Virtual Keyboard controlled by Eye Movements

A project report submitted for BTP phase II

by

Harsh Govil

(Roll No. 180108050)

Under the guidance of

Prof. Manas Kamal Bhuyan

DEPARTMENT OF ELECTRONICS & ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI

November 2021-April 2022

Abstract

The work presented provides a virtual on-screen keyboard which can be controlled using our eyes. This is implemented with the help of an eye gaze estimation model and a blink detection model which will utilize the webcam connected to the system to control the on-screen keyboard. This model will be of great importance for those people who are suffering from motor, neurological or speech disorders. Such people will be able to interact with the people around them in a much better and efficient way. The keyboard design used in this model is based on the idea of QWERTY keyboard and its extensive utility through out the industry. The gaze model along with eye blinking detection techniques enables the user to control the virtual on-screen keyboard.

Contents

List of Figures

Chapter 1

Introduction

People with speech or motor related disabilities suffer various difficulties in expressing themselves in a convenient and effective way, which reduces there ability to communicate with others substantially. Therefore, they rely on external devices and technologies for these activities. However, the aids available are still not very efficient and have a lot of scope off improvement. The external aids are computer devices based on augmentative and assistive communication (AAC) system technologies which help such people with disabilities in performing their day to day tasks. A virtual keyboard (VK) also known as an on-screen keyboard can be used as an input module for such AAC based systems. The keys of such a virtual keyboard are laid on the screen of the computer. The keys are like normal keys on a physical keyboard, though they are on the computer screen virtually. Its still not feasible for people with disabilities to use this virtual keyboard effectively and efficiently. Some may use it for some time only due to motor disabilities and some may not use it at all. Therefore, different modes of access should be present for such keyboards so that every person can access them effectively and efficiently. Scanning and access switches are most common devices which are used in such cases, however, they are also not the best way of moving forward. Brain computer interface (BCI) is used which uses the electrical brain signals inside the user's house to control the computer system. This BCI based communication is very slow and ineffective as compared to the other modes of access.

Thus, for a virtual keyboard both the layout design of the keyboard and the mode of access together decide the over all efficiency and effectiveness of the communication system. We can have keyboards operated using various input modules such as physical control, touch screens,

switches, etc. as be appropriate for users with appropriate needs. The mode of access which we propose to use is the eye gaze estimation technique along with eye blink detection . The keys on the keyboard will be accessed using the eye movements of the user. The web-cam installed at the top or bottom of the computer display will be used to detect the gazing direction of the eye and then using this information we will be considering a click on the keyboard based on the time span of constant gaze. The eye gaze detection algorithm is implemented using OpenCV, Dlib and other various libraries of python. The method, though not very accurate is still very practical. With this system, it is expected the people who have disabilities, will be able to communicate in a better manner so, that the other people can understand their desire or the information which they intend to communicate.

Chapter 2

Literature Review

The idea of developing a virtual on-screen keyboard controlled by eye movements is very ambitious and quite a few have tried working on this topic. Such a keyboard with an efficient mode of access will be of great help for a lot of people who suffer from motor or speech disabilities. The typing efficiency of a person depends heavily on the layout of the keyboard as well as on the mode of access. Several designs have been developed over the years so that the layout can be as ideal as possible. It is a difficult task to arrange such a large number of keys on a single screen, without causing any discomfort for the user. Though, it provides us various benefits as well. Word prediction [10], word completion [11], and personalization [12] can be easily incorporated into the virtual keyboard, thereby, increasing the over all effectiveness of the keyboard. Features like key size optimization [13], detecting error in typing [14] and its correction [14] can be of immense use to the users. Shumim Zhai et al. [15] used a Metropolis random walk algorithm guided by a "Fitts-digraph energy" objective function that quantifies the movement efficiency of a virtual keyboard. Mathieu Raynal and Nadine Vigouroux [16] proposed a genetic algorithm formal framework to optimize character location on a virtual keyboard. A genetic algorithm [20] is a search heuristic that is inspired by Charles Darwin's theory of natural evolution. This algorithm reflects the process of natural selection where the fittest individuals are selected for reproduction in order to produce off-spring of the next generation. All these methods provide an effective method to increase the level of user experience while using the virtual keyboard. The layout which we have used in this project so far is just an experiment to test the efficiency and more robust will be adopted later on into the project.

