Handover Manual

360 Degrees Web Conferencing Device

Project Client: Frederick Chew

Team member:

1. U5629478 - Hengjia Li
2. U5670512 - Hongjian He
3. U5922620 - Jiawei Fan
4. U6051142 - Jireh Mendoza
5. U5604968 - Jose Quiroga Perez
6. U5586843 - Minh Doan
7. U5613613- Yilin Geng

1. Introduction

The aim of this project is to design a 360-degree online conference system, which has the following characteristics:

- Portable
- Plug-and-play
- Easy to use
- Open-source
- Affordable (no more than 300AUD)
- Reproducible
- Upgradable

The desired functions are:

- Output 360-degree panoramic view
- Produce video stream with a minimum resolution of 1280x720
- Output high definition audio
- Reduce noise from undesired sources
- Detect faces of participants of the meeting
- Locate the speakers based on sound
- Focus automatically on the speakers using face detection and sound localization
- Feed a single stream to the user as a normal webcam
- Powered by USB

The product delivered in this project is assessed based on the above characteristics and desired functions in Table 1 & 2. Green blocks mean that we achieved as desired. Yellow ones mean that we achieved with some compromise, but it is considered equivalent and acceptable, while the orange ones indicate possible improvement in the future.

Characteristic	**Assessment**
Portable	Highly portable
Plug-and-play	Patent technology, replaced with IP camera approach.
Easy to use	Assembly is relatively simple. Installing software dependencies takes some effort.
Open-source	Open source under GNU public license
Affordable (no more than 300AUD)	253 AUD (or 300 AUD with extra microphone)
Reproducible	Can be easily reproduced following this manual

Upgradable	Upgrading suggestions are clearly listed in handover documents

Table 1. Product characteristic assessment

Desired function	**Assessment**
Output 360-degree panoramic view	Achieved with de-warped fisheye input
Produce video stream with a minimum resolution of 1280x720	Achieved
Output high definition audio	Achieved with mic-array (more HD audio with extra microphone)
Reduce noise from undesired sources	Partially achieve, cannot cancel echoes
Detect faces of participants of the meeting	Have a solution with rotation-invariant face detection (too resource-consuming for our processor) and a normal cascaded face detection solution
Locate the speakers based on sound	Achieved using the DOA algorithm
Focus automatically on the speakers using the association of face detection and sound localization	Completed, but the lagging due to computational limitation is too much for a valid conference call
Feed a single stream to the user as a normal webcam	Achieved by generating a online stream under IP camera standards and have the user computer recognize it as webcam input
Powered by USB	Achieved, but not recommended as processing ability is throttled by Raspberry Pi without sufficient power.

Table 2. Product desired functions assessment

This manual gives detailed instructions on how to reproduce what we have with suggestion on future improvements. This document can be read in conjunction with Documentation Process to understand the team thought process, design decision and consideration for the project.

2. Assembly

The hardware components of our systems are listed below with the expected cost, these costs are reflective of when the project was completed and may fluctuate in the future. Discussion on component selection can be observed in our conversation log or meeting minutes.

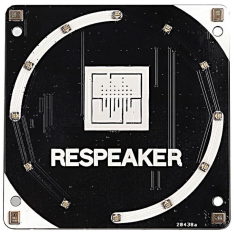
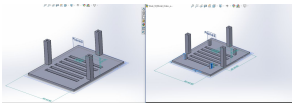
Component	Price (AUD)	Picture	Recommended Source
Raspberry Pi 3 B+ (with adaptor & storage > 16GB)	75		Ebay
210-degree Fisheye camera	92		TaoBao
Sound array	37.35		seeedstudio
Conference Microphone	53		TaoBao
3D printed supporting frame	3		3D printing

Table 3. Details on hardware components used

Functionalities of component

The raspberry pi model will be the computational heart of the device. The 210 degrees fisheye camera captures the 360-degree view of the conference. The sound array detects the direction of the person who is talking during the online conference. The microphone provides high-quality audio input for online conferencing. The system assembled at the end of the project can be seen in Figure 1.

