Facial Tic Detection Using Computer Vision

Christopher D. Leveille

Advisor: Prof. Aaron Cass

March 20, 2014

Abstract

Detecting dishonesty or deception in the features of the face has many useful applications. One such application is in the game of poker, where deceiving one's opponents is a big part of the game. Naturally, an opponent's face can provide a player with information on whether that player is being truthful in their betting or not. In this project, we built a system that employs computer vision techniques to automatically detect one common facial tic: a mouth twitch. The system was evaluated based on its ability to accurately detect and identify occurrences of the tic within a video in real time. Preliminary testing results found the system to be accurate under ideal conditions, but it struggled to correctly detect the tell in a more realistic, less-than-ideal setting.

1 Introduction

Humans are very face-oriented creatures. Reading someone's facial body language is something that comes naturally to humans. In fact it comes so naturally that people take for granted their ability to detect when someone changes their facial expression by raising their eyebrows just a couple of milimeters or twitching their mouth a few degrees. The detection of such fine muscle movements in the face (or the detection of just a face, for that matter) is a much more difficult task for computers, which must do considerable amounts of calculation in order to come close to recognizing the features of a face. That said, creating a computer vision-based system capable of efficiently detecting the movement of a single facial feature makes for a challenging and interesting research problem.

Using computer vision to detect the features of a human face has many useful applications. In human-computer interaction, a system capable of accurately reading and interpreting a person's face would be able to better identify, understand and communicate with the person. Such a system would be useful in social robotics and lie detection systems to name a couple.

In particular, detecting dishonesty or deception in the features of someone's face could be useful in many different areas. For example, in police interrogation knowing whether or not the subject is lying is of utmost importance. The game of poker is also a good example, as deceiving one's opponents is a big part of the game. Naturally, a player's face can provide an opponent with information on whether that player is being "truthful" in their betting or not.

In a hand of poker, each player is dealt a number of private cards that are not visible to their opponents. The core mechanic of the game is to bet on having the best hand, or to make other players give up by betting enough that they do not feel it is worth matching the bet and continuing in the hand. While players may bet for value knowing they likely have the best hand and are hoping to be paid off by a weaker one, they may also choose to **bluff**, where they bet strongly with

a weak hand hoping to make players with a better hand give up so they can win the pot.

When bluffing, players will sometimes exhibit physical mannerisms known as **tells** which can give opponents information about the strength of their hand. While professional poker players train themselves to maintain good control of their body language and keep their tells to a minimum, amateur players are often very readable since bluffing is naturally a "dishonest" course of action. A tell can take many different forms, such as a change in posture, shaking fingers, a visible pulse in the neck, increased blink rate, or covering the face with a hand. Many poker tells can be visible on a player's face. In this project, we designed and built a software system that uses computer vision to detect one specific facial tic that an amateur poker player exhibited when bluffing.

2 Background and Related Work

Viola and Jones [10] [11] proposed a fast and efficient way to model object detection in general using 'Haar-based classifiers'. A Haar-based classifier essentially relies upon the contrast in color between different features of interest. A couple examples are that the color of a face contrasts with the color of the background, or the color of an eye contrasts with the color of the face around it. Instead of using an algorithm that looks for pixels of a specific color (skin color, for example) when finding a face, the approach described in these papers searches for groups of pixels accross a range of scale multipliers that follow a contrast pattern similar to a specific set of Haar-like features (see Figure 1), returning the detections it makes as a list of rectangular areas (see Figure 2). Most often, the rectangle with the greatest number of neighbors closely surrounding it is the one that is finally returned by the classifier. The Haar-like features are the results of running a training routine on a large database of positive and negative images. Because of the nature of this technique, it can theoretically be applied to detecting any distiguishable object in an image as long as the classifier is trained well enough, such as a face of features of the face.

