A Survey on Face Recognition Technology - Viola Jones Algorithm

Ankur Sahitya
Dept of ISE
BMSIT&M, Bangalore
Karnataka, India

Manjunath T N
Dept of ISE
BMSIT&M, Bangalore
Karnataka, India

Veena N
Dept of ISE
BMSIT&M, Bangalore
Karnataka, India

ABSTRACT
Computer vision is gaining momentum towards variety of applications in recent days, in this context, we have done survey of face recognition technology and algorithms for the current trend applications. Here we describe the necessity and adopted methods to detect a human face. Since the data is computed by the computer, many algorithms are developed to detect a face. Some of the key challenges for the process of face detection are discussed. A rapid approach to detect face developed by viola and jones is explained in brief. The 4 main concepts involved in the viola jones method such as haar features, integral image, Adaboost and classifier cascade are highlighted, this work will help image accelerators, enhancements, filmy and military purposes.

Keywords
Adaboost, Face detection, challenges, haar cascade

1. INTRODUCTION
In the past 10 years, there has been an exponential development in the field of image processing. New generation computers are becoming smarter and faster to process Terabytes of data. Object recognition is a field which has gained attention because of its utility in industries such as manufacturing, packing etc. A common example is supermarket that uses barcode to identify a product. A manufacturing unit employs a device that needs to accurately find the position of an object. For example, a bottle filler robotic arm needs to know the relative position of the bottle in which the liquid is filled. Object recognition needs computer vision to detect the structural nature of the object. Computer vision allows the computer to sense the object and process the desired information at blazing speeds. Just like eyes are the source for human, a camera is the undisputed source for computer.

Fig-1: face in a 2D frame

Face detection is also a part of object detection [1].Face detection can be classified into two classes (face and non-face). Most applications are based on face recognition and tracking. These applications need to locate the position of the face in the image or video. Moreover, face detection has added a much needed aspect of security in the recent years. Biometric systems, a front sided camera (Selfie) of a smart phone, human presence detection are some of the key implementations of face detection. Basically face detection senses the presence of the face in a 2D frame. Fig-1. Shows human faces that has a close resemblance with uniform structures. Hence the detection of human face is possible [2]. Several methods and approaches are developed for face detection. Fig-1. The extended Yale face database B. The results generated by the computer depends upon the data feed into it, the correctness and the reliability of the algorithm used. To evaluate the performance of the face detectors some parameters are used as a standard. Some of them are [3]:

a) Learning time: time required to adapt and improve the reliability to differentiate between a face and nonface,

b) Samples needed for training: more the number of samples more is the accuracy of the result obtained sacrificing the speed of detection,

c) The ratio between detection rate and false alarm,

d) Execution time. In section 3, some factor affecting the face detection algorithm is discussed. Section 4 describes the commonly used face detection approaches. Section 5 throws light on viola jones method with adaptive boost learning.

2. LITERATURE SURVEY
During the last decade a number of promising face detection algorithms have been developed and published. Among these three stand out because they are often referred to when performance figures etc. are compared. This section briefly presents the outline and main points of each of these algorithms. Robust Real-Time Objection Detection, 2001 [1] By Paul Viola and Michael J. Jones. This seems to be the first article where Viola-Jones presents the coherent set of ideas that constitute the fundamentals of their face detection algorithm. This algorithm only finds frontal upright faces, but is in 2003 presented in a variant that also detects profile and rotated views [2]. The 'Methods' chapter will elaborate more on the basic version of this algorithm. Neural Network-Based Face Detection, 1998 [3] By Henry A. Rowley, Shumeet Baluja and Takeo Kanade. An image pyramid is calculated in order to detect faces at multiple scales. A fixed size sub-window is moved through each image in the pyramid. The content of a subwindow is corrected for non-uniform lightning and subjected to histogram equalization. The processed content is fed to several parallel neural networks that carry out the actual face detection. The outputs are combined using logical AND, thus reducing the amount of false detections. In its first form this algorithm also only detects frontal upright faces. A Statistical Method for 3D Object Detection Applied to Faces and Cars, 2000 [4] By Henry Schneider man and Takeo Kanade. The basic mechanics of this algorithm is also to calculate an image pyramid and scan a fixed size sub-window through each layer of this pyramid. The content of the sub window is subjected to a wavelet analysis and histograms are made for the different wavelet coefficients. These coefficients are fed to differently train parallel detectors