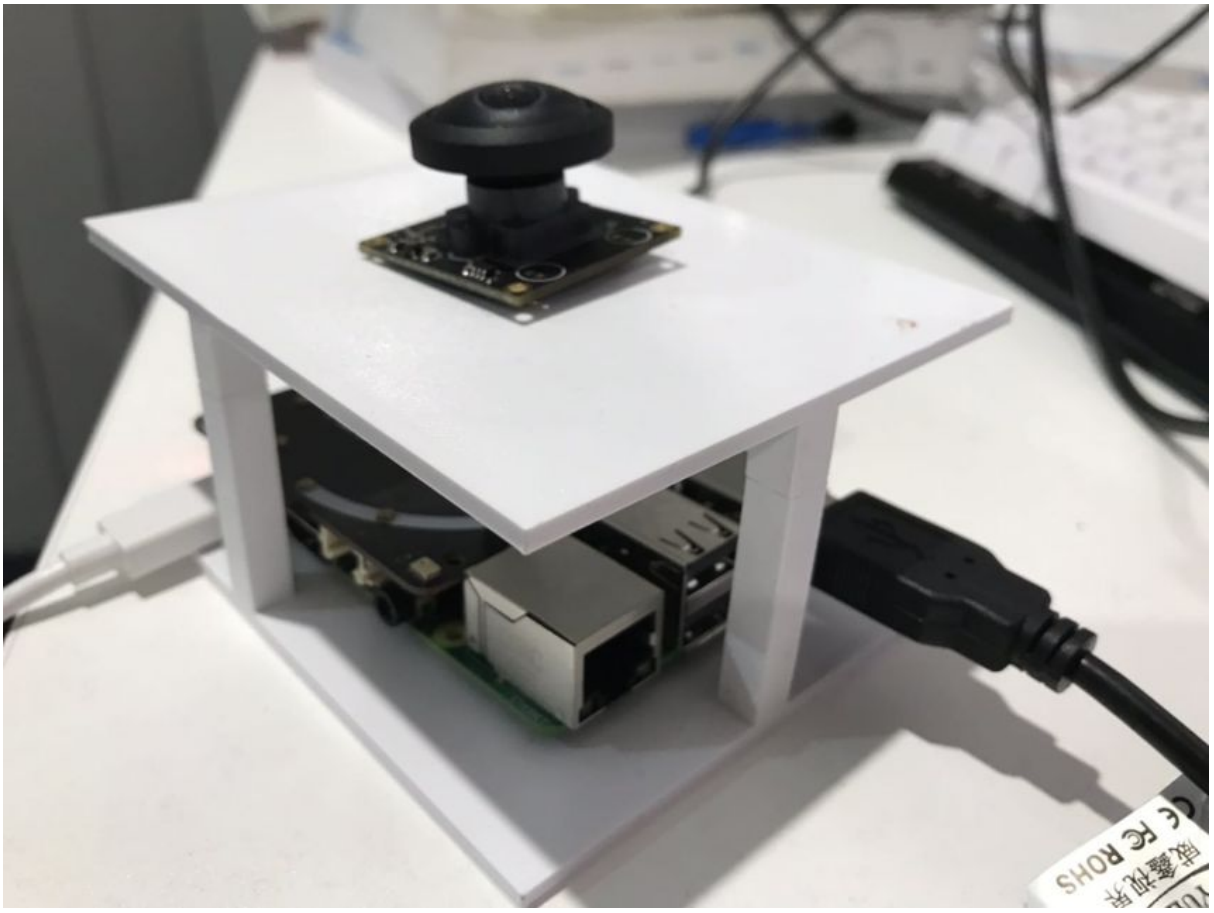


Figure 1. Final design of 360 degrees web conferencing system

3. Design approach

In this section, the critical subsystems of our system will be introduced and corresponding source codes will be explained.