As mode of access of the proposed virtual keyboard, eye gaze estimation techniques are em-

ployed. Eye gaze estimation techniques are of tremendous use in today's world. They can used in detecting disorders in psychological illness, in determining alcoholism [17], for measuring cognitive assessment, and can also be used for enhancing the skills in learning [18]. This technique can also be employed in online proctoring, which is very essential in the present scenario. Eye tracking can be implemented using a lot of equipment like diodes, Infra-Red LED, etc., however, we aim at predicting the eye gaze with the use of a camera,i.e., web cam only. Several researchers have described the use of slow deep neural networks for image processing, such as Fast R-CNN, Faster R-CNN. However, these techniques prove to be in efficient as the process has to work in real time. In this report, we are presenting an algorithm which uses the Dlib and openCV libraries of python for estimating the gaze direction. The facial key-points detector and facial detector models of the Dlib library are used extensively along with various methods of the OpenCV library [19]. This method will be improved further by employing the use of image processing, Haar cascades, and Yolo algorithms. More models like head pose detection are also being considered for the purpose of our project as a more robust method. Experiments will be conducted on the proposed method and more robust techniques will be employed into the project as progress is made.

Chapter 3

Implementation

3.1 Detection Methods

3.1.1 Haar Cascades

Haar feature-based cascade classifier is a machine learning-based approach that is used for object detection. It is a very effective method that uses a lot of positive and negative images to train a cascade function, which is used to identify objects in images. Object detection using the Haar feature-based cascade classifiers was proposed by Paul Viola and Michael Jones in [1]. This technique can be used to detect any object, and it is well known to detect face and other body parts.

We are using Haar Cascade to detect faces and eyes. To use this algorithm, we need a lot of images of faces that will work as positive images in our case and a lot of images without faces, which will work as negative images. Then We need to extract features from these images. A Haar feature considers neighboring rectangular regions in a detection window at a specific location (White and black rectangular regions), sums up the intensities of pixels in each region, and measures the difference between these quantities, which are like the convolutional kernel. There are three types of Haar features that are used.

- Edge Feature
- Line Feature
- Rectangular Feature

These features are shown in the below image .

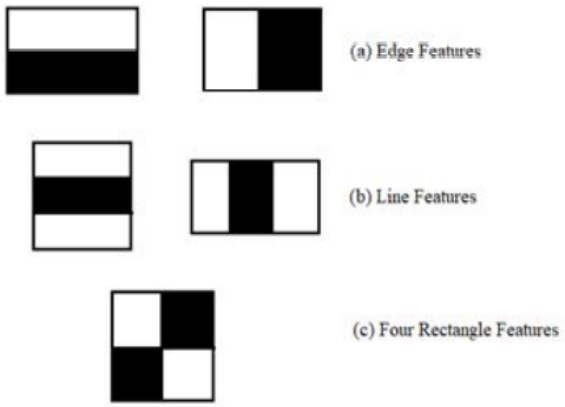

Figure 3.1: Types of Haar Features [4]

To calculate a vast number of features, all possible sizes and location of the detection window need to be used. If we calculate features for all possible cases, it will take a lot of time, and a large amount of computation will be required.

Deep learning-based detectors can be very slow, but when trained correctly, they can outperform Haar cascade and HOG + Linear SVM in accuracy and robustness. Depending upon the depth and complexity of the model, using GPU inference, they can be sped up.

3.1.2 Landmark Detection

Facial landmark identification is the task of identifying and monitoring main features on the face and at the same time being robust to various facial orientations due to movements and expressions. The proper identification of points of reference inside face images is crucial for a number of tasks related to computer vision such as blink detection. While it was an easy and simple task for humans to perform, decades of research and increasing availability of appropriate datasets and a significant improvement in computer processing capacity have been necessary to achieve near-human precision in feature detection. The various locations between the facial components and the facial contour help us in catching non-stiff and rigid facial deformities due to the movement of head and various expressions. Therefore, these locations are crucial for different tasks which require facial analysis.

Finding the face landmarks constitutes of two main tasks :

Step 1: Detection of the face in the input image.

Step 2: Detecting the principal structures of the face.

