Super Mario using eye gestures

Kartikey Shukla : 23679521

Abstract

Mario is a game where Mario jumps, runs, and passes obstacles across each level. If Mario touches an enemy, he dies. If he falls down a hole or misses a jump, he loses a life. Disabled people cannot play this game in such a fashion, as we need a keyboard to play the game. In this paper, I have proposed a way that can let users specifically people with disability to play games like Mario using eye gesture. The user will be able to play Mario by using their eyeball gestures. Instead of pressing the space bar key on the keyboard, the user will be able to jump by blinking their eye. Additionally, we can move Mario towards the right by moving the eyeballs towards the right instead of pressing the right arrow key on the keyboard. Similarly, we can move Mario towards the left by moving eyeballs towards the left, instead of pressing the left arrow key. The goal of my project is to make arcade games like Mario accessible to disabled people.

1 Introduction

Mario is the title character of the Mario video game franchise and the mascot of Japanese video game company Nintendo. Mario has appeared in over 200 video games since his creation. Depicted as a short, pudgy, Italian plumber who resides in the Mushroom Kingdom, his adventures generally center on rescuing Princess Peach from the Koopa villain Bowser[1]. Super Mario is an arcade game in which the player controls the character Mario, which can move towards the left or towards the right and can also jump. The game player is tasked with navigating Mario through stages, clearing hurdles at every stage. Mario can get killed by an enemy whenever it touches the enemy. On the other side, Mario can kill an enemy, by jumping on it. Each successful pass through a stage, awards the player points. Falling down the ground ends the gameplay. At the game over screen, the player is awarded with points and a high score. Mario can be

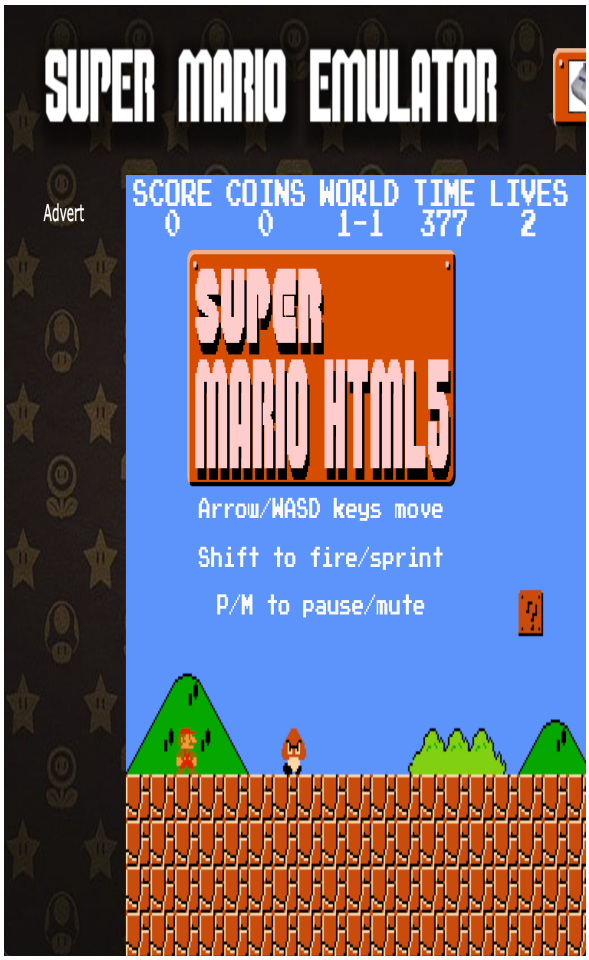


Fig1: Super Mario Emulator

played on computer as well on many consoles. For the implementation of this project, I have used an online Super Mario Emulator. On the Emulator, the UI, sounds and everything else remains the same as the original Super Mario game. Wouldn't it be great if disabled people would be able to play and enjoy famous games like Mario, just like normal people? But they can't as, these games don't provide functionalities for disabled people. To overcome

this, in this paper we are going to built a system by which we can play Super Mario using eye gestures. The game will be completely controlled by eyeball movement and the blinking of the eye. To do so, I have thought of using the concepts of Computer Vision to give the users a completely hands-free experience of playing the game.

2 Motivation

The main motivation behind this project is to add functionalities for disabled people. Gaming is a huge industry, with millions of users playing and buying games every day. Although gaming is such a big industry, they have overlooked the comfortableness of disabled people. Disabled people make up a large population and adding more functionalities to games will benefit the gaming industry, as well as will let the disabled people enjoy the experience of gaming. Thus, to make a small try from my side, I have extended the game functionality for the disabled people so that they can also enjoy playing the game.

