Liveness Detection Using Face Recognition

CS B657 : Final Project Report

Guided by Prof. David Crandall

Prakash Rajagopal (prakraja)

Vishesh Tanksale (viajtank)

Content

1. **Introduction**

Biometrics is being increasingly used for the purpose authentication and authorization. It is based on the various distinguishing human physiological characteristics like fingerprints, retina scans, iris scan, voice and face, which are used to identify a human being. Biometric systems is emerging technology because of its ease of use. But, biometric systems are not totally secure against various computer security threats. With advancement in technologies it is with that these biometric system are subjected to attacks. The general biometric system are designed to identify individual human being, but they are not designed to differentiate between live and non live biometric.

We are concerned with face recognition system in our project. There are multiple methods by which attack on face recognition system can be launched. In our project we have tried to develop a system that will detect a spoofing attack on a face recognition system using photograph. An attacker can obtain a picture of a victim and then use it to gain wrongful access to a system which is protected by a face recognition biometric system. We have tried to implemented liveness detection for face recognition system using two different method. Details about them will be put forth in later sections. There are various way of liveness detection for a face recognition system. Our method of liveness detection is based on the facial variation. Precisely, we are trying to capture the motion in the eye region to determine liveness. Eye region is used for detection because it exhibits lot of variation in it shape and size in small amount of time. This variation in eye region can be attributed to blinking of eyes, continuous movement of eye, change in pupil size.

2. **Background**

Liveness is mainly based on studying patterns on face over a sequence of images. It attempts to detect that a real user is using the system without the help of any extra external hardware/equipment. The main features we have used in liveness detection of the face are Eye detection and change detection of eyes.

2.1 Eye and face detection

Initially detection of face is done using the Haar classifiers built into openCV. Once face is detected, then we detect face feature using Hog over the face and mark out the points around the eye to create a bounding box.

2.2 Normalization [2]

Eye regions might have different orientation and sizes. Before processing eye images, we need to ensure that all eyes are aligned in the same way, have same color, same size and are taken in same light conditions. To do this we take the bounding box of the eye, rotate in the opposite direction of its tilt to make it parallel to the image. Then we resize the bounding box to fixed size of 50x24. Then we finally apply self quotient image to normalize any lighting conditions.Self Quotient Image can be thought of as a high pass filter. It can be defined as below

$$Q = \frac{I}{I'} = \frac{I}{F * I}$$

Where, I’ is low frequency image of the original Image, F is the Gaussian Kernel

$$G = \frac{1}{2\Pi\sigma 2} e^{\frac{i2 + j2}{2\sigma}}$$

$$\tau = Mean\,(I)$$

$$W(i,j) = 0 \; I(I,j) < \tau \text{ or } 1 \text{ otherwise}$$

$$F(i,j) = W(i,j)\, G(i,j)$$

Where, I is the kernel size, I(i, j) is the intensity, F(i, j) is the Gaussian kernel, and W(i, j) is the weight in (i, j).

2.3 Attack methods

One of the most common methods of attack on a face recognition system is with a photo of the user which is then used to spoof and authenticate. Other methods include but not restricted to video spoofing and physical model of user. All the methods covered here are intended to work for photo spoofing and physical model of user. They cannot distinguish between live and a spoofed video before the camera. Video spoofing can be partially countered using background detection which we do not cover in this report.

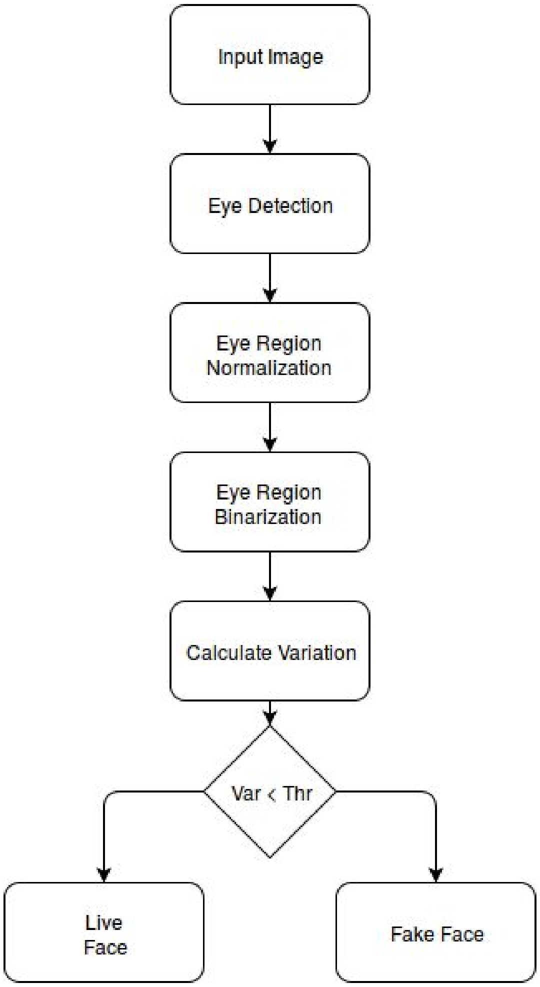
2.4 Motivation and advantages[2]

