Volume: 13 Issue: 02 Pages: 4919-4924(2021) ISSN: 0975-0290

Operating Computer Cursor using Eye and Face Movements

U. Chaitanya
Department of Information Technology, Mahatma Gandhi Institute of Technology, Hyderabad-75
Email: uchaitanya_it@mgit.ac.in
Hansika Garapati
Department of Information Technology, Mahatma Gandhi Institute of Technology, Hyderabad-75
Email : hansika.garapati@gmail.com
S.Priyanka Raj
Department of Information Technology, Mahatma Gandhi Institute of Technology, Hyderabad-75
Email : priyankaraj0223@gmail.com

ABSTRACT

The advent of modern human computer interfaces has seen a considerable progress in Hands-free Human Computer Interaction (HCI) solutions. This project focuses on developing a methodology to facilitate computer cursor control for people with physical disabilities such as Quadriplegics and amputees. The proposed methodology takes real-time video input from the user using OpenCV and performs face recognition. The 68-point landmark algorithm is used to locate the various facial features which can be used for cursor control. Opening/closing the mouth based on Mouth Aspect Ratio (MAR) indicates activation/deactivation of the cursor control. The nose tip is used for controlling and moving the cursor in all 4 directions by moving the head left, right, up and down. Eye Aspect Ratio (EAR) is used to detect eyes and eye flickering. Left and right eye blinks indicate left and right clicks respectively. Squinted eyes indicate scrolling of pages, which is beneficial while working with PDFs and other such documents. The proposed system requires very basic requirements like webcam and a few Python libraries such as OpenCV, Numpy, imutils, dlib and PyAutoGUI. Thus it would help the physically disabled users to efficiently use the computer, thus eliminating the need of a physical mouse interaction.

Keywords - **Human Computer Interaction, face recognition, 68-point landmark algorithm, MAR, EAR.**

Date of Submission: May 24, 2021 Date of Acceptance: June 22, 2021

1. INTRODUCTION

Computers have become an inevitable part of human life. The traditional computer mouse is the basic approach to move the cursor on the screen. But, these systems lose their practicality when it comes to people who have lost their arms, or those suffering from Locked-in syndrome (LIS), Quadriplegia etc. There are systems in place that allow users to communicate with the computer without the need of physical contact such as Eyeball movement based cursor system, but they are not efficient in terms of usage, as the system will not be able to differentiate between intentional and unintentional eyeball movement. Hence the need for a practical and convenient hands-free cursor control system was observed. Thus we have proposed a methodology for operating computer cursor using eye and face movements, which facilitates operations such as left click, right click, move the cursor up, down, left, right, scrolling up and down. The main objective of our project is to overcome the difficulties faced by the disabled users in operating the computer system and to provide a hassle-free cursor control system that can be accessed without the need for physical contact. Our project's problem definition focuses on improving the Human Computer Interaction for users suffering from physical disabilities like Quadriplegia, Locked-in Syndrome, amputations etc. Such users face issues in interacting with a computer system. Since only their neck and head muscles can move properly, we can develop a cursor system that mainly functions on eye and face movements, hence eliminating the need to physically interact with the system.

Few technologies related to eye cursor have already been developed such as the eyeball movement based cursor, which uses an IP camera to capture the picture of an eye frame for cursor movement. In this methodology, Raspberry pi was used for pupil identification, which was the control point for the cursor. OpenCV python module was used to take snaps of the eye and it was compared against the EAR (Eye Aspect Ratio) threshold. The cursor will move on the screen depending on how the eye ball is moving, i.e. by mimicking the movement of eye. But there are few pitfalls to these existing systems, such as the system assumes all eye movements to be intentional, hence decreasing efficiency. It is difficult to move the cursor towards the end of the screen (for example to the full screen button).Another defect is the low practicality as the cursor will keep moving even if the user wants to only read the contents of the screen.To overcome these drawbacks we have introduced a methodology for Controlling the computer cursor using eye and face movements. The concepts of EAR and MAR are widely used for developing Driver Drowsiness Detection systems [1]

Volume: 13 Issue: 02 Pages: 4919-4924(2021) ISSN: 0975-0290

The proposed system takes real-time video input using OpenCV. Face detection is then performed on the input using dlib's built in detector, i.e., Histogram of Oriented Gradients (HOG) based detector which is combined with an image pyramid, a linear classifier, as well as a sliding window detection scheme. Dlib's 68 point landmark estimation algorithm is used to detect the facial landmarks, which are then used to calculate the EAR and MAR values. If the MAR value goes up, it indicates activation. If the EAR value is less than the specified threshold, it will indicate a click or scroll based on the value. The actions that can be performed using the proposed algorithm are : squinting your eyes to activate scroll mode, winking for left and right clicks, moving your head around (pitch and yaw) to move the cursor and opening your mouth to activate/deactivate cursor control. The benefit of using our system is, it provides a Hands-free cursor for the physically disabled. No external sensors or wearable hardware is required for implementation, and the upper hand is that the proposed system can be implemented on laptops/desktop computers easily.

