First Person VR Interface with RC Car

Team 7 – Deniz Yıldırım, Erik Jacobson, Sang Baek Han
ECE 445 Design Document – Spring 2020
TA: Weihang Liang

1 Introduction

1.1 Objective

RC (Radio-controlled) cars are very fun but often lack immersion, and they are hard to control because the person controlling it can lose line of sight. We want to give players the perception that they are a small person inside a RC car, making the experience more immersive and fun. In order to make this happen, we will use a VR (Virtual Reality) headset and a fisheye camera that can display in 180 degrees. The players controlling the RC car would never lose sense of where the RC car is as they would feel like they are inside the car and would be able to see its surrounding by looking around. A steering wheel and hands could be rendered on top of the video feed to let players control the car by holding the virtual steering wheel with their VR controllers.

1.2 Background

While there are some RC cars with cameras in the market, they fail at being immersive because they keep using traditional remote controllers and often do not give users the ability to see the surrounding of RC car as the camera faces only forward [1]. The RC toy vehicles in general do not gain much progress in terms of user experience. In the VR community, there is little interest and research on using VR as an input and output device for robots compared to other areas where VR is used [2]. Thus, this project is a good opportunity to explore new and different ways VR technology can be used. While our project is about RC cars, the idea challenges the usual approach to virtual reality and hopes to widen its use cases. We hope to experiment with the idea that while VR can make us exist in immersive virtual environments, it can also make us exist in real but unreachable environments such as inside an RC car. The same concept can be applied to have a more immersive presence in parts of the world that humans cannot reach as easily as robots do.

1.3 High-Level Requirements

- Camera must be able to display the first-person point of view from radio-controlled car through virtual reality headset with less than 1 second delay of video transmission.
- Virtual reality headset must be able to display the appropriate field of view based on the direction where the user faces, in 30 frames per second with 720p resolution.
- Controller must be able to wirelessly control the four directions and the speed of radio-controlled car.

1.4 Visual Aid

Our product will look very similar to what is shown in Figure 1 below, which is the commercially available RC car with a forward-view camera and the virtual reality headset. The major difference will be the camera itself. The VR headset RC car available on the market has a camera that can only see the front view, not allowing the user to look around the surrounding of RC car. Our product will use 180 degrees fisheye camera instead of single lens camera to provide 180 degrees surrounding view so the user can have a look around the surrounding by simply facing the different direction while wearing the VR headset.

Figure 1. Visual Aid of Commercially Available Radio-controlled car with VR headset

2 Design

The successful working project consists of majorly five sections: a DC motor drive subsystem, a RC controller subsystem, a video transmission subsystem, a VR headset facing direction subsystem, and a power supply subsystem. The entire flow of the signals and the connections are shown in Figure 2 below.

A controller sends the direction and the speed signals to the Bluetooth, which the microcontroller receives them to control the PWM (Pulse-width modulation) signals. Appropriate PWM signals are sent to each H-bridge circuit, which controls the speed and the forward/reverse direction of DC (direct current) motor. The wheel rotates as the DC motor rotates.

A 180 degrees fisheye camera receives the curvilinear image in RGB, which is sent to the single-board computer. The curvilinear image is sent to the computer through Wi-Fi from the single-board computer. The computer converts the curvilinear image to the rectilinear image, then sent to the VR headset to display the appropriate portion of view to the user.

The 7.4V battery is the power source for the components that are on the RC car. The 5V voltage regulator is used to provide appropriate voltage to each component. The controller and VR headset have separate batteries attached on them.