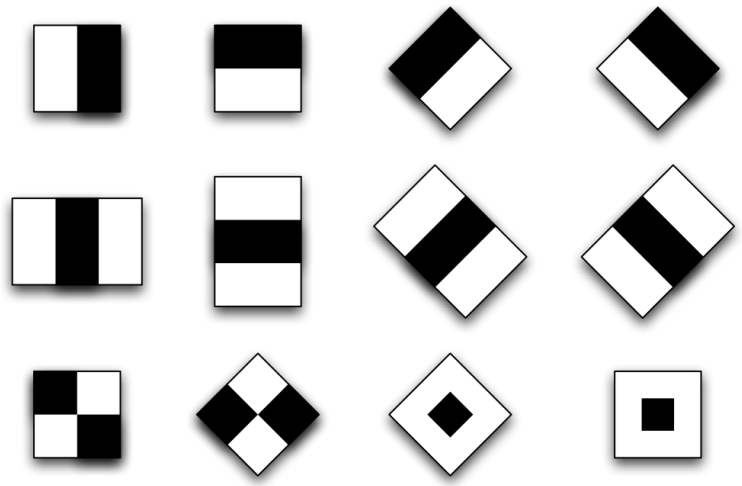

Figure 1: Some examples of haar-like features.

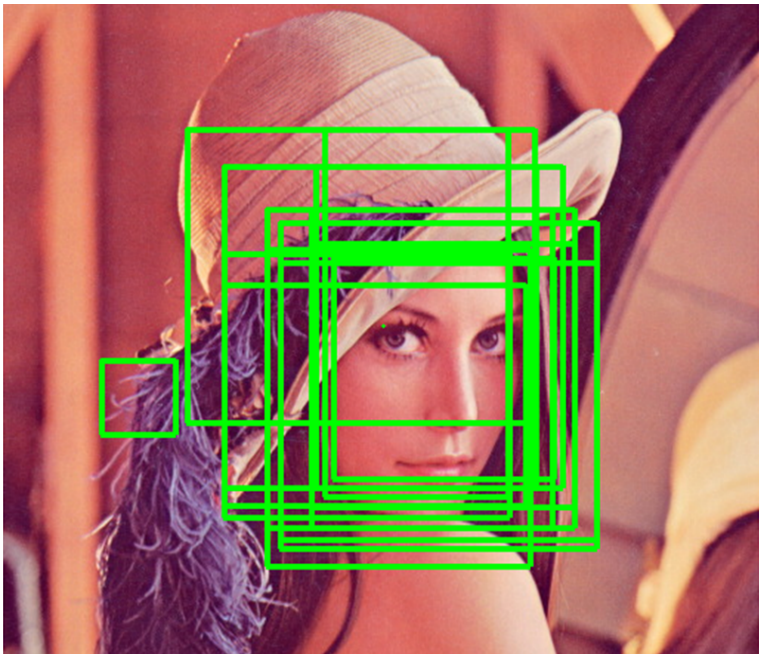

Figure 2: A visual representation of the multiple detections made by a face-trained Haar classifier.

A popular method of face detection in particular is Eigenfaces [6], which involves running principal component analysis on a large database of face images to generate a small set of statistically average faces. These 'eigenfaces' (see Figure 3) are then used to detect a face within a target image by comparing different sections of the target image with the eigenfaces and looking for close matches.

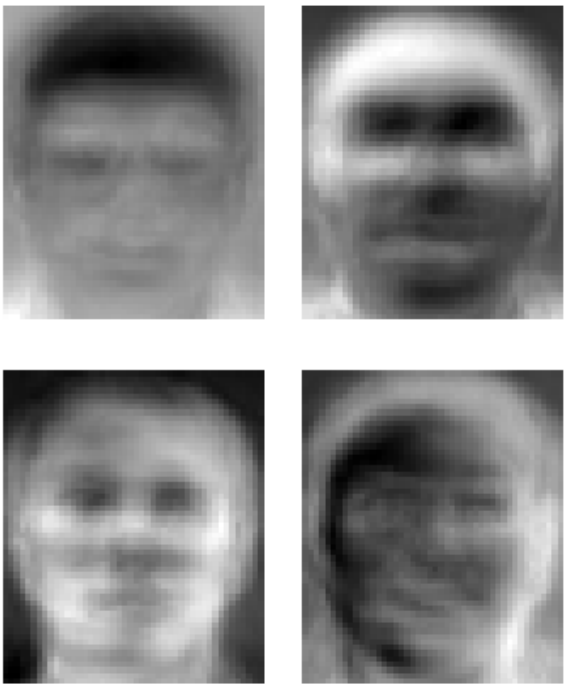

Figure 3: An example set of generated eigenfaces.

