RC-DSLR

Remote Controlled Digital Solution for Live Recording

Lamont Franklin, Igor Gonchar

James Brewer, Jake Nasonov

Sponsor: Children's Sanctuary Church

June, 2016

Table of Contents

RC-DSLR

Project Motivation

The inspiration for the RC-DSLR device was derived from multiple sources concurrently. Several members on our team had specific interests in photography and videography, some on a more elementary level, others at a more professional level. This allowed our team to have a solid fundamental understanding of the market, and common customer r equirements (as we were all customers to some degree). Due to our common interest, we had nearly unrestricted access to professional grade camera equipment to use with testing our device. The motivation for this project originated from our personal experience, and from the realization that the camera industry lacked a market for professional-grade, affordable, automated, tripod attachments. This type of device is especially valuable in live-recording situations, where a professional camera operator is not available. After realizing that there is a gap in the market, we began to design a device to fill it. From our efforts, we are proud to present the RC-DSLR – *the remote controlled digital solution for live recording*.

Customer

The original idea for the RC-DSLR device came from our sponsor, the *Children's Sanctuary Church*, as they initially approached us. Our sponsor is a small, local church located in Redford, OR. The senior pastor approached us for help with a live-recording solution for his sermon series. The church lacked trained volunteers to operate the camera controls, consequently restricting their ability to record live sermons. The pastor asked us to design a device, which would allow him to operate the camera functions remotely, and individually. His project proposal, paired with our background in the industry, led to a mutual decision in further pursuing this device design.

Current Market

The current camera industry definitely lacks a market in affordable, professional-grade, automated tripod-attachment devices. The market is existent, just lacking. Currently, similar devices range from small-scale gimbals, to incredibly expensive stabilization systems with automated controls... but nothing in the middle. A consumer-level, automated tripod head would cost around $100-$300, and support a camera up to 10lbs. These devices are perfect for time-lapse photography enthusiast and other general applications. These devices do not contain the capability of remotely displaying the camera’s live view, or the capability to control the camera's functionality. These devices simply allow a user to automatically rotate the camera body on a dual axis field of motion. At the other end of the spectrum, we find professional-grade devices costing in the tens-of-thousands of dollars. These cinema heads can support cameras up to 50lbs, and can cost upwards of $30,000. These devices are used on a regular basis by crane & jib operators, visual effects studios, and feature film crews worldwide. They are capable of wirelessly operating camera controls, as well as recording and storing live video data. Overall, these devices are great – just insanely expensive and inaccessible to the average filmmaker. Also, these devices are highly complex, and require professional training to operate. Clearly, neither one of these solutions is suitable for our client.

Problem Statement

Freelancers and small organizations that produce live video recordings require at least a single camera operator to physically control the camera equipment during a recording. The employment of a camera operator for these functions can be costly, therefore discouraging. Consequently, many independent creatives prefer to record themselves individually, and lack the capability to control camera movement and be in the video frame at the same time. There is an interest from various parties in the entertainment business in an affordable device to control a camera remotely, and ideally automatically.

Our Solution

Our solution is a single, affordable device, which will allow a user to control the camera's tilt, pan, zoom, and focus operations from an IOS/Android application. This device will allow for live remote monitoring and controlling of camera operations – essentially eliminating the need for a cameraman.

Customer Requirements

In considering our goals for this project, the requirements set out to us by our client were very minimal. We spent time analyzing, as potential users ourselves, the requirements that would make the results of our project useful to our customer. Several members of our team had personal interest in cameras, photography, and videography in different backgrounds, which made it easier to see things from the customer's point of view.

We needed to analyze across several themes that would filter into our objectives: quality, features, usefulness, practicality, and safety. The following table represents our device specifications, and their classification as an objective, constraint, or a function.

