DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
THE UNIVERSITY OF TEXAS AT ARLINGTON

# ARCHITECTURAL DESIGN SPECIFICATION
# CSE 4316: SENIOR DESIGN I
# SPRING 2019

## EYET GUYS
## THE EYE TRACKER

JONATHAN EL-KHOURY
PAUL WAFULA
GEORGE HINKEL
DHARAMPREET GILL
JAY DENTON

## Revision History

| Revision | Date | Author(s) | Description |
|---|---|---|---|
| 0.1 | 04.08.2019 | JE, GH\| | document creation |
| 0.2 | 10.05.2015 | AT, GH | complete draft |
| 0.3 | 10.12.2015 | AT, GH | release candidate 1 |
| 1.0 | 10.20.2015 | AT, GH, CB | official release |
| 1.1 | 10.31.2015 | AL | added design review requests |

# CONTENTS


**1 Introduction** **5**

**2 System Overview** **6**
- 2.1 Hardware Layer Description . . . 6
- 2.2 Software Layer Description . . . 6
- 2.3 Eyetype Layer Description . . . 6

**3 Subsystem Definitions & Data Flow** **7**

**4 Hardware Layer Subsystems** **8**
- 4.1 Controller Subsystem . . . 8
- 4.2 IR Camera + IR Pass Filter Subsystem . . . 9
- 4.3 IR Light Subsystem . . . 10
- 4.4 Character LCD Subsystem . . . 10

**5 Software Layer Subsystems** **12**
- 5.1 Model Controller/State Machine Subsystem . . . 12
- 5.2 Eye Tracking Image Processing Subsystem . . . 13
- 5.3 Calibration Subsystem . . . 14
- 5.4 Server Subsystem . . . 15

**6 Eyetype Layer Subsystems** **17**
- 6.1 Input Layer . . . 17
- 6.2 Control Layer . . . 17
- 6.3 Graphical User Interface Layer . . . 18

# List of Figures


1 The Eye Tracker Overview Diagram . . . 6
2 Subsystem Overview with Data Flow Diagram . . . 7
3 TX2 Subsystem Diagram . . . 8
4 IR Camera + IR Pass Filter Subsystem Diagram . . . 9
5 IR Light Subsystem Diagram . . . 10
6 Character LCD Subsystem Diagram . . . 11
7 Model Controller/State Machine diagram . . . 12
8 Eye Tracking Image Processing diagram . . . 13
9 Calibration Subsystem diagram . . . 14
10 Server Subsystem diagram . . . 15
11 Input subsystem description diagram . . . 17
12 Control subsystem diagram . . . 18
13 GUI subsystem description diagram . . . 18


# List of Tables


2 Subsystem interfaces . . . 9
3 Subsystem interfaces . . . 10
4 Subsystem interfaces . . . 10
5 Subsystem interfaces . . . 11
6 Subsystem interfaces . . . 13
7 Subsystem interfaces . . . 14
8 Subsystem interfaces . . . 15
9 Subsystem interfaces . . . 16
10 Subsystem interfaces . . . 17
11 Subsystem interfaces . . . 18
12 Subsystem interfaces . . . 19

# 1 Introduction

Our system consists of two categories, the Hardware layer and the Software layer. The hardware layer consists of various components that will connect with NUC. These components include the IR camera + IR pass filter, IR Light, and Character LCD. The software components include the eye tracking image processing, calibration, and server subsystems.

As for the hardware layer. There will be only one IR camera we will be using since the client is already used to using a single camera. This also make it easier on our end, as we only need to calibrate to one pupil, instead of calibrating to two. This will filter out just the pupil it self from the rest of the eye. The other component we will be using is the IR Light. This will allow eye tracking in a dark environment. The other component that is included in our hardware layer is the Character LCD, which will be used for debugging purposes.

The other layer is the software layer, which consists of all of the software components that will be run by the model controller. The components that will be controlled by the state machine are eye tracking image processing, calibration, and server subsystems. The Eye tracking image processing software will receive video stream from the filtered IR camera. The calibration will be used whenever the client initially needs to set up the device in the early mornings or whenever they move or make any other adjustments. The server is the software that will communicate between the TX2 and the NUC, which is what the Eye Type team will use to receive data from us. (this is a comment at the end of semester: we eneded up never using the TX2, so we were just using the NUC the calculate the data and the eyetype team received it on NUC also).

# 2 System Overview

The Eye Tracking Remote has two main layers: the Software Layer and the Hardware Layer. The Software Layer implements the eye-tracking algorithm using the image data imported from the Hardware Layer. It also calibrates gaze vector to screen coordinates conversions and wraps up the analyzed data in a packet which will be sent to EyeType's portion of the project. The Hardware Layer consists of the IR light and the camera to capture the user's pupil movements and sends this data to the Software Layer. It also sends the analysed data packet imported from the Software Layer to the EyeType system. Meanwhile, the analyzed data is also shown on the Character LCD for debugging purposes.