While both of these techniques seemed to be feasible, the Haar approach stood out to us because of its convenient integration into OpenCV [1], the computer vision library that we used in building the system. The OpenCV library consists of several pre-trained Haar classifiers for detecting the face and different features of the face. While using the Haar method would solve part of our problem by tracking facial features of interest, it would not solve the problem of detecting a tic

completely because the existing classifiers are trained to work only on single static images. They are not capable of detecting movement of those features in any way. Nonetheless, we decided to use OpenCV's Haar classifiers as part of the system, and find some other way to dynamically detect a tic within the features of the face.

Given that we planned on using poker as a medium for the study, another part of our research invloved looking into what kind of poker tells a player might exhibit in a live game. In *Caro's Book of Poker Tells* [2], the author discusses many different tells in detail and what they might mean in the context of a live poker game. However, he emphasizes that everyone exhibits different kinds of tells, and that they can mean different things for different people. Because of this, we decided to focus the scope of this project on just one tell for one person. This is because attempting to generalize the meaning of a single tell across multiple poker players would pose a threat to the validity of the study. Although this is a narrow approach that may not yield any significant findings, we felt that it was the most realistic option given the limited time period for the project.

3 Deciding on a Facial Tic to Detect

The first decision to make in the project was which tell to design the system to look for. While we could have decided this up front and then tested the system simply by intentionally exhibiting the tell in a video, we decided it would be better to record an amateur player in a live game of poker and see which tells they natually exhibited. Although the experiment was not completely genuine (since the player was aware he was being recorded), it was more candid for testing purposes than intentionally exhibiting tells in a recording.

Over the course of fall term we conducted a series of laboratory trials, in which two amateur poker players were filmed playing against each other. The main goal of these lab trials was to collect video footage that we could later review and know whether or not a player was bluffing in a

given hand. Then we would be able to closely watch the subject's body language during the spots he was bluffing in, and take note of any physical tells that he exhibited. We would then use this information to decide which tell to build the system to detect.

We set up and ran four 40 - 80 minute sessions in which two volunteer participants played no-limit Texas Hold'em poker. It was important that the game was no-limit so that the players could bluff each other more effectively. In a game with a betting limit, it is not as easy to bluff an opponent since you cannot bet more than a certain amount. Only one of these players was recorded (referred to throughout as the **subject player**); the other served as the opponent and full-time dealer. Although it is common to alternate dealing in a game with two players and no dedicated dealer, the opponent volunteer dealt full-time for these trials in order not to provide unnecessary distraction to the subject player being recorded. The games took place on a round, fold-up table in a quiet, well-lit lab environment (see Figures 4 and 5 for a diagram and image of the laboratory setup). Two Logitech Pro 9000 webcams were used for recording. These webcams are relatively inexpensive, accessible, and feature ample resolution (1280x720) for computer vision applications. One camera was positioned on a tripod across the table facing toward the subject, with the the subject's uper body and face clearly visible. The second camera was attached to the top of a tall lab chair, with the lens facing down at the center of the table. The purpose of this camera was to see the community cards on the board, and also for the subject to show his hole cards to at the beginning of each hand (see Figure 6). This was important to provide context information on whether or not the subject was bluffing when later reviewing the footage.

In the lab trials it was important to play as many hands as possible in a given amount of time because we wanted to give the subject as many opportunities to exhibit tells as possible in a reasonable amount of time. To accomplish this we used an automatic deck shuffler for the third and fourth trials, which the opponent volunteer used to quickly shuffle the deck after each hand while a second deck was being dealt out for the next hand. This increased the throughput of hands

