Dual Play
SDP Final Project Report

Yujia Huo, EE; SuoAn Gao, EE; Lina Wu, CSE; Siyu Wu, EE

***Abstract*—Designed to solve the problem where a TV cannot display two sources without sacrificing visual and audio qualities, Dual Play is a system that enables any 3D TV to display two different input sources with full screen simultaneously. Dual Play is able to preserve image proportion and maintain original frame rate, while providing high-quality private audio environment through wireless headphones. This system is composed of five main blocks: content input, computer processing, 3D TV display, remote control, and user experience.**

I. INTRODUCTION

Ever since the invention of the television, it has been advertised as a necessity in the living room. A family of four sitting together on the couch watching TV together has been a common theme of Television advertisement. It is ironic that TV can only display one source at a time, which is ironic for something that is meant to be shared among multiple people. What are the chances that every family member wants to watch the same thing every day?

There are some unsatisfying existing solutions. basic methods are used if different contents are needed to display on the same TV, such as split screen display for multiplayer video games. This solution is a quick and simple fix, but it comes at a huge cost. Half-split TV screen (Figure 1) is often extremely narrow or out of proportion, while quarter-split TV screen (Figure 2) is too small to display all the important details for gameplay. Beside visual compensation, overlaying sound effects coming from different player also makes the video games rather hard to engage.

To counter this issue, LG invented 3D TVs with "dual play" function, which allows full screen display of two different contents. However, these TV costs up to $8,000, and they only work with certain video games that come with half-split-screen mode. What they do is simply take traditional split-screen display as Side-By-Side or Top-and-Bottom 3D format and "stretching" each side into full resolution by interlacing them. As the result, the two player perspectives are extremely unproportional. As seen in Figures 3 and 4, LG TVs' "Dual Play" simply "stretches up" players' perspectives to fill the screen, resulting in, for example, tall zombies and long and narrow subway display screen. Such low quality performance is certainly not fit for the extreme high cost.

Fig.1. Half-split Multiplayer Game Borderlands [1]

Fig. 2. Quarter-split Multiplayer Game Halo [2]

Fig. 3. Traditional Multiplayer Mode [3]

Y. H. Author from Hubei, China (e-mail: yujiahuo@umass.edu).
S. G. Author from Beijing, China (e-mail: suoangao@umass.edu).
L. W. Author from MA, USA (e-mail: lingmengwu@umass.edu).
S. W. Author from Hubei, China (e-mail: siyuwu@umass.edu).

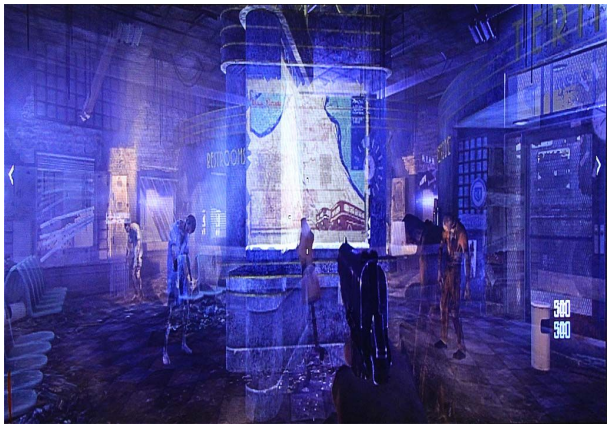

Fig. 4. Multiplayer Mode on LG 3D TV [3]

Our system is able to take two arbitrary source inputs and display them at the full screen simultaneously on any 3D TV that uses Top-and-Bottom 3D format. There are two main parts of our solution: input processing and remote control. Input processing subsystem merges two input into Top-and-Bottom 3D format while maintaining the frame rate of 30 frames rate per second in Python. The resulting video stream is then displayed on a 3D TV. The 3D TV is paired with passive 3D glasses, which are extremely light and comfortable to wear. The remote controller is made up of a microcontroller, a Bluetooth module, and switches. The microcontroller reads data from switches as input from users and ask Bluetooth module to send corresponding data via Bluetooth to the macOs in order to swap sources or switch between modes. All components is packaged using a 3D printed case. More specifications are shown in Table 1.