The first part has already been covered earlier. Also, the algorithm used for detecting a face in the live video does not cause any difference to the next task. The important task of this step is to get a portion that includes a face by using Haar-Cascades, HOG, or any other technique. Now , we have got a portion that surely includes a face, we can start detecting the features which are of interest such as the portion around the human eye. This is precisely the interest of our project. The dataset used for this task comprises of images which include manually marked coordinates around the different structures of the human face. Positions of facial characteristics are found out using intensities of respective pixels by training a group of regression trees. Note that no feature extraction is taking place here. The final result is a facial landmark detector, which can be used with high-quality predictions to identify facial features in real-time. Dlib library consists of a facial landmark detector that is already trained. It can be used to map facial structures on the face.

No matter what data set you're using, the very same dlib framework can be used to train a predictor of shape on your training data — this is helpful to train facial markers or even your customized shape predictors.

3.1.2.1 Detection of Facial features

The dlib landmark detector returns a shape object with the 68 (x, y) coordinates of the regions with the facial feature. We converted this object to a NumPy array so that further implementation becomes easier. Now we start to work for the feature detection part. But first, we have to identify the face in our input picture before we can detect facial landmarks. Once we get the locations of all the faces in the image, we can start applying facial landmark detection on each one of them.

For every face detected in the video frame, we use facial landmark detection, which sends us 68 (x, y) coordinates that map the image with the particular facial features. We will then transform it into a NumPy array of shape (68, 2). A bounding rectangle is drawn around the

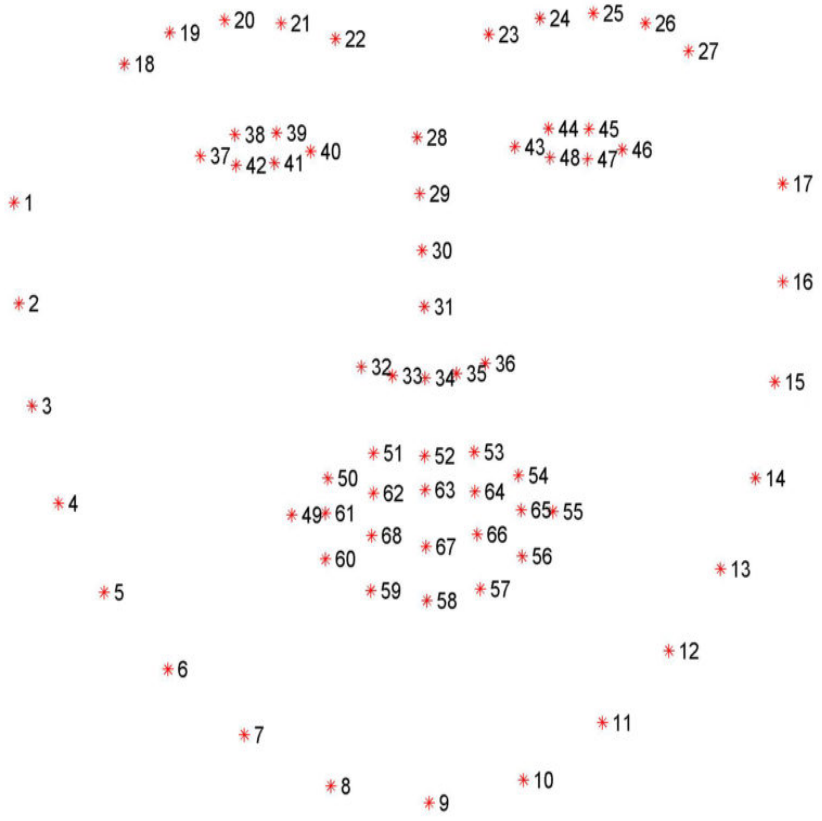

Figure 3.2: Facial Keypoints [4]

detected face on the video frame. Then we loop over the detected facial landmarks and outline both the detected eyes.

By the end of this step, we had a detected face in the frame which was bounded by a green box, and both the eyes were marked by green contours. Our next step would be to track the movement of eyes efficiently in the real-time scenario, which may include challenging situations like head movement or changing facial expressions.

3.2 Blink Detection

3.2.1 Using the Eye Aspect Ratio (EAR)

For detecting the blink of a user's eye, an EAR metric is proposed, which is also known as the Eye Aspect Ratio. It was first introduced in [4]. Conventional Image Processing techniques are typically comprised of:

1. Eye localization.

2. Detecting the sclera of the eye.

3. Tracking whether the white region (sclera) of the eye has vanished which will indicate that a blink has been performed.

