12/12/2015

Computer Vision
Short Project

Design and implementation of a real-time face tracking technique.

Maher NADAR Dimitri LECA

Table of contents

i. Abstract

By merging facial detection process along with feature detection and point tracking techniques, this paper presents a simple yet respectable algorithm that can track multiple human faces in real time. First, we start by scanning the initial frame in order to detect the different present faces and bound them in a rectangular box. Next, we perform corners detection within the cropped image of the detected faces via the 'Shi and Tomasi' logic and initialise these detected features as the tracking reference. Having this at hand, we can track these points in the subsequent video frame and calculate the transformation that link the reference tracked points in both the previous and current frame. Finally, we use this transformation matrix in order to make the bounding boxes shadow the movement of their respective faces.

1. Introduction

Faces analysis in images has been, for the past decade, a major point of interest in the computer vision research field around the world. Explorations in this topic are mainly drawn from incentives such as surveillance monitoring, security assistance, cutting-edge human-machine interaction and virtual reality applications. Some examples of research paths in this general field include facial detection, recognition and tracking, face animation, lip reading and facial expressions identification. In this short project, we will be talking about multiple-face detection on MatLAB program, and further exploring the possibility to track them throughout the video frames.

The proposed algorithm starts initially to skim through the starting frames until it detects camera-facing faces using of the Viola-Jones algorithm, after which the 'Shi and Tomasi' features are retrieved and applied as point tracker initialization. We attempt to follow the reference tracking points to the following frame and, if indeed detected there, retrieve the transformation matrix which describes their path (i.e. rotation and/or translation). With this transformation in hand, we can apply it to the previous frame's bounding box and force it to tail the tracked face. Finally, we update the points to be tracked for the next frame (3rd frame mentioned here), and we repeat the process until the end of the displayed video.

Due to the fact that the Viola Jones procedure requires both eyes to be clearly present, our proposed algorithm needs the face to be facing the camera and without glasses that may hinder the appearance of the eyes. Hence, the tracking of the face only commences when these conditions are satisfied.

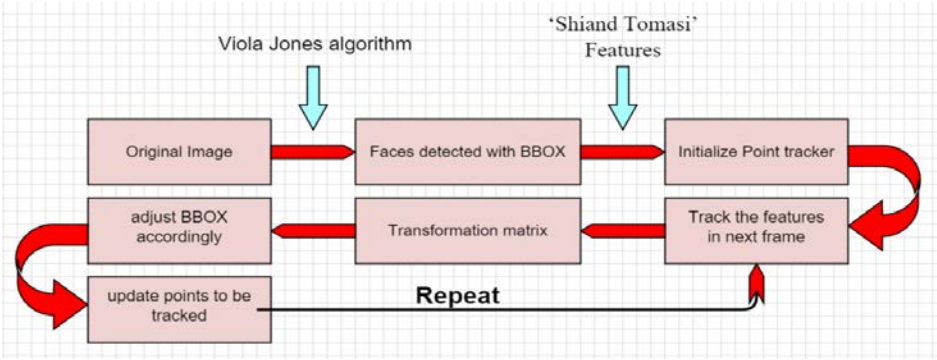


Figure 1. Block diagram of face tracking algorithm

2. State of the Art

2.1 Face detection

The task in this part is merely concerned with the localization of a human face within a certain image. It is a challenging problem to solve, mainly due to changes in the background, lighting, occlusion of the face, and the figuring of a face like object in the image in question. The following pictures, although taken to the extreme, may illustrate the difficulty of finding a face inside an image, or even avoid false positives.

