Eye-Clicker

V and V

SFWR ENG 4G06 / MECHTRON 4TB6
GROUP 8

Adam Gordon 400036676
Bowen Yuan 400005913
Jacky Chan 400015317
Jeffrey Sun 400022051
Leo Li 400032949
Tongfei Wang 001437618

Contents

1 Revision History

Date	Revision	Authors
Feb 16, 2020	0	All group members

2 Purpose

The purpose of the EyeClicker is to give our users an alternate method to use their computer. With our device, a user will be able to control their computer or laptop without having to use a mouse, and instead use their eyes. This is especially helpful to users who have limited mobility with their arms, those who are injured, or simply someone who wants to rest. Given an alternate method of using a computer, our device can help more people use a computer comfortably and be connected to the digital world we live in. Eye tracking has become fairly popular over recent times, and our EyeClicker will utilize this concept to provide a full user experience that mimics the behavior of a mouse. The EyeClicker will identify where the user is looking at to position the cursor to that spot on the screen. It will also track a unique set of simple eye movements/actions as well as certain voice commands to deliver the appropriate response. Whether the user wishes to click, drag or enter text the EyeClicker will provide all those functionalities simply though tracking the eye and voice control.

3 Scope

In-scope functionality items for the EyeClicker including the following:

- Calculate the position the user is looking on the computer display
- Calibration system to increase the accuracy of the EyeClicker
- Allows user to perform a left-click as well as dragging with certain special actions that can be performed with voice commands
- Be able to let user disable the EyeClicker system to prevent misoperation
- Provide a functional GUI to enable/disable the EyeClicker and enter the calibration system

The following items are out of scope: Allows user to control the cursor with voice input

4 Background

The project will be based around tracking user's eye movements and moving the cursor to where the user is looking on the screen. This will be achieved through image processing, more specifically, human eye recognition. To aid with fully replacing a mouse, the project will also have a voice control system where the user will be able to input mouse action commands. Lastly, the project include a GUI where the user could initiate a calibration system. The user would then be prompted to follow the calibration steps and in result would receive greater accuracy with the EyeClicker.

5 Test Cases

5.1 Testing Plans and Testing Factors

To test the Eye-Clicker software, we decided to test each of the 7 modules individually. This allowed us to divide and conquer the testing task. For each of the individual modules, we broke down the module into miniscule sections that each have a single functionality/purpose. This made it easy to determine what input to feed into the system and what to expect as an output. Thus, we can use black box testing to see if the performed output matches what we expect to see on a successful trial.

Component	Test Plan Factors
Eye-tracking module	The eye-tracking module is the largest and all-encompassing module that handles the most critical portion of the software. This means that every single component needs to be tested thoroughly to ensure that the components always produce the expected result so that it does not interfere with the performance of the other components. In order to achieve this, we need to test various scenarios that may occur as well as multiple components working synchronously. The overall idea behind testing this module is to ensure that the software can receive a steady input video stream and track the gaze of the laptop user consistently.
Cursor module	The main goal of testing the cursor module is to ensure that the simple move cursor and retrieve cursor position work accordingly under all conditions, including edge cases. Since it is a small component, we need to guarantee that it has no errors that could affect our judgements about the eye-tracking module. If an error were to occur in the cursor module, we could easily oversee it as an AI training error. This would be catastrophic as it would greatly reduce our ability to completely all the components in a reasonable timeframe. Therefore, we cannot afford to have errors in these trivial components.
Voice module	The testing for the voice module is very important as some functionalities in the software can only be accessed through voice commands. Since there are huge variations in the accent, tone, speed from person to person, we suggest that the user does not speak too quickly and attempt to articulate to the best of their ability. To mimic this situation, we had all our group members test the voice module to ensure that there is enough flexibility to suit more than one specific user. Overall, the idea is to test that the voice module can receive the voice input, transcribe the audio message into text, and match the text to the correct voice commands.
Mouse Action module	The voice action module is tested through feeding it left and right click commands and testing if the correct action was performed. From what we discovered, the best place to run this test was on the X to close any webpage or application. When you right click the X, a menu pops up, and when you left click the X, the page closes.

Calibration module	The calibration module is very challenging to test as it is hard to determine the difference between the original AI model and the newly trained AI model since it is a black box even for us. So far, what we can verify from the tests are that the data generated from the calibration was successful and that we were able to initialize training with the new data.
AI module	Testing the AI module is mainly verified through the eye-tracking module where we can see the user's gaze being mapped appropriately to the computer screen. This is the best way to test whether the training was successful. However, we still conducted many test cases to ensure that the shape of the data can be understood by the training algorithm and that the output of the AI matches the shape we want.
GUI module	The GUI module is simply a UI and most of the testing is visual. This does not require any unit testing as we simply just want to visually identify that the appropriate action has occurred.

5.2 Eye-Tracking Module

Eye-tracking module test plan

Test Number	Description	Requirement Reference	Inputs	Expected Outputs	Actual Outputs	Results
1	Testing video input is captured by webcam	Performance Requirement 1	Webcam feed	Video stream is alive	Video stream is alive	Pass
2	Test that eye-tracking activates upon setting eye_tracking_activated flag to true	Performance Requirement 1	Setting the flag to true	Eye-tracking beings	Eye-tracking beings	Pass
3	Testing that pupil and landmark points are detected	Performance Requirement 1	Webcam feed	Software produces respective points	Software produces respective points	Pass
4	Test that software can determine blinking based on landmark points	Performance Requirement 3	Webcam feed	Detect blinking	Detect blinking majority of the time	Pass
5	Testing that eye-tracking is predicting cursor positions	Performance Requirement 1	Webcam feed	Points are within 1920x1080	Points are within 1920x1080	Pass
6	Test that cursor does not move while eye is blinking	Performance Requirement 3	Webcam feed	Points are ignored	Points are ignored	Pass
7	Visually test with a user that the eye-tracking matches where the user is looking	Performance Requirement 1	Webcam feed	Cursor is moved to where the user is looking	Cursor is moved to where the user is looking	Pass
8	Test tracking if a user's eye is no longer moving	Performance Requirement 2	Webcam feed	Cursor does not move	Cursor does not move	Pass
9	Test timer to track how long the user's eyes have stayed stationary	Performance Requirement 2	Webcam feed	Initiate new timer	Need to implement	Fail