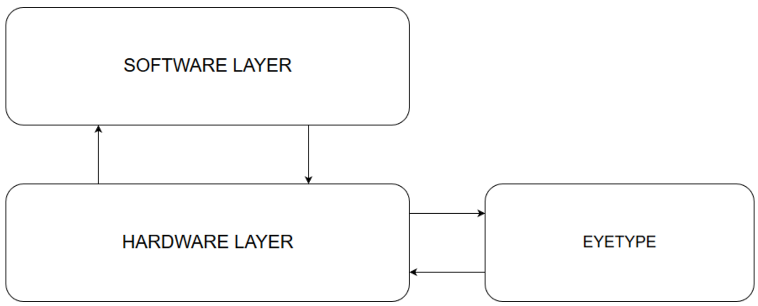


Figure 1: The Eye Tracker Overview Diagram

## 2.1 Hardware Layer Description

The hardware layer in our project consists of the TX2, the character LCD, IR light and the IR camera. Each of these perform the necessary tasks that will be needed for the software layer to send data to the NUC (EyeType team). The TX2 which runs Ubuntu will be used to gather data from the camera and the character LCD which we will be send to our software layer. (We just eneded up using the NUC for both team)

## 2.2 Software Layer Description

The Software layer consists of the Eye tracking image processing, calibration, and the server subsystems. The Eye tracking image processing subsystem receives and processes data from the IR camera which will be used to track Marci's eye. The character LCD will be used for debugging purposes. As for calibration, it will be done initially when we start the program so we can calibrate where Marci's pupil is looking in relation to the screen. As for the server, it will communicate between our layer and the NUC which is the EyeType team's system.

## 2.3 EyeType Layer Description

This layer is the responsibility of EyeType and will be the system that Marci directly interacts with.

## 3 Subsystem Definitions & Data Flow

There will be a total of eight(nine including EyeType Machine) that will comprise our Eye Tracking System. These are divided into two layers: the hardware layer and the software layer. Each layer is then comprised of four subsystems.

In the hardware layer, we have the TX2 itself. This will be the machine that our system runs off of. The TX2 will be running Ubuntu 18.04.2 LTS. The next subsystem is the IR camera + IR pass filter. This will be responsible for receiving video of Marci's eye. The IR pass filter will be used to receive only IR light and block out the visible light spectrum. The next subsystem will be the IR lights. These will produce the IR light that will be needed in order to accurately track Marci's eye. This IR light will pulse at the frame rate of the camera capturing the video. Finally, the last subsystem will be the character LCD. This will provide basic operational information to the user as well as us. This information will be used for trouble shooting purposes.

In the software layer, we have the Model Controller/State Machine. This will be responsible for coordinating all software layer subsystems as well as passing on information to the EyeType machine relating to eye tracking data and gaze vector data. The next subsystem in the software layer will be the Eye Tracking Image Processing subsystem. This will be responsible for processing video received from the camera and accurately tracking Marci's pupil. It will contain the algorithm necessary to first obtain Marci's eye and then accurately determine the location of Marci's pupil at all times. The next subsystem is the calibration subsystem. This is responsible for re-calibrating Marci's calculated gaze vector to the appropriate location on the screen. Without this subsystem, our system will be functionally unusable. Without appropriate mapping, Marci will not be able to reliably use our system. Finally, the last subsystem in the software layer is the server. This will handle receiving and sending data packets to the NUC/EyeType machine which is the machine that Marci will be using directly.

This is a comment at the end of semester: We just ended upi using the NUC to send data to the eyetype team who was responsible for the UI part of this project. If you feel like its necessary to use the TX2 like we had originally planed please do so, otherwise continue on as it is.