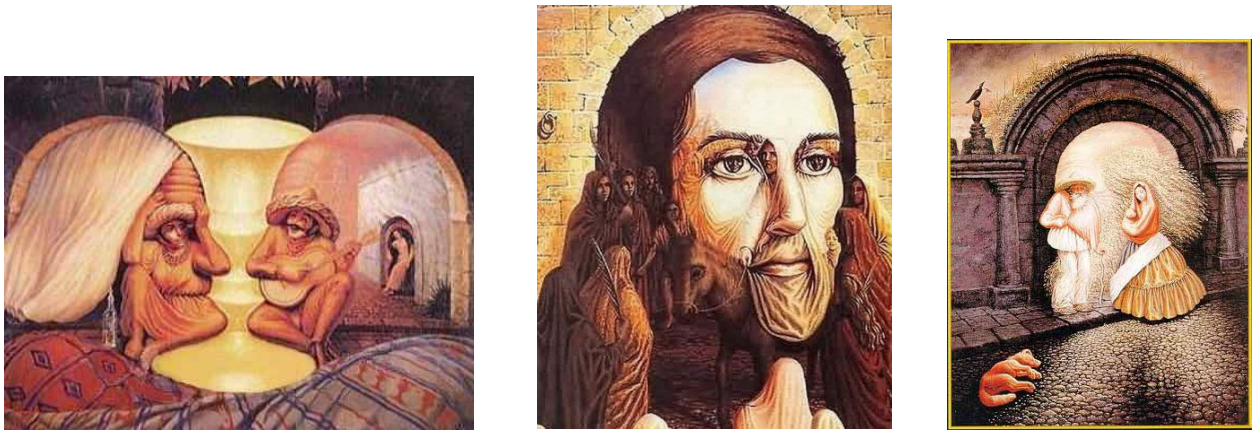


Figure 2: Difficulty of Face Detection

Although the Viola Jones technique (employed in our code), constructed with Haar-like features and combinations of weak classifiers, has a comparatively high level of robustness, we are going to mention a few other methods previously tried.
The '**Colour based'** method deals with the **colour histograms** of the images in order to learn the colour of the skin and use this data to differentiate the faces from the non-faces. The main drawbacks here are that it only applies to coloured images and that some skin colours are poorly detected. The '**Edge based'** edge detection such as the **Canny** type to find faces. Although it can be hard to use by its own when the background is complex, background subtraction may help. Other detection techniques are '**Features based'**. Some of them use facial features such as eyes, nose and lips, but have been found to be weak due to the variety of facial appearances between different people. The most successful ones are the person independent ones such as the Haar-like features.

2.2 Face tracking

The main difficulties in this part of the project is the faces that are tracked may change pose and appearance with time, with the addition to facing temporary partial or whole obstruction and passing through a background (or foreground) with face-like characteristics.

Figure 3: Difficulty of Face Tracking

In order to deal with '**Illumination variations'** with time, one method uses **histogram equalization enhancement** in every frame. Another method takes into account the illumination variation by altering the brightness constancy of the **optical flow.** In order to solve '**Occlusion constraints',** one approach suggests computing the **mean face** of the tracked person. A more elaborate methodology is to generate a **Kalman filter** in order to predict the location of the face in a certain frame based on its behaviour in the previous frames.

3. Face detection: Viola Jones approach

As mentioned earlier, we implemented the Viola Jones face detector in our code, since we found it to be relatively robust and fast to implement. In short, the idea is to subtract the sum of black pixels (in the dark rectangles of the features) from the sum of white pixels (in the light rectangles), which would aid us in detecting target facial texture variations, edges, shade variations and others.

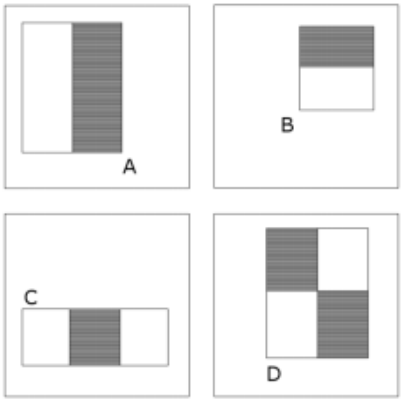
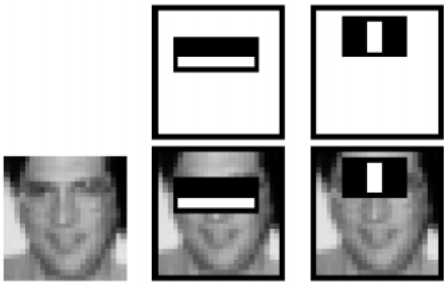