Table 1: This table represents our device's objectives, functions, and constraints

Item Specifications	Objective	Constraint	Function
Pan and tilt camera		**X**	**X**
Adjust camera manual focus		**X**	**X**
Adjust camera manual zoom		**X**	**X**
Attach to an existing tripod		**X**	**X**
Device controlled remotely		**X**	**X**
App to control device			**X**

Live video feed to app			**X**
Operate on rechargeable battery power	**X**		
Operate on DC power			**X**
Onboard video storage	**X**		
Must be completed by June 3rd		**X**	
Cost must be less than $1000		**X**	
Multiple camera control	**X**		
Stable weight on tripod		**X**	
Minimal vibration	**X**		
Smooth camera movement	**X**		
Low noise movement	**X**		
Software focus control			**X**
Motion tracking	**X**		
Facial Recognition Tracking	**X**		
Easy setup	**X**		
Quick lens swapping	**X**		
Compatibility with multiple camera types	**X**		
Minimize Cost	**X**		
Minimize Power Consumption	**X**		
Easy to setup/use	**X**		

Translating Customer Requirements

Objective Analysis

From our requirements, we wanted to frame objectives that were most important to the results of the tripod attachment. To keep the scope simple, we focused on four major themes: Cost, Power, Quality, and Simplicity. These themes can be analyzed in the following objective diagram.

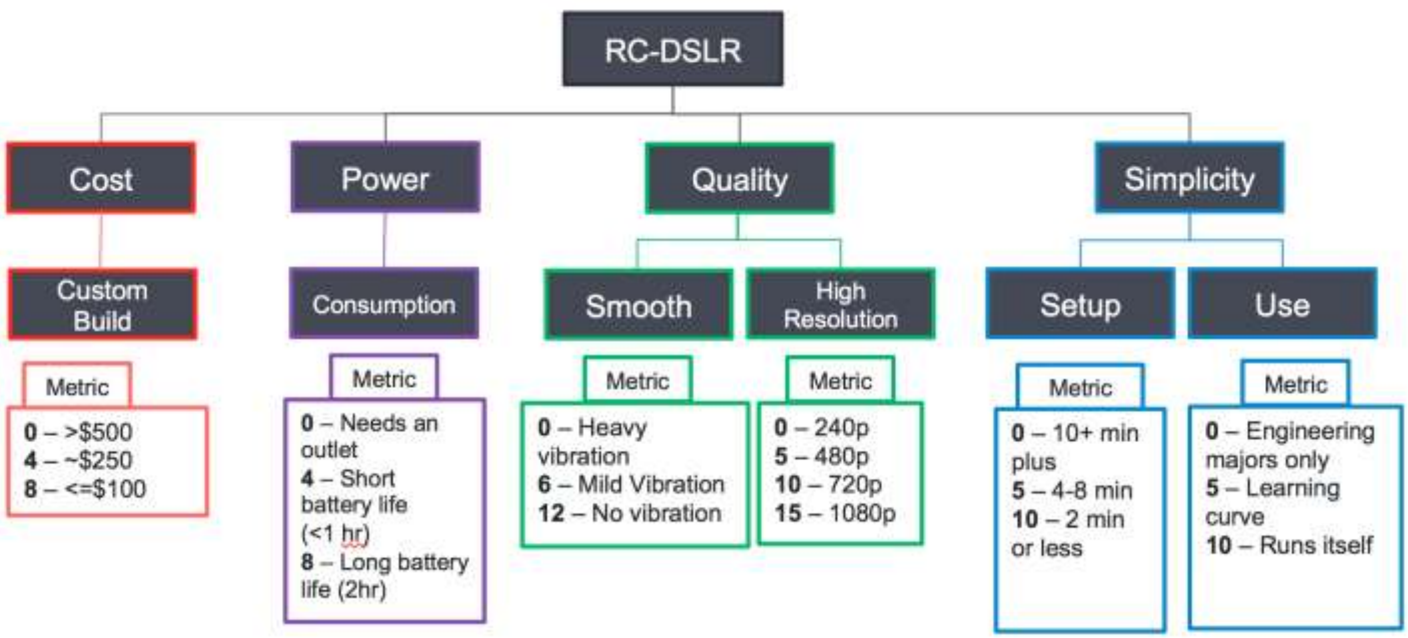


Figure 1: This figure represents our Objective Diagram

Although we had no requirements for a specific cost, we still wanted to take into consideration that we wanted to keep cost minimal in order to stay competitive.

Cost of Build

Overall Cost	Perceived Value	Value
>$500	**Unsatisfactory**	**1**
$300 to $400	**Not Ideal**	**2**
$200 to $300	**Tolerable**	**4**
$100 to $200	**Fair**	**6**
<=$ 100	**Excellent/Ideal**	**8**

Because a tripod is for the most part kept stationary, we weren't as interested in mobility, but from that idea came power consumption based on how much the system could be used with a particular power source. We would be limited by power consumption in a battery, or even if the system requires a power outlet.