2. RELATED WORK

Vandana Khare, et al. proposed cursor control using eye ball movement[2] . An eyeball based cursor system developed on Raspberry Pi with the pupil as the main point. The cursor moves according to the eyeball movements. The Blinks are translated into clicks based on EAR (Eye Aspect Ratio) value. For cursor movement, the system employs an Internet protocol camera to capture a picture of an eye frame.The cursor will move on the screen depending on how the eye ball is moving, i.e. by mimicking the movement of eye. The user has to move his eyes and focus on where he wants the cursor to move, and it will move accordingly. Clicks are performed by eye winking.

Sukrit Mehta , et al. proposed real-time driver drowsiness detection using eye aspect ratio and eye closure ratio[3] .This device detects a driver's level of drowsiness and sends an alert about it.. It makes use of dlib's pre-trained face detector and then captures facial landmarks. These landmarks are used to calculate the EAR value, i.e., Eye Aspect Ratio. If the EAR is lesser than the threshold value, it would indicate a state of fatigue/drowsiness of the driver. ECR value (Eye Closure Ratio) is calculate using the sleep counter (number of times EAR is lesser than the threshold value). If the ECR value exceeds the threshold value, then an alarm is generated to indicate the drowsiness state of the driver.

M. Vasanthan, et al. proposed a facial expression based computer cursor control system for assisting physically disabled person[4] .In this method, five facial gestures are used to guide cursor movement in the left, right, up, and down positions, as well as click. Four illuminated stickers are applied to the user's cheeks, forehead, and mouth. Movement of these markers is detected the coordinate changes on the input video and each of the five expressions is represented by five ASCII characters, which are in turn represented by binary numbers.

Shruti Mohanty, et al. proposed the Design of Real-time Drowsiness Detection System using dlib[5]**.** This paper deals with driver drowsiness detection using Eye Aspect Ratio (EAR) and Mouth Aspect Ratio (MAR). The system makes use of Dlib library for face and landmark detection. The values of the measured aspect ratios are evaluated with the eye and mouth threshold values, i.e., if the EAR value drops down , it is considered to be a state of drowsiness. If the MAR value exceeds the threshold value, it is taken as a yawn and the corresponding alarms are fired.

3. METHODOLOGY

3.1 PROPOSED ARCHITECTURE

The proposed "Operating Computer Cursor using Eye and Face Movements has a very straightforward and easy-to-implement architecture. The overall architecture can broadly be sectioned into five modules. The system takes in real time input from the user, pre-processes it, performs face detection and landmark detection, calculates the EAR and MAR ratios and finally translates the gestures into mouse actions on the screen.

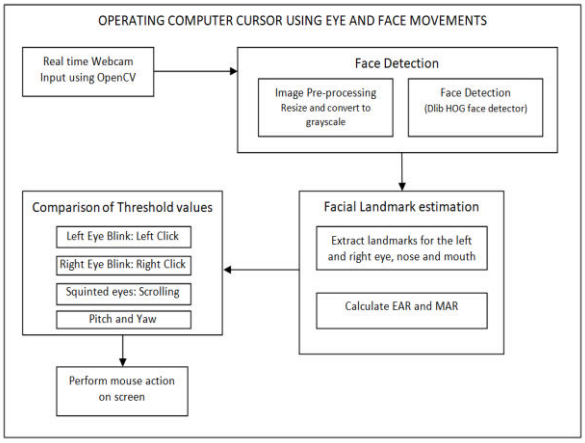

Fig 3.1.1 System Architecture

Fig 3.1.1 shows the architecture of Operating Computer Cursor using eye and Face Movements. The entire system can be viewed as the flow of action between five modules. The first module deals with taking the real time web camera input from the user using OpenCV. This input is then passed on to the next module, which pre-processes it and then performs face detection using dlib's HOG based face detector. The input video with the detected faces is now passed onto the next module which estimates facial landmarks using the 68 point facial landmark estimator. The EAR and MAR values are now calculated using the landmarks around the eyes and mouth. These calculated ratios are now sent to the next module, which compares them against the threshold values, and translates them into the corresponding mouse actions. The last module deals with performing the mouse actions on the screen using PyAutoGUI.

