FACIAL GESTURING FOR MOUSE CONTROL

Aditi Raj Kandoi
Department of Computer Engineering,
Stony Brook University
akandoi@cs.stonybrook.edu

Abstract

In recent years, the interaction between humans and computer takes place in a variety of ways such as using hand gestures, head movements, facial expressions, voice, touch, and so on. The project describes a method for facial gestures detection in order to communicate between man and machine. It uses image processing techniques like face detection, eye detection and face tracking real-time. Based on the detection of facial characteristics that involve eyebrows, eyes and mouth, activities such as head motions and eye blinks are done so that the mouse cursor can perform required actions on computers. Having a hands-free interface helps specially-abled people to access computer applications as well.

1 Introduction and motivation

Almost everything in the twenty-first century is computerized. As a result, the computer has had a wide-ranging impact on society and the environment. Personal computer (PC) application is widely distributed throughout all domains such as communication, entertainment, education and research and is utilized by many people. Human computer interface is an essential field in artificial intelligence because it utilizes human motions to interact with the computer.

For this interaction, the human gestures are recognized using the vision-based multimodal interface techniques, which only require face detection and tracking, thus making it less expensive. The muscular movements made by the face are vital in deciding or expressing human behavior, such as speaking, facial expression, and gesture. According to prior studies (Becker, 1953), neuronal circuits involving the hands and face take up a disproportionate amount of sensory and motor cortical space. As a result, it can be believed that facial motions might play an essential complementary or supplemental function in machine interaction to the part provided by the hands.

All standard gadgets require human operation and are inaccessible to those with limited mobility. Physically challenged persons, such as those who are bedridden or paralyzed, face a significant obstacle in utilizing computers in their daily lives. Therefore, there is a need to develop alternative techniques of communication between humans and computers that are suited for people with motor disabilities. The techniques for the head and face detection is the initial stage in recognizing facial expressions. Therefore, the aim of this project is to create a virtual interactive module that allows users to control the mouse cursor with face movements.

The project offers a software approach which uses face detection, face tracking, eye detection, and interpretation of a sequence of eye blinks in real-time. To better interact with computers, the traditional way of using a mouse is replaced by facial gestures. This approach will assist persons who are paralyzed or physically challenged, particularly those who do not have hands, in computing effectively and effortlessly. A webcam is required to obtain photographs of face movements captured by the camera. Aside from that, no extra hardware or sensors are required.

2 Literature Survey

The most crucial task in face analysis is automated facial point detection. Several approaches have been developed, and their findings on databases in both limited and unconstrained settings have been reported. However, most of the datasets contain annotations with various mark-ups, and in some cases, the correctness of the reference points is questioned. Furthermore, due to the limits mentioned above and the lack of an assessment mechanism, comparing performance between several systems. The purpose of the paper (Sagonas et al., 2013) was to evaluate the performance of multiple algorithms on

a newly produced dataset using the same assessment technique and mark-up, culminating in the development of the first standardized benchmark for facial landmark localization.

The authors in (Soukupová and Cech, 2016) provide a real-time method for recognizing eye blinks in a video series acquired by a standard camera. Landmark detectors trained on in-the-wild datasets show exceptional resilience to camera head direction, variable light, and facial expressions. The paper demonstrates that the landmarks are recognized correctly enough to determine the eye-opening level reliably. As a result, the proposed technique assesses landmark positions and produces a single scalar value Eye-Aspect-Ratio (EAR) representing each eye-opening frame. The SVM classifier detects eye blinks as a pattern of EAR values in a temporary temporal structure, and on two typical datasets, the fundamental technique outperforms the state-of-the-art results. This study mainly focuses on predicting the facial landmarks of a given face, and this detected eye blinks in a video can be used as gestures for mouse pointers.

The article (Kazemi and Sullivan, 2014) delves into the topic of Face Alignment for a single image. The authors illustrate how an ensemble of regression trees may be used to directly estimate the positions of a face's landmarks using a sparse subset of image intensity, resulting in near-real-time performance and high-quality predictions. They developed a general framework for learning an ensemble of regression trees using gradient boosting that minimizes the sum of square error loss and accepts missing or partially labeled data. They additionally illustrated how using priors to leverage the structure of image data benefits successful feature selection.

The author of the paper (Lyons) examines numerous vision-based interface technologies that depend on facial action using the concept of human-computer interaction (HCI). Text entry, artistic and musical expression, and assistive technology for users with physical disabilities are a few of the domains and categories covered. The results of research using conventional control tasks show that simple, fast, and durable systems for monitoring head motions and facial shapes provide a high-level of interactive control.

The research (Meena et al., 2020) presents an algorithm that executes mouse activities by enabling individuals and computers to interact without using their hands. Actions may be determined based on motion by using computer vision to distinguish unique expressions on a person's face and compare them to previously recorded expressions. This algorithm will aid physically challenged individuals in executing mouse operations by utilising their face and eye motions.

