Eye Region Detection

Using facial landmark detection technique

Presented By :- Atul Sahay
Summer Research Intern (C.P.D.M) I.I.Sc.

Advisor : - Dr Pradipta Biswas
Assistant Professor (C.P.D.M) I.I.Sc.

Contents

BACKGROUND

Face detection is a computer technology being used in a variety of applications that identifies human faces in digital images. Face detection also refers to the psychological process by which humans locate and attend to faces in a visual scene.

- Face detection can be regarded as a specific case of object-class detection. In object-class detection, the task is to find the locations and sizes of all objects in an image that belong to a given class. Examples include pedestrians, and cars.
- Face-detection algorithms focus on the detection of frontal human faces. It is analogous to image detection in which the image of a person is matched bit by bit. Image matches with the image stores in database. Any facial feature changes in the database will invalidate the matching process.

PROPOSED METHODOLOGY

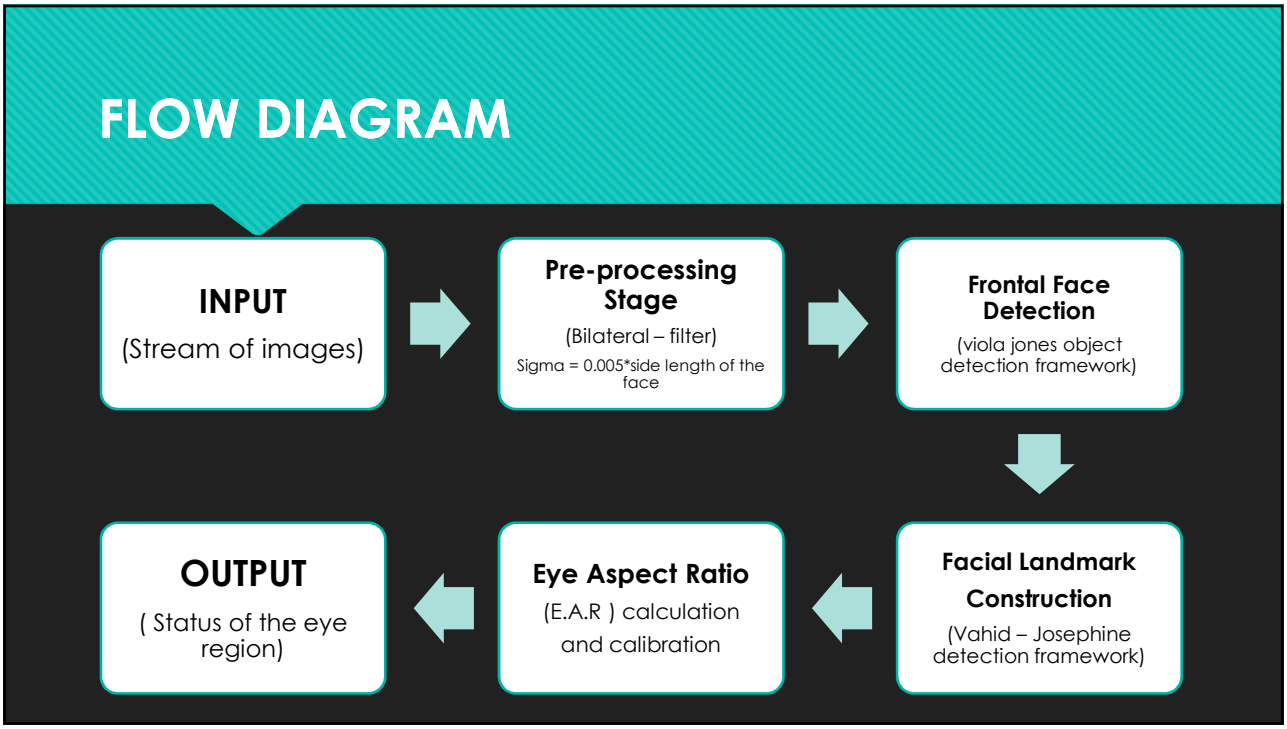
Image processing based eye region detection (in real time)

- using facial landmark detection technique

- Eye Region can be detected with this methodology in a real time, module can easily tell how much area is enclosed by the user , like whether it's wide open , normal ,slightly open or closed.
- In proposed methodology , a stream of images were taken from the real time video footage taken from the webcam under suitable light conditions.
- The Pre-processing step was made by applying the bilateral filter on the taken images stream to smooth the images without smoothing the sharp edges.
- The frontal face was then detected using the viola jones object detection framework.
- After the probable face detection process , next we did the facial landmark construction using the method of ensemble of regression tree proposed by Vahid – Josephine
- Now , eye aspect ratio is then calculated and calibrated to the particular user to predict the state of the eye region

viz., Wide ,normal ,slight open or closed .