3.1. Video subsystem

This subsystem processes the fisheye video input and output from the audio subsystem in Python. We use OpenCV to do face detection and image processing. There is no available open source code for dewarping fisheye images in Python 3. Thus, we reproduce the algorithm in Python 3 (referring to the data_receiver.dewarp method), from the available Python 2 source code.

We have three options of face detector:

1) Built-In cascaded face detector in OpenCV
2) Neural Network based face detector in Tensorflow
3) Rotation-invariant face detector based on Progressive Calibration Network (PCN)

The third option has the best performance on PC, since it can detect faces directly from fisheye images and it is robust to the face orientation. However, it is implemented in Pytorch and it takes too much computational resource in Respi. Similarly, Raspi cannot afford the second option. Therefore, we use the built-in cascaded face detector in our system. However, such face detector has two limitations: wrong detections on background and missing target detections (especially it cannot give detections on side faces, cropped faces on panorama).

According to the speakers direction detected by the audio subsystem, the speaker is cropped out from the panorama and .

A focused view of the participants who is talking will cropped out and displayed under the panorama view, based on the direction determined by both the face detection and sound localization, which is the output of the audio subsystem.

3.2 Audio subsystem

We use Direction of Arrival combined with Voice Activation Detection to determine the location of people who is speaking in the conference. Refer to the original github project page.

3.3. I/O subsystem

The I/O subsystem aims to combine the input signals from the fisheye camera and sound array, and to feed a video output to the user computer.

Input from fisheye camera
We use v4l2 Linux package to import the image stream from fisheye camera to our video subsystem. This v4l2 package can be enabled in ***OpenCV***, while building the library.

Input from the sound array
There is a Linux driver (***seeed-voicecard***) for importing audio signals from sound array to our audio subsystem.

Video output

The output of the video subsystem is a stream of numpy array in Python. We use V4L2 loopback package to claim a virtual video device on Raspi. Then, the output of the video subsystem (the stream of numpy array) is broadcasting into the virtual device created on Raspi. Finally, we use an open-source Linux package called ***mjpg-streamer*** to streaming the virtual video device over the Internet under the IP camera standard.

We use the IP camera as the output standard. Raspi streams the video output from our video subsystem to an IP address. Thus, the user's PC can detect the IP camera and use it as a virtual webcam for commercialized softwares, like skype. In Windows, Mac OS and Linux/Linux-like OS, there is much free software capable of converting IP camera as a virtual webcam.

4. Future improvements

In the future, the conferencing device could be improve in terms of code efficiency by using a more efficient programming language. By converting the code to system programming language such as Go, C or C++, it could reduce the computational load placed on the Raspberry Pi, and increase the uploaded frame rate on the HTTP server. However, there are limitation to this method, as analysing codes and improving efficiency is a lengthy process.

Utilisation of more powerful hardware can significantly increase the performance, as the current Raspberry Pi 3 B+ only utilise a single core 1.2 GHz processor. By switching to a platform with duo, quad or even hexacore cores with higher clock rates, the computation requires to perform DOA, Dewarping, Face Detection and Directional Focusing can be divided and process in parallel more efficiently.

Possible options of hardware include:

1. UDOO X86
2. NanoPi Neo64
3. Rock Pi 4
4. Libre Computer AML-S905X-CC

These platforms can be more expensive than the Raspberry Pi 3 B+ due to offering better hardware and this should be taken into consideration to ensure the device is as cheap as possible. The only concerns with switching to a more powerful platform is determining which chip is widely supported and updated by the open source community. As this enables ease of improvement and implementation of new features by other contributors in the future.

Currently, due to the latency of images being uploaded to the HTTP server, the face detection was not operating alongside the DOA algorithm during the demonstration. This is to increase the frame rate to improve user's experience. However, we have delivered a successful integration between the Face Detection and DOA algorithm, albeit not the most efficient even when the face detection is run as a thread (background process). Future teams could consider improving how these algorithms can be integrate more efficiently and devise better function architecture.