The aspect ratio is a very efficient solution requiring very simple calculations based on the distance ratio between the facial features of the faces. This method is simple, effective, and easy to implement for blink-eye detection. The first step of blink detection involves calculating the eye aspect ratio and using it to find whether a blink has occurred or not. Now we keep tracking facial landmarks and detecting the blinks. Since a person can also blink without intention, we held a certain threshold to differentiate them from the voluntary blinks. To detect blinks, we are only concerned about two facial features, that is both the eyes. We are only concerned about two sets of face structures when it comes to blink-detection– the eyes. The 6 (x, y)-coordinates of each eye start on the left-hand side of the eye, then move in the direction of the clock around the rest of the region.

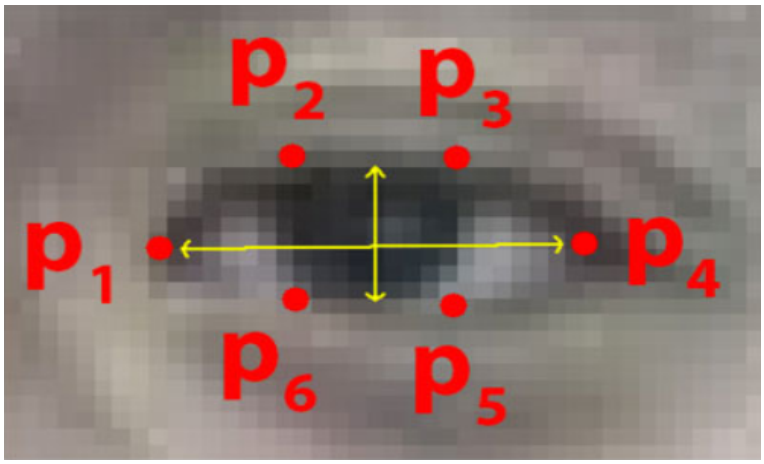

Figure 3.3: Facial Landmarks associated with the eye [3]

The width and height of these coordinates are related. A corresponding formula, the Eye Aspect Ratio (EAR), can be developed.

$$EAR = \frac{||p2 - p6|| + ||p3 - p5||}{2||p1 - p4||} \tag{3.1}$$

Where p1, . . . , p6 represents facial landmark locations for eyes in the 2-dimensional plane.

The numerator of the above expression measures the vertical distance between the vertical eye landmarks (p2,p6,p3,p5) while the denominator does the same for the horizontal eye landmark points (p1,p4). The Denominator is normalized accordingly since there are two sets of points for the numerator of the equation and only one set of points for the denominator.

Figure 3.4 shows that the eye aspect ratio (EAR) remains almost constant as long as the eye is open and falls sharply when it is closed. This observation has been used for the blink detection.

Implementation using the EAR blink detection algorithm faced several problems like brightness issues and face orientation issues, therefore, a CNN model was implemented which reduced these problems to a large extent and improved the system.

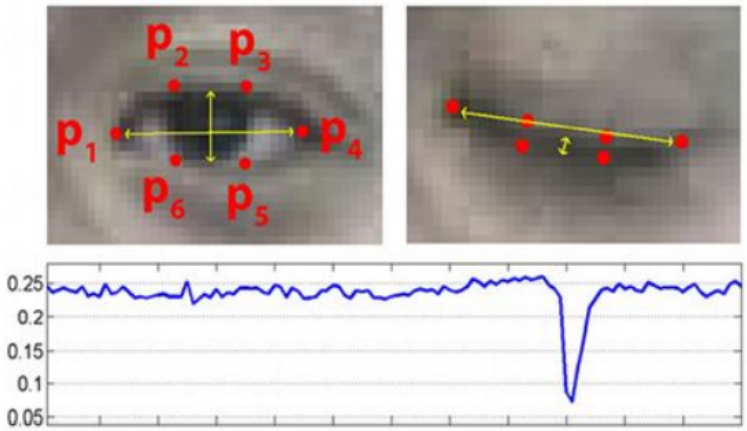

Figure 3.4: Change in Eye Aspect Ratio on blinking [3]

3.2.2 CNN Model for Blink Detection

The current version of the project uses a Convolutional Neural Network (CNN) model for blink detection. The model used has been trained over a custom dataset [21] of 7000 images. The CNN model used works better than the Eye aspect ratio method and incorporates seamlessly in the proposed keyboard layout.