3 Methodology

The focus of this project is on correctly assessing the facial descriptors of a given person and employing the markers to interact with the computer. It is possible to build features that will allow the model to detect certain actions, such as using the eye-aspect-ratio to detect a blink or wink and using the mouth-aspect-ratio to detect opening and closing of the user's mouth. The gestures that are implemented in the project can be seen in table 1:

- The mouse control will be activated or deactivated by the mouth movements. When the user opens their mouth the first time, the control activates. Similarly, if they want to deactivate the control, user should wide open the mouth.
- The mouse scroll will be activated or deactivated by eye and eyebrow movements. When the user wants to enable mouth scroll, they squint their eyes, the same should be done to disable the scroll.
- There are eight possible mouse movements which is performed by moving the head in the required direction.
- To allow left or right click, the user has to wink the left or right eye respectively.

Table 1: Facial Gestures with corresponding mouse control

Gesture	Mouse Control
Open mouth	Activate/Deactivate mouse control
Eye squint	Activate/Deactivate mouse scroll
Move head in 8 directions	Mouse scroll movement
Wink left/right eye	Left/Right click

3.1 Basic Implementation

The video is taken in real-time using a webcam. A grayscale image is produced by inverting the input stream and translating it to a picture frame. The face is discovered by examining the frame. Face

features are obtained as soon as a face is recognized. User gestures may then be recognized by comparing them to the training model. The action on the mouse pointer, or any other action initiated by the system, is dependent on the gesture made.

3.2 OpenCV

OpenCV currently has a large number of pre-trained classifiers for various facial features such as eyes, emotions, etc. Using Haar cascades to recognize faces is a machine learning-based strategy in which a cascade function is trained using a collection of input data before being used. I use it to OpenCV's *VideoCapture* function to capture video through the default webcam of the system and process it frame-by-frame. Using the camera, I create an endless loop and continue reading frames until a keyboard interrupt is hit.

First, the function *cv2.flip()* is used to flip a 2D array of an image-frame horizontally, around the y-axis. The detecting algorithm is only effective on grey photos. Because of this, it is necessary to transform the color frame to grayscale.

I am also using the *ConvexHull* function to approximate the contours for eyes and mouth and then draw it on the frame for the user to see it. OpenCV is also used to display the rectangular box of the detected face in real-time and write the text on the frame.

3.3 Dlib package

Dlib is a facial classifier that uses pre-trained models to predict the placement of 68 vectors (x, y) representing the facial attributes on a person's face, as shown in the figure 1.

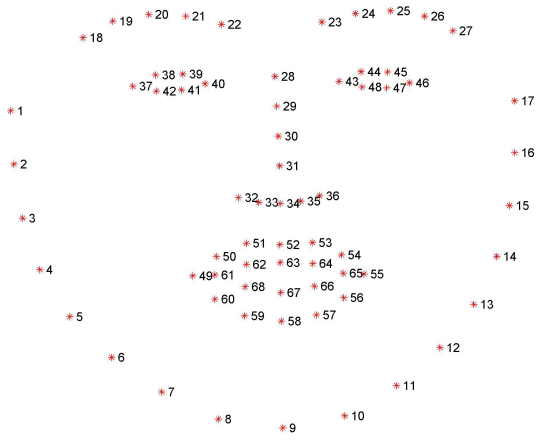

Figure 1: Landmarks identified by the dlib predictor

Faces are detected by using a Histogram of Oriented Gradients and Linear SVM based facial recognition algorithm. With this function within the variable, we can receive the face image detector from **dlib.get_frontal_face_detector**.

Detecting feature points on faces with the help of a predictor Inside the variable, I have placed the **dlib.shape_predictor_function**. The model is built on ensemble regression trees since the model will predict continuous quantities. It is required that the argument for this function be a pre-trained model which I have selected as *"model/shape_predictor_68_face_landmarks.dat"*. The model extracts the landmarks as a coordinate array which is then converted to NumPy array for computing ratios.

3.4 Eye Aspect Ratio

After detecting the face, I calculate EAR using the formula(1) to determine the condition of the eyes openness.

$$L.EAR = \frac{\| 38 - 42 \| + \| 39 - 41 \|}{2* \| 37 - 40 \|}$$

$$R.EAR = \frac{\| 44 - 48 \| + \| 45 - 47 \|}{2* \| 43 - 46 \|}$$

$$EAR = \frac{(L.EAR + R.EAR)}{2}$$

$$DIFF\_EAR = \| (L.EAR - R.EAR) \| \quad (1)$$

where the numbers represent the labels from Fig.1

When the eye is opened, the ratio remains constant, but when the eye is closed, the ratio progressively decreases until it is almost equal to zero. The greater the EAR, the greater the extent to which the eye is open. A threshold EAR value is determined (=0.19) and then used to evaluate whether or not the eye was closed using (EYE_AR_CONSECUTIVE_FRAMES = 15).

