifier, which is to look the face region as a class of models, and use a large number of "Face" and "non-face" training samples to construct the classifier.

The method has strong adaptability and robustness, however, the detection speed needs to be improved, because it requires test all possible windows by exhaustive search and has high computational complexity. The AdaBoost algorithm arose in recent years; it trains the key category features to the weak classifiers, and cascades them into a strong classifier for face detection. The method has real-time detection speed and high detection accuracy, but needs long training time. For object tracking, there are several mainstream methods, including the Kanade- Lucas-Tomasi (KLT) [1], the mean shift [2] and the particle filtering [3] algorithms. Most videos face tracking approaches are based on one or more of these basic methods. For example, in [4], the KLT tracker is used to track points of interest throughout video frames, and then each face track is formed by the faces detected in different frames that share a large enough number of tracked points. The method was tested on eight broadcast video sequences and achieved a face track detection rate of 94.17% with a processing speed of 3.8 FPS (frames per second).

The CamShift method is an extension of the mean shift algorithm with an adaptive region-resizing step. In [5], to enhance the accuracy of CamShift, three constraint items are posed, including evaluation of location precision, scale of face area and dynamic histogram updating, to ease problems caused by surrounding regions with color similar to the skin tone and illumination changes across frames. Also, tracked faces in one frame are matched with faces in previous frames using the LBP (local binary patterns) features to determine if a face is new or has appeared before. Tested on three videos captured by their own, the tracking part took 1.6~4.7ms per frame with an error rate around 12%. The face matching part, however, has rather high error rates with 83.33% for false negatives and 81.48% for false positives.

A particle filter based approach is proposed in [6] that uses an adaptive target model. The method was tested on YouTube videos with resolution from 180x240 to 240x320. This paper describes a system that can detect and track human face in real time using haar-like features where the detection algorithm is based on wavelet transform. In computer vision, low level processing involves image processing tasks in which the quality of the image is improved for the benefit of human observers and higher level routines to perform better [Viola & Jones, 2001]. Intermediate level processing involves the processes of feature extraction. With an AdaBoost face detector and 1500 particles per target, an average frame rate of 6 FPS was reported on Pentium IV 3GHZ processor. It was shown that with the method, some face detection errors can be removed and faces may be recovered after a total occlusion.

Video Face Recognition

Several methods have been suggested for face recognition over the past few years and a recent survey could be found in [4]. The most common techniques used in face

recognition are Principal Component Analysis (PCA), Neural Network, Template Matching, and Model Matching. The choice of using a particular method is specified by its suitability for a specific application [5].In face recognition by Template Matching [6], salient regions of the facial image are extracted, and then these regions are compared on a pixel-by-pixel basis with an image in the database. The advantage of this method is that the image preprocessing is simple, but the database search and comparison are computationally expensive. In face recognition by Neural Network [7] based on learning of the faces in the training phase, the learning set of faces should be large enough in number to realize the variations in real life situations. Neural Network solutions model the face recognition problem very well, but they take significant training time. In Local Feature Analysis (LFA) [8] technique, dozens of features from different regions of the face and the relative location of these features are utilized and incorporated to identify and verify the face image. Although LFA method offers robustness in carrying out a match with local variations on the facial image, the technique is not robust against global facial attributes. In [9] Hidden Markov Models (HMM) and Wavelets for face recognition are used, and during the learning model the best matching model, offers a query image. Success of Model Matching methods depends mainly on building realistic representative model. Eigenface [2] method is one of the well- known face detection and face recognition algorithms.In our proposed methodology we use a multi algorithm combining PCA with eigen face method using Haar-like features applied on the same facial data to decide the identity of a subject based on the above summary, it can be seen that there is still a big gap in getting real-time multi-face tracking and recognition in high resolution video. Much more work needs to be done in several aspects to make efficient recognition of multiple faces utilizing multiple views of the face.

2. OVERVIEW OF THE SYSTEM

The main target of this application is to build a real-time system that could be used in real-world such as the

environment, where many technical systems require natural human-computer interfaces using different kind of cameras installed into everyday living and working environment.