FLOW DIAGRAM



PRE – PROCESSING STAGE

Filtering

- Filtering is perhaps the most fundamental operation of image processing and computer vision. In the broadest sense of the term "filtering", the value of the filtered image at a given location is a function of the values of the input image in a small neighbourhood of the same location.

For our case we used a bilateral filter [2]

- **Bilateral Filter [2]**
  - A **bilateral filter** is a non-linear, edge-preserving and noise-reducing smoothing filter for images.
  - The intensity value at each pixel in an image is replaced by a weighted average of intensity values from nearby pixels. This weight can be based on a Gaussian distribution.
  - Crucially, the weights depend not only on Euclidean distance of pixels, but also on the radiometric differences (e.g. range differences, such as colour intensity, depth distance, etc.).
  - This preserves sharp edges by systematically looping through each pixel and adjusting weights to the adjacent pixels accordingly

VIOLA - JONES OBJECT DETECTION ALGORITHM

The **Viola–Jones object detection framework[1]** is the first object detection framework[2] to provide competitive object detection rates in real-time proposed in 2001 by Paul Viola and Michael Jones. Although it can be trained to detect a variety of object classes, it was motivated primarily by the problem of face detection.

The Process of Viola –jones algorithm has four stages:

- Haar Feature Selection
- Creating an Integral Image
- Adaboost Training
- Cascading Classifiers

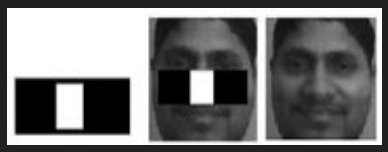
HAAR LIKE FEATURES

- **Haar-like features** are digital image features used in object recognition. They owe their name to their intuitive similarity with Haar wavelets and were used in the first real-time face detector.
- Haar Features – All human faces share some similar properties. These regularities may be matched using **Haar Features**.

- **A few properties common to human faces:**
  - The eye region is darker than the upper-cheeks.
  - The nose bridge region is brighter than the eyes.
  - Composition of properties forming matchable facial features:
  - Location and size: eyes, mouth, bridge of nose
  - Value: oriented gradients of pixel intensities

CONTINUED

- The four features matched by this algorithm are then sought in the image of a face Rectangle features:
  - Value = Σ (pixels in black area) - Σ (pixels in white area)
  - Three types: two-, three-, four-rectangles, Viola & Jones used two-rectangle features
  - For example: the difference in brightness between the white &black rectangles over a specific area
  - Each feature is related to a special location in the sub-window



Three rectangle haar like feature

Two rectangle haar like feature

INTEGRAL IMAGE

- **Given that the base resolution of the detector is 24×24, the exhaustive set of features is quite large, 160,000.**
- **A very efficient way to compute them is needed ->** **integral image**

INTEGRAL IMAGE –

- The integral image at location x , y contains the sum of the pixels above and to the left of x , y, inclusive:

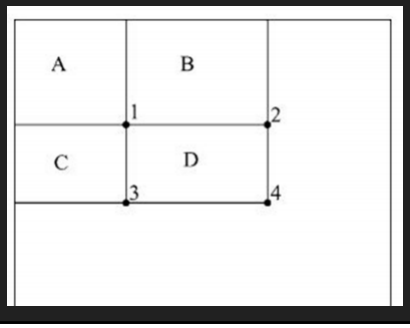
$$ii(x, y) = \sum_{x' \le x, y' \le y} i(x', y'),$$

- Using the following pair of recurrences:
  - **S(x,y) = S(x,y-1) + i(x,y)**
  - **ii(x,y) = ii(x,y) + s(x-1,y)**
  - where s(x , y) is the cumulative row sum, the integral image can be computed in one pass over the original image.

CONTINUE

- The sum of the pixels within rectangle D can be computed with four array references:
  - The value of the integral image at location 1 is the sum of the pixels in rectangle A. The value at location 2 is A + B, at location 3 is A + C, and at location 4 is A + B + C + D.
  - The sum within D can be computed as 4 + 1 − (2 + 3).

