AI Eye-track enabled video game for paralyzed patients

Dr. Yangyang Tao and Dr. Junxiu Zhou

School of Computing and Analytics | Northern Kentucky University

Abstract:

Paralyzed patients caused by stroke or accidents lost certain functions, such as movement or thinking. The phenomenon responsible for paralyzed patients' recovery is neuroplasticity. Recently, video games have offered an effective form of therapy because they help keep users engaged and encourage repetitive practice of specific skills. For this project, we want to develop AI eye-track-enabled video games for patients. The design uses AI models to capture the eye movements of the patients and then convert those movements to directions for video games. In this way, the patient's brain nerves function actively towards potential recovery.

Introduction:

System Architecture:

More and more paralyzed patients caused by stroke or accidents are seeking help from video games to recover. Some of them even formed an expert team that had amazing achievements [1]. Some recent works like human-computer interfaces have been developed to help paralyzed patients recover. However, for some patients, the human-computer interface is still expensive and unstable. In this proposed work, we want to develop an AI eye-tracking-enabled video game for paralyzed patients. First, the eyeball movement will be captured by the camera on the computer. Then, the data will be processed through an AI model to produce the coordinates on the computer screen. Lastly, the coordinates will be used to control the video game. Since the AI model is used to handle eye tracking, no other auxiliary appliances are needed for this process. So, our design only requires a computer with a workable camera.

System Architecture Cont.:

Figure 1 shows our AI eye-tracking system. The platforms can be any device that has a camera attached. The face information collected from the camera will be processed by MediaPipe [2]. The intermedia data is the FaceMesh [3] dataset. We will use the subset from FaceMesh. The reason is we want to use less data to achieve better training performance. The main component of the subset of face information contains the basic iris data (pupil position) and the basic face position. Then we train a machine learning model to predict the eye positions. After the training process. The trained model can be deployed to any device that has a runtime environment. The last step is feeding the eye positions to the video games to control the game. This project has two tasks: (1) machine learning model preparation. (2) interface for feeding the eye-tracking output to video games. The detailed timeline for these two tasks is shown in the next subsection.