The architecture of the CNN model used is:

- Convolutional layer comprising of 32 nodes with a kernel size of 3
- Convolutional layer comprising of 32 nodes with a kernel size of 3

- Convolutional layer comprising of 64 nodes with a kernel size of 3
- Fully Connected layer : 128 nodes

The final layer used in this model is also a completely connected layer comprising of two nodes. A Relu activation function is used in all the above mentioned layers except the output layer in which we have used a Softmax function.

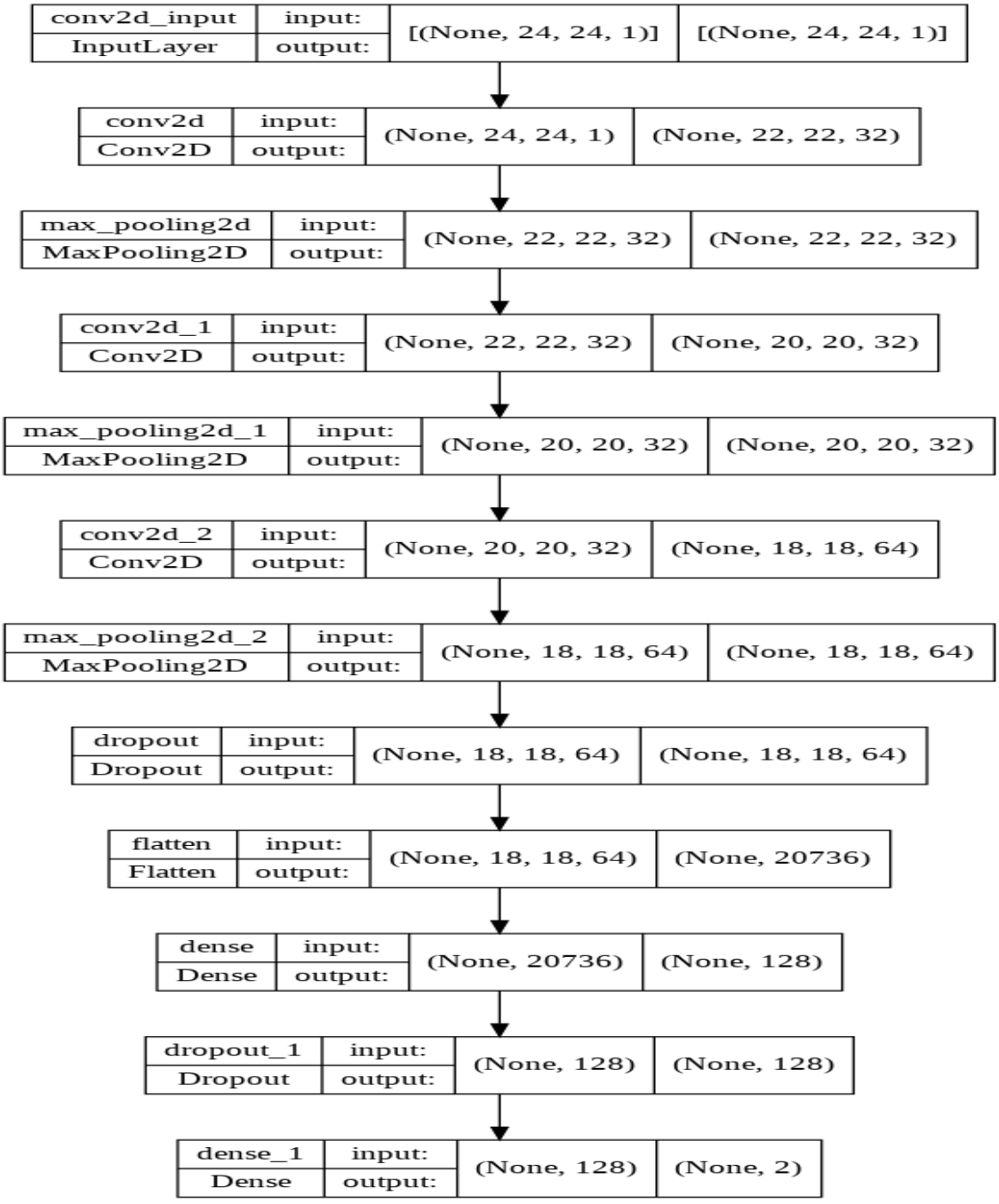

Figure 3.5: Plot of the model

3.3 Gaze Estimation

The direction of Gaze in the human eye can be estimated by the pupil and center of the eyeball where the center of the eyeball is unobservable in 2-D images. The relative position of the pupil will make the task of estimating the gaze direction very easy. However, the task of calculating this relative position is very difficult. The network which is now being mentioned has proved to be successful and will be of quite importance in our project. The network will be given a 3D input, which will then give us the gaze direction. The process of directly using the 3D input to estimate the gaze direction would be very tedious and computationally expensive, therefore, this model uses an intermediary state which will make the process of gaze estimation easier. The intermediary state will be the 2D projection of the 3D input with all of its features retained. Then, this 2D projection can be used to estimate the gaze direction. This process is explained thoroughly here [3]. The approach is basically divided into two halves 1) Converting the input image of the human eye into its 2-Dimensional Projection. 2) Using this produced projection to predict the gaze of the user. The first process is performed by the use of Stacked Hourglass architecture from Newell et al. [2]. This architecture is basically used for body pose estimation but can also be tweaked to be used for performing 2D projection. 1×1 convolutional layers are used at the end of this architecture to generate gazemaps which will be used further for estimation the gaze. The gazemaps genrated through the above process will then be used as input to the DenseNet to estimate the gaze direction.