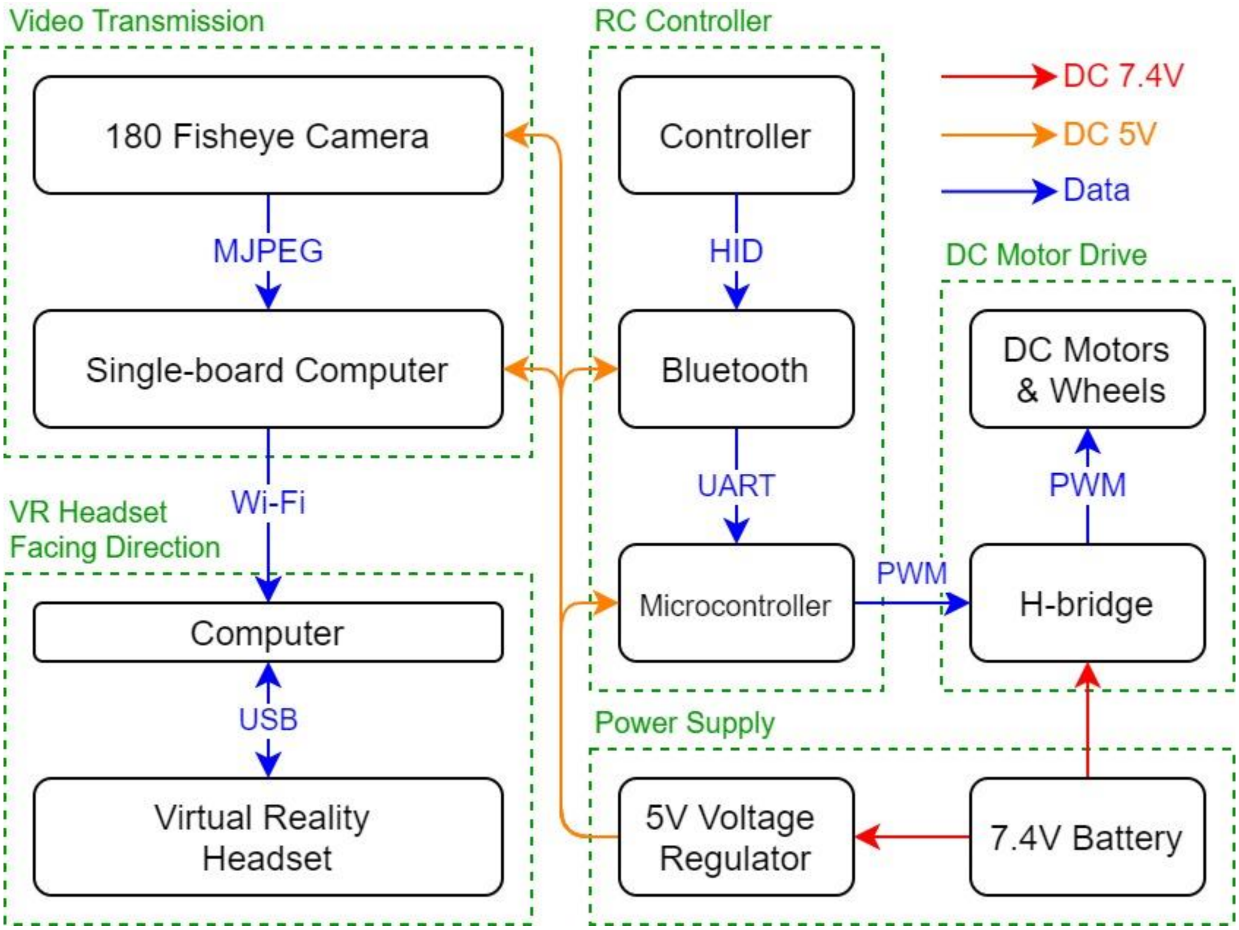

Figure 2. Block Diagram of Radio-controlled Car, Controller, and Virtual Reality Headset

The RC car has rest of the components attached besides from the controller and the VR headset. Figure 3 below is the physical diagram that shows the approximate locations of where the components will be located on the RC car. Four wheels are directly attached each DC motor. Four DC motors are directly connected to each H-bridge on the PCB (printed circuit board). Four H-bridge circuits are connected to each output port of the microcontroller. Appropriate output ports on the microcontroller send out appropriate PWM signals to each H-bridge circuit. Bluetooth chip is attached on PCB, which is connected to the microcontroller, to receive the speed and the direction signal from the controller.

7.4V battery is connected to the PCB to provide appropriate voltages to each component. On the PCB, 6V battery is connected to two parts: H-bridge circuits and 5V voltage regulator. 5V voltage regulator is connected to the microcontroller, the Bluetooth chip, and the single-board computer to provide 5V voltage supply. The single-board computer is connected to the fisheye camera to provide 5V voltage supply to the fisheye camera and to receive the curvilinear image from it.

The controller, the computer, and the VR headset are not designed by ourselves, so these are not shown in the physical diagram.