that are sensitive to various orientations of the object. The orientation of the object is determined by the detector that yields the highest output. Opposed to the basic Viola Jones algorithm and the algorithm presented by Rowley et al. this algorithm also detects profile views. One of the fundamental problems of automated object detection is that the size and position of a given object within an image is unknown. As two of the mentioned algorithms demonstrate the standard way to overcome this obstacle is to calculate an image pyramid and scan the detector through each image in the pyramid. While being relatively straightforward to implement this process is rather time consuming, but in their paper Viola-Jones presents a novel approach to this problem.

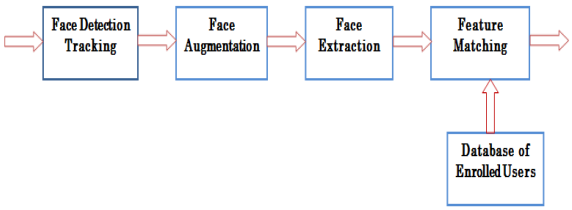


Fig-2: Face recognition process flow

2.1 The Viola-Jones face detector

Here we discuss about the concern of implementation of the Viola-Jones face detection algorithm. The first part elaborates on the methods and theory behind the algorithm. In order to avoid copying the original Viola-Jones paper this section is kept relatively short, but still the most important points are explained. Secondly interesting aspects of the actual implementation are emphasized and presented together with results and comments on performance. This structure is preferred since many intermediate results have affected implementation decisions and vice versa. The basic principle of the Viola-Jones algorithm is to scan a sub-window capable of detecting faces across a given input image. The standard image processing approach would be to rescale the input image to different sizes and then run the fixed size detector through these images. This approach turns out to be rather time consuming due to the calculation of the different size images. Contrary to the standard approach Viola-Jones rescale the detector instead of the input image and run the detector many times through the image – each time with a different size. At first one might suspect both approaches to be equally time consuming, but Viola-Jones have The scale invariant detector The first step of the Viola-Jones face detection algorithm is to turn the input image into an integral image. This is done by making each pixel equal to the entire sum of all pixels above and to the left of the concerned pixel. This is demonstrated in Fig-3

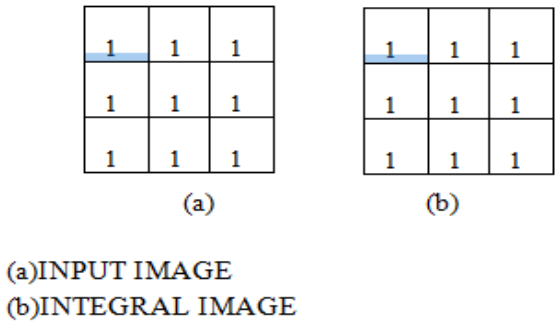


Fig-3: The integral image

This allows for the calculation of the sum of all pixels inside any given rectangle using only four values. These values are the pixels in the integral image that coincide with the corners of the rectangle in the input image. This is demonstrated in Fig-3.

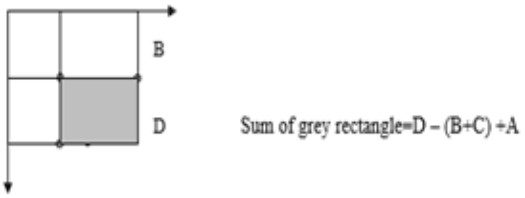


Fig-4: Sum Calculation

Since both rectangle B and C include rectangle A the sum of A has to be added to the calculation. It has now been demonstrated how the sum of pixels with in rectangle of arbitrary size can be calculated in constant time. The Viola-Jones face detect or analyzes a given sub-window using features consisting of two or more rectangles. The different types of features are shown in Fig-5.