Figure 3: example of Haar-like features



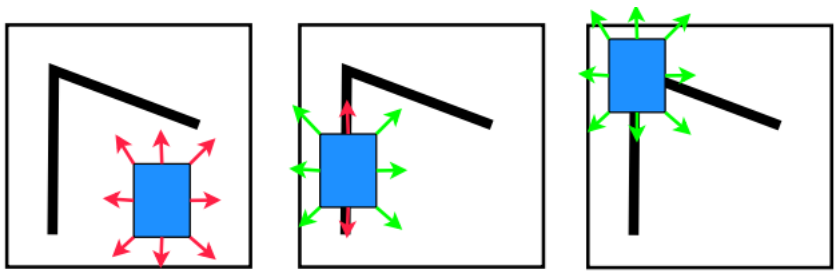
$$f(x,y) = \sum_i p_b(i) - \sum_i p_w(i)$$

Figure 4: best 2 Haar-like features chosen by the algorithm

The following Matlab embedded function that uses this approach in order to detect the faces in the concerned frame of our video, with the addition of generating a bounding box around each face:

4. Feature selection: Shi and Tomasi Approach

As it turns out, the best (and most unique) points to track in a certain image are the detected corners. In Figure 5, one can observe why this is the case, by perceiving that these special points are the ones with significant changes in all directions.



(a) Flat (b) Edge (c) Corner

Figure 5: differences in behaviour of pixels derivatives

The basic idea behind this methodology is to apply a special 'autocorrelation matrix' on the Hessian (Figure 6) of the image over a window around every point and study the eigenvalues of the obtained matrix. In fact, it was found that comparing the smallest of the 2 eigenvalues λ1 and λ2 to a pre-set threshold, we can directly detect if the point in question is a corner or not.

$$H(f) = \begin{bmatrix} \frac{\partial^2 I}{\partial x^2} & \frac{\partial^2 I}{\partial x \partial y} \\ \frac{\partial^2 I}{\partial x \partial y} & \frac{\partial^2 I}{\partial y^2} \end{bmatrix} \qquad M(x,y) = \begin{bmatrix} \sum_{-K \le i,j \ge K} w_{i,j} I_x^2 & \sum_{-K \le i,j \ge K} w_{i,j} I_x I_y \\ \sum_{-K \le i,j \ge K} w_{i,j} I_x I_y & \sum_{-K \le i,j \ge K} w_{i,j} I_y^2 \end{bmatrix}$$

Figure 6: a) Hessian matrix; b) Autocorrelation matrix

The following Matlab embedded function that uses this approach in order to retrieve the Shi and Tomasi Features within the BBox of the multiple tracked faces:

5. Face Tracking code overview and script fragments + example Frame

INITIALISATION

→ *In the first frame, detect all the faces, and return the bounding boxes*

```
faceDetector = vision.CascadeObjectDetector();
bbox = step(faceDetector, videoFrame);
```

→ *For each bounding boxes, detect the initial markers sets inside*

```
cornerDetector=vision.CornerDetector('Method','Minimum eigenvale(Shi&Tomasi)');
points = step(cornerDetector, rgb2gray(imcrop(videoFrame, bboxn)));
```

→ *Initialize the PointTracker in Matlab*

```
initialize(PointTracker, double(points), rgb2gray(videoFrame));
```

FOR EACH FRAME N

FOR EACH BOUNDING BOX IN THE INITIAL FRAME

→ *Compute all the markers inside and around this bounding box.*

→ *Match these markers with the markers of the frame N-1*

```
[points, isFound] = step(PointTracker, rgb2gray(videoFrame));
```

→ *Compute the geometrical transformation from the set of markers in the N-1 frame to the set of markers in the N frame.*

```
[xform, geometricInlierIdx] = step(geometricTransformEstimator,
double(oldInliers), double(visiblePoints));
```

→ *Apply this transformation to the current bounding box.*

```
boxPoints = boxPoints * xform;
bboxPolygon_n = reshape(boxPoints', 1, numel(boxPoints));
```

→ *Remove all the markers with are not matching with the previous ones.*

```
visiblePoints = visiblePoints(geometricInlierIdx, :);
oldInliers = oldInliers(geometricInlierIdx, :);
```

→ *Update the markers for the next iteration.*

```
setPoints(PointTracker, abs(oldPoints));
cell_oldPoints{n} = oldPoints;
```

