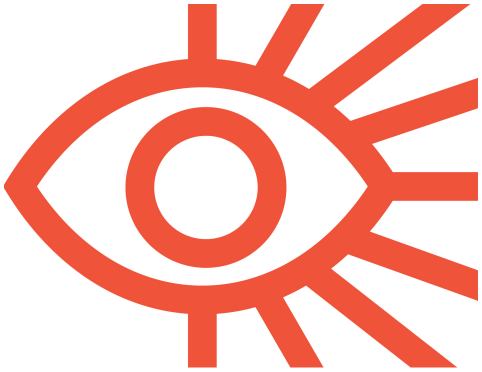

eyeris

USER MANUAL

THE TWO PIPELINES

1. GUMSTIX - PANDABOARD - SERVER - CLIENT COMPUTER
   This is the mobile pipeline that is defined in the project. The advantage to this design is that it is mobile and you can attach a battery to it. This was the intended pipeline for the project. Its downsides, however, are there that the user cannot see the eye-cameras as he/she is putting on the glasses, so it is hard to know if the eyes are visible or not. This pipeline uses ducati h.264 encoding, which has specialized hardware and can be very fast.

2. “PANDASTIX” = LAPTOP - SERVER - CLIENT COMPUTER
   This is the pipeline that makes testing and development a lot easier. The user can use this project to combine both the Gumstix’s and the Pandaboard’s processes on to the same computer. With this pipeline, the user can easily adjust the glasses easily by looking at the eye-camera feeds and making sure the eye is completely visible (more information on this in the *CALIBRATION* section). This requires a computer with at least 2 different USB buses, because the eye-cameras in conjunction with the outward-facing camera require too much USB bandwidth and must be each on different USB buses. This also requires a pretty high-performance computer because it uses x.264 encoding, which is processor-based and may slow down the computer.

SETUP

1. Start the server up and begin the server process.
2. Connect the two eye cameras to a USB hub, and then into the Gumstix. Plug the outward facing camera into the Pandaboard. (For PandaStix, plug the USB hub with the eye-cameras into one of the USB buses on your computer, and the outwarding facing camera into another. Then skip to step 8).
3. Give power to both the Pandaboard and the Gumstix (via a battery, or plug in the power adapters to a wall outlet).
4. Get the Pandaboard hooked up to a network via WiFi.
    a. Refer to *PARAMETERS* for more information.
5. Connect the Pandaboard and Gumstix together via an Ethernet cable.
6. The Gumstix process can be started via a serial connection. This will eventually be changed to start the process upon boot.
7. Once the Gumstix process is started, the Pandaboard process may now be started. The easiest way to access the Pandaboard is via an SSH connection.
8. Open up the client GUI on any computer(s), enter in the server IP, and click Connect.
    a. A video screen will pop up.
    b. refer to *CALIBRATION* for more information.

CONFIGURATION PARAMETERS

There are a few parameters that might be needed to changed for this to work (these can be changed on each system individually in the *settings.configurations.eyeris* file, and some of these can be edited in the *Defaults.h* file that defines the default values).

1. IP addresses
    a. The Pandaboard-Gumstix communication should not need to be modified but can be if preferred.
    b. The Pandaboard-Server communication will need adjustments. The Pandaboard IP is currently registered at Iowa State as ‘eyeris-panda.student.iastate.edu’ but that is currently registered to a specific student’s Net-ID and will need to change when this student leaves. The server’s IP will need to be the IP number (or hostname, for simplicity) of what the server is running on.
2. Ports
    a. These will probably not need to change unless it is preferred.
3. The majority of the other parameters are for the eye-tracking algorithm and can be adjusted as needed according to their function.

CALIBRATION

1. Expected order of the cameras plugged in is the following:
    a. Left eye camera must be plugged in first (/dev/video0)
    b. Right eye camera must be plugged in second (/dev/video1)
    c. World cam is (/dev/video0) on the Pandaboard.

    d. PandaStix ONLY: world cam is expected to be plugged in third (/dev/video2)
        i. WARNING: your computer may have a built-in camera that will be defaulted to /dev/video0. If so, you will have to modify the cameras expected by the PandaStix code.
2. Put on glasses and make sure that the user's eyes are easily viewable by the cameras. If running on the Gumstix-Pandaboard pipeline (versus just a computer), you will have to look at the ellipses in the top-left corner of the client's video screen to determine if the algorithm is able to see the pupils very well or not. If you are using the PandaStix pipeline, you can enable the PANDASTIX_SHOW_EYECAMS define at the bottom of *Defaults.h*.

TROUBLESHOOTING

It's not working! Well, let's figure out why not...
Common things that you may have forgotten:

- Plug the cameras in in the correct order (see *CALIBRATION*). Make sure that if you are using the PandaStix pipeline, you are plugging in the eye-cameras and the outward-facing camera into different USB buses.
- Are the cameras able to get a good image of the person's eyes? Is it flooded with outside light?
- Make sure that all of the IPs and ports match up. Note that all of the network communications are:
    - Gumstix - Pandaboard (this is statically defined and should never be conflicted)
    - Pandaboard - Server
    - Server - Client$_i$

Well... it's working, but it's going really slow! What's going on?
First, understand that the full pipeline usually has a ~3 second delay, and the PandaStix project can have anywhere between a 1 and a 10 second delay (depending on hardware). However, the video IS supposed to be quite smooth from both the full pipeline and the PandaStix branch. If there is every a full "graying-out" of the video

- Is the server fast and good at graphics processing? Remember that the server needs to decode h.264, use OpenCV to analyze the frames, paint the eye data, and then re-encode h.264.
- If you are using the PandaStix pipeline, is your computer fast enough and good at graphics processing? This doesn't do as much graphic processing as the server so doesn't require quite as much power, but you still need some.
- Is the network that you are connected to fast and free of congestion? This requires a lot of traffic use - there is a lot of information (both information, data points, and control messages) being sent over the network very quickly, so you must ensure that the network you are connected to can handle all of this traffic. If you are having problems during the day, you might want to try later at night when fewer people are connected to it.

To debug through the actual code, we recommend using an IDE debugger, like Netbeans. Watching what the server is printing is very useful for understanding what is making it to the server and what is being forwarded on correctly.

About

Project Eyeris is team May 13-20 for the 2012-2013 school year at Iowa State University. It is comprised of six members three computer engineers, two software engineers and one electrical engineer. The system cost $538.20. It is designed with open source software and made in one year by this team. If you want any more information please visit our website: http://seniord.ece.iastate.edu/may1320/.