ESLAB HW2 Report

Li-Wei Chen, Ya-Liang Chang

B04901014 EE3, B03901014 EE4

Input Direction

Camera as Input

We use the *tessel-av* module of *tessel2* for our input system to handle the USB camera inputs. As the camera recieves images from the outer world, it pipes the image as stream and send it to our webserver. To achieve this, the access point is created by the *teesel2* and linked to the server.

Stream and Face Detection

To make use of the high efficieny of the stream module, we send the stream object by websocket using the *socket.io-stream* module instead of sending each frame of images. Before the stream is piped to the web, we utilize the face detection function of *opencv*, which is implemented by cascade classifiers. Once a image is detected to have human face by the classifier, it renders a rectangle around the face to the image. Then after doing such processing, the image is send to the server for display.

Reason of Local Server

The reason we are using a local server instead of launching the server on *tessel2* is that the backend of *opencv* is C++, and it seems difficult for

mounting the precompiled binaries to the *tessel2*. Thus we use a workaround, using a local server as a backend to process the images.

Resposive Client

If the web is not responsive, we have to refresh the page manually to get the new image displayed. This is solved by using *react*, which we launched on localhost and fetched data from the server. Utilizing the resposive characteristic of *react*, the realtime streaming display can be realized.

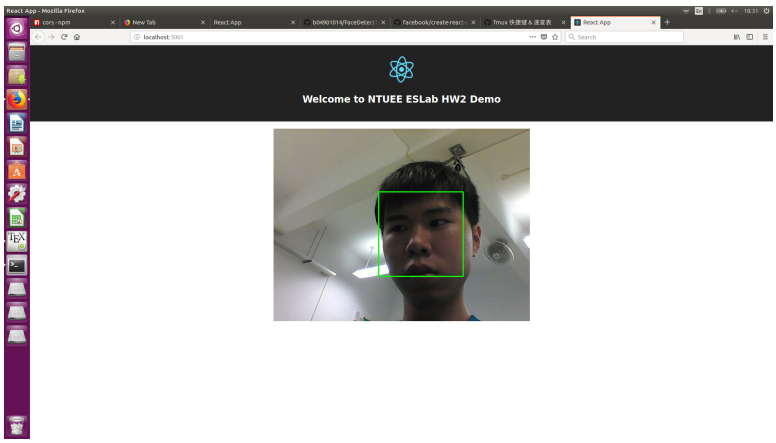

Figure 1: Snapshot at the client output

Output Direction

For the output direction, once a face is detected, the server sends a websocket to the *teesel2*. Then it activates the function to turn on the green LED which is on the other side of the three main LEDs for 0.5 seconds. This part is implemented by the *tessel-led* module.