The use of a more powerful processor can potentially allow the implementation of rotational-invariant face detection instead of the simpler cascade algorithm. This will increase the accuracy of the program to detect the face and can allow more advance video interface to be produced.

Furthermore, a cooling system for the housing frame may need to be implemented to reduce throttling due to increasing CPU temperature. Possible considerations include a heatsink in the 3D model frame to dissipate unwanted heat, and/or a small fan to cool the microprocessor. If a fan is implemented, the future team must consider the noise of the fan affecting the DoA algorithm and live sound capture. There are already existing design on the

market, such as those produced by Adafruit, the included heatsink and fan can reduce the temperature on the CPU by almost 50% as seen in Figure 2.

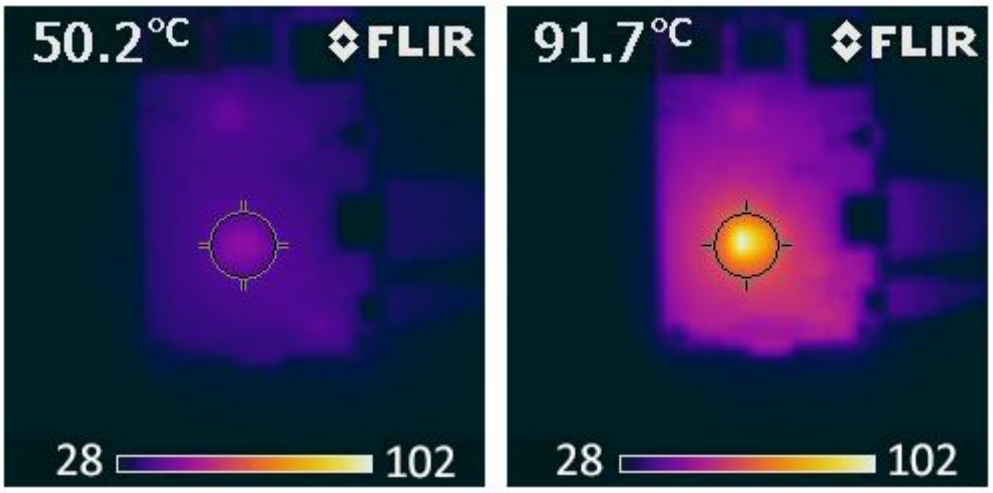


Figure 2. Before and after application of cooling package by Adafruit (Microsoft, 2018) [1]

In closing, the largest improvement that could be made by future teams is the capability for the device to be “plug and play”. To accomplish this, teams must consider a new approach for the I/O subsystem instead of the current need to install Open Broadcaster Software (OBS) to obtain and transmit the video from the HTTP server as a virtual webcam. Currently, plug-and-play technologies are held under intellectual properties by company and requires specific hardware or adapter to be purchased. Further research into how the system can be adapted to enable USB Video Class (UVC) to allow direct real-time streaming to the PC. This can potentially be done by setting up a USB Gadget Mode with UVC.

By converting the device into a plug-and-play system, it will be simpler for the technology to be adopt by individuals or organisations with access to the open source space. This will also reduce the difficulties of setting up the device by having no installation required on the user’s PC.

[1] https://microsoft.github.io/ELL/tutorials/Active-cooling-your-Raspberry-Pi-3/