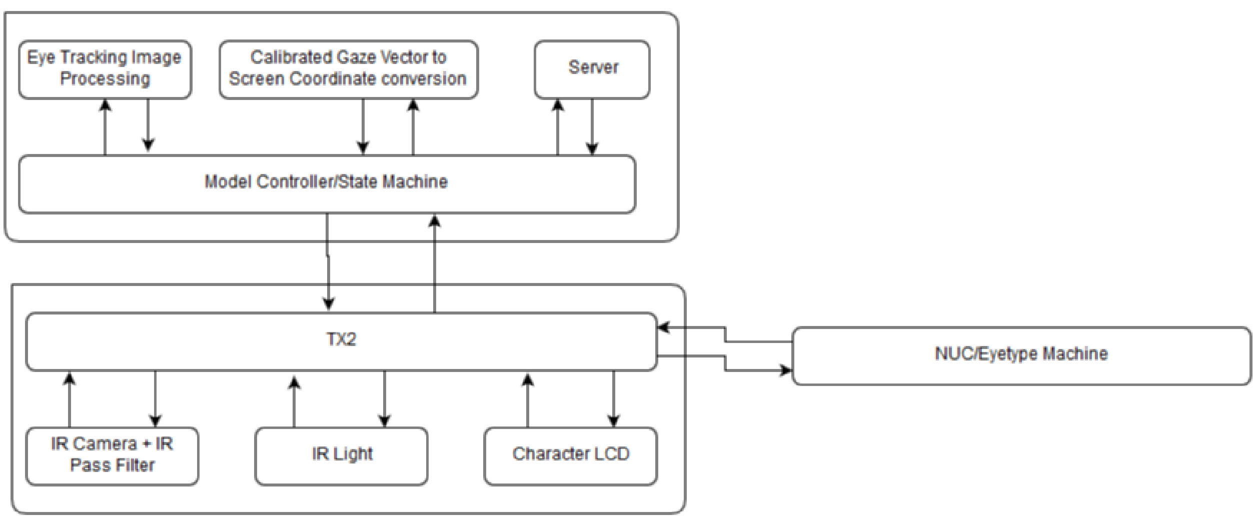


Figure 2: Subsystem Overview with Data Flow Diagram

# 4 Hardware Layer Subsystems

## 4.1 Controller Subsystem

The Controller Subsystem is the parent subsystem of all of the rest of the hardware and software. This is the subsystem containing the micro controller on which all of the software will be running, the TX2, and the hardware peripherals will be interfacing with. The Controller Subsystem will be communicating with the camera, IR light, character LCD, Model Controller, and Eyetype Machine subsystems.

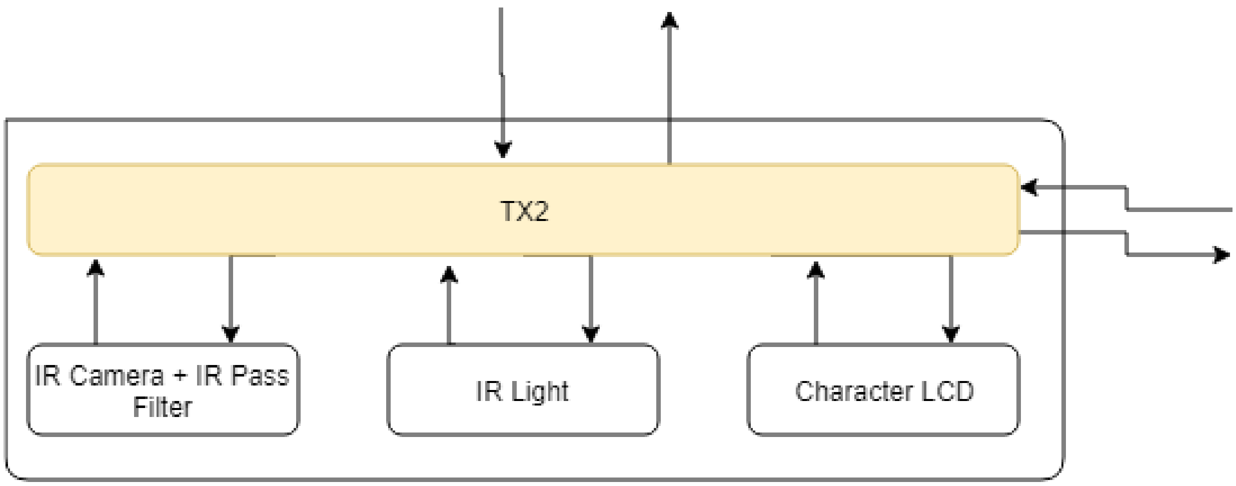


Figure 3: TX2 Subsystem Diagram

### 4.1.1 Assumptions

It will will run Ubuntu 18.04.
It will be enough to meet the performance requirements.

### 4.1.2 Responsibilities

The Controller subsystem is responsible for interfacing with the camera and collecting the data. It is responsible for running the software that analyzes the data doing the image processing. The Controller will be the hardware running a server and interfacing over USB to facilitate the network that connects to the Eyetype layer to deliver the eye tracking data. The Controller will also drive the IR light and possibly flash it as a set rate in sync with the camera frame rate. The Controller will also interface with the character LCD display to display status messages regarding the TX2, the eye tracking, the camera, and connection to the Eyetype machine.

### 4.1.3 Subsystem Interfaces

Over interface 1, which will be CSI or USB, the controller will connect to the camera and send commands to start or stop capture, and the camera will stream raw capture data back. Over interface 2, the controller will connect to the Eyetype machine and HTTP server data will flow to and from the server on the controller to and from the server on the Eyetype machine. Over interface 3 the controller will send either GND or VCC in order to either ground or drive the LED to turn on. Over interface 4, the controller will communicate over various GPIO wires in order to send status messages from the controller to the character LCD.

Table 2: Subsystem interfaces

| ID | Description | Inputs | Outputs |
|---|---|---|---|
| #01 | CSI/USB interface between camera and controller | Camera | Model-Controller |
| #02 | USB network interface | Server | Eyetype Machine |
| #03 | Analog Wire to IR LED | Model/Controller | GPIO Pins |
| #04 | Analog wire bus to Character LCD | Model/Controller | GPIO Pins |

## 4.2 IR Camera + IR Pass Filter Subsystem

The IR Camera subsystem is the subsystem that contains the camera sensor with an IR pass filter and will collect the image data necessary to perform eye tracking.