The Data flow Diagram

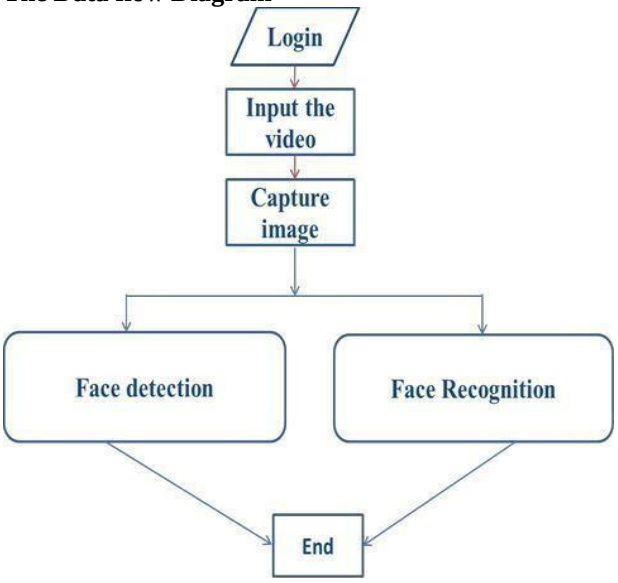

Fig .1.The proposed data flow diagram isshown

In the software implementation, the user logins to the system ,the login form is created using visual studio. once the users credentials are validated then the application runs.

The input to the application is video, so the web cam is used. The image data of the video frame is acquired and the frame is preprocessed, including normalization, gradient computation and integral image calculation .Face detection is applied for every frame followed by face tracking. Face detection forms the basis for both face tracking and face recognition .The face recognition technique uses the eigen face vector and pca algorithm to identify the face in the video frame .Finally the recognized face is tagged with the name. This process continues for each and every face and each and every frames simultaneously.

Open CV

OpenCV (*Open* Source Computer Vision Library) is a library of programming

functions mainly aimed at real time computer vision, developed by Intel and now supported by Willow Garage. It is free for use under the open source BSD license. The library is cross-platform. It focuses mainly on real-time image processing. It is written in C++ and its primary interface is in C++, but it still retains a less comprehensive though extensive older C interface .In the proposed system C# programming language is used as back end .

EMGU CV

Emgu CV is a cross platform .Net wrapper to the Intel OpenCV image processing library. Allowing OpenCV functions to be called from .NET compatible languages such as C#, VB, VC++, Iron Python etc. The wrapper can be compiled in Mono and run on Linux / Mac OS X. In other words Emgu CV is an awesome Wrapper, this let make very interesting things and tasks of computer vision, this library is used in the implementation of software., EmguCV have many functions that let us work with CPU and GPU increases the performance dramatically.

Major Image Processing Modules

The facial recognition process normally has four interrelated phases or steps. The first step is face detection, the second is normalization, the third is feature extraction, and the final cumulative step is face recognition. These steps depend on each other and often use similar techniques. Each of these steps poses very significant challenges to the successful operation of a face recognition system.

Face Detection

Face detection is conducted occasionally to adjust locations of tracked objects and to discover new objects in video as well. A multi-view face detector is developed based on the Viola-Jones method (i.e. AdaBoost cascade classifier with Haar-like features). Three classifiers are trained and applied for frontal, half-profile and profile faces, respectively. And with each classifier, it works at multiple directions (e.g. up-right,
±30°) to deal with in-plane rotations and over multiple scales. In this module we are going to detect the face of the characters. In this module we are using the emgucv library we must install the emgucv library. After installing the emgucv lib in our project we need to add reference with the name emgu.cv, emgu.cv.util, emgu.cv.ui. When you will complete the references you will get the emgu controls in the toolbox.

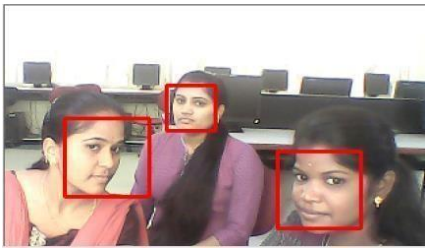

Fig. 2.Sample face detection results (red bounding boxes).

Normalization

Once the face has been detected (separated from its background), the face needs to be normalized. This means that the image must be standardized in terms of size, pose, illumination, etc., To normalize a probe image, the key facial landmarks must be located accurately. Using these landmarks, the normalization algorithm can (to some degree) reorient the image for slight variations. Such corrections are, however, based on statistical inferences or approximations which may not be entirely accurate. Thus, it is essential that the probe is as close as possible to a standardized face.

Feature Extraction and Recognition

Once the face image has been normalized, the feature extraction and recognition of the face can take place. In feature extraction, a mathematical representation called a biometric template or biometric reference is generated, which is stored in the database and will form the basis of any recognition task. It is important for successful recognition that maximal information is retained in this transformation process so that the biometric template is sufficiently distinctive. To improvise the system and provide additional features, the following algorithms are used:

ECGM Algorithm

ECGM algorithm is being used to detect the faces in the video and to reduce the noises in the complex scenes. ECGM is a powerful tool for graph matching with distorted inputs. It has various applications in pattern recognition and computer vision. Through error correcting graph matching, we can define appropriate graph edit operations according to the noise investigation and design the edit cost function to improve the performance.