Power Consumption

Overall Power	Perceived Value	Value
Needs an Outlet (12+ W)	**Unsatisfactory**	**1**
Needs an Outlet (10 W)	**Tolerable**	**2**
Short Battery Life (5 W)	**Fair**	**5**
Long Battery Life (2.5 W, 500 mA)	**Excellent/Ideal**	**8**

With videography and photography, we wanted to focus mainly on having a professional looking result, which means that high video quality was a major target to achieve. We wanted to focus on the motion of the camera, as well as what kind of resolution we could handle for recording.

Video Smoothness

Overall Smoothness	Perceived Value	Value
Heavy Vibration or Worse	**Unsatisfactory**	**1**
Noticeable Vibration	**Not Ideal**	**3**
Jerky Movement	**Tolerable**	**6**
Twitchy Movement	**Fair**	**9**
Smooth Motion	**Excellent/Ideal**	**12**

Video Resolution

Overall Resolution	Perceived Value	Value
240p or less	**Unsatisfactory**	**1**
360p	**Not Ideal**	**3**
480p	**Tolerable**	**5**
720p HD	**Fair**	**10**
1080p HD	**Excellent/Ideal**	**15**

Simplicity is a moderate factor in what we wanted to target in our design. Based on some knowledge of camera and accessory setup time with other devices, we knew that setup time is a big factor in how easy a device or accessory is easy to use. While setup time is easy to quantify, usage itself is somewhat subjective.

Setup Time

Average Time	Perceived Value	Value
Greater than 15 Minutes	**Unsatisfactory**	**1**
10-15 Minutes	**Not Ideal**	**3**
6-10 Minutes	**Tolerable**	**5**
2-6 Minutes	**Fair**	**7**
2 Minutes or Less	**Excellent/Ideal**	**10**

Usage

Overall Simplicity	Perceived Value	Value
Rocket Science	**Unsatisfactory**	**1**
Engineering Majors Only	**Not Ideal**	**3**
Camera Professionals Needed	**Tolerable**	**5**
Manual Detail Adjustments	**Fair**	**7**
Practically Runs Itself	**Excellent/Ideal**	**10**

Constraint analysis

Pan and Tilt camera

- Pan must allow for a minimum of 90 degrees of motion.
- Tilt must allow for a minimum of 45 degrees of motion.
  - Having minimums for the range of motion for the camera is essential to the project. Ideally we can get far past the minimums in an initial design, but any less than those minimums and our project becomes impractical.

Adjust Camera Manual focus and zoom

- Manual focus and zoom functionalities must utilize full range of camera capabilities.

Attach to an existing tripod

- Device must attach to standard tripod legs.
  - This is a constraint based on compatibility. The purpose of the project is to use as much of the customer's existing equipment as possible. Forcing them to purchase a completely new, specific tripod would be cost prohibitive.

Device controlled remotely

- Device must be controllable via a wireless connection of an accepted standard.

Stable weight on tripod

- Device must not tip or fall during regular operation.
  - This is an essential constraint for both the safety of the camera and to anyone near the tripod as well. Camera equipment can be heavy in some cases, and very fragile.

Functional Analysis

For each of our functions in the device, we wanted to set goals to establish what the customer and we would consider satisfactory. In terms of panning and tilting, we extended some of the specifications past their constraints to meet what would be considered satisfactory. Ideally, some specifications would be able to exceed that in terms of range of motion or control features per some of the remote control functions.

Table 2: This table represents the performance specifications of our design

Function	Performance Specifications
Pan and Tilt Camera	Should be able to pan a minimum of 170 degrees and tilt a minimum of 90 degrees (or deemed safe for the camera)
Adjust Camera’s Manual Focus	Should smoothly turn the lens focus ring within it’s acceptable range (Not turning past it’s limits)
Adjust Camera’s Manual Zoom	Much like the focus, should smoothly turn the lens zoom within it’s acceptable range
Attach to a Preexisting Tripod	A mount including the servos and the microcontroller should fit safely on an acceptable tripod
IOS/Android App to Control Device	Should be able to control pan, tilt, zoom, and focus
Live Video Stream to the App	Should be able to see a live feed from the camera, sent wirelessly to the app

We also considered many extra functions for our project beyond the initial scope that would add a much deeper and advanced layer of usefulness to our project that customers wouldn't see in other devices. Some functions like tracking are more experimental based on using pre-existing cameras not considered for functions like that.