Specification	Value
3D TV Glasses Weight	< 10g
3D TV Size	27 inches
3D TV Resolution	1080p (1092px × 1080px)
Microcontroller	ATmega32
Bluetooth Module	HC-05
Remote Control Weight	< 100g
Remote Control Clock	4MHz
Remote Control Baud Rate	9600 bits/sec
Remote Control Battery Life	7 hours
Input Video Stream Frame Rate	30 fps
Input Video Stream Resolution	1440 pixels x 1080 pixels
Output Video Stream Frame Rate	30 fps
Output Video Stream Resolution	1440 pixels x 540 pixels

Table 1. System Specifications

The basic requirements for Dual Play are a 3D TV, a computer, and well internet connection. The users can use Dual Play by simply connecting ipads to the computer and connecting the computer to the 3D TV.

Please notice that the idea and goal of this project is similar to an existed patent, '*3D Shutter Glasses with Mode Switching Based on Orientation to Display Device*, patent number US20100177174A1' [4], but still there are some obvious technical differences between the two projects despite that they're trying to solve a similar problem. For example, the main optical technique of the existed patent is Active Shutter Filter Glasses, versus us, Polarization Filter Glass. The main system in the existed patent is built in hardware, but our project's main system is built in software. We believe that along your reading through this project report, more difference between the two projects will be discovered.

II. Design

A. Overview

A 3D LCD TV is used to display two video streams simultaneously. The sources come from two iPads, and the video streams are collected on a MacBook laptop using a third-party screen mirroring software. In order to maintain the input frame rate, another third-party software is used to generate a virtual camera that streams the two input video streams. Resulting camera video stream is read and processed by our python software. The resulting video stream is then sent into the HDMI input of the 3D LCD TV, which projects the two sources using different polarized light. Two pairs of 3D glasses filter out the corresponding video for each user, and sounds for each video are sent through the corresponding set of bluetooth headphones. In addition, users can use a remote controller to switch between "dual play" and "single source" modes, and swap video sources. Figure 5 shows an overview of our system.

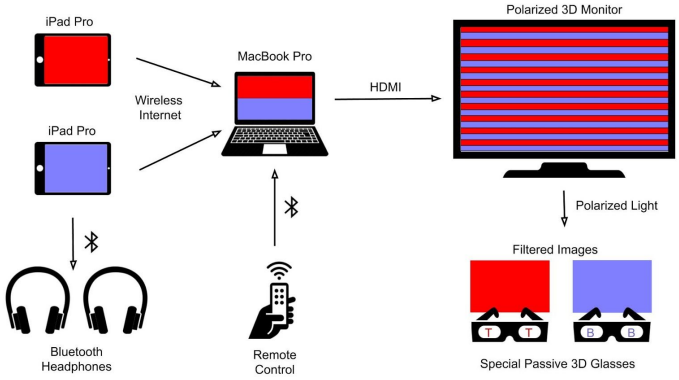

Fig. 5. Overview of Dual Play System

There are five blocks in our system, as shown in Figure 6. The first block is "Input". The iPad screen contents are sent to computer using a third-party software called Reflector 3 [5], which is a screen mirroring software. The resolution for each video source is 1440x1080 pixels, and the frame rate is 30 frame per second. Time delay is imperceptible. The second block is computer processing. It collects video sources from Reflector 3 and uses third-party software ManyCam[6] to

Y. H. Author from Hubei, China (e-mail: yujiahuo@umass.edu).
S. G. Author from Beijing, China (e-mail: suoangao@umass.edu).
L. W. Author from MA, USA (e-mail: lingmengwu@umass.edu).
S. W. Author from Hubei, China (e-mail: siyuwu@umass.edu).

create a virtual camera streaming the two input sources. Our python program reads video stream from the virtual camera and process the video stream directly. The resulting video stream is sent to the 3D LCD TV through HDMI. The delay caused by input processing is imperceptible.