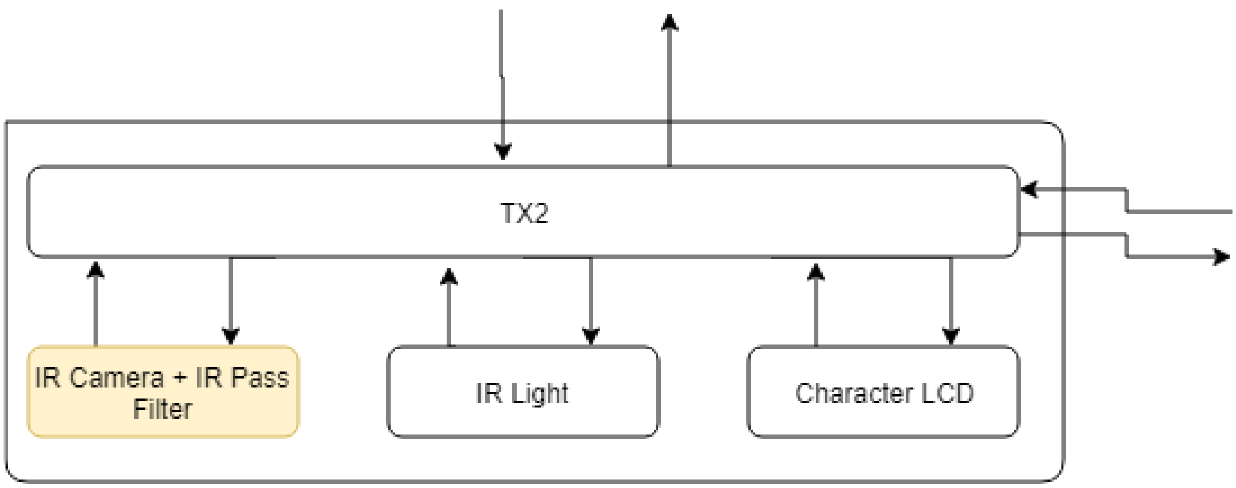


Figure 4: IR Camera + IR Pass Filter Subsystem Diagram

### 4.2.1 Assumptions

We assume that the resolution of the camera is high enough to meet the performance requirements of the eye-tracking.
We also assume that a normal visual light camera with an IR pass filter is sufficient to meet the requirements.

### 4.2.2 Responsibilities

The IR Camera subsystem is responsible for collecting the image data of the customer's eye and streaming that raw data to the controller subsystem to be utilized in the eye tracking algorithm.

### 4.2.3 Subsystem Interfaces

Over interface 1 the IR Camera subsystem will communicate over CSI or USB with the controller subsystem to send the raw video stream.

Table 3: Subsystem interfaces

| ID | Description | Inputs | Outputs |
|---|---|---|---|
| #01 | USB or CSI between camera and controller | Light into the sensor | Video data to the controller |

## 4.3 IR Light Subsystem

The IR light subsystem contains an array of IR LEDs which will be used to illuminate the customer's eye for capture by the camera with an IR pass filter.

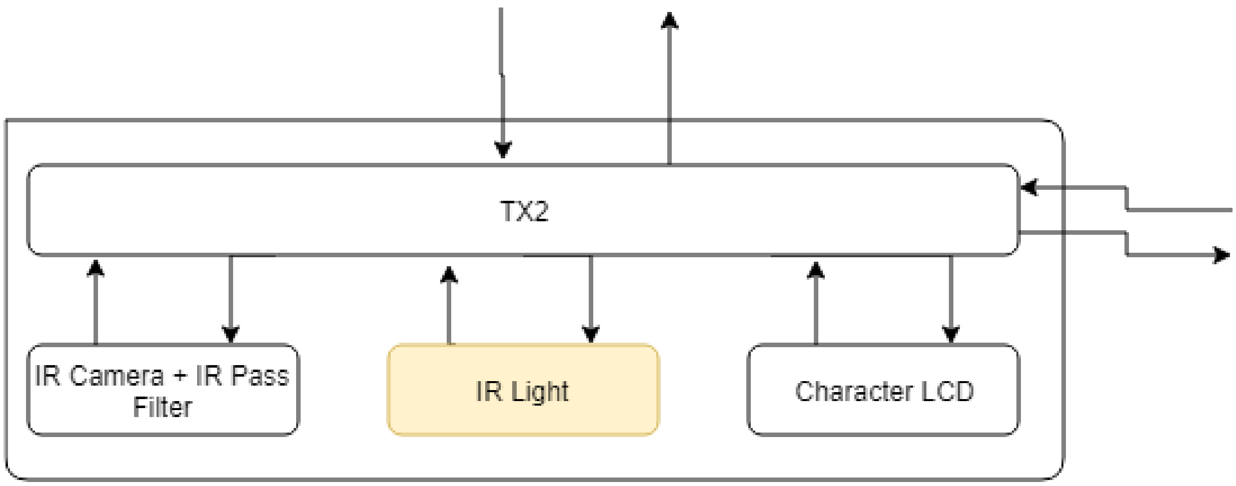


Figure 5: IR Light Subsystem Diagram

### 4.3.1 Assumptions

We assume that the IR LEDs will be luminous enough to sufficiently illuminate the customer's eye.

### 4.3.2 Responsibilities

The IR Light subsystem will be responsible for illuminating the customer's eye.

### 4.3.3 Subsystem Interfaces

Over interface 1 the IR Light subsystem will be driven or grounded over the wire interface.