Volume: 13 Issue: 02 Pages: 4919-4924(2021) ISSN: 0975-0290

3.2 METHODOLOGY

The following steps are to be followed in order to implement the proposed system successfully:

- Real Time Video Input
- Image Pre-processing and Face Detection
- Facial Landmark Estimation
- Comparison of Threshold values
- Perform Mouse Action

3.2.1Real Time Video Input

Real time video input is taken from the user with the use of Python's OpenCV library. VideoCapture() function is used take the input from either the integrated camera or web camera.

3.2.2 Image Pre-processing and Face Detection

The real time video input must be preprocessed before moving on to the next step in the algorithm pipeline. Face detection can then be performed on the preprocessed input. Image pre-processing: The input needs to be pre-processed, i.e., flipped, resized and converted to greyscale. As the frame received from the web camera is automatically flipped, we need to re-flip it using OpenCV flip() function. Resizing is done using imutils's resize() function. We convert the input to greyscale using OpenCV cvtColor(), as the complexity of dealing with greyscale pixels is lesser than compared to the coloured ones.
Face Detection: Dlib's HOG algorithm based face recognition is implemented for detecting faces in the pre-processed video input. In this algorithm, the gradient orientation of the localized areas in an image are used to build the histograms. HOG is favored because it is more reliable than Haar cascades and has a lower false positive percentage. It also ignores the movement of the user and hence is more practical. A call to the get_frontal_face_detector() function is enough to activate the face detector.

3.2.3 Facial Landmark Estimation

Dlib library's 68 point facial landmark algorithm based predictor is used. This algorithm locates 68 distinct landmarks around the face. We make use of the shape_predictor() function in order to initialize it.

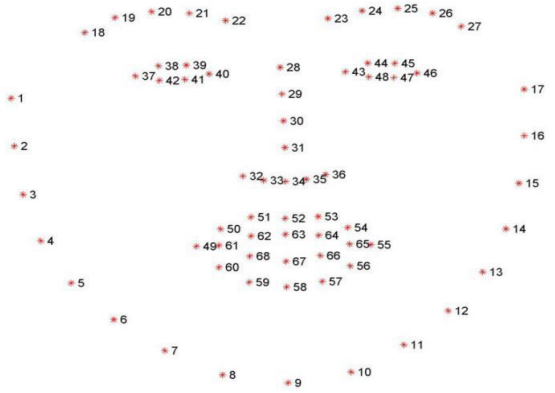

Fig 3.2.3.1 68 facial landmarks mapped on the face [6]

The 68 distinct landmarks estimated by dlib's 68 point landmark estimation algorithm are shown in Fig 3.2.3.1. Out of these, we are interested in the landmarks around the eyes mouth, and the nose

We Extract the left and right eye coordinates, mouth coordinates and the nose pointer ,then draw contours around eyes and mouth using OpenCV's drawContours() function.
Using the extracted landmarks, the Eye Aspect Ratio (EAR) and Mouth Aspect Ratio (MAR) values are calculated.

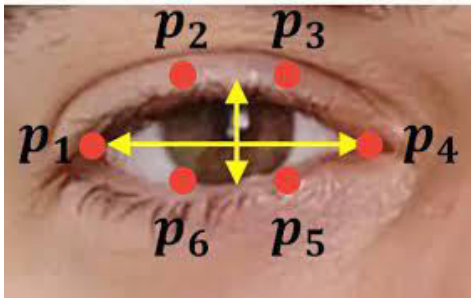

Fig 3.2.3.2 Landmarks around the eye [7]

Fig 3.2.3.2 depicts the 6 distinct landmarks around the eyes which are used to calculate the EAR value. It has two pairs of vertical landmarks and one pair of horizontal landmarks.