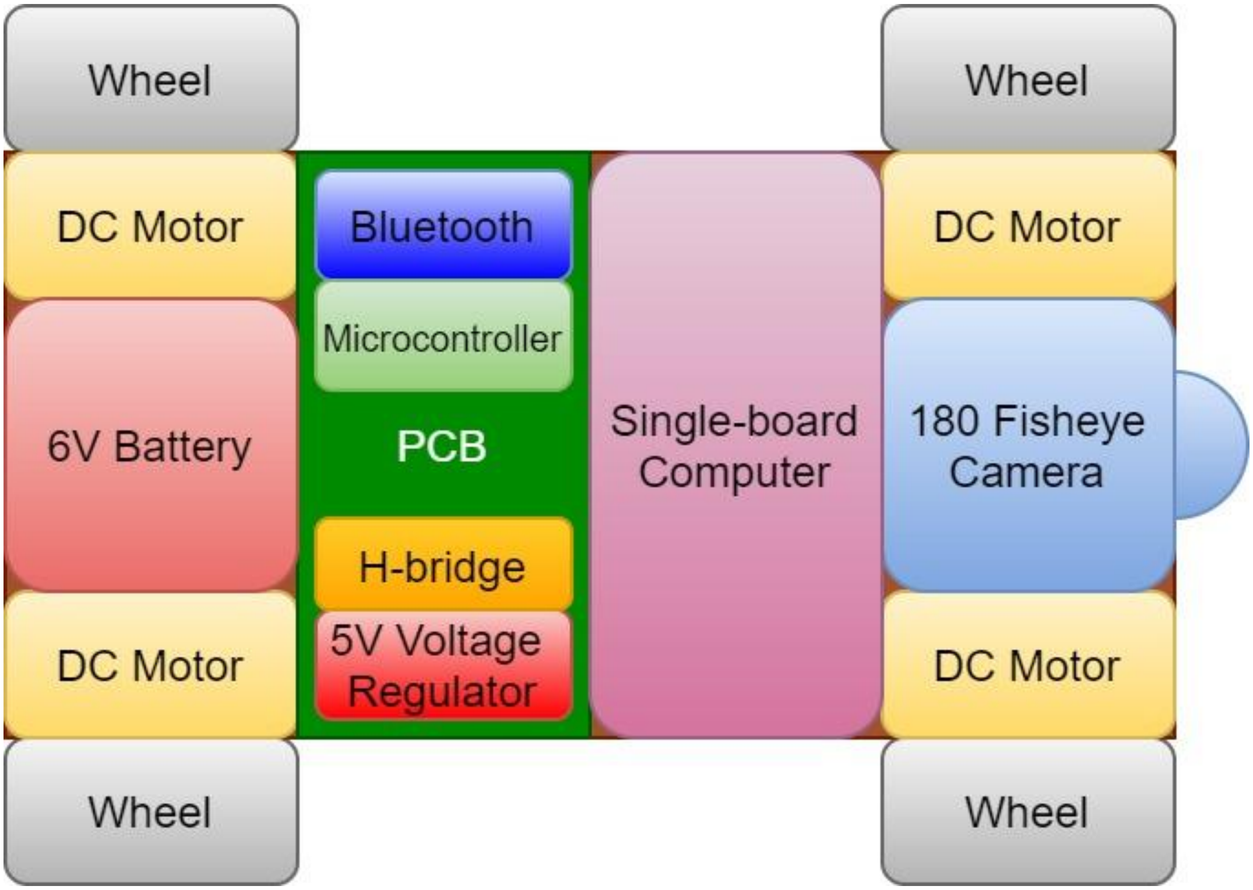

Figure 3. Physical Diagram of Radio-controlled Car

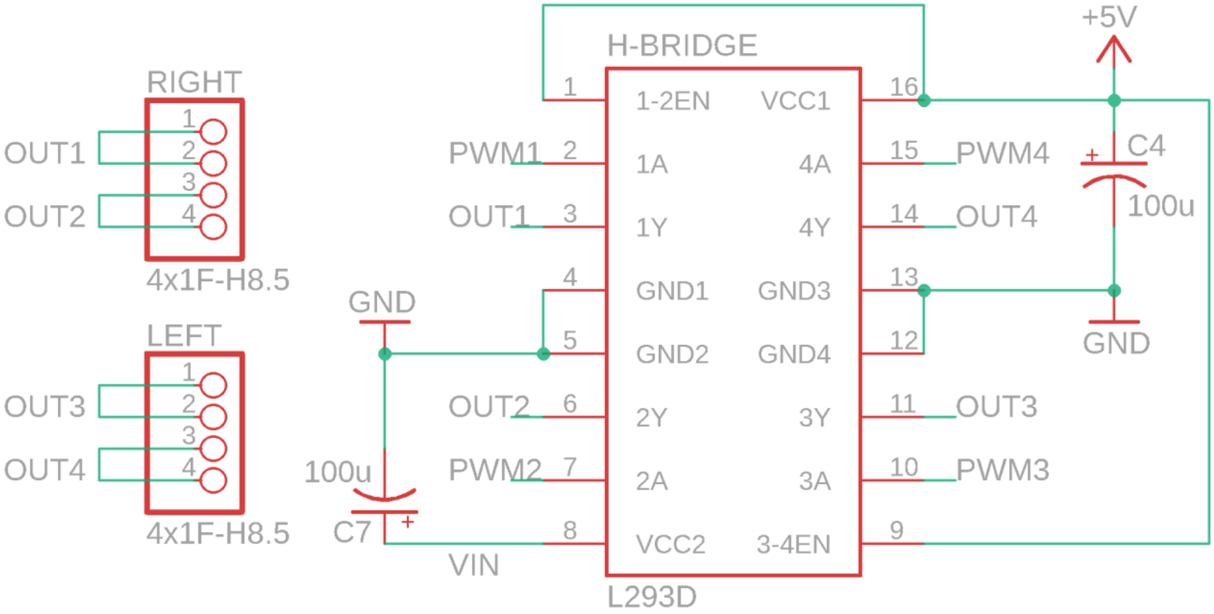

Figure 4. DC Motor Drive Schematic

2.1 DC Motor Drive Subsystem

The DC motor drive subsystem is the body of RC car. Four wheels are attached on four DC motors, which are connected to H-bridge to control the clockwise or the counterclockwise rotation. PWM signals sent to each H-bridge circuit to control the speed of DC motors.