5.3 Cursor Module

Cursor module test plan

Test Number	Description	Requirement Reference	Inputs	Expected Outputs	Actual Outputs	Results
1	Test cursor is moving to the specified points	Performance Requirement 1	Window co-ordinates	Cursor moves to specified location	Cursor moves to specified location	Pass
2	Test retrieving cursor position	Performance Requirement 6	Get cursor location	Actual cursor x, y positions	Actual cursor x, y positions	Pass

5.4 Voice Module

Voice module test plan

Test Number	Description	Requirement Reference	Inputs	Expected Outputs	Actual Outputs	Results
1	Test voice input is received	Performance Requirement 5	Mic recording	Recording is received by the system	Recording is received by the system	Pass
2	Test that voice input can be transcribed into text	Performance Requirement 5	Mic Recording	Text generated matches speech	Text generated matches speech	Pass
3	Test that text commands can be matched with system commands	Performance Requirement 6	Transcribed text	Command matched	Command matched	Pass
4	Test that the respective system action is performed upon	Performance Requirement 6	Transcribed text	Mouse action performed	Mouse action performed	Pass
5	Test deactivation of eye-tracking with speech	Performance Requirement 7	Transcribed text	Disable eye-tracking	Need to implement	Fail
6	Test that random speech does not invoke an action	Performance Requirement 6	Transcribed text	Ignore random command	Ignore random command	Fail

5.5 Mouse Action Module

Mouse Action module test plan

Test Number	Description	Requirement Reference	Inputs	Expected Outputs	Actual Outputs	Results
1	Test that right click is performed on right click requested	Performance Requirement 6	Right click command	Right Click	Right Click	Pass
2	Test that left click is performed on left click requested	Performance Requirement 6	Left click command	Left Click	Left Click	Pass
3	Test that no action is performed when there is no command	Performance Requirement 6	Null	No action	No action	Pass

5.6 Calibration Module

Calibration module test plan

Test Number	Description	Requirement Reference	Inputs	Expected Outputs	Actual Outputs	Results
1	Test that calibration initialization is successful	Performance Requirement 4	Init calibration Set to true	Calibration starts	Calibration starts	Pass
2	Test that the calibration process runs through all the points	Performance Requirement 4	Init calibration is true	Calibration executes	Calibration executes	Pass
3	Test that calibration records the data for calibration points	Performance Requirement 4	Webcam feed	Data is recorded	Data is recorded	Pass
4	Test that calibration initiates training the model to improve accuracy	Performance Requirement 4	Pupil and landmark data	AI model starts to train	AI model starts to train	Pass

5.7 AI Module

AI module test plan

Test Number	Description	Requirement Reference	Inputs	Expected Outputs	Actual Outputs	Results
1	Test that AI module can initialize with the give data	Performance Requirement 1	Pupil and landmark data	AI model starts to train	AI model starts to train	Pass
2	Test that the AI module trains a model successfully	Performance Requirement 1	Training Set	Training completed	Training completed	Pass
3	Test that the AI module can predict reasonable x, y co-ordinates	Performance Requirement 1	Test Set	Can make reasonable predictions	Can make reasonable predictions	Pass

5.8 GUI Module

GUI module test plan

Test Number	Description	Requirement Reference	Inputs	Expected Outputs	Actual Outputs	Results
1	Test that the software base page initializes	Performance Requirement 8	Initialize program	UI appears	UI appears	Pass
2	Test that the buttons on the UI can initialize the calibration and eye-tracking modules	Performance Requirement 9	Click buttons	Module starts	Module starts	Pass

6 Testing Philosophy

6.1 Code Walkthroughs

Since our group has 3 software and 3 mechatronics students, we divided ourselves into two teams based on our Engineering streams. The software individuals mainly worked on the AI, Calibration, and Eye-tracking modules while the mechatronics teammates worked on Eye-tracking and the remaining modules. Therefore, we needed to have weekly meetings to review the progress among team. Since we did not work on the exact piece of code the majority of times, we had many code walkthroughs to ensure all team members are aware of the progress of other members. This also helped us stay on track as there was is certain degree of responsibility for each team member as we needed to present what progress we had made.

6.2 Code Reviews

In each of the code walkthroughs, if there were major code changes performed by any other two groups, a member of the other group will review the code to provide a fresh pair of eyes and see if everything looks reasonable. This also helps us understand how to integrate the various components together as we have an idea of how all the code is implemented.

7 Traceability

Traceability "matrix" table

Test Cases:	Performance Requirement:
Eye-tracking Module	Performance Requirement 1
	Performance Requirement 2
	Performance Requirement 3

Test Cases:	Performance Requirement:
Cursor Module	Performance Requirement 1
	Performance Requirement 6

Test Cases:	Performance Requirement:
Voice Module	Performance Requirement 5
	Performance Requirement 6
	Performance Requirement 7

Test Cases:	Performance Requirement:
Mouse Action Module	Performance Requirement 6

Test Cases:	Performance Requirement:
Calibration Module	Performance Requirement 4

Test Cases:	Performance Requirement:
AI Module	Performance Requirement 1

Test Cases:	Performance Requirement:
GUI	Performance Requirement 8
	Performance Requirement 9