The python software is also responsible for receiving signals sent from the remote control. Two processes, video processing and signal receiving, are running in parallel using multi-threaded processing. The output video has the same frame rate, 30 frames per second, as the input video streams. The third block is the 3D LCD TV, which displays merged video from the computer in two different polarizations respectively. The resolution of the video each user receives is half of the resolution of original source, 1440x540 pixels. In our case, the 3D LCD TV is a second-hand LG 3D monitor.

The fourth block is "Remote Control", built with microcontroller, Bluetooth Module and switches. The microcontroller reads status from switches and tells Bluetooth module to send bits to the computer by Bluetooth. The remote control has a clock of 4MHz and a baud rate of 9600 bits per second, which means there won't be any perceivable delay between users pressing the switches and the computer receiving the signals. It is also very lightweight, less than 100g, and has a battery life of 7 hours. The fifth block is the user side. Each user receives the wanted video through 3D glasses with polarizer filters, and sounds are directed to users through bluetooth headphones connected directly to the iPads.

To summarize, each user receives a video resolution of no less than 1440x1080 pixels and a frame rate of 30 frame/second. The latency is imperceptible. There is no delay when users use the remote to swap sources. Additionally, excellent audio environment is provided through private headphones.

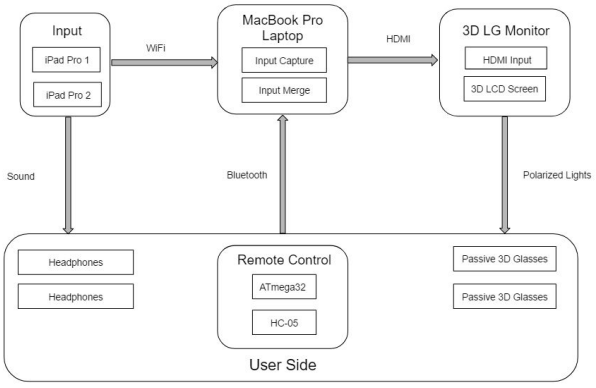

Fig. 6. Block Diagram

B. Input

This block illustrates the input of our system, which are two independent iPads. Users can do anything on their iPads. The screen contents of two iPads are sent to the macOs simultaneously and wirelessly via "AirPlay". "Airplay" is a feature on all Apple products, which allows users to stream or mirror the screen of an apple device, the ipads in our case, on any "Airplay" receive enabled devices.

In our design, we use a third party software "Reflector 3", which uses the "Airplay" feature to mirror two iPad screens. "Reflector 3" runs on a computer and is used to capture inputs for our system. As shown in Figure 7, screen contents of two iPads are captured on macOs.