3.5 Mouth Aspect Ratio

Similar to EAR, I calculated MAR using the formula 2, which increases as the mouth opens and vice-versa.

$$MAR = \frac{\| 62 - 68 \| + \| 63 - 67 \| + \| 64 - 66 \|}{2* \| 61 - 65 \|} \quad (2)$$

where the numbers represent the labels from Fig.1

3.6 Imutils

Imutils are a series of functions that carry out various operations on basic images such as rotation, translation, resize, rotation and display Matplotlib images easier with OpenCV. In order to detect faces, the frames are first captured from the user's video file stream and then passed through the functions to resize it in order to further convert to gray scale form and detect the faces.

3.7 PyAutoGUI

PyAutoGUI is a library using which one can automate the movements of mouse cursor and keyboard. In this system, the goal is to automate the mouse controls. The current position of the cursor is taken into account and by using **pag.moveRel(drag, 0)**, the cursor is moved left and right relative to this previous position depending on the value of drag. Similarly, **pag.moveRel(0, drag)** moves the cursor up and down relative to this previous position depending on the value of drag.

4 Results

In this work, the goal of the system is to detect the facial gestures and perform the corresponding mouse action.

Figure 2: Mouse control activated

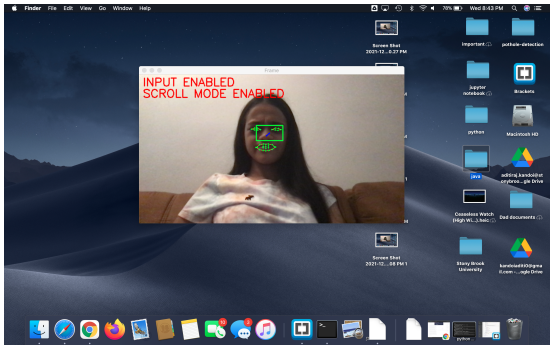

Figure 3: Mouse scroll enabled

It can be observed in Figure 2 that first the face landmarks were predicted and using these landmarks and the mouth aspect ratio, the mouth movement was recognized which activated the mouse control on the laptop. Similarly, in Figure 3, it can be seen that with the help of eye landmarks and the eye aspect ratio, the movement of the eyes was captured. Since in the video, the eyes were squinted, thus, the system performed the corresponding action of enabling the mouse scroll.

Figure 4: Mouse scroll enabled

As per the methodology of the system, the movement of the mouse cursor is based on the movement of the head. From the Figure 4, it can be seen that as the user's head tilted towards the right-down diagonal, the cursor also moved in that direction.

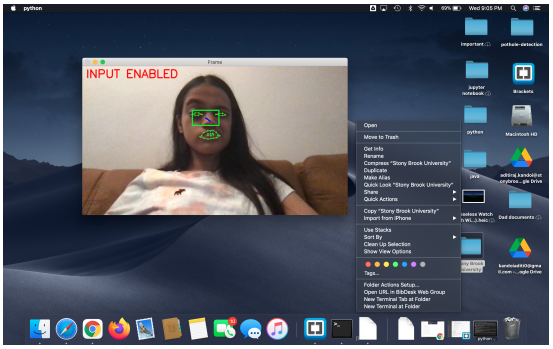

Figure 5: Right click

In the proposed system, the mouse action of right and left click is carried out by winking the right and left eye respectively. Figure 5 gives an example of a user winking the right eye when the cursor is at the folder 'Stony Brook University'. This enables the right click action on the folder which gives us a variety of options to choose from.

5 Conclusion

The proposed system deals with facial gestures taken by the web camera to operate the computers,

allowing users to browse without having to touch the computer or installing any hardware. It is easy to use and efficient for cursor control. To move the mouse, the system employs head movement, and to initiate the click, it uses eye and mouth gestures. The system's usability is quite good, particularly when used with desktop programs. It has precision and speed that are sufficient for many real-time applications and allow disabled people to work on various computing applications.

6 Future work

The proposed work can be expanded further to detect facial gestures efficiently even in low lighting conditions. Moreover, a voice command control for mouse cursor can also be incorporated and the user can decide which control commands they want to use. This may be used to create user-friendly programs that allow people to perform actions like playing virtual games, messaging and painting.

References

R. Frederick Becker. 1953. The cerebral cortex of man. by wilder penfield and theodore rasmussen. the macmillan company, new york, n.y. 1950. 248 pp. *American Journal of Physical Anthropology*, 11(3):441–444.

Vahid Kazemi and Josephine Sullivan. 2014. One millisecond face alignment with an ensemble of regression trees. *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1867–1874.

M.J. Lyons. Facial gesture interfaces for expression and communication. *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No.04CH37583)*.

Kshitij Meena, Manish Kumar, and Mohit Jangra. 2020. Controlling mouse motions using eye tracking using computer vision. In *2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS)*, pages 1001–1005.

Christos Sagonas, Georgios Tzimiropoulos, Stefanos Zafeiriou, and Maja Pantic. 2013. 300 faces in-the-wild challenge: The first facial landmark localization challenge. *2013 IEEE International Conference on Computer Vision Workshops*, pages 397–403.

Tereza Soukupová and Jan Cech. 2016. Real-time eye blink detection using facial landmarks.