Table 3: This table represents the extended performance specifications of our design

Extended Functions	Performance Specifications
Onboard video storage	If camera feed quality is higher definition, it should store a copy of the video feed on our device
Rechargeable battery	The battery should be sufficient to power the device for at least half an hour, and be easily recharged
Automatic focusing	Software should control the focus automatically using the video feed and focusing algorithm
Motion tracking	Should track an object and move the camera to follow it
Face recognition tracking	Should track a face or person automatically as they move

To keep our methods of achieving these functions, at first we pictured the means as simple categories. This would help us analyze each need that needs to be filled with other various options in detail, as each would have several options that would clutter up the graph far too much.

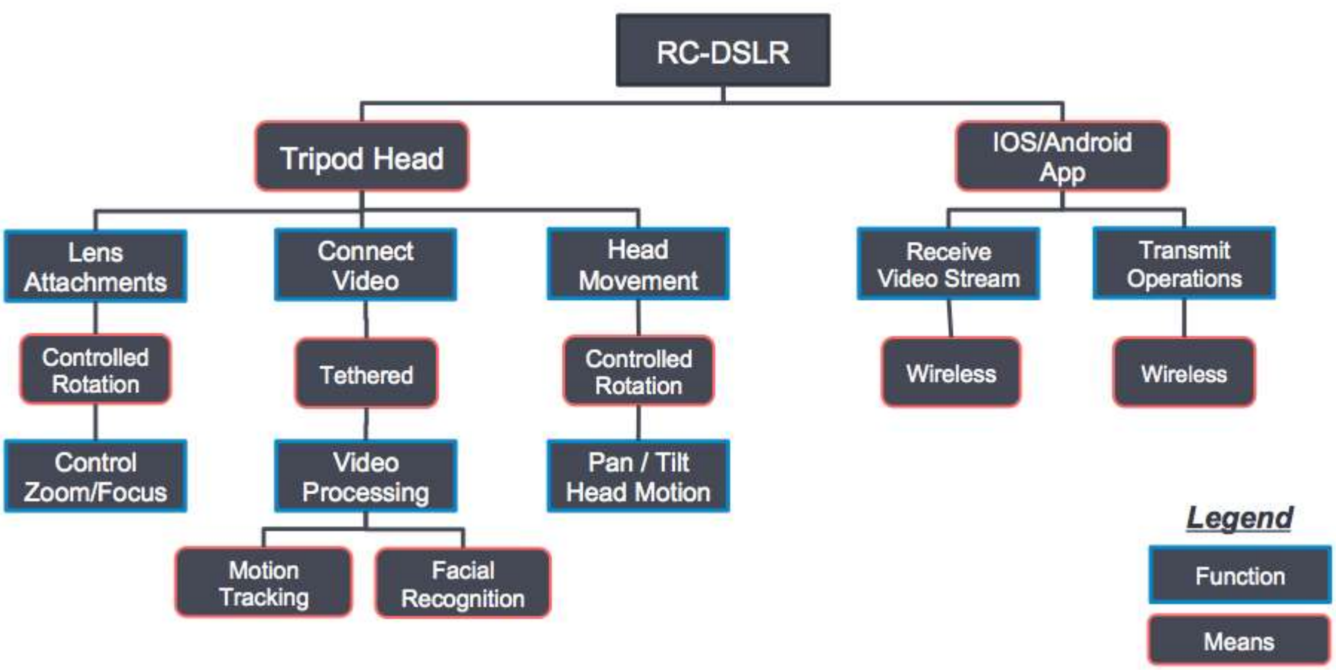


Figure 2: This figure represents our function-means tree

Design Space

Method of Choice

When it comes to our design space, there is a large potential for it to become very broad. To not become overwhelmed with possibilities, we kept ideas limited to what we knew would be simple to work with and fit the needs of our target clientele (churches and hobbyists). If left unchecked, the possibilities would be exponentially greater.

One of the advantages with our project is that it's highly modular. Most means can be interchanged without affecting other parts of the system. The biggest dependent factor on the means we choose is what kind of software we will have to develop or implement.