Figure 4 above shows the schematic of DC Motor Drive Subsystem. Appropriate decoupling capacitors are attached to the ground and the supply voltage. PWM1, PWM2, PWM3, and PWM4 are the input PWM signals from the microcontroller. OUT1, OUT2, OUT3, and OUT4 are the output PWM signals to the DC motors. VCC1 is supplied with DC 5V while VCC2 is supplied with DC 7.4V, which is referred as Vin on the schematic.

2.1.1 H-bridge

H-bridge controls the direction and the speed that the DC motor will rotate. PWM signal from the microcontroller is fed into the appropriate H-bridge circuits to control the speed of DC motors. Besides the control of forward and reverse rotation of DC motor, H-bridge also prevents short circuits. For H-bridge circuit on our PCB, L293D quadruple half-h drivers will be used.

Requirement	*Verification*
1. $I_{out} = 600mA \pm 30mA$ at 5V	1. A) Attach 8.33Ω resistor as a load on the output of L293D B) Attach the oscilloscope across the load. C) Input 5V supply into L293D. D) Measure the current through oscilloscope and ensure the output current is within 570mA and 630mA.

2.1.2 DC Motors and Wheels

The wheels of same size are attached to the direct current motors of same products. DC motors rotate in forward or reverse direction based on the H-bridge. The speed of DC motors is controlled by PWM signal. TT Motor from Adafruit will be used as a DC motor. TT motor runs at voltage of 3V to 6V with the current of 150mA. The operating speed is 90RPM to 200RPM.

Requirement	*Verification*
1. The wheels must be same size with same weight to maintain the balance. 2. The DC motors must be same products to have a same speed output when moving forward or reverse.	1. A) Measure the diameter of each wheel with the ruler and ensure the diameter is $\pm 1\%$ from each other. B) Measure the weight of each wheel with a scale and ensure the weight is $\pm 1\%$ of each other. 2. Observe the RC car move in a straight line when the RC car is moving forward or reverse only.

2.2 Radio-controlled Controller Subsystem

The radio-controlled controller subsystem is the brain of the RC car. The controller allows the user to control the direction of the movement and how fast the RC car moves. The Bluetooth chip is connected to the microcontroller to receive these control signals from the controller. The microcontroller decides which H-bridge circuit will get the PWM signals to rotate the DC motors based from the control signals received from the Bluetooth chip.

Figure 5 below shows the schematic of Radio-controlled Controller Subsystem. Appropriate decoupling capacitors and resistors are attached to the ground and the supply voltage. PWM1, PWM2, PWM3, and PWM4 are the output PWM signals to the H-bridge chip. TXD is connected with the voltage divider circuit to produce 3.3V signal for HC-06 Bluetooth chip.