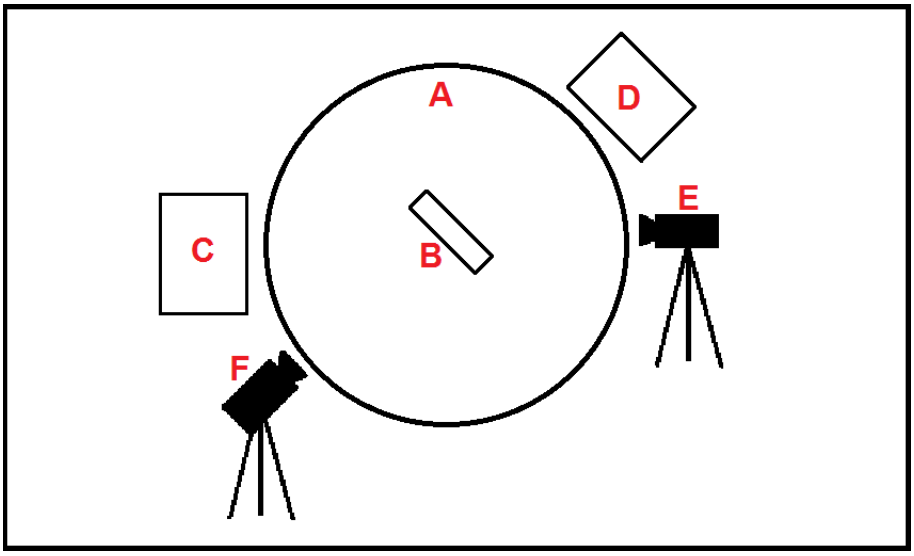

Figure 4: A top-down view diagram of the laboratory setup. **A:** The round, plastic, table used as a surface to play poker on. **B:** A strip of blue tape used to mark the location of the community cards for the dealer. **C:** The chair where the subject player sat. **D:** The chair where the opponent player/dealer sat. **E:** The camera aimed at the subject player. **F:** The camera aimed at the community cards on the table.

Figure 5: The laboratory setup as recorded in Morae Recorder.

Figure 6: An example shot of the the subject player showing his hole cards to the camera.

played by cutting down on the time between hands.

For recording software we used Morae Recorder [9]. We chose to use Morae mainly because of the 'picture-in-picture' recording format, which displayed the 'board camera' as a small overlay on top of the larger 'main camera' picture in a single video file. This format proved to be very useful when later analyzing the recordings as the footage from each camera was perfectly synchronized. Also, we didn't have to fumble over pausing and playing two different video files at the same time while keeping them in sync.

Once we were done conducting the trials, the video recordings were reviewed and each hand was analyzed for any tells that the subject player exhibited. It was also noted whether or not the subject player was bluffing wherever possible. All information collected by reviewing the videos was stored into a table, an excerpt of which can be seen in Table 1. Based on our analysis of the

footage and research into effective computer vision techniques, we decided to pursue building a system to detect **twitching of the mouth**. This is a facial tic that the subject player exhibited multiple times during the recorded lab sessions when bluffing (see Figure 7).

Figure 7: The mouth of the subject is in a normal state on the left, and is twitching on the right.

Table 1: Excerpt from Lab Trial 1 Hand Annotations

Hand #	Time	Subject's Cards	Opponent's Cards	Flop	Turn	River	Notes
1	0:09:22	A J	4 Q	Ah 7c 10c	-	-	Value bet on flop, open mouth
2	0:11:02	Qs 4c	As 4s	Js 5d Kd	8c	2d	Mouth twitch on turn (bluffing)
3	0:13:00	As Jd	6h 7d	2c Kh Ad	-	-	Value bet fail
4	0:14:20	??	??	Qh 7d 3h	Qd	10c	-
5	0:16:30	6d 3d	Kc 3c	7h As 3s	4c	8d	Closed mouth, puffed cheeks on flop bet
6	0:18:52	Jc 6c	6h 5s	4d 8c 4c	Qd	7d	Eye rub, bluff raise on flop: mouth twitch
7	0:20:24	??	??	Kh 5h 8s	As	Kd	-
8	0:22:08	7s 8h	2d 3s	5d 2d Jh	7c	3h	Squinting on river bet
9	0:23:58	Kc 4c	Ad 5h	Ks 9h Kh	As	Qs	Mouth pursed on flop raise
10	0:27:01	9d 6h	Kd 9s	4s Ks 6c	3c	4c	-

4 System Implementation

Now that we had decided on a tell for the system to detect, the first problem to be solved in implementing the system was locating the mouth within the face of a person. To do this, we chose

to make use of the OpenCV library. As one of the most popular computer vision libraries out there, we felt confident that it would provide us with the tools we needed to accomplish our goal. OpenCV is also very well documented, which helped a lot in learning how to implement the things we needed in a time-effective way. We used the most recent version at the time of the project, OpenCV 2.4.8.