END FOR

END FOR

Frame Example:

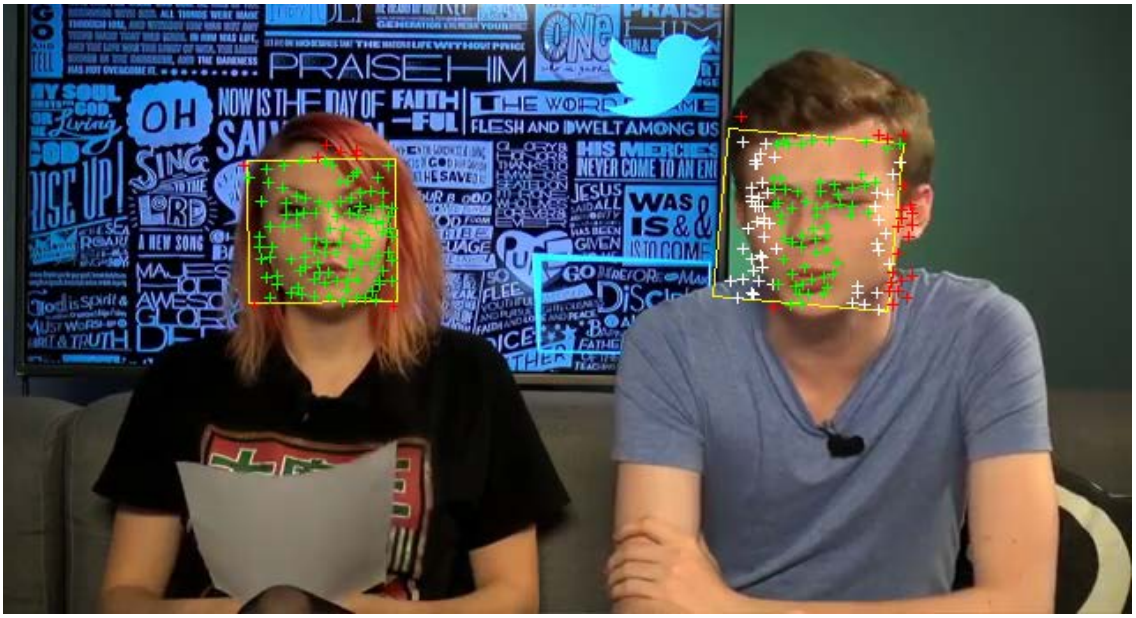


Figure 7 example frame that shows the different types of features in frame N

In Figure 7, we can see how our algorithm works:

- **Green points** are markers detected in the current frame, and which match with markers in the previous frame.
- **White points** are markers detected in the current frame, but not found in the previous frame. In this part of the video, the man is turning his head.
- **Red points** are markers which are outside the bounding boxes.