The main motivation for using liveness based algo on eyes is that

- No extra hardware is required.
- Eyeblink behaviour is the prominently distinguishing character of a live face from a facial photo, which would be much helpful for liveness detection only from the generic
- Needs no user calibration

3. **Method**

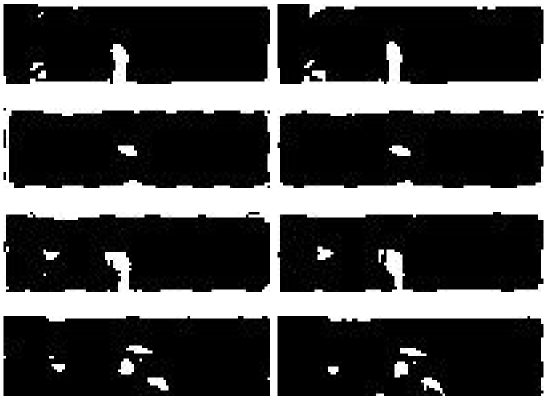
3.1 Using Binarization and hamming distance



Multiple input images are captured in a short amount of time. On each image face detection and eye detection is applied. After detection of eye region Self Quotient Image (SQI) is applied. The image is converted into a binary image. This conversion is necessary for comparing sequence of images captured with each other.

Eye region binarization is carried out so that each pixel will have either 0 or 1 value. The value of 0 or 1 is assigned to each pixel by comparing its value with a threshold. The threshold is obtained for each region individually. It is the average of all the pixel intensities in that region. Few examples of binarized eye regions are shown below.

After eye region binarization each eye region is compared with other eye regions. The comparison is made by calculating the hamming distance between eye region binarized images. Average of the hamming distances across a sequence of images is the liveness score of that sequence of image. If the liveness score is below certain threshold then the image is fake image, else it is a live image. The idea behind this is that if the sequence of images are not a real face then there is little or no difference between them. But if it is a real face there will lot more difference between each of those images attributed to the motion of eyes on a normal human face.



Binary Real Images

Binary Fake Images

3.2 CRF / HMM

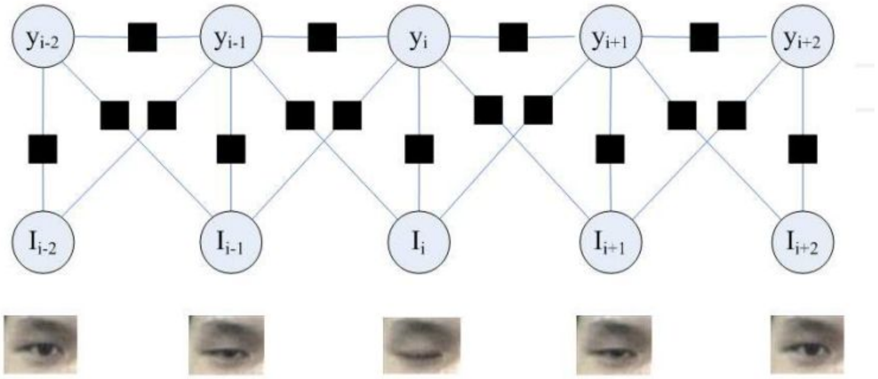
This method basically tries to deduce eye blink sequences to prove liveness. Accordingly we can classify eye status as one of the three :

- Open
- Closed
- Ambiguous

An eye blink action is series of Open -> Ambi -> Closed -> Ambi -> Open or sequences starting with closed. To classify the image into one of the 3 categories, we define a parameter called eye closity.