The OpenCV library contains several Haar cascade classifiers pre-trained to detect faces and features of the face. We began by using these classifiers to track the location of the face within each frame of a video. During the development of the system, we tested this live in front of a webcam. We then tried running the mouth-trained Haar classifier on each frame to find that it frequently mistook other objects in the image for a mouth, other than the mouth itself. We addressed this problem by first running a face-trained Haar classifier on each frame, and then running the mouth-trained classifier on just the area where the face was detected. This eliminated the problem of the mouth-trained classifier detecting false-positives outside the area of the face, but it was still not working well enough. The system still frequently highlighted one of the subject's eyes when looking for mouths. We dealt with this issue by further restricting the search space of the mouth classifier. Now, instead of searching the image of the entire face, the mouth classifier would only ever search the bottom half of the face area returned by the face classifier. Since the mouth is the only object in the bottom half of a (normal) person's face anyway, the system was now able to detect and highlight the mouth within the image with considerable accuracy. (See Figure 8).

Now that the system was capable of detecting the mouth within a person's face, we set our sights on finding a way to detect a mouth twitch within a video. A big part of this problem was defining what a mouth twitch really is. In the lab trial videos, the mouth twitches that the subject exhibited were sharp but subtle displacements of the mouth that took place in less than a second. Originally we thought that this would have something to do with training a new Haar classifier to detect a mouth in a twitched state. This classifier could then be run on video, displaying output

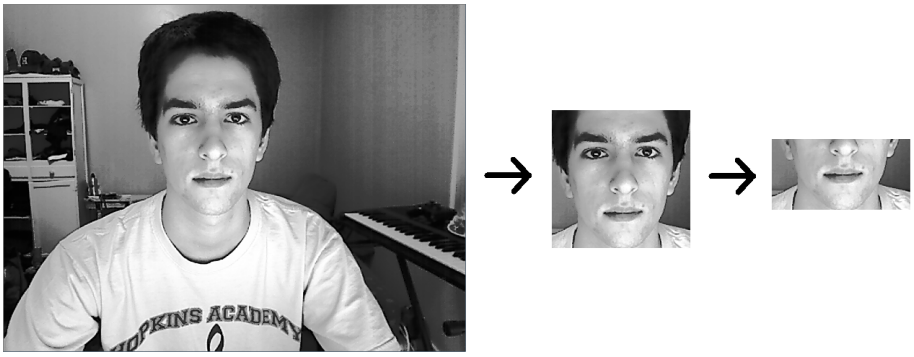

Figure 8: The process by which the system narrows the search space for the mouth classifier.

when it detected a twitching mouth. However, we decided to not pursue this course of action for a couple of reasons. It takes an extremely large database of positive and negative images to train a new Haar classifier. While there are plenty of images of people's faces available on the internet, there are very few of people with their mouths in a twitched state. Because of this, we would have had to capture all of the positive images ourselves, which would have taken a long time with no guarantees as for the accuracy of the classifier once it had been trained. Also, the training tool itself can take days to run on a large image database.

Something that we noticed early on when testing the system during development is that the mouth classifier would temporarily lose track of the mouth when the subject twitched their mouth, only to detect it again once the twitch was finished. We figured that the only time that the mouth would not be detected within the face would be when it was in an abnormal position, twitching. Because of this, we chose to define a twitch (with regards to the system) as a frame in a video where the system succeeds in detecting the face, but fails to detect the mouth within the face. We modified the system to check for this condition, and display a message with a timestamp if it occurs (see Figures 9, 10, 11).