3 Objective

The main objective of the project is to:

- To use computer vision to implement the and add the functionality.
- To detect the face of the user by using python libraries like dlib and OpenCV.
- To recognize the eye gestures of the user, through which the game will be controlled.
- To use python libraries like pynput to handle Keyboard interrupts

4 Implementation

I have used python programming language to code the functionality.

4.1 Technologies Used:

- PyCharm IDE (Python 3.9):
  I have used python programming language to create the project and have used PyCharm to compile, run and test the code.
- Libraries used:

a) OpenCv: OpenCV (Open Source Computer Vision Library) is a library of programming functions mainly aimed at real-time computer vision.
OpenCV is a great tool for image processing and performing computer vision tasks. It is an open-source library that can be used to perform tasks like face detection, objection tracking, landmark detection, and much more.[2]

b) Dlib: It's a landmark's facial detector with pre-trained models, the dlib is used to estimate the location of 68 coordinates (x, y) that map the facial points on a person's face like image below.[3]

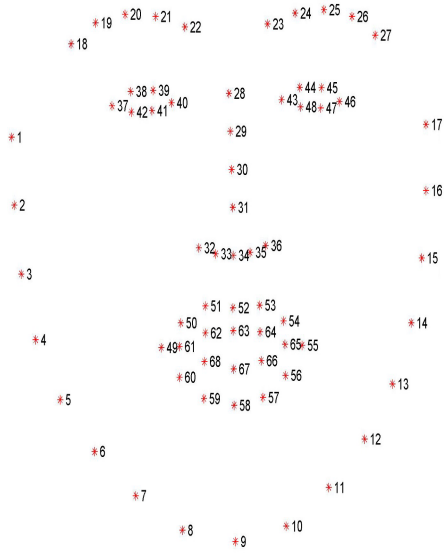


Fig2: Facial points

c) Pynput: This library allows you to control and monitor input devices. I have used this library to handle all the keyboard interrupts that will be made during the execution of the project.

d) Os: This library provides a portable way of using operating system dependent functionality.

e) Math: This library provides access to the mathematical functions defined by the C standard.

f) Time: This library provides various time-related functions.

- Super Mario Emulator: I have used an online emulator to run the game Super Mario [4].

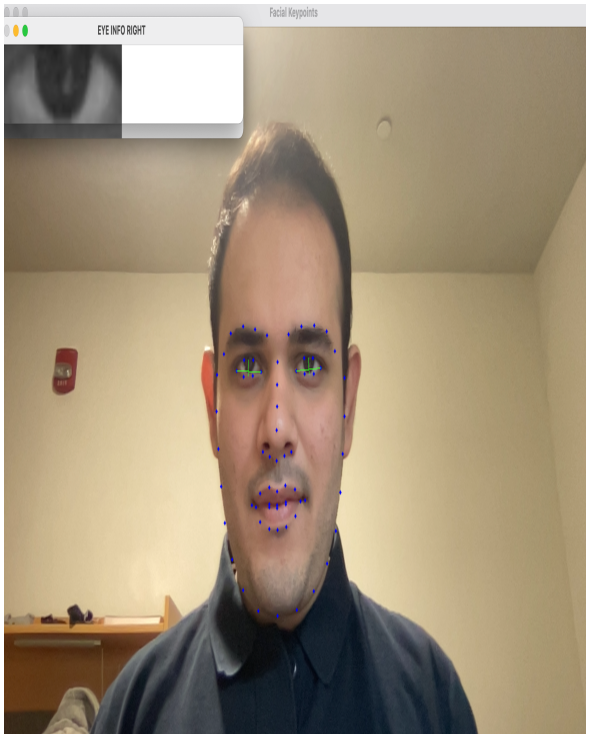
4.2 Flow of the System

As discussed above, the main objective of our system is:

- To use computer vision to implement the project.
- To detect the face of the user by using python libraries like dlib and OpenCV.
- To recognize the eye gestures of the user, through which the game will be controlled.
- To use python libraries like pynput to handle Keyboard interrupts.

4.3 Face Detection

- We need to detect face as the first step.
- To do so, we use OpenCV and dlib libraries to detect face of the user and this is used for real time eye tracking.
- For this will use a pre-trained network in the dlib library which can detect ’68 key points’ of the face.
- This pretrained network detects all the 68 points on the face correctly, as shown in the figure below.
- Once we get the landmarks of the face, we now detect the left and right eye.