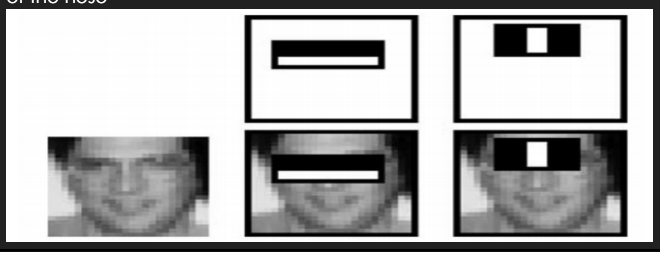
Figure demonstrating the use integral Image →



ADABOOST TRAINING

- A variant of the learning algorithm AdaBoost is used to both select the best features and to train classifiers that use them.
- This algorithm constructs a "strong" classifier as a linear combination of weighted simple "weak" classifiers.
- The first two features selected by AdaBoost for the task of face detection are easily interpreted:
  - First feature: the region of the eyes is often darker than the region of the nose and cheeks
  - Second feature: the eyes are darker than the bridge of the nose

Figure demonstrating the feature selection →



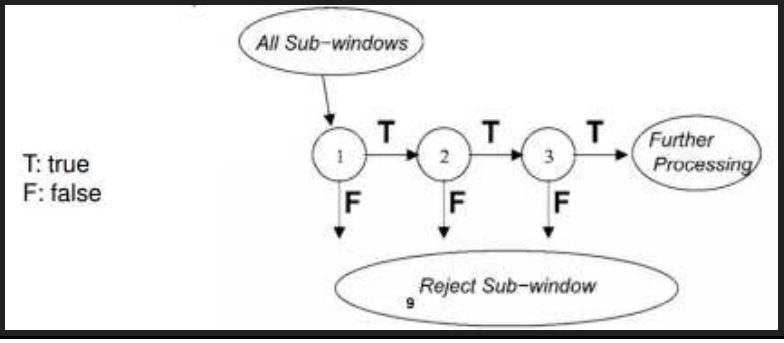
BOOSTING ALGORITHM

- Given example images $(x_1, y_1), \ldots, (x_n, y_n)$ where $y_i = 0, 1$ for negative and positive examples respectively.
- Initialize weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for $y_i = 0, 1$ respectively, where m and l are the number of negatives and positives respectively.
- For $t = 1, \ldots, T$:
  1. Normalize the weights,
  $$w_{t,i} \leftarrow \frac{w_{t,i}}{\sum_{j=1}^{n} w_{t,j}}$$
  so that w_t is a probability distribution.
  2. For each feature, j, train a classifier h_j which is restricted to using a single feature. The error is evaluated with respect to w_t, $\epsilon_j = \sum_i w_i |h_j(x_i) - y_i|$.
  3. Choose the classifier, h_t, with the lowest error ϵ_t.
  4. Update the weights:
  $$w_{t+1,i} = w_{t,i} \beta_t^{1-e_i}$$
  where $e_i = 0$ if example x_i is classified correctly, $e_i = 1$ otherwise, and $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$.
- The final strong classifier is:
$$h(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \\ 0 & \text{otherwise} \end{cases}$$
where $\alpha_t = \log \frac{1}{\beta_t}$

CASCADED CLASSIFIER

- Smaller, and therefore more efficient, classifiers can be constructed which reject many of the negative sub windows while detecting almost all positive instances:
  - simpler classifiers are used to reject the majority of sub windows
  - then, more complex classifiers are called upon to achieve low false positive rates.

Figure demonstrating the cascading of classifier →



FACIAL LANDMARK CONSTRUCTION

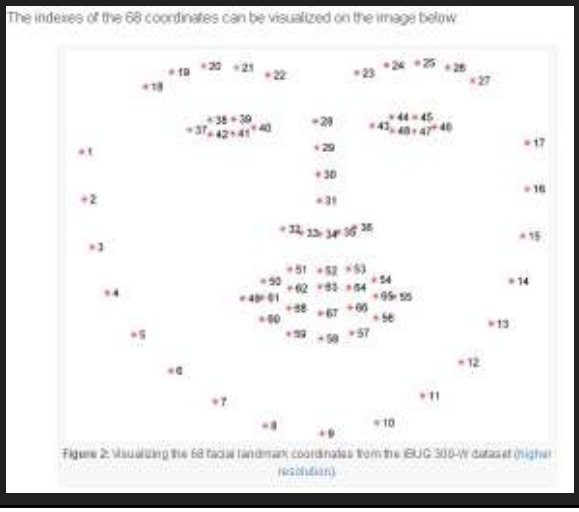
- Facial landmarks can be used to align facial images to a mean face shape, so that after alignment the location of facial landmarks in all images is approximately the same.
- The pre-trained facial landmark detector inside the dlib library is used to estimate the location of ***68 (x, y)-coordinates*** that map to facial structures on the face.
- implementation is based on excellent paper from Computer Vision and Pattern Recognition Conference: "one Millisecond Face Alignment with an Ensemble of Regression Trees by Vahid Kazemi and Josephine Sullivan"[3]
- Basically , the predictor is implemented using the ensemble of regression tree , iterative cascading is used to deal with the estimate of the shape and to extract reliable features.
- iBUG 300-W dataset is used to train the predictor .