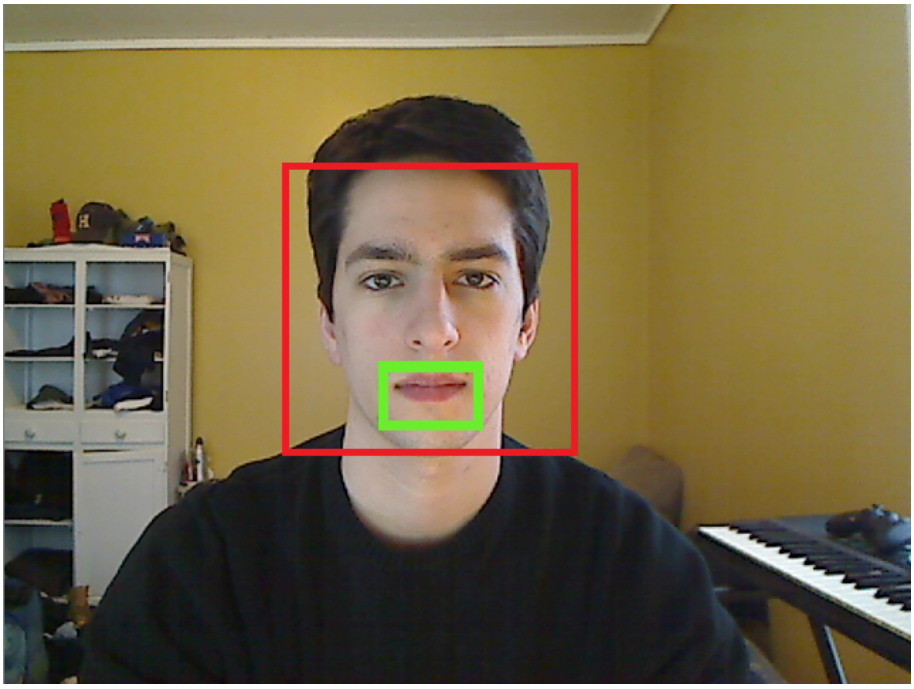

Figure 9: The system detecting the face and the mouth.

5 Evaluation

Preliminary testing has shown that the system works well in detecting mouth twitches under controlled conditions. The face must be upright, facing toward, and clearly visible to the camera with ample lighting in order for the system to work effectively. However, it did not work as effectively on the lab trial videos. We speculate that this is because the subject‘s attention was mostly invested in the game and not on keeping his head upright and facing the camera at all times, which seems to be necessary for the systems ability to consistently track the face. Also, the mouth twitches exhibited by the subject in the lab sessions were very subtle, making it difficult for the system to recognize them given the distance at which the subject was filmed from.

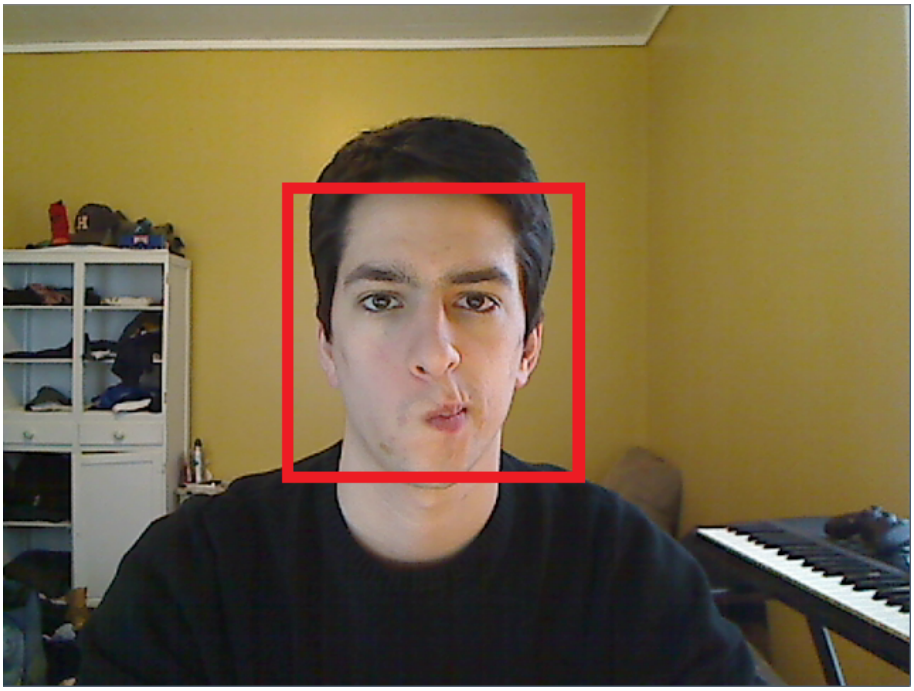

Figure 10: The system detecting the face but not the mouth while the subject's mouth is twitching.