Table 4: Subsystem interfaces

| ID | Description | Inputs | Outputs |
|---|---|---|---|
| #01 | Analog wire GPIO controller to IR subsystem | Voltage over wire | IR light |

## 4.4 Character LCD Subsystem

The character LCD subsystem consists of a basic character LCD capable of displaying text messages regarding the status of the system.

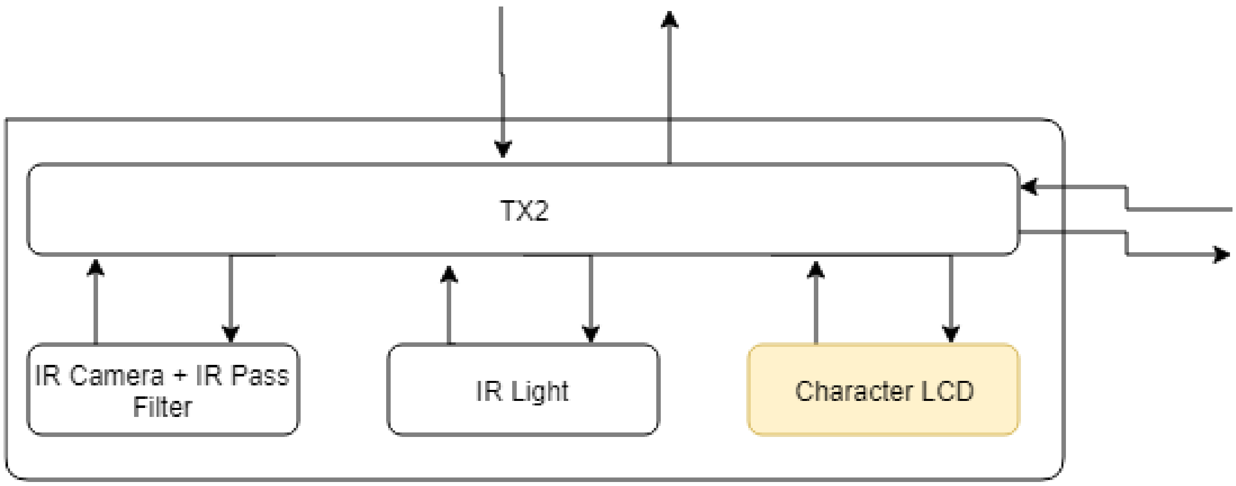


Figure 6: Character LCD Subsystem Diagram

### 4.4.1 Assumptions

We assume the LCD will be able to interface with controller.

### 4.4.2 Responsibilities

The Character LCD subsystem will be responsible for displaying status messages about all parts of the system for trouble shooting purposes.

### 4.4.3 Subsystem Interfaces

Over interface 1 the controller will interface over a serial port to send the status message data.

Table 5: Subsystem interfaces

| ID | Description | Inputs | Outputs |
|---|---|---|---|
| #01 | Serial port wires | Controller | Screen |

# 5 Software Layer Subsystems

## 5.1 Model Controller/State Machine Subsystem

The Model Controller/State Machine Subsystem is the parent subsystem of the software. This is the subsystem that transforms the data flow from all the software subsystems to each other, and connects the Software layer to the Hardware layer.

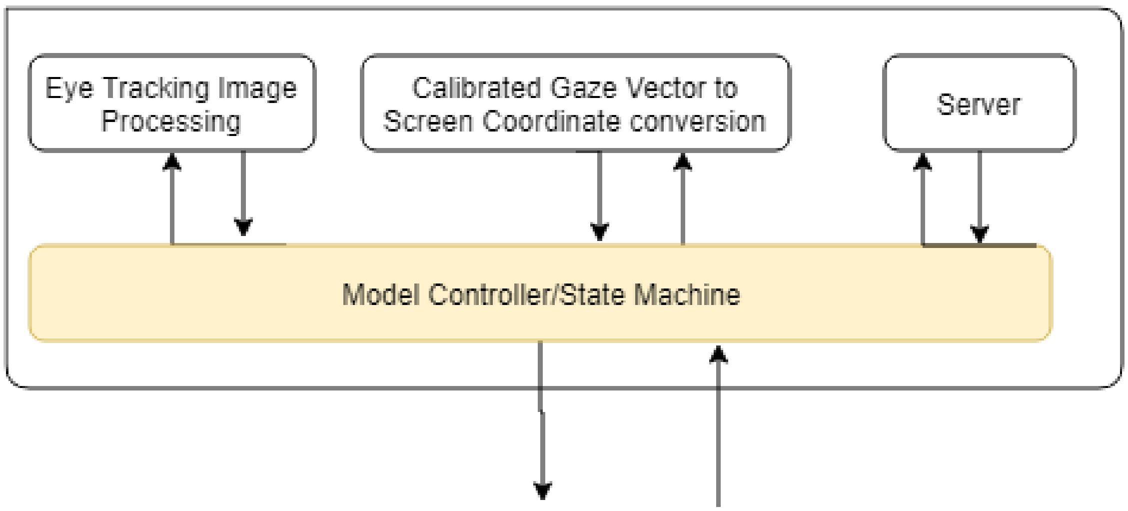


Figure 7: Model Controller/State Machine diagram

### 5.1.1 Assumptions

We assume the state machine can communicate between different computer languages.