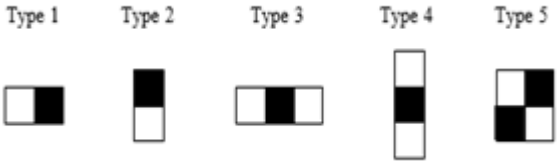


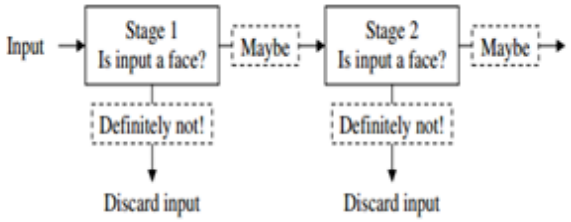
Fig-5: The different types of features

Each feature results in a single value which is calculated by subtracting the sum of the white rectangle(s) from the sum of the black rectangle(s).Viola-Jones have empirically found that a detector with a base resolution of 24*24 pixels gives satisfactory results. When allowing for all possible sizes and positions of the features in Figure4a. total of approximately 160.000 different features can then be constructed. Thus,the amount of possible features vastly out numbers the 576 pixels contained in the detector at base resolution. These features may seem overly simple to perform such an advanced task as face detection, but what the features lack in complexity they most certainly have in computational efficiency. One could understand the features as the computer's way of perceiving an input image. The hope being that some features will yield large values when on top of a face. Of course operations could also be carried out directly on the raw pixels, but the variation due to different pose and individual characteristics would be expected to hamper this approach. The goal is now to smartly construct a mesh of features capable of detecting faces.

2.2 The cascaded classifier

The basic principle of the Viola-Jones face detection algorithm is to scan the detector many times through the same image – each time with a new size. Even if an image should contain one or more faces it is obvious that an excessive large amount of the evaluated sub-windows would still be negatives (non-faces). This realization leads to a different formulation of the problem: Instead of finding faces, the algorithm should discard non-faces. The thought behind this statement is that it is faster to discard a non-face than to find a face. With this in mind a detector consisting of only one (strong) classifier suddenly seems inefficient since the evaluation time is constant no matter the input. Hence the need for a cascaded classifier arises. The cascaded classifier is composed of stages each containing a strong classifier. The job of each

stage is to determine whether a given sub-window is definitely not a face or maybe a face. When a sub-window is classified to be a non-face by a given stage it is immediately discarded. Conversely a sub-window classified as a maybe-face is passed on to the next stage in the cascade. It follows that the more stages a given sub-window passes, the higher the chance the sub-window actually contains a face. The concept is illustrated with two stages in Fig-7.



- Given examples images $(x_1,y_1),\ldots,(x_n,y_n)$ where y_1=0,1 for negative and positive examples.
- Initialize weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for y_1=0,1, where m and l are the numbers of positive and negative examples.
- For t=1,...,T:

1) Normalize the weights, $w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$

2) Select the best weak classifier with respect to the weighted error:

$$\varepsilon_t = \min_{f,p,\theta} \sum_i w_i \left| h(x_i, f, p, \theta) - y_i \right|$$

3) Define $h_t(x) = h(x, f_t, p_t, \theta_t)$ where f_t, p_t and θ_t are the minimizers of ε_t.

4) Update the weights:

$$w_{t+1,i} = w_{t,i} \beta^{1-e_i}$$

where $e_i = 0$ if example x_i is classified correctly and $e_i = 1$ otherwise, and $\beta_t = \frac{\varepsilon_t}{1-\varepsilon_t}$

- The final strong classifier is:

$$C(x) = \begin{cases} 1 \text{ if } \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 \text{ otherwise} \end{cases}$$

where $\alpha_t = \log \frac{1}{\beta_t}$

Fig-6: Cascade Classifier

In a single stage classifier one would normally accept false negatives in order to reduce the false positive rate. However, for the first stages in the staged classifier positives are not considered to be a problem since the succeeding stages are expected to sort them out. Therefore Viola-Jones prescribes the acceptance of many false positives in the initial stages. Consequently the amount of false negatives in the final staged classifier is expected to be very small. Viola-Jones also refers to the cascaded classifier as an attentional cascade.