3.4 Workline

The model implemented uses the landmark detection algorithm along with the custom CNN model for eye blink detection to control the on-screen virtual keyboard. We will be taking the frames from the webcam in real time. Then using the landmark detection algorithm, the eyes of the user will be recognized. Then blink detection algorithm (CNN) will recognize whether the user is trying to select an alphabet or not. The gaze estimation algorithm used in the implemented model is based on the idea that the eye detected will be broken into two parts and then the part of the eye with more sclera will decide the direction of the gaze (Figure 3.6).

Figure 3.6: Gaze Estimation

The Figure 3.7, shows the layout of the keyboard used. It is comprised of two parts and each part contains one half of the qwerty keyboard. By making use of the gaze estimation, the system will identify the direction in which the user is gazing, i.e., left or right, and then that part of the qwerty keyboard will be displayed on the screen.

Figure 3.7: Layout of Keyboard Used

Both the halves of the keyboard contains several squares which contain the various alphabets (as in Figure 3.7). A cursor continuously travels over these squares and when ever the user wishes to select the alphabet on which the cursor is present, the user must close his/her eyes till a buzzer is heard and then the alphabet is printed on the output panel. The keyboard resets to the left and right option tab and the user has to again choose the side for further typing. This way the user can continue to type using the virtual keyboard. The options of space and backspace are also present in the keyboard layout along with the alphabet keys. Furthermore, The various numerical keys, and special symbols can be further be added for more appropriate typing. There are some difficulties which the user faces in the beginning in getting accustomed to the method proposed but with practice the system works fine and this can be improved with the use of the

gaze estimation technique explained above. The pictures given below shows the picture of the whole console on the user's screen:

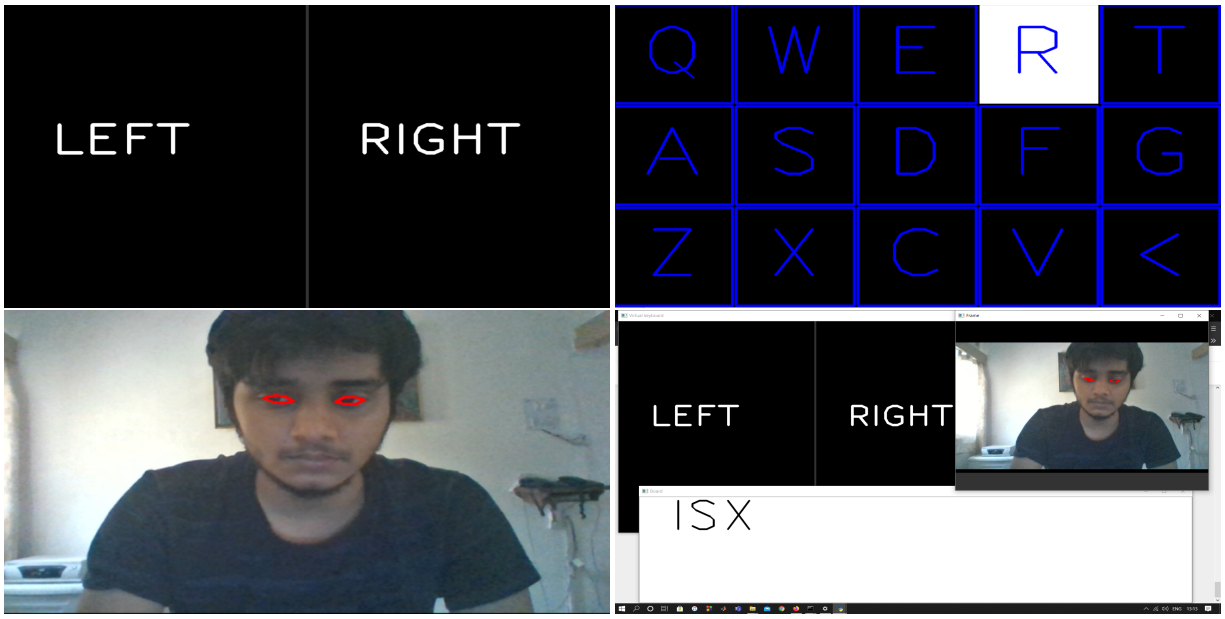

Figure 3.8: Implemented Model

Chapter 4

Observations and Results

In order to calculate the number of words a user can write per minute, we placed the laptop in front of the user's face at a distance of 60 c.m. The experiment was started with a timer for 5 minutes. The number of words a user could write in the given interval were noted. This process was repeated 10 times and then the average number of words the user could type in a minute were calculated.

The words used for the experimentation are:

" Hi ", " App ", " Data ", " Hello ", " Calculate ".

The words were used in the same order for all the test cases presented below.

Results using Eye Aspect Ratio Algorithm :

	Test1	Test2	Test3	Test4	Test5	Test6	Test7	Test8	Test9	Test10
No. of words typed in 5 mins.	3	3	2	2	3	2	2	1	3	2
Words per minute	0.6	0.6	0.4	0.4	0.6	0.4	0.4	0.2	0.6	0.4

Results using the custom CNN model :

	Test1	Test2	Test3	Test4	Test5	Test6	Test7	Test8	Test9	Test10
No. of words typed in 5 mins.	4	3	4	3	3	3	3	3	3	3
Words per minute	0.8	0.6	0.8	0.6	0.6	0.6	0.6	0.6	0.6	0.6

It was observed that the custom CNN model used provided better results than EAR algorithm. Moreover, the system was able to capture blinks for different lighting conditions and face orientations. Though, the gaze estimation algorithm implemented suffers some setbacks in different lighting conditions and with different face orientations.

Varying the distance of the face from the screen and the brightness on the face are also very crucial as it affects the process of facial landmark recognition which in turn affects the process of eye blink detection. Also, sometimes problem with eye gaze detection were faced as the user was not able to select the left or right side of the keyboard menu or selected the wrong option due to ineffectiveness of the algorithm. This can be improved with the employment of the above mentioned gaze estimation technique.

The average number of words typed according to the above table are quite less and can be improved with the user of better algorithm and proper practice of the user.

Chapter 5

Conclusion and Future Work

The demonstration shown uses dlib library, OpenCV, CNNs along with landmark detection to achieve the motivation behind this project. The outputs are promising and can be improved further by using better gaze estimation techniques. The keyboard layout model can further be optimized for improving the typing rate of the user. This software can provide an easy way of communication for the disabled people who are not able to voluntarily use parts of their body . Using the proposed on screen keyboard, these people can express themselves without the involvement of other people, thereby making them independent to some extent. The idea proposed does not require high end hardware and software devices to work, therefore it can be made easily accessible to people from all walks of life. Moreover, the idea of eye-tracking is commonly used in psychological experiments such as implicit connection test, Iowa Gambling Assignment, as well as in paradigms of glance contingency. In medical settings, it may also be necessary to monitor an individual's gaze. Research findings have shown the possible predictive strength of eye tracking in autism treatment, and in many other neurological conditions. Future uses in healthcare settings will be seeing the application of eye-tracking data in providing optimum patient care.