CONTINUED

Figure demonstrating the facial landmarks[4] →

Python Script to take on useful landmarks :-

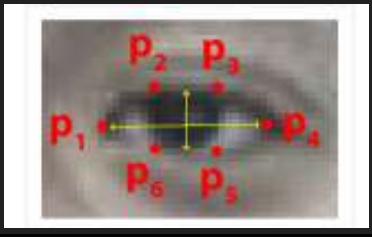
```
JawLine_Points = list(range(0,17))
RightEyebrow_Points = list(range(17,22))
LeftEyebrow_Points = list(range(22,27))
Nose_Points = list(range(27,36))
Right_Eye_Points = list(range(36,42))
Left_Eye_Points = list(range(42,48))
Outline_Mouth_Points = list(range(48,61))
Inner_Mouth_POints = list(range(61,68))
```



EYE ASPECT RATIO

- Using the facial landmark feature we can show that precisely enough reliable to estimate the level of the eye opening.
- For this we'll be computing a metric called the *eye aspect ratio* (EAR), introduced by Soukupová and Čech in their 2016 paper[5], *Real-Time Eye Blink Detection Using Facial Landmarks*.
- Each eye is represented by 6 *(x, y)*-coordinates, starting at the left-corner of the eye (as if you were looking at the person), and then working clockwise around the remainder of the region:

Figure showing the landmarks we constructed →



CONTINUED

$$\text{EAR} = \frac{\|p_2 - p_6\| + \|p_3 - p_5\|}{2\|p_1 - p_4\|}$$

Figure 4: The eye aspect ratio equation

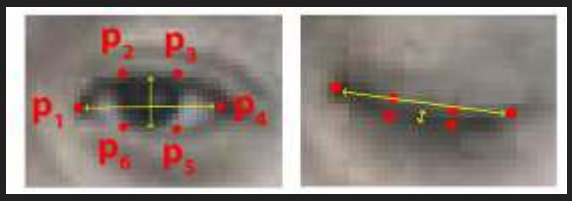
Where *p1, …, p6* are 2D facial landmark locations[6]

→ The numerator of this equation computes the distance between the vertical eye landmarks while the denominator computes the distance between horizontal eye landmarks, weighting the denominator appropriately since there is only *one* set of horizontal points but *two* sets of vertical points.

→ The scale is (0,1)

Well, as we'll find out, the eye aspect ratio is approximately constant while the eye is open, but will rapidly fall to zero it is closed.