2.3 Modified AdaBoost Algorithm

Generating positive examples

In order to train the different stages of the cascaded classifier the AdaBoost algorithm requires to be fed with positive examples – that is, images of faces. The faces used in this project were taken from two different databases:

FERET: The Facial Recognition Technology Database. A database administered and distributed by the American National Institute of Standards and Technology. Contains 2413 facial images taken under controlled conditions [6].

LFW: Labeled Faces in the Wild. A database made and distributed by The University of Massachusetts with the intend of studying the problem of unconstrained face recognition. Contains more than 13.000 facial images found on the internet. All faces in the database were detected by a Viola-Jones face detector.

A Matlab script called "algo01.m" capable of conditioning the database images into positive examples was constructed. The script does the following:

1. Opens a facial image and transforms to grayscale if needed.
2. Displays the image and lets the user place a bounding box around the face.
3. Rescales the face to 24*24 pixels.
4. Saves the rescaled face as an image and as a variance normalized data file.

The variance normalization is suggested by Viola-Jones as a mean of reducing the effect of different lighting conditions. In Fig-7 an image from the LFW database is shown after the user has defined the face

Fig-7: Generating Positive Example

A total of 5000 positive examples should be sufficient to train the individual stages of the staged classifier, but these examples should be chosen with care. The final detector will only detect faces similar to those in the training set so the training set should represent the most common facial variation. For the first stage in the cascade Viola-Jones advice constructing a simple pre-screening filters containing only two features. Since these two features are the first in the entire cascade they encode the most fundamental differences between the faces in the training set and the non-faces used for training. Consequently the two features also make it possible to draw conclusions regarding primarily the positive training set and secondly also the negative training set. For training of the cascade a total of three slightly different datasets were constructed using the above-mentioned databases. Together with the same one negative dataset the three datasets were used to train three different first stages. In the following the three data sets are presented together with a graphical representation of their respective two first features:

Dataset-A

Contains 5500 positive examples.No.1–1343 are taken from FERET. No.1344–5500 is taken from LFW.

From the FERET database only the frontal upright headshots were used.

The positive examples were made by having a user place a point just above the eye brows and between the mouth and the chin. These two points were used to create a rectangle which was enlarged by 50%. This is what is shown in Figure7.

The first two features are shown in Fig-8.

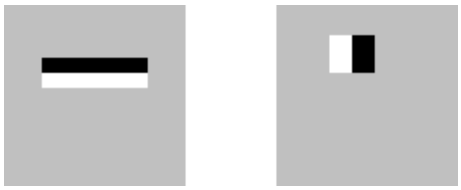


Fig-8: The first two features for dataset A.

Dataset-B

Contains 5234 positive examples. No.1–1243 is taken from FERET. No.1244–5234 is the 'goodones' from LFW.

In this training set only frontal upright headshots from both databases were used. The user placed a point between the eyes and a point just below the chin. The mark below the chin was the bottom line in a square box where the eyes were 1/3 from the top. The first two features are shown in Figure9.

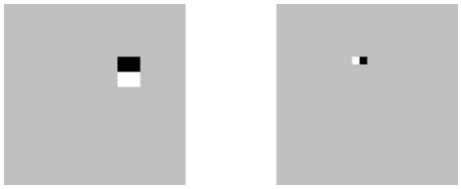


Fig-9: The first two features for datasetB.

Dataset-C

Contains 6000 positive examples. No.1–6000 are randomly taken from LFW. The examples were generated as in datasetA. The first two features are shown in Figure10.

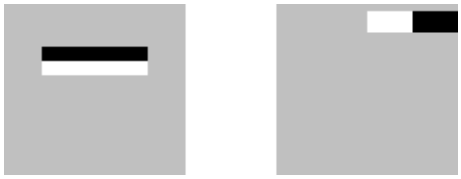


Fig-10: The first two features for data set C.