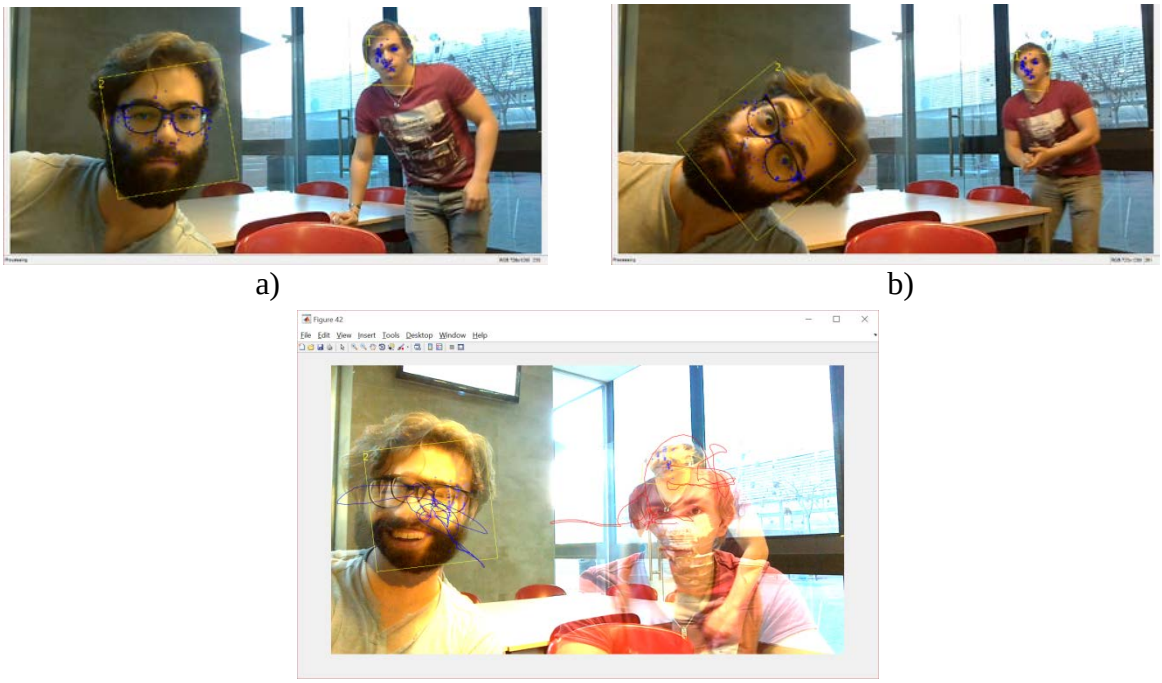
6. Experimental Results

- Single face Tracking with only tilting of the Face

Figure 8: Single Face Tracking; Face always facing Camera

In Figure 8, the tracking faces no major challenges since the face is always directed towards the camera, which keeps allot of tracking points apparent in the subsequent video frames.

- 2-Face Tracking with more significant movements and turning of the neck



a) b)

c)

Figure 8: a) and b) show frames in mid video with different poses. c) Initial and final frames superposed along with trajectories of the Bounding Boxes of the faces

- Multiple face tracking with a variety of movements

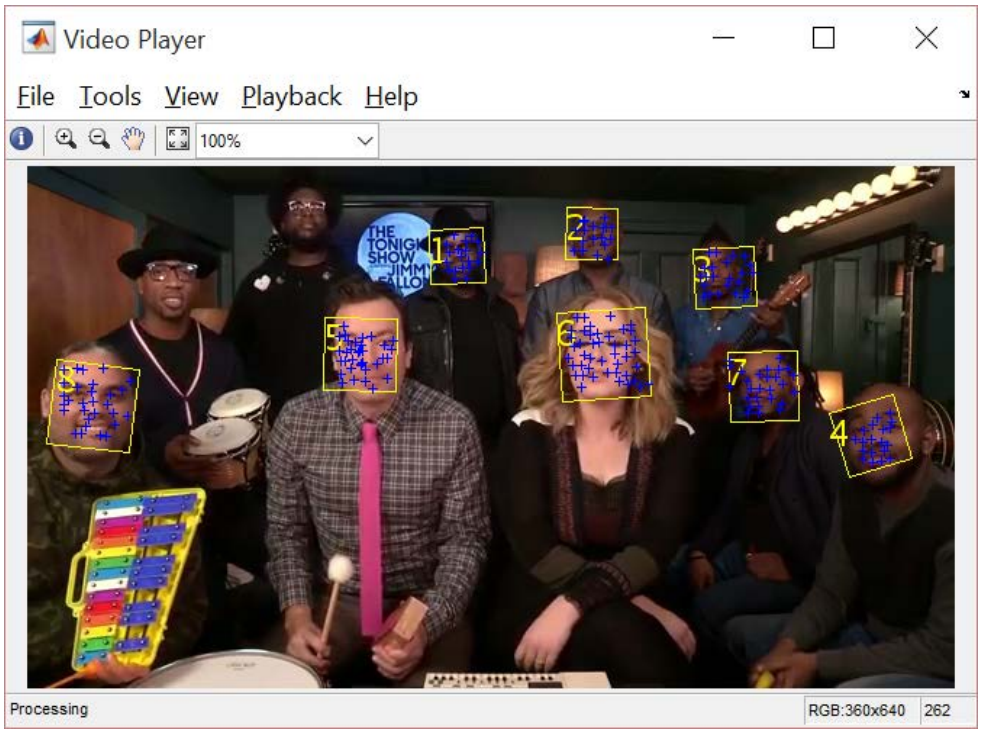


Figure 9: Multiple face tracking with respectable results

In Figure 9, we can see that the tracking is working fairly well, with the exception of the two men on the top left. This is mainly due to the fact that they are wearing shaded glasses and a hat, which may have hindered the performance of the face detection Viola Jones Algorithm.