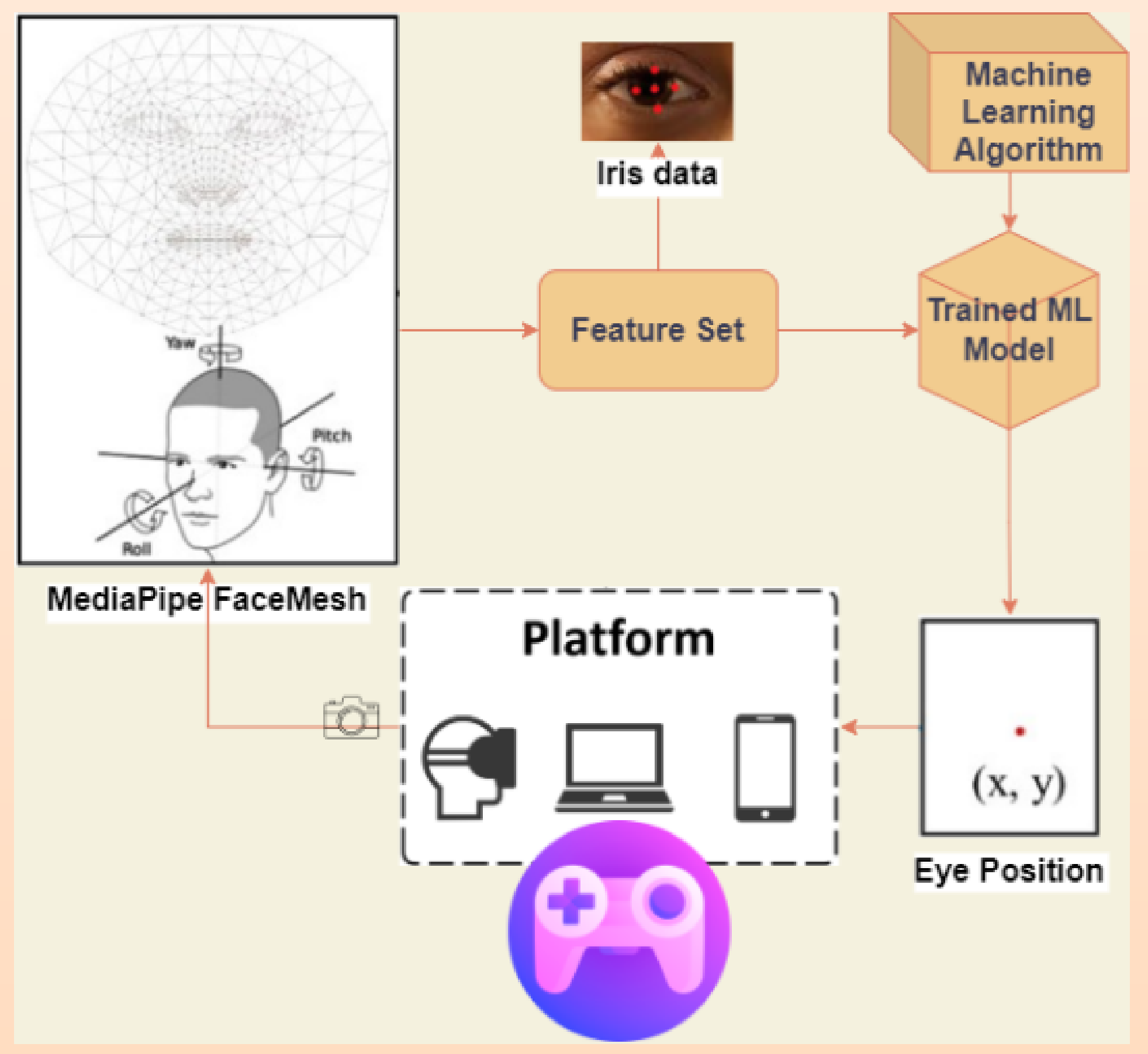

Fig. 1 The eye tracking system

Implementation:

- **Eye tracking:** The data set of eye features contains almost all of the information obtained from the MediaPipe, FaceMesh, and Iris, including the center of the user's face, the bounding box surrounding the user's face, face posture, facial geometry, and all the eye-related features in the landmarks from FaceMesh, and Iris eye coordinates. The iris-eye coordinates are the most important ones.

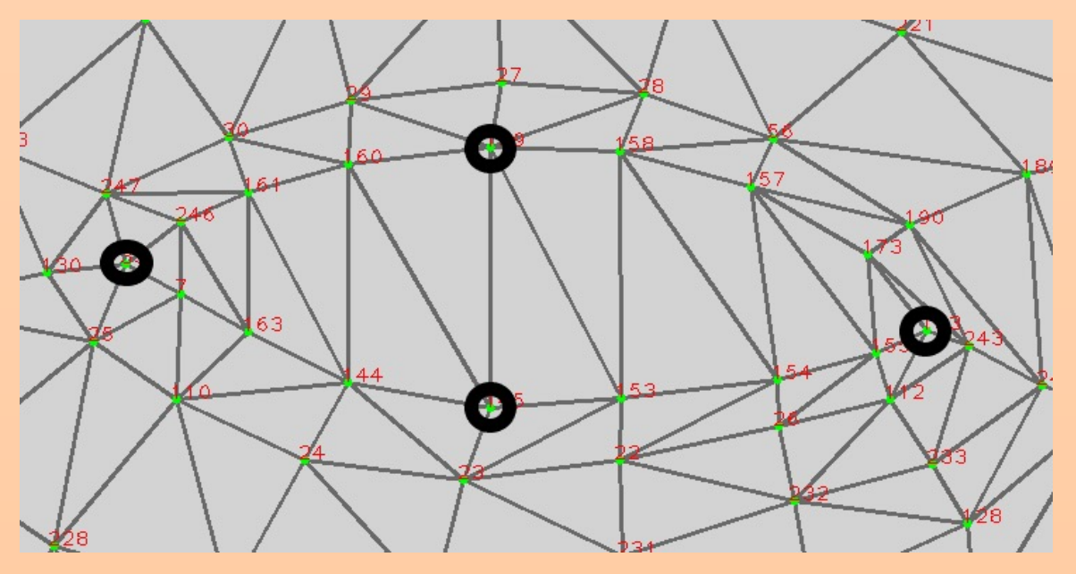

Fig. 2 Iris eye coordinates

- **Machine learning models:** to evaluate the performance of the system, we employed four commonly used machine learning algorithms, linear regression, ridge regression, LASSO regression, and Artificial Neural Networks. We compared the performance of the models in a preliminary experiment and show the result in the next section.
- **Eye tracking games:** The game selection and incorporation of the eye tracking model into the game is the next step. We will select a suitable game for demonstration purposes. The best machine learning model will be used in the game to predict the eye concentration of the patients

Experimental Results:

The project is divided into two parts. For the current stage, we have implemented the first part which is the machine learning model evaluation. The result is shown below. The best model is ridge regression. In the table, four important factors are displayed. MAE (mean absolute error), MSE (mean square error), training time, and testing time. The MLP has the largest MAE which means it is not the suitable model that we should use for this type of task. However, we believe when the volume of the dataset grows, MLP might have the potential to outperform the other models.

Model	MAE	MSE	Time_Train	Time_Test
RR	229.152458	83880.176637	0.145292	0.000001
Lasso	230.887122	83319.622588	1634.004820	0.000001
LR	235.321159	94386.376270	0.450090	0.000001
MLP	248.179031	108001.117188	2700.745783	0.000058

Conclusion:

In this project, we implemented the basic architecture of our design. The first step that build basic machine learning models for eye tracking has been finished with a promising result. Through the experiment, we tested the performance of different machine-learning algorithms. This is an important step for us to implement the next step, incorporate the machine learning model into a game.

Future Direction:

We will incorporate the machine learning model into a video game to test the performance.

References:

[1]https://www.cnet.com/science/features/the-new-gods-of-esports-are-paralyzed-from-the-neck-down/
[2]https://google.github.io/mediapipe/
[3]https://google.github.io/mediapipe/solutions/face_mesh.html