The three figures shown above allow for some conclusions to be made. Most noticeable is that set A and C generate the same first feature. This feature represents the fact that the eye region is generally darker than the (light) upper cheek region on the training subjects. Also, the relatively large width of the feature is explained by variations in face sizes and proportions. This variation is suspected to stem primarily from the LFW images. The light bridge of the nose and the dark region of the training subjects 'lefteye. For dataset B two plausible explanations for the size and position of these cond features arise. The first observes that the position and type of the feature is similar to these cond features in dataset-A. Furthermore the smaller size should be attributed to the very homogeneous nature of the dataset. That is, the position of the light bridge of the nose and the dark eye region are almost constant in the examples. These cond explanations take as a starting point that a feature at minimum size (1*2 pixels) doesn't encode much meaningful information. As a result of perhaps a very good discrimination by the first feature it should then be seen as a coincidence that these cond features becomes so small. Lastly these cond features of datasetC represent the fact that here usually is a contrast between the training subjects 'light forehead and the generally darker background. For the training of the succeeding stages dataset-A is chosen. This dataset seems to offer a good compromise between being too specific–like dataset-B and being too 'loose'–like set C. In addition dataset-A is also the only one where both features encode prominent facial characteristics. Of the 5500 positive examples contained in the dataset 5000 are used for training and the last 500as evaluation set.

The Matlab script that generated dataset-A can be found on the enclosed DVD. The size of the first feature in dataset-B is in stark contrast to size of the first feature in set A and C. A very probable reason is that set B is the most homogeneous set and apparently can be discriminated from the negative training set utilizing a small feature representing the left eye of the training subjects .Looking at these cond feature for dataset A the conclusion is that it represents a contrast between

2.4 Training a stage in the cascade

In the cascade a stage is actually just a strong classifier trained by the modified AdaBoost algorithm. During the course of this project two major different implementations of the modified AdaBoost algorithm were constructed. The sequential version made as a mex function. Programmed in C. This is more or less a straight forward implementation of the modified AdaBoost algorithm described in Figure-6, Figure-6. The first version loaded the examples from the hard disk drive. This proved to be very inefficient because of the many times each example has to be evaluated. The second version uploaded all examples to memory and instantly proved more efficient albeit more memory consuming. As described earlier the best performing feature is determined by a brute force search through all possible features. This approach requires a sorting of all examples for every feature evaluated. This sorting was initially done by a classic bubble sort, but was soon replaced by a build in quick sort, which also contributed to a faster execution. The parallel version made as a Matlab script that calls four mex functions. The mex functions are programmed in C. In the sequential version a for-loop runs through all features and for each feature all examples are evaluated. Since the iterations in the for-loop are independent of each other the algorithm can be easily parallelized. This was done using the Matlab R2006b Parallel Computing Toolbox. The sequential version was executed on a Zepto laptop containing 1 GB RAM and a 2 GHz Intel Centrino Dothan CPU. The parallel version was executed on IMM's Linux cluster consisting of 30+ servers each running a 64 bit version of Linux on a 1 GHz dual core AMD CPU. Recorded performance results are listed in Table 2.

Table-2: Training Performance

System	Features $[F]$	Positives $[P]$	Negatives $[N]$	Total time $[T]$	Parallel jobs $[J]$	$\frac{T}{F\cdot(P+N)}$	$\frac{JT}{F\cdot(P+N)}$
Zepto	3	5000	23560	13334 s	1	0.1556	0.1556
Zepto	10	5000	20160	39143 s	1	0.1556	0.1556
Zepto	10	5000	23829	45121 s	1	0.1565	0.1565
Zepto	10	5000	23829	45466 s	1	0.1577	0.1577
Cluster	10	5000	20627	4464 s	16	0.0174	0.2787
Cluster	100	5000	5185	24999 s	16	0.0245	0.3927
Cluster	140	5000	5892	28970 s	18	0.0190	0.3420
Cluster	260	5000	5028	56700 s	18	0.0217	0.3914
Cluster	320	5000	5436	62025 s	18	0.0186	0.3343

3. APPLICATIONS

Facial Recognition & Identity Resolution

Fig-11: Military Application

Military, Intelligence and Homeland Security, Advanced 3D Facial Recognition and Identity Verification. Mobile and Video Identity Resolution and Surveillance Applications

Centralized and Secure Facial Identity Management Solution

3.1 Investigative Face Recognition & Analysis

Fig-12: Moving vehicle- recognition

Law Enforcement, Private Security and Criminal Investigations. Powerful 2D to 3D transformation of video or photo facial images and fast, accurate and analytical comparison of multiple facial images for precise facial recognition and criminal screening.