### 5.1.2 Responsibilities

The Model Controller/State Machine subsystem is responsible for sooth communication between the Software layer and the Hardware layer.

### 5.1.3 Subsystem Interfaces

Table 6: Subsystem interfaces

| ID | Description | Inputs | Outputs |
|---|---|---|---|
| #01 | Get the video stream from the Hardware layer for processing in Software layer | Hardware layer | Eye tracking processing |
| #02 | Converts the calibrated gaze vector to screen coordinate | Eye tracking processing | Calibrated gaze vector to screen coordinate conversion |
| #03 | Packet the converted screen coordinates ready for the Server | Calibrated gaze vector to screen coordinate conversion | Server |
| #04 | Server communicates the Eye Tracker and the Eye-type | Server | Hardware layer |

## 5.2 Eye Tracking Image Processing Subsystem

The Eye Tracking Image Processing Subsystem is responsible for processing video sent in from the hardware layer and tracking Marci's eye. As her eye moves, the subsystem will continuously identify her eye movements and accurately track her pupil.

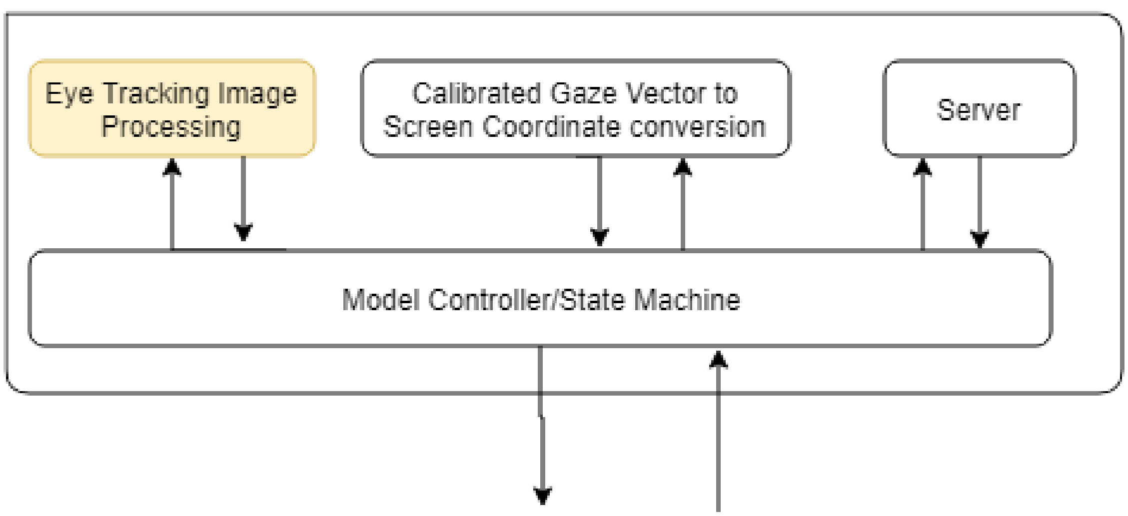


Figure 8: Eye Tracking Image Processing diagram

### 5.2.1 Assumptions

The Eye Tracking Image Processing subsystem will be written in C++ using the OpenCV library. It will accurately track Marci's eye as it moves.

### 5.2.2 Responsibilities

As stated above in the assumptions, this is the primary subsystem responsible for tracking Marci's eye movements. It will be written in C++ using the OpenCV library in order to facilitate quick calculation and response time. Initially, the subsystem will be prototyped in Python but will eventually be ported to C++ in the final product. It will receive data from the Model Controller from the hardware layer in the form of streaming video. It will then process the video and accurately measure and track the location of Marci's pupil.

### 5.2.3 Subsystem Interfaces

Table 7: Subsystem interfaces

| ID | Description | Inputs | Outputs |
|---|---|---|---|
| #01 | Data will be transferred through the software layer (likely in the form of packets) | Model Controller - State Machine | Model Controller - State Machine |

## 5.3 Calibration Subsystem

This subsystem is of extreme importance and is responsible for the calibration of the eye tracking capabilities of the system. It will allow the correct mapping of Marci's gaze vector to the monitor in which she'll be using.