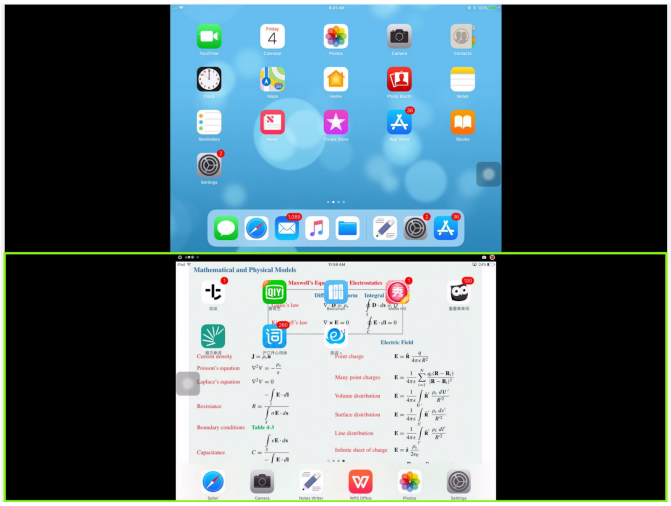

Fig. 7. Captured Screen Contents of Two iPads in One of Windows of ManyCam

Then software "ManyCam" is used to generate a virtual camera for video processing program to read streams of the source. The frame size of the streams from the camera is 1440x1080 pixels for each source. The frame rate is set to be 30 frames/second. Using this way to pass the video streams can gain good the frame rate and frame size to achieve our real time video processing. There are four windows in ManyCam. The content of virtual camera can be chosen from any of these four window The first one is set to two sources Top-and-Bottom. The second one is set the same as the first one but sources are swapped. The third one is set to be only one source. And the fourth one is set to be the other source. So we can use python to switch the content of the virtual camera that ManyCam generates to achieve the change between Dual Play mode and single source mode of our system.

Y. H. Author from Hubei, China (e-mail: yujiahuo@umass.edu).
S. G. Author from Beijing, China (e-mail: suoangao@umass.edu).
L. W. Author from MA, USA (e-mail: lingmengwu@umass.edu).
S. W. Author from Hubei, China (e-mail: siyuwu@umass.edu).

C. *Software, Image Processing System Drive*

Our goal is to build a image processing system software to combine two streams of videos together and organize them in the Top-and-Bottom configuration as shown in Figure 8, if the remote controller chooses dual-play mode. Also, display each individual source separately if other mode was chosen. The platform that we've used to conduct our software is a macOS laptop. As we mentioned before, we collect video sources by frame using a third party software called 'Reflector 3' via Apple Airplay. It transfers what is playing on each of the two iPads' screen to our macOS platform as a window. Then, we use another software called ManyCam to configure our two original inputs as one formatted input with the demanded format base of the 3D monitor and generate it as a virtual camera. This formatted input will display what displays on the two iPads. Our program then read the formatted input as source frames directly from the virtual camera.

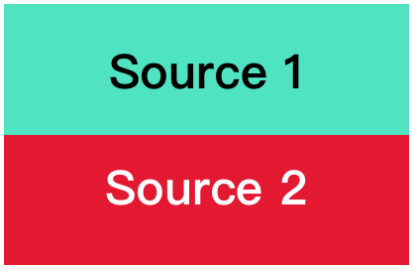

Fig.8. Each Frame of Top-and-Bottom Format 3D Video

The main program is written in Python [7] with several external libraries employed, including: *Numpy* [8], *opencv* [9], *PySerial* [10], *and PyAutoGui* [11]. *Numpy* is the most efficient matrix calculation library in Python. *OpenCV* is an image and video processing library been widely used in computer vision. *PySerial* is the library we use to interact with the bluetooth serial port. *PyAutoGUI* will operate interaction with ManyCam.

We know that a piece of video is nothing more than frames of images displayed sequentially. In order to combine two streams of videos all together, what we must do is simply combine all of their frames of images, which is the formatted input we mentioned above. We did this by using the Top-and-Bottom configuration in ManyCam as in Figure 7. However, this configuration also produced some undemanded margins which are not from our original inputs (the black area in Figure 8). We need to get rid of them.

Before we do anything about these formatted input images, we must have those images beforehand. We use *OpenCV* library (imported as "cv2" in python), "cv2.videoCapture" to get frames by frames of formatted images from formatted video stream.

We now are able to deal with the undemanded margin of each frame of formatted input video stream. We have used *Numpy* to turn the formatted images into a numpy array, then chopped off the undesired columns as shown in Figure 9.

Fig. 9. Top-and-Bottom Configuration of ManyCam without Margins

We also need to get rid of all of the odd rows to keep the original ratio. Numpy once again will be our tool to complete this operation. In Figure 10, you will see the final display on the laptop screen. Since each source was chopped half of the original size to keep the 1440 * 1080 display on 3D monitor, output resolution of each view would become 1440 x 540. And since we directly process the video stream, there is no latency during the processing. Due to the relatively low performance of an integrated graphic processor, the system FPS rate (33 fps) is lower than the original (60 fps). However, it still meet our system specification of more than 30 fps.