.

Fig3: Facial Detection on real face

4.4 Left Eye Detection

- We calculate the all the landmarks near the left eye.
- Then, we calculate the horizonal distance between the leftmost part of the left eye and right most part of left eye by using hyplot, which calculates the Euclidean distance between the same.
- Then, we calculate the vertical distance between the top of the left eye and the bottom of the left eye by using hyplot, which calculates the Euclidean distance between the same.
- Then, we calculate the left eye open ratio by dividing the horizontal distance and vertical distance.(This is later used to check if the eye is blinking or not)
- Then, to figure out if we have looked towards left or the right we convert color image to gray image and then we convert the gray image to binary parameter with a particular threshold.

Fig4: Left Eye

4.5 Right Eye Detection

- We calculate the all the landmarks near the right eye.
- Then, we calculate the horizonal distance between the leftmost part of the right eye and right most part of right eye by using hyplot, which calculates the Euclidean distance between the same.
- Then, we calculate the vertical distance between the top of the right eye and the bottom of the right eye by using hyplot, which calculates the Euclidean distance between the same.
- Then, we calculate the right eye open ratio by dividing the horizonatal distance and vertical distance.(This is later used to check if the eye is blinking or not)
- Then, to figure out if we have looked towards left or the right we convert color image to gray image and then we convert the gray image to binary parameter with a particular threshold.

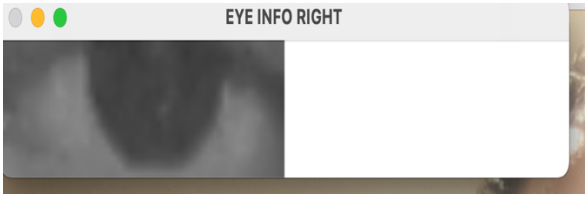


Fig5: Right Eye

4.6 Blinking and Keyboard Interrupts

- We find if the eye is blinking by calculating the ratio of sum of right eye open and left eye open divided by 2. If this ratio is more than the set threshold, we will use the keyboard.press function to press the key, i.e spacebar .
- We will use the keyboard.press function to press the key, i.e right or left based on the direction which is most indicative of the eyeball moving(more number of 1's above the threshold than 0's).

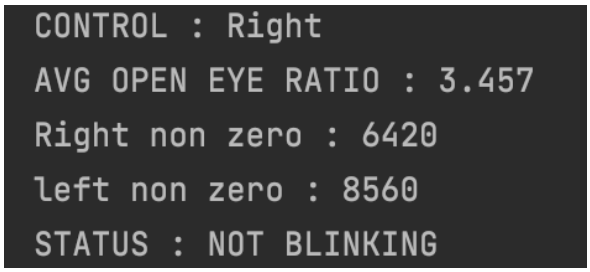


Fig6: Status of Eye

4.7 How to run the game

1. Run the main python file.
2. Before running the file, open the Super mario simulator by using a web browser.
3. Then, after running the main python file, open the browser window and start using eye gestures to control Mario without using the keyboard.

.

5 Evaluation

Evaluation of the system is a major step of project building. This step is important, as it points out all the problems in our project and helps us to evaluate the accuracy of our project.
For our project evaluation, we have used qualitative evaluation method.
I have selected five users to test the system. Each user will move their eye to the left , move their eye to the right and try to jump using blink five times each.
Below are the computed results from the testing.

User	Left	Right	Jump	Accuracy
User1	3	4	3	66.67%
User2	2	3	4	60%
User3	4	3	3	66.67%
User4	2	1	4	46.67%
User5	1	3	5	60%

The table contains values when Mario successfully moved to the left or right or jumped respectively.
As seen above, the total accuracy of our system is about 60%.

6 Future Work

We can improve the scope our project and make it more general by adding more additional functionalities such as jump when we open our mouth, etc.
Also, we can add voice-based recognition using speech recognition library to start and end the game, etc.
We need to do extensive research, on how we can improve the accuracy of the system.

7 Conclusion

Thus, we have successfully used computer vison to create a functionality by which we can use eye gestures to move to the right, left and jump our character while playing arcade game like mario. We used libraries like dlib and OpenCv to make this possible. After the evaluation of the system, we come at a conclusion, that our system can still improve.

References

[1] https://en.wikipedia.org/wiki/Mario.

[2]https://en.wikipedia.org/wiki/OpenCV

[3] http://dlib.net/

[4] https://supermarioemulator.com/supermario.php