7. Limitations

Several limitations are faced using our algorithm: Other than the fact that the **Viola Jones** needs **particular face orientation** to detect the face, the major limitation to our algorithm is the encountering of the face with a **background or foreground that has a face or a skin like feature.** For instance, as you can see in Figure10, when the girl tries to fix her hair using one of her arms, her face tracking is lost. Another example is shown in Figure 11, where president Obama, walking by another person in the frame (both faces meet), both faces' tracking are distorted. **Total occlusions** may also distort the tracking process if occurring for a relatively long time.

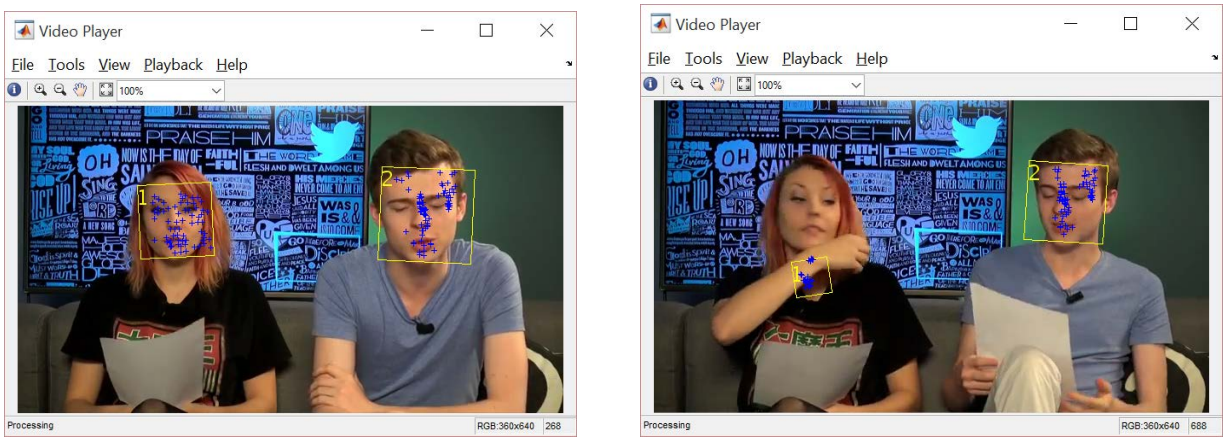


Figure 10: arm passing in front of the face distorts the tracking

Figure 11: when faces meet, the tracking is distorted

8. Conclusion

In this Computer Vision short project, we discussed a technique of real time face tracking using the Viola Jones face detection algorithm along with Shi-Tomasi Feature detection method in order t extract the interest points that would be tracked from to frame. We tested our algorithm on several video sequences, which showed respectable multiple-face tracking with face tilting, neck turning and abrupt movement. Nonetheless, problems arose sometimes due to the face detection, where false positives emerged in some cases, and faces would not be detected because when satisfying the special face poses that need to figure in order for the Viola Jones algorithm to properly spot it. Moreover, with major occlusions, the tracking of the faces would stop due to loss of features.

A further continuation of our work might consist of using a Kalman filter that would aid in predicting the faces locations in the frames based on the data acquired by the faces in the previous frames, and thus theoretically solving the occlusion problem.

9. References

Dan Casas, G. (2009). Real-time face tracking method (Memoire of the final project). Universitat Autónoma de Barcelona

Minyoung K, Kumar, S.,Pavlovic, V. and Rowley, H. (2008). Face tracking and recognition with visual constraints in real-world videos. Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on , vol., no., pp.1-8, 23-28

Mathwork (2015). Face Detection and Tracking Using the KLT Algorithm. Retrieved December 3, 2015, from http://es.mathworks.com/help/vision/examples/face-detection-and-tracking-using-the-klt-algorithm.html