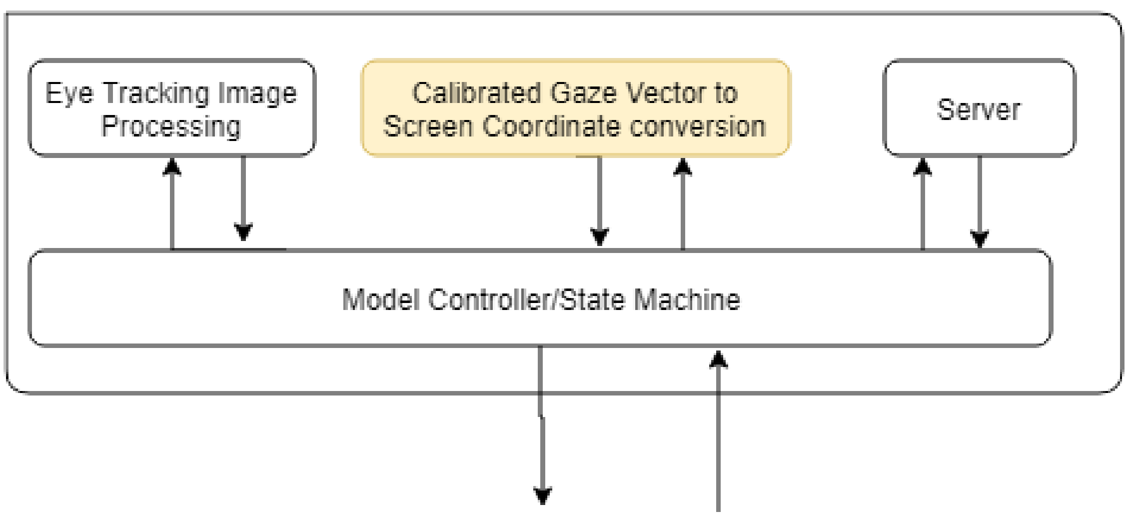


Figure 9: Calibration Subsystem diagram

### 5.3.1 Assumptions

The system will work as intended and properly map her pupil location to an appropriate location on her monitor. Calibration may be written in Python or C++ using OpenCV. Calibration will not be needed until Marci is either re-seated or the camera is bumped/moved.

### 5.3.2 Responsibilities

As stated above, the responsibilities of this subsystem are of incredible importance. The subsystem will be responsible for accurately mapping Marci's pupil direction and gaze vector onto the correct and appropriate location on the screen. This data will then be passed along to the EyeType team to allow smooth and reliable use of the system. Without this subsystem, Marci will not be able to use the system reliably and will functionally render our Eye Tracking system unusable.

### 5.3.3 Subsystem Interfaces

Table 8: Subsystem interfaces

| ID | Description | Inputs | Outputs |
|---|---|---|---|
| #01 | Data will be passed along through USB to EyeType system | Model Controller - State Machine | Model Controller - State Machine |

## 5.4 Server Subsystem

The Server subsystem is to communicate from the Eye Tracker and the Eye-Type software.

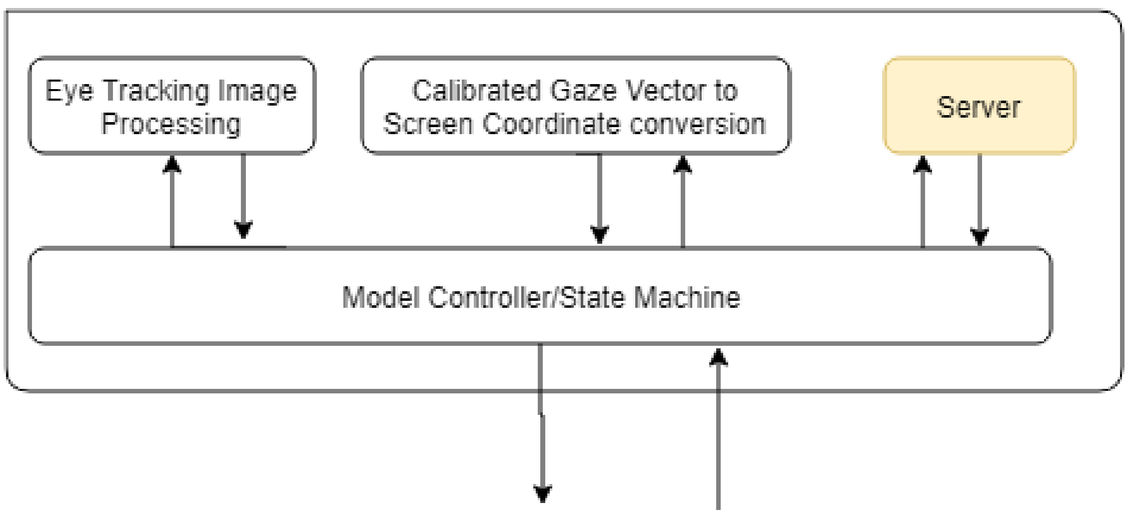


Figure 10: Server Subsystem diagram

### 5.4.1 Assumptions

The network region is reachable for Eye Tracker and Eye-Type.

### 5.4.2 Responsibilities

The responsibility of this subsystem is to wrap up the analyzed screen coordinates converted from the pupil movements and send the packet to the Eye-Type software.

### 5.4.3 Subsystem Interfaces

Table 9: Subsystem interfaces

| ID | Description | Inputs | Outputs |
|---|---|---|---|
| #01 | Wrap up the analyzed screen coordinates and send the pocket to Eye-Type | Model Controller - State Machine | Model Controller - State Machine |

# 6 EyeType Layer Subsystems

The EyeType layer is the part of the system that Marci will directly interact with. This will be the computer that Marci uses on a daily basis whether to browse the internet, complete her work duties, or to communicate with her family.

## 6.1 Input Layer

Subsystem that is primarily used for interaction with the system. Will take in the data coming from the TX2 in order to allow user to make choices on how system will carry out functions.