Appendix 1. Instructions on installation

A1.1 Open cv

Step 1: Update Packages

```
sudo apt -y update
sudo apt -y upgrade
```

Step 2: Install OS Libraries

```
sudo apt-get -y remove x264 libx264-dev

## Install dependencies
sudo apt-get -y install build-essential checkinstall cmake pkg-config yasm
sudo apt-get -y install git gfortran
sudo apt-get -y install libjpeg8-dev libjasper-dev libpng12-dev

sudo apt-get -y install libtiff5-dev

sudo apt-get -y install libtiff-dev

sudo apt-get -y install libavcodec-dev libavformat-dev libswscale-dev
libdc1394-22-dev
sudo apt-get -y install libxine2-dev libv4l-dev
cd /usr/include/linux
sudo ln -s -f ../libv4l1-videodev.h videodev.h
cd $cwd

sudo apt-get -y install libgstreamer0.10-dev libgstreamer-plugins-base0.10-dev
sudo apt-get -y install libgtk2.0-dev libtbb-dev qt5-default
sudo apt-get -y install libatlas-base-dev
sudo apt-get -y install libmp3lame-dev libtheora-dev
sudo apt-get -y install libvorbis-dev libxvidcore-dev libx264-dev
sudo apt-get -y install libopencore-amrnb-dev libopencore-amrwb-dev
sudo apt-get -y install libavresample-dev
sudo apt-get -y install x264 v4l-utils

# Optional dependencies
sudo apt-get -y install libprotobuf-dev protobuf-compiler
sudo apt-get -y install libgoogle-glog-dev libgflags-dev
```

```
sudo apt-get -y install libgphoto2-dev libeigen3-dev libhdf5-dev doxygen
```

Step 3: Install Python Libraries

```
sudo apt-get -y install python3-dev python3-pip
sudo -H pip3 install -U pip numpy
sudo apt-get -y install python3-testresources
```

We are also going to install virtualenv and virtualenvwrapper modules to create Python virtual environments.

```
cd $cwd
# Install virtual environment
python3 -m venv OpenCV-"$cvVersion"-py3
echo "# Virtual Environment Wrapper" >> ~/.bashrc
echo "alias workoncv-$cvVersion=\"source
$cwd/OpenCV-$cvVersion-py3/bin/activate\"" >> ~/.bashrc
source "$cwd"/OpenCV-"$cvVersion"-py3/bin/activate
###############
```

Next, we create the Python virtual environment.

```
############# For Python 3 #############
# now install python libraries within this virtual environment
sudo sed -i 's/CONF_SWAPSIZE=100/CONF_SWAPSIZE=1024/g'
/etc/dphys-swapfile
sudo /etc/init.d/dphys-swapfile stop
sudo /etc/init.d/dphys-swapfile start
pip install numpy dlib
# quit virtual environment
Deactivate
```

Step 4: Download opencv and opencv_contrib

```
git clone https://github.com/opencv/opencv.git
cd opencv
git checkout $cvVersion
cd ..

git clone https://github.com/opencv/opencv_contrib.git
cd opencv_contrib
git checkout $cvVersion
cd ..
```

Step 5: Compile and install OpenCV with contrib modules

First we navigate to the build directory.

```
cd opencv
mkdir build
cd build
```

Next, we start the compilation and installation process.

```
cmake -D CMAKE_BUILD_TYPE=RELEASE \
        -D CMAKE_INSTALL_PREFIX=$cwd/installation/OpenCV-"$cvVersion" \
        -D INSTALL_C_EXAMPLES=ON \
        -D INSTALL_PYTHON_EXAMPLES=ON \
        -D WITH_TBB=ON \
        -D WITH_V4L=ON \
        -D
OPENCV_PYTHON3_INSTALL_PATH=$cwd/OpenCV-$cvVersion-py3/lib/python3.5/site-
packages \
      -D WITH_QT=ON \
      -D WITH_OPENGL=ON \
      -D OPENCV_EXTRA_MODULES_PATH=../../opencv_contrib/modules \
      -D BUILD_EXAMPLES=ON ..
```

```
make -j$(nproc)
make install
```

Step 6: Reset swap file

Once we are done with installing heavy Python modules like Numpy, it's time to reset the swap file.

```
sudo sed -i 's/CONF_SWAPSIZE=1024/CONF_SWAPSIZE=100/g'
/etc/dphys-swapfile
sudo /etc/init.d/dphys-swapfile stop
sudo /etc/init.d/dphys-swapfile start
```