Eigen Faces using PCA

The PCA technique converts each two dimensional image into a one dimensional vector. Each component (eigenface) represents only a certain feature of the face, which may or may not be present in the original image. A probe image is compared against a gallery image by measuring the distance between their respective feature vectors. For PCA to work well the probe image must be similar to the gallery image in terms of size (or scale), pose, and illumination. It is generally true that PCA is reasonably sensitive to scale variation. The system supports manual tagging of a new subject. The user only needs to type the name of a subject appearing in the video.

3. FACE DETECTION

In this section the base algorithm used to detect the face is discussed ,AdaBoost algorithm is discussed.

4.1 AdaBoost

Adaboost is an algorithm for constructing a strong classifier as linear combination. Adaboost, short for Adaptive Boosting, is a machine learning algorithm , formulated by Yoav Freund and Robert Schapire[14].It is a meta-algorithm, and can be used in conjunction with many other learning algorithms to improve their performance. Adaboost is adaptive in the sense that subsequent classifiers built are tweaked in favour of those instances misclassified by previous classifiers. Adaboost is sensitive to noisy data and outliers. In some problems, however, it can be less susceptible to the over fitting problem than most learning algorithms. The classifiers it uses can be

weak (i.e., display a substantial error rate), but as long as their performance is slightly better than random (i.e. their error rate is smaller than 0.5 for binary classification),they will improve the final model. Even classifiers with an error rate higher than would be expected from a random classifier will be useful, since they will have negative coefficients in the final linear combination of classifiers and hence behave like their inverses [14]. Adaboost generates and calls a new weak classifier in each of a series of rounds .For each call, a distribution of weights is updated that indicates the importance of examples in the data set for the classification. On each round, the weights of each incorrectly classified example are increased, and the weights of each correctly classified example are decreased, so the new classifier focuses on the examples which have so far eluded correct classification.

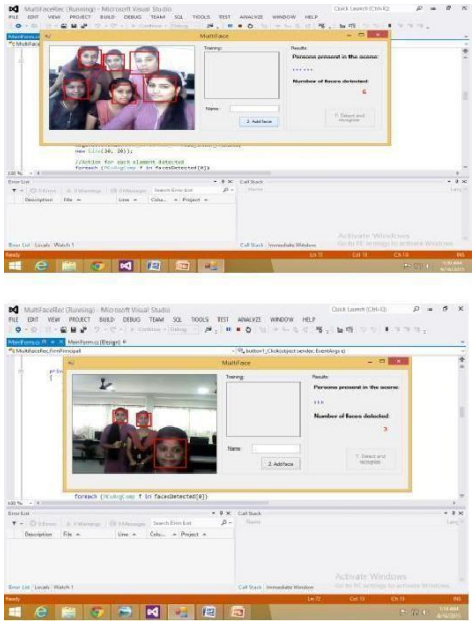

Fig. 3.Robust detection in case of pose anddistance changes .

4. FACE RECOGNITION

We use EMGU to perform Principle Component Analysis (PCA) to achieve multiple face recognition. To perform PCA several steps are undertaken:

Stage 1: Subtract the Mean of the data from each variable (our adjusted data)

Stage 2: Calculate and form a covariance Matrix

Stage 3: Calculate Eigenvectors and Eigen values from the covariance Matrix Stage 4: Choose a Feature Vector (a name for a matrix of vector

Stage 5: Multiply the transposed Feature Vectors by the transposed adjusted data

Stage 1: Mean Subtraction

This data is fairly simple and makes the calculation of our covariance matrix a little simpler now this is not the subtraction of the overall mean from each of our values as for covariance we need at least two dimensions of data. It is in fact the subtraction of the mean of each row from each element in that row.

(Alternatively the mean of each column from each element in the column however this would adjust the way we calculate the covariance matrix)

Stage 2: Covariance Matrix

The basic Covariance equation for two dimensional data is:

$$cov(x,y) = \frac{\sum_{i-1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{(n-1)}$$

Which is similar to the formula for variance however, the change of x is in respect to the change in y rather than solely the change of x in respect to x . In this equation x represents the pixel value and x̄is the mean of all x values, and n the total number of values.