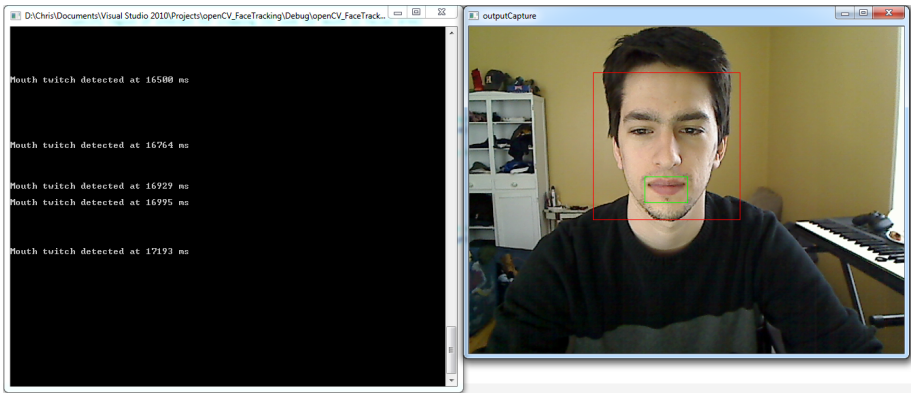

Figure 11: The system displaying output messages when twitches are detected. The messages seen here are from twitches that occurred within a few seconds before this screenshot was taken.

6 Future Work

Given the relatively small scope of this project, there are many prospects for future work. One such prospect is to formally test the system's accuracy by creating a video file under controlled conditions and having the subject intentionally exhibit mouth twitches at specific times in the video. The system could then be evaluated on its ability to detect mouth twitches at the correct times. Also, testing the system's accuracy across many different subjects instead of just one would be important for exapanding the study‘s significance.

Another application for future work is to create a second system that detects mouth twitches in a different way, and then compare its accuracy to the original system‘s accuracy. One idea that we did not have time to pursue is creating a new Haar classifier trained to detect mouths in a twitched state. A future project might consist of gathering a database of positive and negative images, training the classifier, and testing its ability to detect when a person is twitching their mouth in an image.

Also, future work prospects might involve expanding the existing system to support the detection of other poker tells. These might include tracking blink rate, facial expression, or a player's posture at the table. A system able to detect multiple common poker tells would be very useful in human vs. robot poker projects in computer science.

References

[1] Gary Bradski and Adrian Kaehler. *Learning OpenCV - Computer Vision with the OpenCV Library.* O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472, 2008.

[2] Mike Caro. *Caro's Book of Poker Tells.* Cardoza Publishing, Las Vegas, NV, March 2003.

[3] Erik Hjelmås and Boon K. Low. Face Detection: A Survey. *Computer Vision and Image Understanding*, 83(3):236–274, September 2001.

[4] J. Kovac, P. Peer, and F. Solina. Human skin color clustering for face detection. In *EUROCON 2003. Computer as a Tool. The IEEE Region 8*, volume 2, pages 144–148 vol.2. IEEE, 2003.

[5] M. Pantic, M. Tomc, and L. J. M. Rothkrantz. A hybrid approach to mouth features detection. In *Systems, Man, and Cybernetics, 2001 IEEE International Conference on*, volume 2, pages 1188–1193 vol.2. IEEE, 2001.

[6] A. Pentland, B. Moghaddam, and T. Starner. View-based and modular eigenspaces for face recognition. In *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR '94., 1994 IEEE Computer Society Conference on*, pages 84–91. IEEE, June 1994.

[7] D. Reisfeld and Y. Yeshurun. Robust detection of facial features by generalized symmetry. In *Pattern Recognition, 1992. Vol.I. Conference A: Computer Vision and Applications, Proceedings., 11th IAPR International Conference on*, pages 117–120. IEEE, August 1992.

[8] H. A. Rowley, S. Baluja, and T. Kanade. Rotation invariant neural network-based face detection. In *Computer Vision and Pattern Recognition, 1998. Proceedings. 1998 IEEE Computer Society Conference on*, pages 38–44. IEEE, June 1998.

[9] Author Unknown. Morae Components and Features, March 2014. http://www.techsmith.com/morae-features.html.

[10] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–511–I–518 vol.1, Los Alamitos, CA, USA, April 2001. IEEE.

[11] Paul Viola and Michael J Jones. Robust Real-Time Face Detection. *International Journal of Computer Vision*, 57(2):137–154, May 2004.

[12] Phillip I. Wilson and John Fernandez. Facial Feature Detection Using Haar Classifiers. *J. Comput. Sci. Coll.*, 21(4):127–133, April 2006.

[13] Xiangxin Zhu and D. Ramanan. Face detection, pose estimation, and landmark localization in the wild. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2879–2886. IEEE, June 2012.