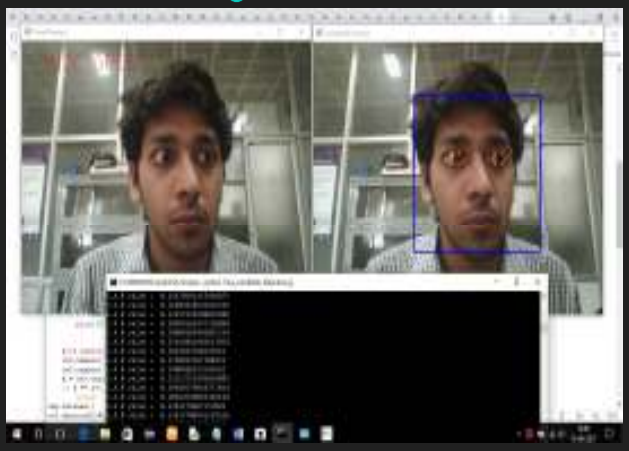
CONTINUED

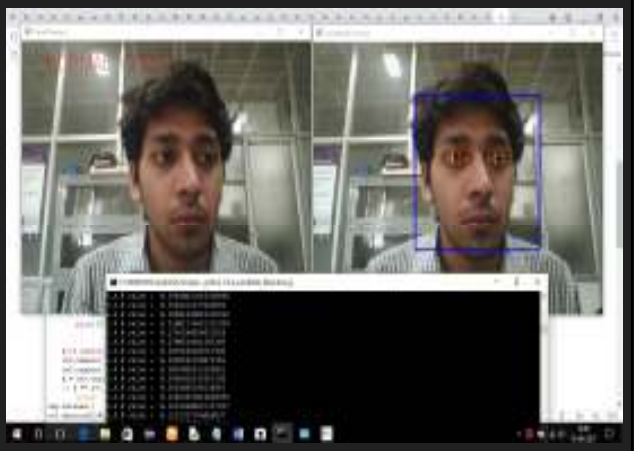


→ Using the calibration, for an individual we can predict how much his eyes are open .

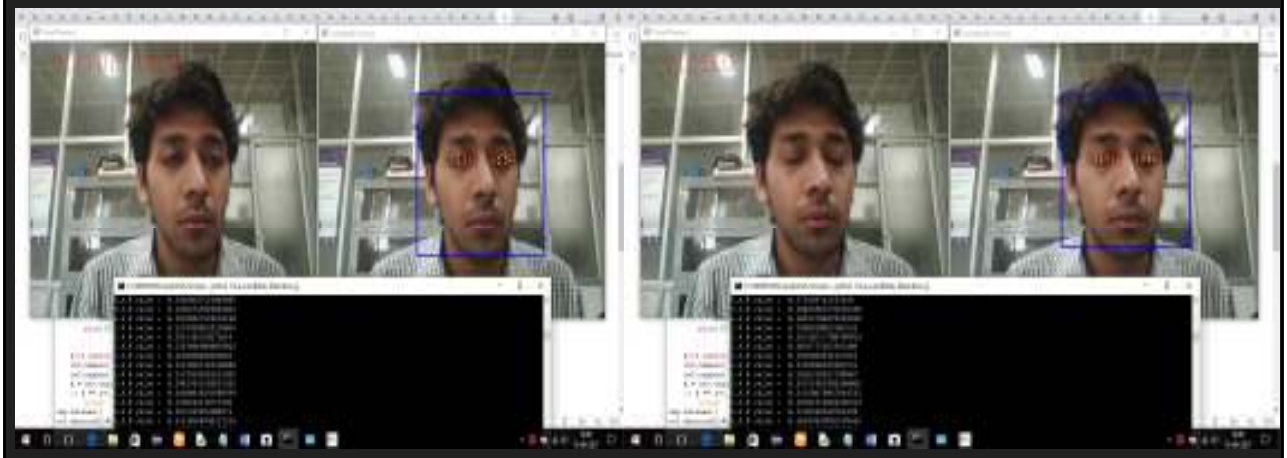
→ Because of constant time calculation , complexity is very much reduced.

FINAL OUTPUT





FINAL OUTPUT



TIME COMPLEXITY ANALYSIS

- Time Complexity Analysis :-
  - Among all the steps ,face detection step has the highest complexity.
  - Complexity of it can be understand in following ways :-
    - For each round of boosting:
      - Evaluate each rectangle filter on each example
      - Select best threshold for each filter.
      - Select best filter/threshold combination .
      - Reweight examples .

Computational complexity of learning: O(MNT) →
M filters, N examples, T thresholds

CONTINUED

- 38 layers, total of 6061 features
- Average of 10 features evaluated per window on test set [7]
- So for window of 640 x 480 , and 2.4 GHz Intel Core i5-4210U with Intel HD Graphics 4400
- The face detector can process a 640 X 480 pixel image in about .037575 seconds"
- So approximately → 27 Hz, or 27 samples per second.

REFERENCES

[1] Rapid object detection using a boosted cascade of simple features

[2] Jump up ^ Viola, Jones: Robust Real-time Object Detection, IJCV 2001 See pages 1,3.

[3] One Millisecond Face Alignment with an Ensemble of Regression Trees
Vahid Kazemi and Josephine Sullivan
KTH, Royal Institute of Technology
Computer Vision and Active Perception Lab

[4] https://pdfs.semanticscholar.org/d78b/6a5b0dcaa81b1faea5fb0000045a62513567.pdf

[5] http://blog.dlib.net/2014/08/real-time-face-pose-estimation.html

[6] https://vision.fe.uni-lj.si/cvww2016/proceedings/papers/05.pdf

[7] http://cs.nyu.edu/courses/fall12/CSCI-GA.2560-001/FaceRecognitionBoosting.pdf