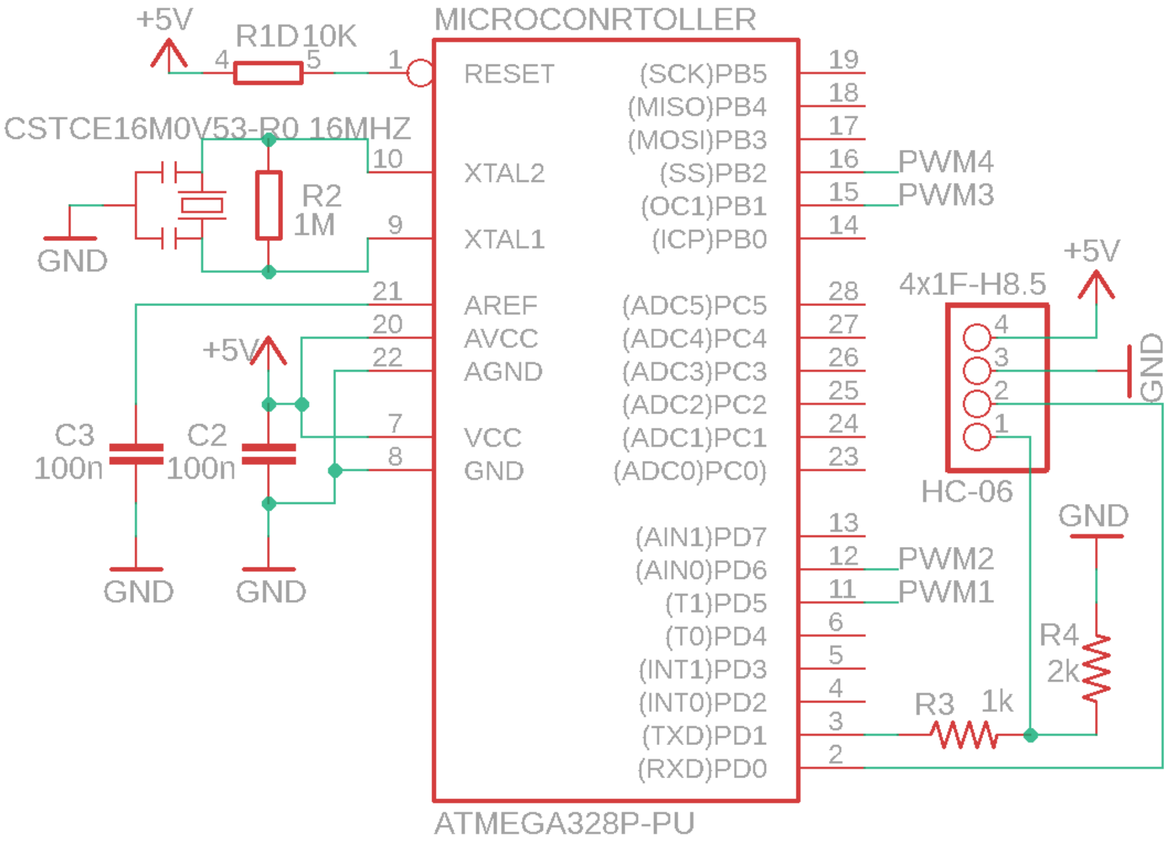

Figure 5. Radio-controlled Controller Schematic

2.2.1 Microcontroller

The microcontroller handles the PWM signal and which DC motors to rotate. It communicates with the Bluetooth chip, which receives the speed signal and the direction of RC car movement from the controller. The speed signal is converted to the PWM signal in appropriate ratio and the direction of RC car movement is controlled based on the decision of which DC motors to rotate forward and reverse. ATmega328P-PU will satisfy our requirement. This chip operates at 5V and outputs PWM signals to L293D H-bridge chip.

Requirement	*Verification*
1. The microcontroller must be able to produce PWM signal from the speed signal given by the controller. 2. The microcontroller must be able to understand which DC motors to rotate based from the direction signal given by the controller.	1. A) Power the microcontroller with 5V. B) Attach oscilloscope probe to PWM signal output pins on the microcontroller. C) Upload the code setting appropriate output pins to square wave with 50% duty cycle. D) Verify the square wave with 50% duty cycle from the oscilloscope. 2. A) Power the microcontroller with 5V. B) Attach oscilloscope probe to PWM signal output pins on the microcontroller. C) Upload the code that controls which pins to output PWM signals. D) Verify the PWM signal is shown only at the time that the controller gives the directional signal.

2.2.2 Bluetooth

The Bluetooth chip receives the signals that determine the speed and the direction of movement of the RC car from the controller. These signals are sent to the microcontroller to control the DC motors. HC-06 Bluetooth module will be used for our project. This module operates at 3.3V to 6V and can work within the range of 9m. Yet, RXD of HC-06 need to be 3.3V. The voltage divider circuit is used to make 5V signal to 3.3V signal as shown in Figure 5 above.

Requirement	*Verification*
1. The Bluetooth chip must be able to communicate over HID (Human Interface Device) with the controller and UART (Universal Asynchronous Receiver/Transmitter) with the microcontroller.	1. A) Assemble Bluetooth chip on PCB. B) Observe the speed of RC car movement when the controller input different speed signals. C) Observe which DC motors run when the controller input different directional signals.

2.2.3 Controller

The controller allows the user to decide which direction to move the RC car at how fast the user want. It communicates in HID with the Bluetooth chip. There will be two version of the controllers: the smartphone version for prototyping and the VR controller as a final product.

Requirement	*Verification*
1. The controller must be able to control the speed and the direction of RC car movement.	1. A) Observe the change in speed of RC car when the controller input different speed signal. B) Observe the change in direction of RC car movement when the controller input different directional signal.

2.3 Video Transmission Subsystem

The 180 degrees fisheye camera is attached on the front of RC car, which provides the 180 degrees view to the user. The fisheye camera sends the curvilinear image to the microcontroller and the microcontroller transmits the curvilinear image to the VR headset through Wi-Fi.