Newer and better designs of virtual keyboards can be researched upon to get the user more easily acquainted with the layout and for their convenience. The new keyboard layout can be based on the frequency of alphabets used in the English dictionary. Also, the virtual keyboard can be developed for various regional languages as well. Recommendation system for words can also be added to the keyboard layout so that user may get the option to select the recommended words based on the already typed alphabets. This way the efficiency of the proposed keyboard can be increased.

Bibliography

[1] Paul Viola, Michael Jones, "Rapid Object Detection using a Boosted Cascade of Simple Features," IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2001

[2] Newell, A., Yang, K., Deng, J.: Stacked hourglass networks for human pose estimation. In: European Conference on Computer Vision. pp. 483–499. Springer (2016)

[3] Avish Kabra, Chandan Agarwal, H Pallab Jyoti Dutta, M.K. Bhuyan, "Vision Based Communicator", 2020.

[4] Tereza Soukupová, Jan Čech, "Real-Time Eye Blink Detection Using Facial Landmarks," 2016

[5] G. W. Lesher, B. J. Moulton, and D. J. Higginbotham, "Optimal character arrangements for ambiguous keyboards," *IEEE Trans. Rehabil.Eng., vol. 6, no. 4, pp. 415–423, Dec. 1998.*.

[6] J. L. Arnott, "Text entry in augmentative and alternative communication," *in Proc. Efficient Text Entry, 2005 [Online]*.

[7] S. Bhattacharya, D. Samanta, and A. Basu, "Performance Models for Automatic Evaluation of Virtual Scanning Keyboards," *IEEE Transactions On Neural Systems And Rehabilitation Engineering, Vol. 16, No. 5 pp. 510-519, October 2008.*.

[8] Dasher-Information-Efficient Text Entry-Hanna Wallach, University of Cambridge/University of Pennsylvania

[9] Vijit Prabhu and Girijesh Prasad,"Designing a Virtual Keyboard with Multi-Modal Access for People with Disabilities," *2011 World Congress on Information and Communication Technologies*.

[10]] C. Aliprandi, N. Carmignani, and P. Mancarella,"An inflected-sensitive letter and word prediction system," *International Journal of Computing Information Sciences,* vol. 5, no. 2, pp. 79 – 85, 2007.

[11] J. O. Wobbrock and B. A. Myers,"From letters to words: Efficient stroke-based word completion for trackball text entry," *in In Proceedings of the ACM SIGACCESS Conference on Computers and Accessibility* (ASSETS 06), October 2225 2006, p. 29.

[12] J. Himberg, J. H. P. Kangas, and J. Mantyjarv, "On-line personalization ¨ of a touch screen based keyboard," in *Proceedings of the 8th international conference on Intelligent user interfaces*, ACM New York, NY, USA, 2003, pp. 77–84.

[13] I. S. MacKenzie and S. X. Zhang, ""The design and evaluation of a highperformance soft keyboard," in *Proceedings of the CHI 99 Conference on Human Factors in Computing Systems*, 1999.

[14] K. Kukich, "Techniques for automatically correcting words in text," in *ACM Computing Surveys*, ACM New York, NY, USA, 1992, pp. 377–439.

[15] Shumin Zhai, Michael Hunter and Barton Allen Smith, "Performance Optimization of Virtual Keyboards".

[16] Mathieu Raynal and Nadine Vigouroux, "Genetic Algorithm to Generate Optimized Soft Keyboard".

[17] Maurage, P., Masson, N., Bollen, Z., Hondt, F., 2020, "Eye tracking correlates of acute alcohol consumption: A systematic and critical review", Neuroscience Biobehavioral Reviews, 108, 400–422. doi:https://doi.org/10.1016/j.neubiorev.2019.10.001.

[18] Sun, J., Hsu, K., "A smart eye-tracking feedback scaffolding approach to improving students' learning self-efficacy and performance in a c programming course". Computers in Human Behavior, 95, 66–72.

[19] *https://github.com/vardanagarwal/Proctoring-AI Github code for the implementation proposed.*

[20] *https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3 Genetic Algorithms*

[21] *https://www.kaggle.com/datasets/serenaraju/yawn-eye-dataset-new Dataset used by the pre-trained model*