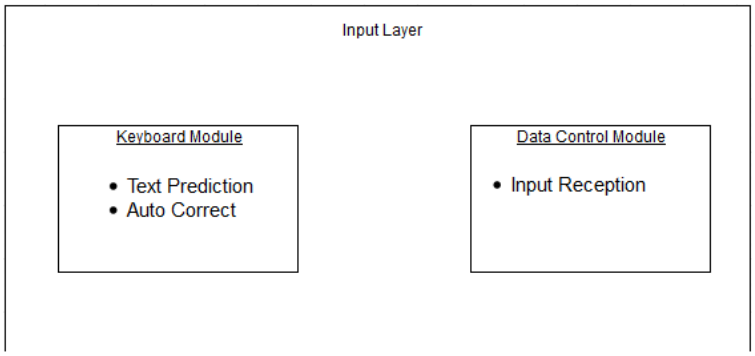


Figure 11: Input subsystem description diagram

### 6.1.1 Assumptions

System will always be able to take in data coming from the TX2, and that data will always be transmitted correctly.

### 6.1.2 Responsibilities

This is where the system will start to create the response that is desired from the user. If the input layer is corrupted, the whole system will be of no actual use, because we assume that the desired actions from the user will be transmitted from the TX2 correctly.

For the Keyboard Module, Text Prediction will be employed in order to help the user type out their words faster, along with Auto Correct to help deal with possible misspelling of words.

### 6.1.3 Subsystem Interfaces

Table 10: Subsystem interfaces

| ID | Description | Inputs | Outputs |
|---|---|---|---|
| #01 | System used to transmit input data | TX2 | Information for Control Layer |

## 6.2 Control Layer

### 6.2.1 Assumptions

Properly directs the software system based on what is coming from the Input Layer

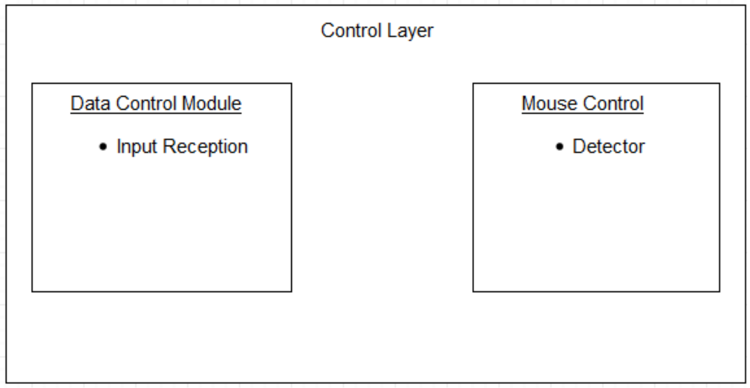


Figure 12: Control subsystem diagram

### 6.2.2 Responsibilities

The Control Layer is what takes the input and turns it into a tangible command for the system. For example, if the user wants to type, it must realize that that is the present command and print out the correct text. It allows the user to determine what action is taken by the system.

Mouse Control is for the user to select from the variety of options that are available.

### 6.2.3 Subsystem Interfaces

Table 11: Subsystem interfaces

| ID | Description | Inputs | Outputs |
|---|---|---|---|
| #01 | Commands coming from Input Layer | Data from Input Layer | Information to drive GUI |

## 6.3 Graphical User Interface Layer

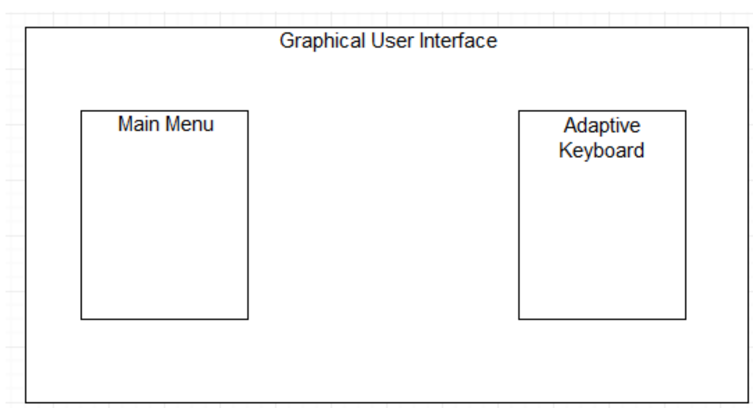


Figure 13: GUI subsystem description diagram

### 6.3.1 Assumptions

Properly displays some type of confirmation of the command chosen by the user on a screen

### 6.3.2 Responsibilities

This layer will be used to show that an action has been taken by the user. For example, if the action desired is typing, there will be a place to show the output of the word that is typed. If another action is chosen, some type of result or confirmation will be shown on the screen.//

Main menu will display the main options available to a user, while the adaptive keyboard will display the text that is being typed.

### 6.3.3 Subsystem Interfaces

Each of the inputs and outputs for the subsystem are defined here. Create a table with an entry for each labelled interface that connects to this subsystem. For each entry, describe any incoming and outgoing data elements will pass through this interface.

Table 12: Subsystem interfaces

| ID | Description | Inputs | Outputs |
|---|---|---|---|
| #01 | Commands from Control Layer | Data from Control Layer | Command Confirmation on Screen |

## REFERENCES