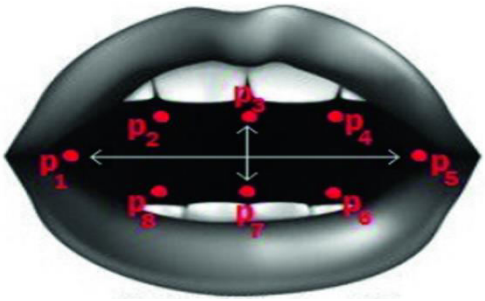

Fig 3.2.3.3 Landmarks around the mouth [8]

Fig 3.2.3.3 depicts the 8 distinct landmarks around the mouth which are used to calculate the MAR value. It has 3 pairs of vertical landmarks and 1 pair of horizontal landmarks.
EAR can be calculated using the formula:
$EAR = \|p_2-p_6\|+\|p_3-p_5\| / 2*\|p_1-p_4\|$

Similarly, MAR can be calculated using the following formula:
$MAR=\|p_2-p_8\|+\|p_3-p_7\|+\|p_4-p_6\|/ 2*\|p_1-p_5\|$

3.2.4 Comparison of Threshold values

The calculated MAR and EAR values are now compared against the threshold values and the corresponding mouth action is performed.

- If MAR is greater than the Mouth threshold value the input is accepted by the system, i.e., the cursor control

Volume: 13 Issue: 02 Pages: 4919-4924(2021) ISSN: 0975-0290

system is activated. The user's facial gestures will now be converted to mouse actions on the screen.

- If left_EAR < right_EAR, then left click is performed. Similarly, if left_EAR > right_EAR, then right click is performed.
- If the aspect ratio of both the left and right eye together is less than the threshold value, i.e., if the user squints his/her eyes, the scroll mode is implemented. The user can now bend their head down or lift their head up to scroll the document down and up respectively.
- Pitch and Yaw movements of the head will result in moving the nose pointer, and eventually the cursor in all 4 directions.

3.2.5 Perform Mouse Action

The eye and face movements performed by the user are translated into the corresponding mouse actions on the screen using PyAutoGUI library.

4. PROPOSED ALGORITHM

The main aim of this project is to bridge the gap between a disabled user and the computer system, by providing a hands-free computer cursor control system. In order to develop the proposed algorithm, the following steps are to be followed:

- Let "np" be the nose point and "ap" be the anchor point.
- "w" is width , 'h' is the height
- Mar is the MOUTH ASPECT RATIO and ear is the EYE ASPECT RATIO
- A,B,C are the distances between the mouth's vertical landmarks, D is the distance between the mouth horizontal landmarks. M,N are the Euclidean distances between the eye vertical landmarks and P is the Euclidean distance between the eye's horizontal landmarks

/*Real time Video Input*/
video=cv2.VideoCapture(0)
/*Image Preprocessing*/
frame=cv2.flip(frame)
frame=resize()
gray=cvtColor(frame)
/*Face Detection*/
Detector= dlib. Get_frontal_face_detector()
/* Landmark Estimation */
predictor = dlib.shape_predictor()
/*Aspect Ratio Calculation*/
mar= A+B+C/2D
ear=M+N/2P
/* Activating Cursor Control*/
if mar> MOUTH_AR_THRESHOLD:
then cursor_control_activated
if
MOUTH_COUNTER>=MAR_CONSECUTIVE_FRAMES
then deactivate_cursor_control

/*Comparison with threshold values and mouse action*/
if leftEAR < rightEAR:
then click(button='left')
else if leftEAR> rightEAR:
then click(button='right')
dir= direction(np,ap,w,h)
if dir== 'right':
then move right
else if dir=='left':
then move left
else if dir=='up':
  if SCROLL_MODE:
    then scroll up
  else move up
else if dir=='down':
  if SCROLL_MODE:
    then scroll down
  else move down

5. RESULTS

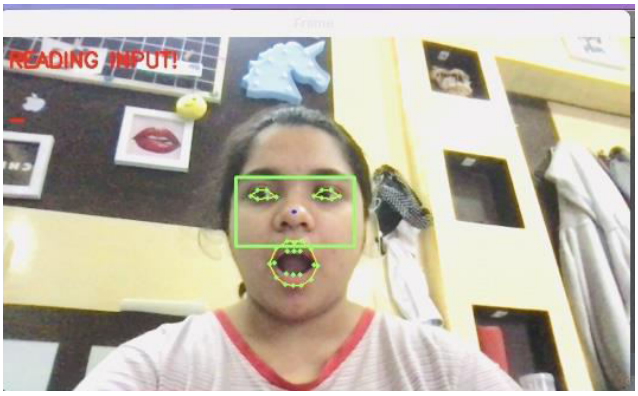

Fig 5.1 Reading Input

Fig 5.1 depicts activating the cursor by opening the mouth widely. The system is now ready to read input .If mouth is opened for the second time, the cursor will be deactivated.