2.3.1 180 Fisheye Camera

The 180 degrees fisheye camera inputs the image in curvilinear. This need to be converted to rectilinear image so that the view does not look distorted to the user, but this will be done on the computer to avoid any delay on the microcontroller. OV2710 camera will be used, which support MJPEG video compression in 30 frames per second at 720p.

Requirement	*Verification*
1. The 180 degrees fisheye camera must be able to send the curvilinear image to the microcontroller. 2. The 180 degrees fisheye camera must be able to capture 720p image.	1. Verify that the framerate is suitable for VR, above 30 FPS. 2. Check for packet loss in the transmission and ensure that it is receiving all data sent (which is 720p).

2.3.2 Single-board Computer

The SBC (single-board computer) receives the curvilinear image from the fisheye camera, then transmits to the VR host computer through Wi-Fi. Since the curvilinear to the rectilinear conversion takes place in the VR host computer, there is no image processing needed to be done on the microcontroller. We will use ASUS Tinker board for our SBC.

Requirement	*Verification*
1. The SBC must be able to communicate over MJPEG to receive and send the curvilinear image. 2. The SBC must be able to communicate over Wi-Fi with IEEE 802.11 protocol.	1. Check through our program that the data is indeed in MJPEG format in 30 frames per second at 720p. 2. Using python libraries to communicate through IEEE 802.11 and checking that communication speed is up to specification.

2.4 VR Headset Facing Direction Subsystem

We need to be able to show the video feed to the user by converting the video from curvilinear to rectilinear and displaying it on a VR headset. The video should be reactive to the movements of the headset and the view should not be distorted.

2.4.1 Virtual Reality Headset

A VR headset that connects to a computer is required to show the video feed to the user. The VR headset should be compatible with the Unity software. Head movements should be tracked, and the appropriate portion of the view must be shown according to the direction the headset is facing. Because a VR headset is relatively expensive, we are planning to lease one from the library which limits our options to Oculus Rift, Oculus Go, Oculus Quest and HTC Vive. We believe Oculus Rift is the best option for this project since it works well with Unity, is wired to the computer and is easy to work with. We chose Unity as it has downloadable VR libraries that include not only compatibility, but also handles the linear and rotational movements of the user's head.

Requirement	*Verification*
1. The field of view on the VR headset should be at least 90 degrees. 2. Both rotational and linear movements should be detected by the headset 3. The display must at least be capable of 720p resolution and 30fps. 4. The Unity program itself must run at 90 fps to avoid nausea when reacting to user movements.	1. Physically test when objects move out of the FOV and ensure it is wide enough. 2. Check by physically moving and observing the reaction. 3. Check packet size and receive-rate through Unity and ensure it is adequate. 4. Unity has built in tools to detect framerate.

2.5 Power Supply Subsystem

A power supply is required to run the entire system while the power is turned on. The battery directly feed voltage to the H-bridge circuit and the voltage regulator. The voltage regulator regulates 5V to the rest of the chips. The controller and the VR headset have separate batteries attached to them.

Figure 6 below shows the schematic of Power Supply Subsystem. Appropriate decoupling capacitors are attached to the ground and the supply voltage. Two 3.7V Lithium-ion batteries are connected in series to provide 7.4V to Vin. NCP1117 voltage regulator will output DC 5V and this will be supplied to all the chips on PCB. However, L293D H-bridge chip will get two voltage supplies: DC 5V from NCP1117 voltage regulator and DC 7.4V from the two 3.7V Lithium-ion batteries connected in series.

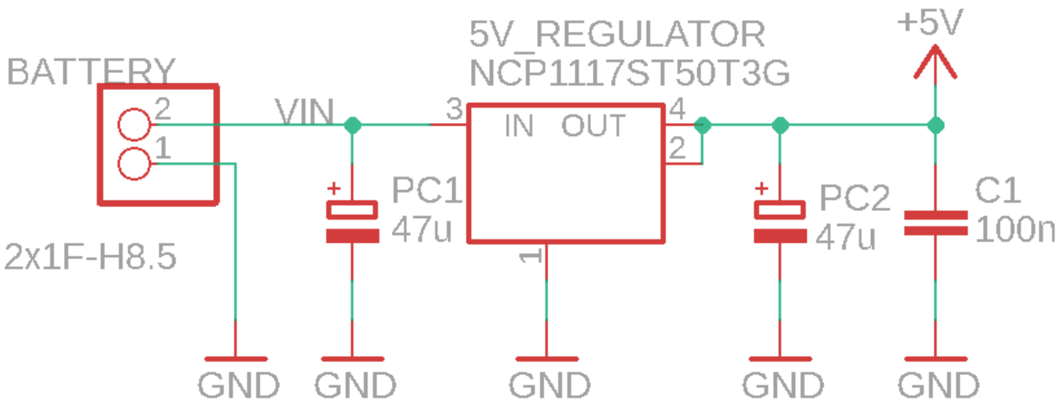

Figure 6. Power Supply Schematic

2.5.1 Battery

Two Lithium-ion batteries will be used to provide supply voltage of 7.4V to 8.4V to the voltage regulator. Two Li-ion batteries are connected in series and the size of battery will be 18650. Each battery has battery life of 2.6AH. We expect the battery to run 3 hours when fully charged.