3.2 Gaming

Image and face recognition is bringing a whole new dimension to gaming. Microsoft's Kinect's advanced motion sensing capabilities have given the Xbox 360 a whole new lease of life and opened up gaming to new audiences by completely doing away with hardware controllers. Meanwhile, startup Viewed recently launched a game that uses face recognition to decide whether you're a human or vampire, setting the stage for a battle between the two species. We're sure to see many more examples face recognition in games in the future too with all kinds of interesting possibilities.

3.3 Image Search

Google recently introduced the ability to search for images by comparing them to others. By uploading an image or giving Google an image URL, it will show you where that image is used on the Web, and display similar images too. The ability to search for matching images is a boon for photographers looking to check where their images have been used. It's also great for checking if an image is what it said it was. During the recent London riots, a rumor spread on Twitter that a tiger had been set loose from London zoo. Searching using Google's tool revealed that the supposed photographic evidence came from a 2008 tiger escape in Italy and the streets of London were free of wild animals (well, of the non-human variety, at least).

3.4 Security

Face recognition could one day replace password logins on our favorite apps – imagine logging in to twitter with your face

Fig-13: Image processing in security

4. CHALLENGES

4.1 Image Quality

The primary requirement of face recognition system is suspect's good quality face image and a good quality image is one which is collected under expected conditions. For extracting the image features the image quality is important. Without the accurate computations of facial features the robustness of the approaches will also be lost. Thus even the best recognition algorithms deteriorate as the quality of the image declines.

4.2 OTHERS

Sliding window detector must evaluate tens of thousands of location/scale combinations Faces are rare: 0–10 per image. For computational efficiency, we should try to spend as little time as possible on the non-face windows. A megapixel image has ~106 pixels and a comparable number of candidate face locations

5. CONCLUSION

Paul Viola and Michael Jones presented an approach for object detection which minimizes computation time while achieving high detection accuracy. The approach was used to construct a face detection system which is approximately 15 faster than any previous approach. Preliminary experiments, which will be described elsewhere, show that highly efficient detectors for other objects, such as pedestrians, can also be constructed in this way.

6. REFERENCES

[1] Manisha V. Borkar, Bhakti Kurhade, A Research – Face Recognition by Using Near Set Theory, International Journal of Advanced Research in Computer Science and Software Engineering, Volume 5, Issue 4, 2015 ISSN: 2277 128X.

[2] Asit Kumar Datta, Madhura Datta, Pradipta Kumar Banerjee, Face Detection and Recognition: Theory and Practice, CRC Press, 28-Oct-2015.

[3] White paper by Animetrics, Inc, Facial Recognition & Identity Resolution, 2012.

[4] Paul Viola, Michael J. Jones, Robust Real-Time Face Detection, International Journal of Computer Vision 57(2), 2004.

[5] Paul Viola, Michael J. Jones, Fast Multi-view Face Detection, Mitsubishi Electric Research Laboratories, TR2003-096, August 2003.

[6] Henry A. Rowley, Shumeet Baluja, Takeo Kanade, Neural Network-Based Face Detection, IEEE Transactions on Pattern Analysis and Machine Intelligence, Volume 20, Issue 1, Jan 1998. Digital Object Identifier 10.1109/34.655647.

[7] Henry Schneiderman, Take o Kanade,A Statistical Method for 3D Object Detection Applied To Faces and Cars, IEEE Conference on ComputerVision and PatternRecognition 2000 proceedings, Volume1.Digital Object Identifier 10.1109/CVPR.2000.855895.

[8] Christopher M.Bishop,Pattern Recognition and Machine Learning, first edition, Springer 2006.

[9] GaryB.Huang,Manu Ramesh,TamaraBerg,ErikLearned-Miller,LabeledFacesintheWild: A Database for Studying Face Recognition in Unconstrained Environments,University of Massachusetts,Amherst,TechnicalReport07-49,October2007.

[10] Brian W. Kernighan, Dennis M. Ritchie, The C Programming Language, second edition, PrenticeHallSoftwareSeries,1988