Fig. 10. Refined Configuration, Rows are Chopped.

Y. H. Author from Hubei, China (e-mail: yujiahuo@umass.edu).
S. G. Author from Beijing, China (e-mail: suoangao@umass.edu).
L. W. Author from MA, USA (e-mail: lingmengwu@umass.edu).
S. W. Author from Hubei, China (e-mail: siyuwu@umass.edu).

Since this system should be able to change between dual play mode and single mode based on the signal sent from the remote controller via bluetooth, Our program also needs a function that could read signal data from remote controller - Mac bluetooth channel. As we introduced before, PySerial will solve this problem. By using the serial port, we could store received signal to a variable and check it in a while loop. However, in Python, it's not allowed to access serial port together from two method. Which means you could either use OpenCV's videoCapture method, or PySerial's serial method, but not at the same time. To solve this problem, we changed our original structure and implemented it in multi-threading structure. Figure 11 is the output console of the program while reading signal from the bluetooth channel and reading data from the video stream at the same time. Where, b'1' is the signal received from the bluetooth channel, and 'Read Frame' indicates that our program is reading data from video stream

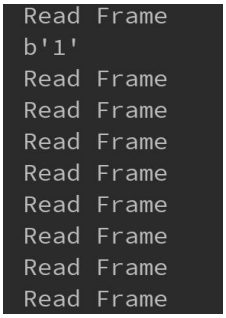

Fig.11. Output Console of the Program.

Next, we will use PyAutoGUI to response base on the signal. We pre-designed four ManyCam configurations. Each one of the four configurations represents one of the four system mode (Dual Play1, Dual Play2, Single1, Single2). PyAutoGUI will switch between the four ManyCam configurations.

D. 3D LCD Screen

The 3D screen displays the iPad screen contents for users. In Dual Play mode, from HDMI, it takes Top-and-bottom format video with the total resolution of 1440x1080 pixels and interlaces pixel columns from top half and bottom half of the video (Figure 12). Odd pixel rows will display one source using clockwise circular polarized light and even pixel rows will display the other source using counter clockwise circular polarized light. However, the resolution of each image becomes half of the original source. In single source mode, it onlys display one source. What we have is a LG 3D LCD Monitor D2792P, with the resolution of 1080p. So it is able to display a source with 1440x1080 pixels.

We the glasses that we buy, the vertical angle of this 3D TV is pretty small. The eye slight should be at the same height of the 3D to get the best view without 2 sources overlapped seen in one pair of glasses. We have measured that the horizontal angle range which should be within ±22.5 degrees whiling facing the TV. And the best view point should be at around 6 feets in front of the TV.

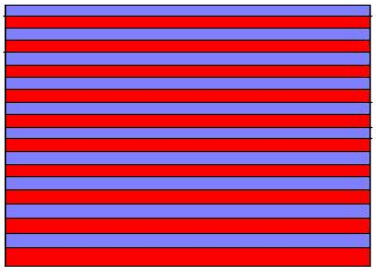

Fig.12. What will display on 3D Screen

E. Users

This block illustrates the user side in our program, which consists of three parts: Remote control, headphones and glasses.

1) Remote Controller

The remote controller allows users to change modes or swap sources. There are three switches. One is the power ON/OFF. Other two are used to change modes and sources. One is the ON/OFF switch for "dual play" mode, and the other one is for swapping sources. There are total four modes. In addition, each switch has a LED When a switch is toggled, the remote controller sends one byte representing a state to the computer that is producing and sending the output video steam. When "dual play" mode is on, the merged video stream is displayed on the 3D TV, and the second switch can swap the sources corresponding to each pair of glasses. When "dual play" mode is off, 3D LCD screen only display one of the unprocessed video sources, and the second switch can swap between the two sources.

As shown in Figure 13, there are two main components of the remote controller: microcontroller ATmega32 and Bluetooth module HC-05. ATmega32 uses a 4MHz clock. Two switches that are responsible for modes are attached to ATmega32, which reads switch status and send them to the computer using HC-05. HC-05 acts as an serial port and sends one byte data to the remote macOs once it is paired with macOs bluetooth. The baud rate of the bluetooth transmission and macOs bluetooth receiver is set to be 9600 bits/s.