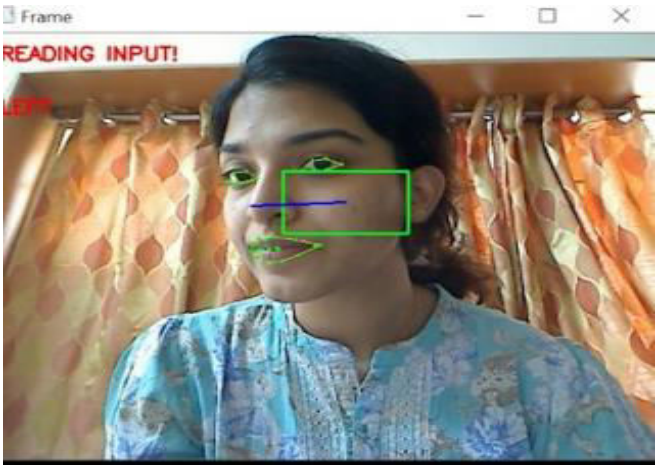

Fig 5.2 Moving cursor to the left

Volume: 13 Issue: 02 Pages: 4919-4924(2021) ISSN: 0975-0290

Fig 5.2 depicts moving the cursor towards left. If head is slightly moved left, the distance between nose pointer and anchor pointer increases and the cursor starts moving left.

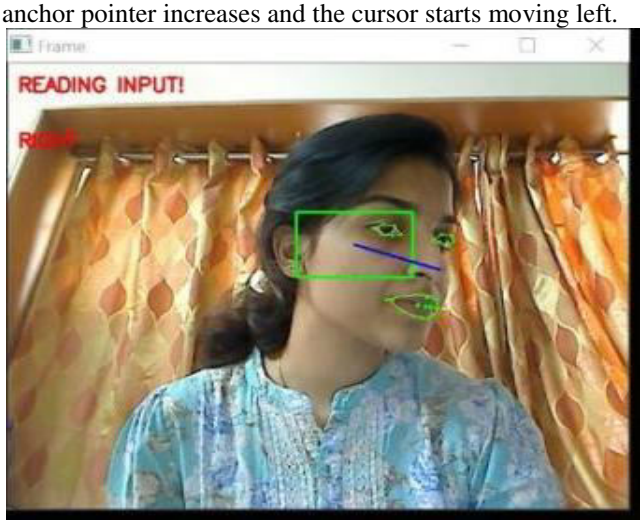

Fig 5.3 Moving cursor to the right

Fig 5.3 depicts moving the cursor towards right. If head is slightly moved right, the distance between nose pointer and anchor pointer increases and the cursor starts moving right

Fig 5.4 Scrolling Down

Fig 5.4 depicts scroll action. To activate scroll mode, user's eyes have to be squinted. Now scrolling down is performed by moving head downwards.

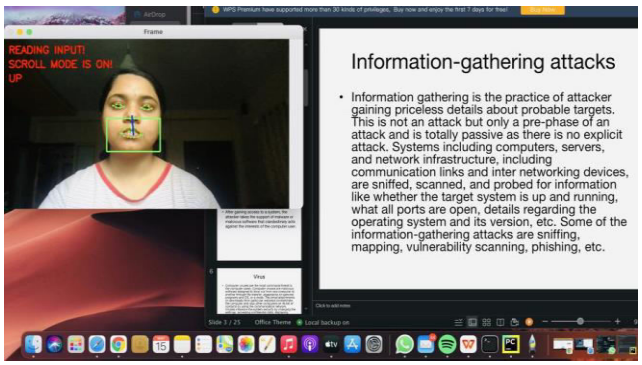

Fig 5.5 Scrolling up

Fig5.5 depicts scroll up action. While scroll mode is on, user must move their head upwards to scroll up. To deactivate scrolling user must squint their eyes again