Eye closity measures the degree of eyes closeness. We use Adaboost algorithm to train a series of weak classifiers from the ZJU eyeblink data which has categorized eyes scaled to 24x24 for each eye[7]

The Eye closity value acts as the observed value for an image in the HMM.



Eye Blinking CRF graphical model [Reprinted from 2]

We can now classify a new image in the sequence using simple dynamic programming algo on the HMM - i.e Viterbi algorithm. Using this we can now get sequence of blink states.

We now determine liveness to be true for any sequence of images with a blink activity or with a number of blinks greater than fixed threshold. The number of blinks we've chosen as threshold in our implementation is 3 (arbitrarily based on tries).

4. **Results**

4.1 Binarization

We test our implementation on number fake and live faces. Table below shows the result of liveness score for each set of live and fake images.

Training	**Average**	**Min**	**Max**	**Number of Eyes Detected**
Live Face 1	105	40	258	41
Live Face 2	124	96	167	65
Live Face 3	69	19	144	68
Live Face 4	71	12	123	44
Fake Face 1	20	4	66	54
Fake Face 2	53	22	90	14
Fake Face 3	54	9	136	88

Fake Face 4	48	10	204	80

Depending on the liveness scores observed above, we have decided the threshold to be 70. We have observed that the score has large variation. We have discussed this more in the next section.

4.2 CRF results

Training :

We run “init_data.sh” and Create a model file eye_closity.model from closed eyes images and open eye images in data folder using Train_EyeClosity program.

Testing :

Using it we apply it on the images sequences in raw folder (same as used in Normalization). We assume that sequence of images are in order of their time. The only issue in the sample data in the raw folder is that some of the sequences do not have any closed eyes probably because of missing frames. Hence the accuracy is lower than expected

Due to missing closed eye frames in the data, the algo leads to lot of false positives (i.e fake when real) , but this makes it better for real world security applications where a false negative can be far more costly than a false positive.

5. Discussion

Issues in Hamming distance

While Hamming distance can generally work well for a very naive spoofing method of simply moving a static image, but it seems to perform much worse when it comes slanted images and rotations. This is because of the inability of the program to normalize in such a case. This leads to greater number of false positives.

Certain spoof techniques like moving the eyes out of focus and bringing it back do not work mainly because we first determine face and features before proceeding to hamming. Our algo fails to detect face in such cases thereby making it better despite it being an implementation detail. But the algo does not actually actively work to prevent it.

6. **Conclusion**

We have studied methods of spoofing face detection methods used for authorization and have mainly dealt with methods for dealing wit h static spoof attacks - i.e photo based spoofing.
Of the 2 techniques, the hamming distance seems to give a lot more false negatives (i.e. accepts spoofs) due to the failure of normalization or self quotient image at certain conditions like back tilting. The HMM created by the normalization fails to create a major difference in accuracy possibly because of bugs in implementation rather than the failure of the actual model. The algo should and does[2] in fact do better than hamming distance, erroring towards false positives making it more secure and applicable in real life scenarios.

7. **Reference**

1) Face feature detection tutorial :http://www.learnopencv.com/facial-landmark-detection/
2) Liveness Detection for Face Recognition : Gang Pan, Zhaohui Wu and Lin Sun, Department of Computer Science, Zhejiang University
3) Liveness Detection for Embedded Face Recognition System :Hyung-Keun Jee, Sung-Uk Jung, and Jang-Hee Yoo
4) Automatic Recognition of Eye Blinking in Spontaneously Occurring Behavior: T Moriyama, T Kanade, JF Cohn, J Xiao, Zara Ambadar, Jiang Gao, Hiroki Imamura
5) OpenCV docs for face detection http://docs.opencv.org/3.1.0/d7/d8b/tutorial_py_face_detection.html#gsc.tab=0
6) Motion-Based Counter-Measures to Photo Attacks in Face Recognition : Andre Anjos, Murali Mohan Chakka and Sebastien Marcel
7) ZJU Eyeblink database from http://www.cs.zju.edu.cn/~gpan/database/db_blink.html with actual download from http://parnec.nuaa.edu.cn/xtan/data/ClosedEyeDatabases.html