Programming on ATmega32 is similar to the Project 2 in ECE 353, Computer System Lab I: a C program is written and

Y. H. Author from Hubei, China (e-mail: yujiahuo@umass.edu).
S. G. Author from Beijing, China (e-mail: suoangao@umass.edu).
L. W. Author from MA, USA (e-mail: lingmengwu@umass.edu).
S. W. Author from Hubei, China (e-mail: siyuwu@umass.edu).

loaded onto the microcontroller. In the code, the states of two switches are read. The asynchronous transmitter (TXD) of ATmega32 and receiver (RX) of HC-05 are connected. Microcontroller will keep detecting whether switch state has changed. Once the state has changed, TXD will send one byte to HC-05. And HC-05 will send the data to macOs. Switch states, modes, and the one byte data are listed in the table 2 below.

Switch states	Mode	One-byte Data
00	Dual Play	character '1'
01	Dual Play	character '2'
10	Single Source	character '3'
11	Single Source	character '4'

Table 2. Four Modes of the Remote Control

In addition, a 4 MHz clock is used for microcontroller. Taking the size and weight of the remote control into consideration, only two triple-A batteries are used as power supply. Therefore, a step up voltage converter is used to convert 3V to 5V. We have measured that the peak total current is about 160 mA. The total charges of one triple-A batteries is 1.15 Ah [12]. The output voltage is 1.5 V. The total capacity of two batteries is calculated below.

$$E=1.15\ A \times 3600\ s \times 1.5\ V \times 2 = 12420\ J$$

The life time of our remote control is,

$$12420\ J / 3\ V / 160\ mA \approx 7\ hours$$

Therefore our remote control can run around 7 hours.

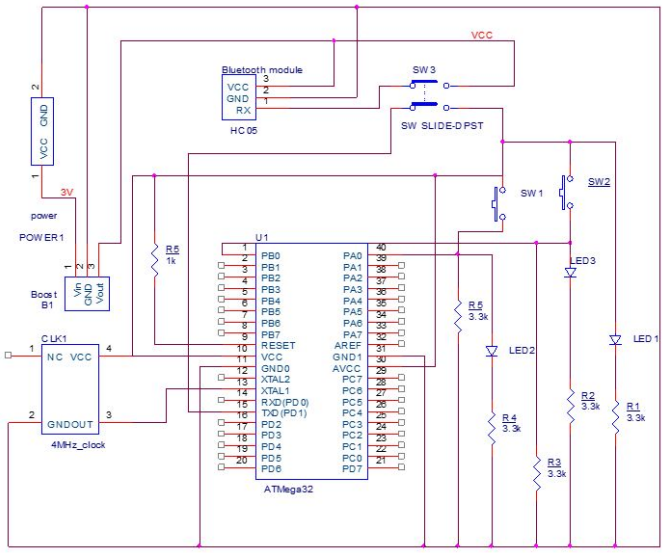

Fig. 13. Schematic of Remote Controller

The remote control is composed of customized PCB (Figure 14) and a 3D printing case (Figure 15). We used OrCAD PCB Editor to design the PCB. It has two layers. The size is 100mm x 100mm. Then the Gerber files were sent to the company JLCPCB to do manufacture. The outer dimension of 3D printing case is 106mm x 106mm x 28.5mm. Including batteries, the weight of the remote control is less than 100g. There is no perceptible delay for our system to switch the display modes.