Requirement	*Verification*
1. The battery must be able to store enough charge to provide at least 1A at 7.4V to 8.4V for 3 hours once fully charged. 2. The battery must be unable to discharge if left plugged.	1. A) Attach 8.4Ω resistor as a load. B) Measure the current through oscilloscope and ensure the output current is approximately 1A. C) Discharge battery for 3 hours and measure to see whether the voltage output is still above 3.7V. 2. A) Fully charge battery and plug the battery on the RC car. C) Allow to sit for 5 days. D) Measure the cell voltage and ensure it is above 3.7V.

2.5.2 Voltage Regulator

The 5V voltage regulator must be able to provide 5V to the rest of chips besides from the H-bridge circuit, the controller, and the VR headset.

Requirement	*Verification*
1. The voltage regulator must provide 5V±5% at 1A.	1. A) Attach 5Ω resistor as a load. B) Supply the regulator with 7.4V DC. C) Measure and ensure the output voltage remains between 4.75V and 5.25V using the oscilloscope.

2.6 Software

2.6.1 Video Feed Injection to Unity

The video feed from the camera is sent to the SBC via a USB port. The SBC sends this video in packets, in MJPEG format, using Wi-Fi to our PC. Once the PC receives the video, we project the feed to a surface in the Unity Engine after correcting distortions caused by the fisheye lens, and send it to the VR headset which is handled automatically by the Oculus SDK in Unity.

We will need some form of software that will receive the video feed from the SBC and prepare it for display in Unity. We can either create a necessary Wi-Fi connection in C# (and therefore directly in Unity) or an external program that will more easily receive the data stream and inject it into Unity. Python seems more adept at networking than C#, and there are libraries to create virtual webcams in python and read and project them in Unity.

2.6.2 Curvilinear to Rectilinear Conversion

Because of the fisheye camera we use, the video will be in curvilinear perspective. The barrel distortion caused by the lens requires rectilinear correction which should be applied in Unity before sending the video to the VR display. In this stage, any other distortions encountered is also fixed to make the video suitable for VR. We will accomplish this by mapping the video feed to an adjustable and curved surface in Unity's virtual environment.

2.6.3 Head pointing Direction and the Field of View

The Oculus integration in Unity has tools to automatically collect rotational and positional data from the VR goggles as well as VR controllers and moves the camera inside the Unity Engine accordingly. This means that on the software side there is little that needs to be implemented in terms of using the VR goggle looking direction and FOV.

2.7 Tolerance Analysis

Ideal framerate for VR is 90 frames per second, fewer than this may lead to nausea in users [3]. This can be broken into two aspects, the framerate of the video feed as well as the framerate of VR headset itself. VR headset or Unity program will need to run above this framerate to respond to the motion of the user and avoid nausea. However, for the video feed, as we have a slow-moving car and the feed is unlikely to induce nausea, we require only 30 fps.

Curvilinear to rectilinear conversion is difficult to quantify and therefore analyze tolerance, as our projection screen will be mostly created by "guess and check" method. However, we can analyze different angles in the field of view in both the real world and the virtual environment. We propose a fairly loose tolerance for this aspect, mostly because of the lack of precision measurements. Furthermore, a slight warping in user perspective might not be all that disorienting, and this might even add to the feel of being inside of a small toy (however, major warping will more than likely create nausea). For these reasons, we propose a max tolerance of ±5 degrees at any given point in the project. We will measure this using physical observation as well as some sort of verification rig to measure angles to the camera in the real world.

3 Costs

We are expected to work at least 10 hours a week for 16 weeks during one semester. Our hourly rate is \$50 per hour. Putting everything into the equation shown below, the total labor cost is \$48,000.

$$3 \times \frac{\$50}{1\ hour} \times \frac{8\ hours}{1\ week} \times 16\ weeks \times 2.5 = \$48{,}000$$

Parts	Quantity	Unit Price	Cost
Amazon RC Car Chassis (DC motors, wheels, frames)	1	$17.99	$17.99
ASUS Tinker board (Single-board Computer)	1	$63.89	$63.89
OmniVision OV2710 180 degrees Fisheye Camera	1	$40.99	$40.99
Microchip ATMEGA328P-PU Microcontroller	1	$2.08	$2.08
Texas Instruments L293D Quadruple Half-H Drivers	1	$5.79	$5.79
Olimex HC-06 Bluetooth Module	1	$2.99	$2.99
ON Semiconductor NCP1117 5V Voltage Regulator	1	$0.45	$0.45
SparkFun 18650 Li-on Battery 3.7V 2.6AH	2	$5.95	$11.90
Amazon 2x 18650 Battery Holder	1	$1.39	$1.39
Oculus Rift VR Headset	1	$599.00	$599.00
Total Cost:	$746.47		

Figure 7. Table of Parts, Quantity, Unit Price, Cost, and Total Cost

Considering the labor cost and the cost of all the parts, we expect total of \$48,746.47 to produce a single product.