Finally, we also need to add a simple statement to make sure that **VideoCapture(0)** works on our Raspberry Pi.

```
echo "sudo modprobe bcm2835-v4l2" >> ~/.profile
```

A1.2 v4L2 loopback

```
git clone https://github.com/umlaeute/v4l2loopback

cd v4l2loopback

make && sudo make install

sudo depmod -a
```

A1.3 mjpg-streamer

```
git clone https://github.com/jacksonliam/mjpg-streamer

sudo apt-get install cmake libjpeg8-dev

sudo apt-get install gcc g++

cd mjpg-streamer-experimental
```

```
Make && sudo make install

cd mjpg-streamer-experimental

make distclean

make CMAKE_BUILD_TYPE=Debug

sudo make install
```

A1.4 seeed-voicecard

A1.4.1. In the raspberry pi terminal, follow the steps to download an enable the necessary drivers for the 4-mic array.

Step 1: Get the seeed voice card source code.

```
sudo apt-get update
sudo apt-get upgrade
git clone https://github.com/respeaker/seeed-voicecard.git
cd seeed-voicecard
sudo ./install.sh
reboot
```

Step 2: Then select the headphone jack on Raspberry Pi for audio output:

```
sudo raspi-config
# Select 7 Advanced Options
 # Select A4 Audio
 # Select 1 Force 3.5mm ('headphone') jack
 # Select Finish
```

Step 3: Check that the sound card name looks like this:

```
pi@raspberrypi:~/seeed-voicecard $ arecord -L
null
        Discard all samples (playback) or generate zero samples (capture) playback
capture
dmixed
array
ac108
default:CARD=seeed4micvoicec
  seeed-4mic-voicecard,
Default Audio Device
```

sysdefault:CARD=seeed4micvoicec
seeed-4mic-voicecard,
Default Audio Device
dmix:CARD=seeed4micvoicec,DEV=0
seeed-4mic-voicecard,
Direct sample mixing device
dsnoop:CARD=seeed4micvoicec,DEV=0
seeed-4mic-voicecard,
Direct sample snooping device
hw:CARD=seeed4micvoicec,DEV=0
seeed-4mic-voicecard,
Direct hardware device without any conversions
plughw:CARD=seeed4micvoicec,DEV=0
seeed-4mic-voicecard,
Hardware device with all software conversions

A1.4.2. Enable SPI pins

Activate SPI
Sudo raspi-config
Go to “Interfacing Options”
Go to “SPI”
Enable SPI
Exit the toop

A1.4.3. Pixel ring

Step 1: get pixel ring
pi@raspberrypi:~ $ cd /home/pi
pi@raspberrypi:~ $ git clone https://github.com/respeaker/4mics_hat.git
pi@raspberrypi:~ $ cd /home/pi/4mics_hat
pi@raspberrypi:~/4mics_hat $ sudo apt install python-virtualenv
pi@raspberrypi:~/4mics_hat $ virtualenv --system-site-packages ~/env
pi@raspberrypi:~/4mics_hat $ source ~/env/bin/activate
(env) pi@raspberrypi:~/4mics_hat $ pip install spidev gpiozero

Step 2: test pixel ring
(env) pi@raspberrypi:~/4mics_hat $ python pixels_demo.py

A1.4.4. DOA algorithm

Option 1: Original DOA with threshold and noise reduction added
Run DOA_test_1.py

Option 2: DOA with ability to detect human voice and threshold and noise reduction added
Run vad_doa.py

A1.5 pyaudio

You can install using pip:

```
$ pip3 install pyaudio
```

A1.6 numpy

You can install using pip:

```
$ pip3 install numpy
```

A1.7 queue

You can install using pip:

```
$ pip3 install queuelib
```

A1.7 v4l2 and fcntl

```
pip install v4l2 fcntl
```

Appendix 2. IP camera software - (OBS stream recommended)

There are many free IP camera software that is able to make an IP camera into a virtual webcam for Skype. OBS Studio is recommended since it is cross-platform. Download the software in https://obsproject.com/download and install it.