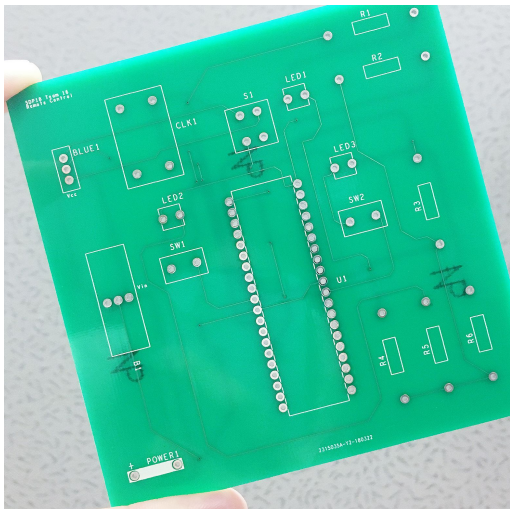

Fig.14. PCB for Remote Control

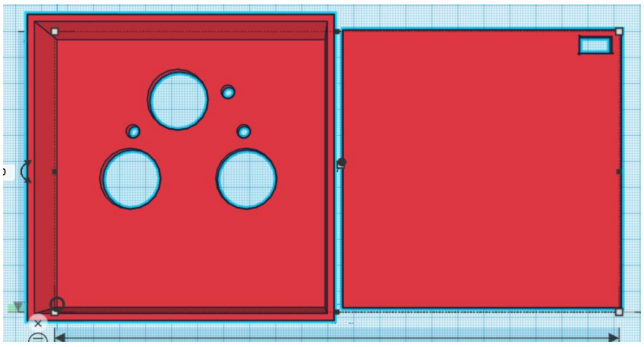

Fig.15. 3D Printing Remote Control Case

2) Headphones and Glasses.

For headphones, we use one pair of bluetooth headphones for each user. For glasses, we use "dual play" passive 3D glasses. Different from typical passive 3D glasses, the two lenses filters on each pair of glasses are the same, which means that both lenses allow same clockwise light (or counterclockwise) and block counter-clockwise (clockwise) light. Each user can only watch one image in full screen in dual-play mode (Figure 15). And Figure 16 shows the image seen from different glasses

Y. H. Author from Hubei, China (e-mail: yujiahuo@umass.edu).
S. G. Author from Beijing, China (e-mail: suoangao@umass.edu).
L. W. Author from MA, USA (e-mail: lingmengwu@umass.edu).
S. W. Author from Hubei, China (e-mail: siyuwu@umass.edu).

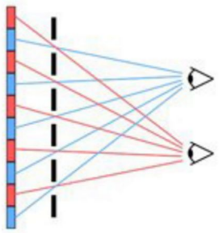

Fig.15. Contents that each user receives

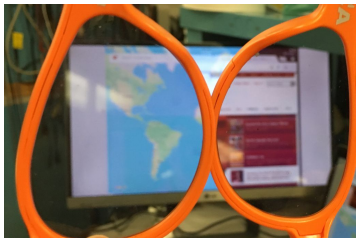

Fig. 16. Contents Seen by Each Pair of Glasses

III. Project Management

In the second semester, we have completed the real-time video processing program. Also for hardware, we complete remote controller circuit using Bluetooth module HC-05 and microcontroller ATmega32. Then we achieve the successful communication between Mac and remote controller. Details could be found in Table 3.

Task	state
Python program for real-time video processed	Finished
Remote controller circuits design using Bluetooth	Finished
Bluetooth module communicate with PC	Finished
Remote controller circuit PCB layout	Finished
Remote controller case	Finished

Table 3. FDR Goals

Every person in our team has contributed to this project. We helped each other to figure out the problem and exchange our ideas while working on the project. We had weekly group meetings and advisor meeting.

When our group encountered any difficulties, we worked together to find proper solution. We are divided into two subgroups: hardware and software. Siyu and Yujia are responsible for the hardware part, because they are more familiar with microcontroller and circuits. Lina and SuoAn are responsible for software. They are skilled in programming. Great division of work increases the efficiency and weekly meetings help to communicate to make project go smoothly. Table 4 illustrates the timeline of this project.

Y. H. Author from Hubei, China (e-mail: yujiahuo@umass.edu).
S. G. Author from Beijing, China (e-mail: suoangao@umass.edu).
L. W. Author from MA, USA (e-mail: lingmengwu@umass.edu).
S. W. Author from Hubei, China (e-mail: siyuwu@umass.edu).