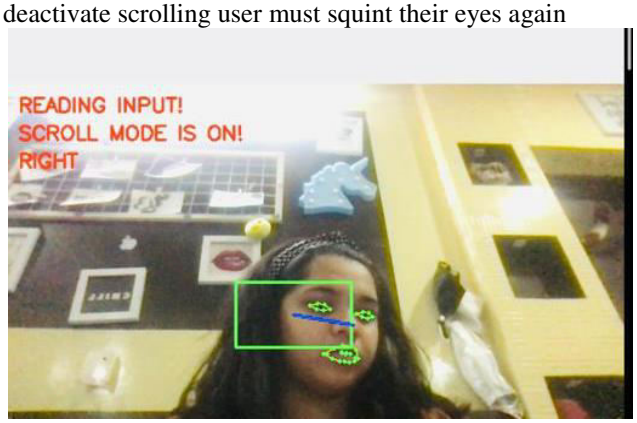

Fig 5.6 Scrolling right

Fig 5.6 depicts scrolling right. When scroll mode is on, user must move their head towards right to scroll right the document. User can similarly move their head towards left for scrolling left

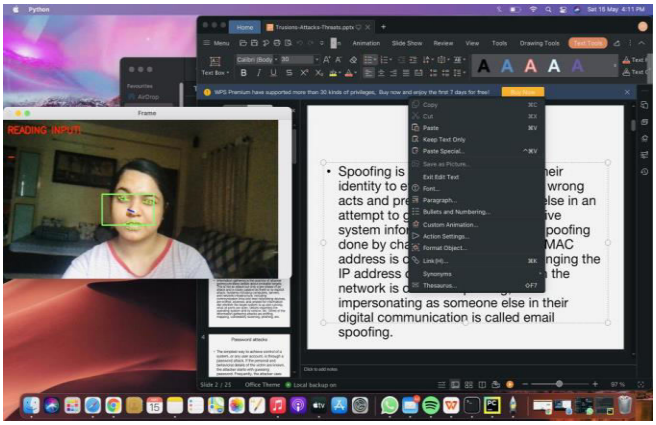

Fig 5.7 Right Click

Fig 5.7 depicts performing right click action. To perform click action user must slightly tilt their head left and wink right eye for right click and left eye for left click

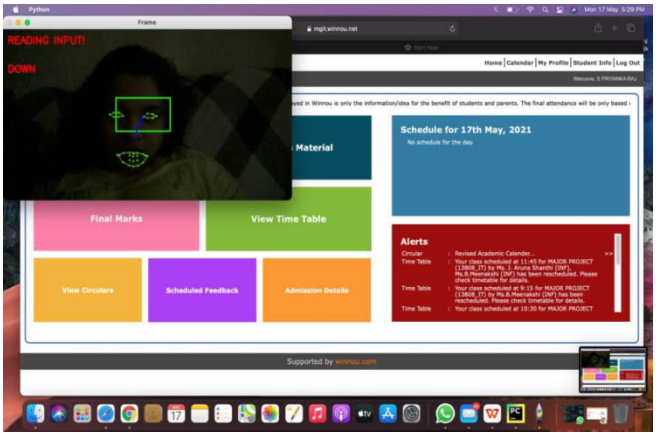

Fig 5.8 Operating cursor control system in dim light

Volume: 13 Issue: 02 Pages: 4919-4924(2021) ISSN: 0975-0290

Fig 5.8 depicts operating the proposed system in a dim light condition. The system is able to successfully operate and perform the required actions

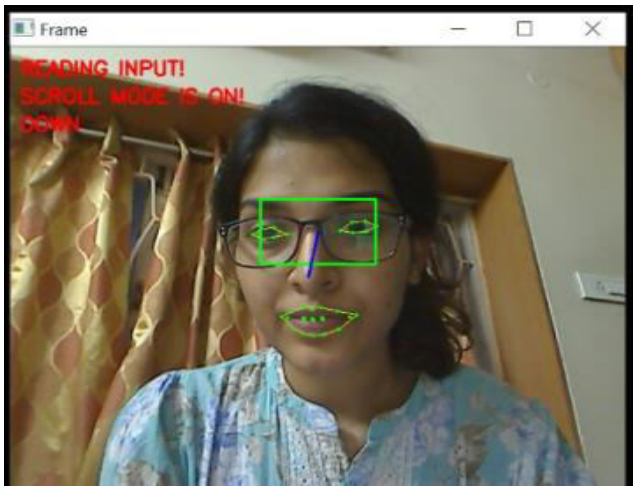

Fig 5.9 Operating cursor control system by a user wearing glasses

Fig 5.9 demonstrates operating the proposed system when the user is wearing glasses. The system is able to precisely locate the landmarks and perform the required action.