The covariance matrix that is formed of the image data represents how much the dimensions vary from the mean with respect to each other. The definition of a covariance matrix is:

$$C^{n*n} = (C_{i,j}, C_{i,j} = cov(Dim_i, Dim_j))$$

Now the easiest way to explain this is but an example the easiest of which is a 3x3 matrix.

$$C_{mat} = \begin{pmatrix} cov(x,x) & cov(x,y) & cov(x,z) \\ cov(y,x) & cov(y,y) & cov(y,z) \\ cov(z,x) & cov(z,y) & cov(z,z) \end{pmatrix}$$

$$I1 = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix} \qquad C(I1) = \begin{pmatrix} cov(1,1) & cov(2,5) & cov(3,9) \\ cov(4,1) & cov(5,5) & cov(6,9) \\ cov(7,1) & cov(8,5) & cov(9,9) \end{pmatrix}$$

Now with larger matrices this can become more complicated and the use of computational algorithms essential.

Stage 3: Eigenvectors and Eigenvalues

Eigenvalues are a product of multiplying matrices however they are as special case. Eigenvalues are found by multiples of the covariance matrix by a vector in two dimensional space (i.e. a Eigenvector). This makes the covariance matrix the equivalent of a transformation matrix. It is easier to show in a example:

$$Covarience\ Matrix = \begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix}$$

$$Eigenvector = \begin{bmatrix} 6 \\ 4 \end{bmatrix}$$

Multipied:

$$\begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix} * \begin{bmatrix} 6 \\ 4 \end{bmatrix} = \begin{bmatrix} 24 \\ 16 \end{bmatrix} = 4 \begin{bmatrix} 6 \\ 4 \end{bmatrix}$$

Eigenvectors can be scaled so ½ or x2 of the vector will still produce the same type of results. A vector is a direction and all you will be doing is changing the scale not the direction.

$$\begin{bmatrix} 2 & 3 \\ 2 & 1 \end{bmatrix} * \begin{bmatrix} 3 \\ 2 \end{bmatrix} = \begin{bmatrix} 12 \\ 8 \end{bmatrix} = 4 \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

Eigenvectors are usually scaled to have a length of 1:

$$\begin{bmatrix} A \\ B \end{bmatrix} \quad becomes \quad \begin{bmatrix} A/(\sqrt{A^1 + B^2}) \\ B/(\sqrt{A^1 + B^2}) \end{bmatrix}$$

The Eigenvalue is closely related to the Eigenvector used and is the value of which

the original vector was scaled in the example the Eigenvalue is 4.

Stage 4: Feature Vectors

Now a usually the results of Eigenvalues and Eigenvectors are not as clean as in the example above. In most cases the results provided are scaled to a length of 1. So here are some example values calculated using Matlab:

$$Covarience\ Matrix = \begin{bmatrix} 0.212 & 0.345 \\ 0.621 & 0.111 \end{bmatrix}$$

$$Eigenvector = \begin{bmatrix} 0.6271 \\ 0.3041 \end{bmatrix}$$

$$Eigenvalues = \begin{bmatrix} 0.6392 & -0.557 \\ 0.7691 & 0.3814 \end{bmatrix}$$

Once Eigenvectors are found from the covariance matrix, the next step is to order them by Eigenvalue, highest to lowest. This gives you the components in order of significance. Here the data can be compressed and the weaker vectors are removed producing a flossy compression method, the data lost is deemed to be insignificant.

$$Resultant\ Eigenvalues = \begin{bmatrix} 0.6392 \\ 0.7691 \end{bmatrix}$$

Stage 5: Transposition

The final stage in PCA is to take the transpose of the feature vector matrix and multiply it on the left of the transposed adjusted data set (the adjusted data set is from Stage 1 where the mean was subtracted from the data).

5. EXPERIMENTAL RESULTS

The above solution has been implemented to build a software that works on both live and recorded videos. When the software is running, a user may intervene at any time to tag a detected face in a frame. A feature vector of the tagged face is then computed

and saved in the trained image folder for different variations in pose, illumination
,distances etc. The training data default location is within the Trained Faces folder of the application path. It has a single XML file that contains tags for the name of the person and a file name for the training image. However, due to the lack of standard benchmark videos, it is unpractical to compare detection accuracy with prior arts.