	Sept.	Oct.	Nov.	Dec.	Feb.	Mar.	Apr.
G2: Python program to collect videos							
G2: Python program to convert videos to frames							
G2: Python program to convert frames to videos							
G1: Remote controller circuits design using Wi-Fi							
G1: Remote controller circuits design using Bluetooth							
G1: Sender and receiver programming of remote controller							
G2: Python program for real-time video processing							
G2: Program for communication between PC and Remote controller							
G1: Remote controller PCB layout and case							

Group 1 (Hardware): Yujia Huo, Siyu Wu
Group 2 (Software): SuoAn Gao, Lina Wu

Table 4. Project Timeline

IV. Conclusion

We've achieved all specification of our project. We programmed the real-time video processing and designed our remote controller. Bluetooth is used to communicate with Mac to control the displayed mode.

During this year, we continually observed and solved technical problems. We helped each other and also we got a lot of assistances from professors staffs.

V. Acknowledgment

Professor Zlatan Aksamija
Professor Wayne Burleson
Francis Caron
Shira Epstein
Professor Christopher V. Hollot
Professor Ramakrishna Janaswamy
Keith Shimeld
Professor T. Baird Soules

All faculty and staff in University of Massachusetts Electrical and Computer Engineering Department who ever assisted with this project.

VI. References

[1] "Decline of Split Screen Games". Gigaventure. Available at: http://www.gigaventure.com/2012/02/24/decline-of-split-screen-games/. [Accessed: 7 May. 2018]

[2] "The State of Split-Screen Gaming, Fall 2016". The Verge. Available at: https://www.theverge.com/tldr/2016/10/14/13225992/split-screen-gaming-gears-of-war-4-couch-star-wars-battlefront. [Accessed: 7 May. 2018]

[3] "Split Screen Game Full Screen". Tweaking4All. Available at: https://www.tweaking4all.com/video/gaming/split-screen-game-full-screen/. [Accessed: 7 May. 2018]

[4] "3D Shutter Glasses with Mode Switching Based on Orientation to Display Device" , patent number: US20100177174A1 https://patents.google.com/patent/US20100177174A1/en [Accessed: 7 May. 2018]

[5] "Screen Mirroring Features for Mac and Windows | Reflector 3." *Squirrels - Apps to Go Nuts for.* www.airsquirrels.com/reflector/features/mac-and-windows/. [Accessed: 7 May. 2018]

[6] "Your Live Streaming Experience Enhanced." *Free Webcam Software and Screen Recorder | ManyCam*, manycam.com/.[Accessed: 7 May. 2018]

[7] "Python"created by Guido van Rossum and first released in 1991. Available at: https://www.python.org/. [[Accessed: 7 May. 2018]

[8] "OpenCV"python graphical and video editing library. OpenCV (Open Source Computer Vision Library) is released under a BSD license and hence it's free for both academic and commercial use. could be find in https://opencv.org/ [Accessed: 7 May. 2018].

[9] "Numpy" is the fundamental package for scientific computing with Python. Available at: http://www.numpy.org/ [Accessed: 7 May. 2018].

[10] "PySerial" https://pypi.org/project/pyserial/[Accessed: 7 May. 2018]

[11] "PyAutoGUI" https://pyautogui.readthedocs.io/en/latest/[Accessed: 7 May. 2018]

[12] "Duracell MN2400BKD". Allied Electronics Automation. Available at: https://www.alliedelec.com/duracell-mn2400bkd/70149226/. [Accessed: 7 May. 2018]

Y. H. Author from Hubei, China (e-mail: yujiahuo@umass.edu).
S. G. Author from Beijing, China (e-mail: suoangao@umass.edu).
L. W. Author from MA, USA (e-mail: lingmengwu@umass.edu).
S. W. Author from Hubei, China (e-mail: siyuwu@umass.edu).