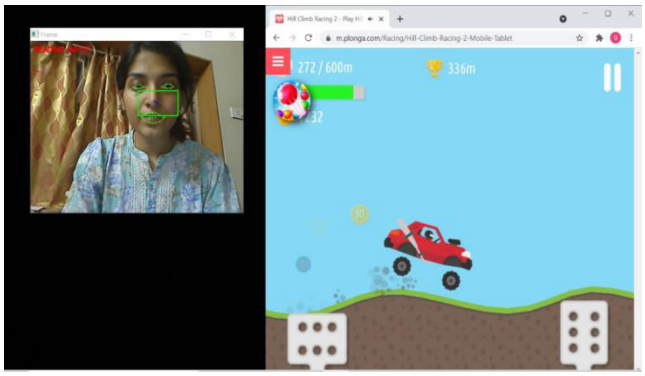

Fig 5.10 Playing a simple online game

Fig 5.10 depicts operating the proposed system in a simple online video game. The system was able to perform all the required actions successfully

6. CONCLUSION

The proposed "Operating Computer Cursor using Eye and Face movements" is a solid and practical system to offer computer cursor control for the physically disabled user. To conclude, we have built an efficient, practical and accurate cursor system which overcomes the challenges of the existing systems. This project is devised to replace the conventional computer cursor devices for the use of disabled users, which promotes operational independence. The future work could be to enhance the system to facilitate controlling the home appliances such as lights, fans, TV sets etc. This system can be infused with speech recognition technology to further increase the field of usage. The proposed system could also play a major role in virtual reality and gaming applications in the future.

REFERENCES

[1] Yuvan.M, Varun Ramesh Kumar, et.al. Real Time Driver Drowsiness Detection Using Open CV, International Journal Of Advanced Networking & Applications (IJANA), 2019, special issue - 2019 : 96-99.

[2]V. Khare, S. G. Krishna and S. K. Sanisetty, "Cursor Control Using Eye Ball Movement," 2019 Fifth International Conference on Science Technology Engineering and Mathematics (ICONSTEM), 2019, pp. 232-235, doi: 10.1109/ICONSTEM.2019.8918780.

[3]Maruthapillai, Vasanthan & M, Murugappan & Nagarajan, R. & Ilias, Bukhari & Letchumikanth, J.. (2012). Facial expression based computer cursor control system for assisting physically disabled person. Proceeding - COMNETSAT 2012: 2012 IEEE International Conference on Communication, Networks and Satellite. 172-176. 10.1109/ComNetSat.2012.6380800.

[4] Mehta, Sukrit and Dadhich, Sharad and Gumber, Sahil and Jadhav Bhatt, Arpita, Real-Time Driver Drowsiness Detection System Using Eye Aspect Ratio and Eye Closure Ratio (March 20, 2019). Proceedings of International Conference on Sustainable Computing in Science, Technology and Management (SUSCOM), Amity University Rajasthan, Jaipur - India, February 26-28, 2019

[5] S. Mohanty, S. V. Hegde, S. Prasad and J. Manikandan, "Design of Real-time Drowsiness Detection System using Dlib," 2019 IEEE International WIE Conference on Electrical and Computer Engineering (WIECON-ECE), 2019, pp. 1-4, doi: 10.1109/WIECON-ECE48653.2019.9019910.

[6] A. ROSEBROCK, FACIAL LANDMARKS WITH DLIB, OPENCV, AND PYTHON, APRIL 2017, AVAILABLE: HTTPS://WWW.PYIMAGESEARCH.COM/2017/04/03/FACIAL-LANDMARKS-DLIB-OPENCV-PYTHON/

[7] How to detect eye blinking in videos using dlib and OpenCV in Python, June 2020, Available: http://datahacker.rs/011-how-to-detect-eye-blinking-in-videos-using-dlib-and-opencv-in-python/

[8] Relangi S.P.K., Nilesh M., Kumar K.P., Naveen A. (2020) Full Length Driver Drowsiness Detection Model—Utilising Driver Specific Judging Parameters. In: Reddy A., Marla D., Simic M., Favorskaya M., Satapathy S. (eds) Intelligent Manufacturing and Energy Sustainability. Smart Innovation, Systems and Technologies, vol 169. Springer, Singapore. https://doi.org/10.1007/978-981-15-1616-0_77