4 Schedule

Week	Deniz Yildirim	Erik Jacobson	Sang Baek Han
2/24/2020	Test with fisheye camera and Tinker board.	Test with VR headset and PC display.	Test RC Car with Arduino.
3/02/2020	Test with Wi-Fi transmission of video.	Test with VR headset direction sensor.	Design PCB schematics.
3/09/2020	Program curvilinear to rectilinear conversion and test with VR headset and the fisheye camera video transmission through Wi-Fi.		Design PCB board layout.
3/16/2020			Solder and test PCB.
3/23/2020	Adjust the curvilinear to rectilinear conversion to reduce the nausea and to maintain constant frames per second while moving around manually.		Revise PCB if necessary.
3/30/2020			Finalize PCB.
4/06/2020	Finalize the program for fisheye camera, Tinker board, PC, and VR headset.		Assemble RC car with PCB.
4/13/2020	Assemble fisheye camera and Tinker board on RC car along with PCB. Test Bluetooth controller and the video transmission from fisheye camera working fine as the RC car moves around.		
4/20/2020			
4/27/2020	Prepare for the final demo and adjust any bugs or issue from the previous weeks.		
5/04/2020	Present the final demonstration and work on the final paper.		

Figure 8. Project Schedule and Task Allocation

5 Ethics and Safety

As with any RC drone or RC car, there arises a problem of users abusing the system to access or view locations which are otherwise off-limits to non-credentialled humans. This could lead to a leak in potentially harmful or confidential data, especially as our project deals with video feed. This definitely goes against the IEEE Code of Ethics # 9, which states that "avoid injuring others, their property, reputation, or employment by false or malicious action" [4]. We cannot fully prevent this from happening, but we try to minimize this by limiting the controller range to max of 9m. This might bring less freedom in controlling, but significantly reduce the chance of false ethical usage of our product.

Furthermore, our RC car can remain motionless and appear power off while still recording people without proper consent. Even while in motion, bystanders may be unaware of the functions of our product and therefore unaware of their presence on a camera. We can prevent this by flashing the LEDs on the single-board computer every few seconds.

Any time you work with remote systems and batteries, a battery explosion could not only endanger yourself, but also civilians. Since we will be using two Lithium-ion batteries, we need pay attention to the potential hazard situation. To prevent any damage to lithium cell, we will avoid over charging, over discharging, draining over current, exposing to excessive heat, shorting the circuit, or abusing mechanically [5].

The possibility of losing control of the vehicle or being unaware of your surroundings is very real; while the vehicle will move very slow, the risk of falling off a ledge or a person tripping on it is not trivial. It might bump into a person, but we will minimize the damage to the person by limiting the maximum speed of RC car.

Any VR experience comes with a risk of nausea. VR sickness has symptoms similar to motion sickness and is theorized to be caused by sensory conflict. The duration and severity of the nausea differs from person to person. We will try to eliminate nausea by following VR Best Practices given by Oculus [6].

References

[1] "Virtual Reality Car," Amazon. [Online]. Available: https://www.amazon.com/dp/B076T48JNW/. [Accessed: 13-Feb-2020].

[2] "AR/VR (XR) Discussion," Unity Forum. [Online]. Available: https://forum.unity.com/forums/ar-vr-xr-discussion.80/. [Accessed: 13-Feb-2020].

[3] "The Importance of Framerates," IrisVR. [Online]. Available: https://help.irisvr.com/hc/en-us/articles/215884547-The-Importance-of-Frame-Rates. [Accessed: 24-Feb-2020].

[4] IEEE.org, "IEEE IEEE Code of Ethics," 2020. [Online]. Available: http://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: 24-Feb-2020].

[5] "Safe Practice for Lead Acid and Lithium Batteries," University of Illinois ECE445 Course Staff, 13-Apr-2016. [Online]. Available: https://courses.engr.illinois.edu/ece445/documents/GeneralBatterySafety.pdf. [Accessed: 24-Feb-2020].

[6] "Oculus Best Practices Guide," Oculus Developer Website. [Online]. Available: https://developer.oculus.com/design/book-bp/. [Accessed: 13-Feb-2020].