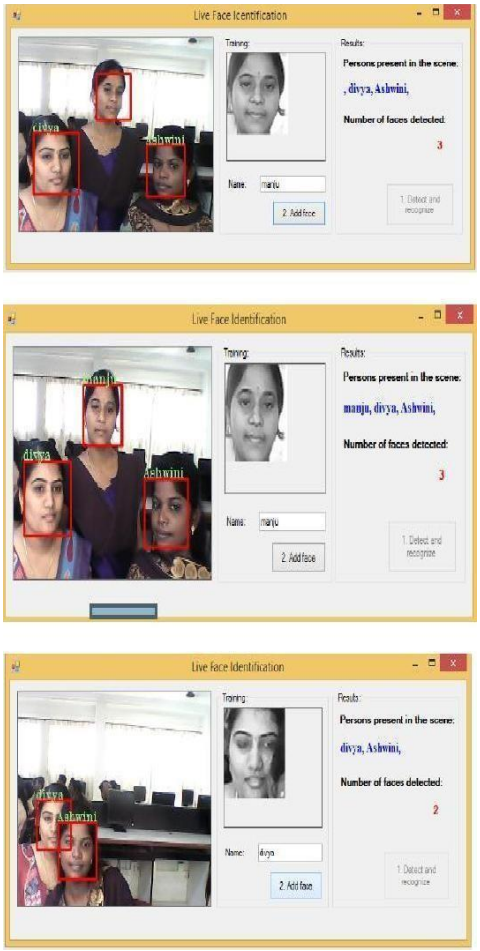

Fig.4. Snapshots of our developed application for human face detection and recognition in real time video.

6. CONCLUSIONS AND FUTURE WORK

In this paper, a human detection and recognition system for real time use cases was presented. Especially, using a open source computer vision called Open CVwhich is a library of programming functions mainly aimed at real time computer vision and with the help of EMGU CV and innovations were made to image analysis modules for both efficiency and robustness. A working application written in C# was developed which allows instant detection and recognition of multiple people in relatively high-resolution videos.

Further research will be done in a number of aspects to improve this work. First, video processing pipeline can be used that integrates the image processing modules. Second, a solution will be made for dealing with large number of images in trained folder. Third, human body detection and tracking will be integrated into the system so that face and body information can be used together.

7. ACKNOWLEDEMENT

We would like to extend our gratitude to the Management and the Principal Dr. T.V. Govindaraju, K.S. Institute of Technology, Bangalore, for facilitating to complete the project. we thank Mrs. Rekha.B.Venkatapur, Mr.Pradeep Kumar G.H, Dr.Naveen N.C and Mr.Pradeep K.R, Department of Computer Science and Engineering, K.S. Institute of Technology, Bangalore, for their support and guidance. The project deliverables were timely scheduled so as to encourage a steady development and thereby leading to the timely completion of the project. The lectures of the course Computer vision and image processing provided a good understanding of the various concepts involved in face recognition.

8. REFERENCES

[1] B. Lucas, T. Kanade, "An iterative image registration technique with an application to stereo vision," *Intl'JointConf.onArtificial Intelligence*,p.674-679, 1981.

[2] D. Comaniciu, V. Ramesh, P. Meer, "Real-time tracking of non-rigid objects using mean shift," *IEEE Conf. onCVPR*, p.142-149, 2000.

[3] M. Arulampalam, S. Maskell, *et al.* "A tutorial on particle filters for onlinenonlinear/non-Gaussian Bayesian tracking,"*IEEE Trans. on Signal Processing*,Vol.50, No.2, Feb. 2002.

[4] T. D. Ngo, D.-D. Le, *et al.* "Robust face track finding in video using tracked points," *IEEE Intl' Conf. on Signal ImageTechnology and Internet Based Systems*, 2008.

[5] X. Wu, L. Li, *et al.* "A framework of face tracking with classification using CAMShift-C and LBP," *Fifth Intl'Conf.on Image and Graphics*, p.217- 222, 2009.

[6] M. Kim, S. Kumar, V. Pavlovic, *et al.* "Face tracking and recognition with visual constraints in real-world videos," *IEEE Conf. on CVPR*, 2008.

[7] V. Belagiannis, F. Schubert, *et al.* "Segmentation based particle filtering for real-time 2d object tracking," *Euro.Conf.on Computer Vision*, Vol. IV,p.842-855, 2012.

[8] Z. Kalal, K. Mikolajczyk, *et al.* "Face- TLD:tracking-learning-detection applied to faces," *IEEE Intl' Conf.onImage Processing*, 2010.

[9] E. Maggio, E. Piccardo, *et al.* "Particle PHD filter for multi-target visual tracking," *IEEE Conf. on Acoustics,SpeechandSignal Processing*, p.15-20,2007.

[10] R. G. Cinbis, J. Verbeek, *et al.* "Unsupervised metric learning for face identification in TV video," *IEEE Intl' Conf. onComputer Vision*, p.1559– 1566, 2011.