Table 4: This table represents our morph chart

Function	Means		
Pan and Tilt Camera	Servo Motors	Stepper Motors	
Manual focus/zoom	Servo Motors	Stepper Motors	
Attach to an existing tripod	Separate connection	Connection built into device	
Device controlled remotely	WIFI	Bluetooth	RF
App to control device	Web app	Mobile App	
Live video feed to app	WIFI	Bluetooth	
Software Focus Control	OpenCV	Custom Video processing	Alternative open source application
Motion Tracking	OpenCV	Custom Video processing	Alternative open source application
Facial Recognition Tracking	OpenCV	Custom Video processing	Alternative open source application

Candidate Designs

In looking at our candidate designs, as mentioned before our project is highly modular. Having multiple candidates would become somewhat redundant, or even trivial. Instead, we wanted to evaluate each means separately and pick the best out of those.

Table 5: This table represents the preferred candidate design

Function	Means
Controlled Rotation	Servo motors
Tripod Attachment	Built-in
Video Processing	UV4L
Device Controlled Remotely	WIFI
Micro-Controller	Raspberry Pi 3
Video Feed	WIFI
App to control device	Android App

Evaluation of Design

Alternative solutions

Although much of our design is modular, one of the biggest major alternative solutions would be building the device from scratch.

Figure 3: This is an image of our device construction with the Android app.

This is the base model of the tripod head. The cost of this device will be greater than rebuilding already existing tripod head, due to the fact that we would have to buy the materials and spend time to cut out to the sizes. It would be easier to build a tripod head from scratch, rather than modifying a preexisting head. However, as mentioned before, it is much cheaper to repurpose a previous design.

Function and Constraint Evaluation

Pan and Tilt camera

- Using servos gives us an acceptable range of motion and positional control. They will meet our minimum ranges of motion and can be modified to exceed that if needed.

Adjust Camera Manual focus and zoom

- Servos are also acceptable for a good range of motion for the lens as well. They can also easily be limited programmatically to not go past a certain point.

Attach to an existing tripod

- By including a universal type thread, we can attach our device to any tripod.

Device controlled remotely

- This constraint/function is easily met through using Wi-F, which is a highly accepted standard.

Stable weight on tripod

- The RC-DSLR is quite heavy at 3.7 lbs. However, it has the solid build that is required for smooth camera movements. The design of the RC-DSLR must support the latest range of small, professional digital cameras. We will be modifying the Manfrotto 501HDV fluid head to contain motorized rotation capabilities.

Live Video Feed to App

- The platforms and microcontroller we've looked at should support the communication to send a video feed.

Onboard video storage

- The Raspberry Pi has ample places to store video (SD and USB). However, this is highly dependent on the quality of the video feed itself whether it should be stored or not.

Rechargeable battery

- Using a rechargeable battery may be possible, but it would add much to the cost as our power consumption increases.

Automatic focusing

- Using servos and a focusing algorithm (likely through OpenCV), we can implement this function easily through our design.

Motion tracking

- OpenCV supports object and feature detection, which we can use to track objects in a feedback loop.

Facial recognition tracking

- OpenCV also supports various face detection and recognition algorithms, but implementing it on a larger scale efficiently will be a challenge. If time allows, we believe it is possible.

Metric Evaluation

In evaluating the types of motors we would include, we wanted to focus on smoothness, power consumption and cost. Both categories have their own ranges and without having specific models, some things are untested for the time being. From observation and general research however, we have analyzed how they fit into some of our metrics. Generally, servo motors have been better with cost and power consumption, and with certain models, smoothness of motion.

Controlled Rotation	Servo Motors	Stepper Motors
Video Smoothness	Untested	Untested
Power Consumption	Constant Medium	Intermittent High
Development Time	Low	Medium

In video processing, we realized there are many alternatives and we tried to limit our search to open source software. Our initial idea was to implement OpenCV; however, due the large overhead associated with OpenCV we decided that it would be too much for the Raspberry Pi to handle. This is why we decided to go with UV4L, which contains a lower footprint and also comes with similar great features like low latency video streaming, facial recognition, motion tracking, etc.

Video Processing	OpenCV	Custom Algorithm	Alternate Application (UV4L)
Development Time	Low	High	Medium

For housing the mechanism to control the lens manually, we considered two options and how they would factor into ease of use. Both would be simple to implement, but having the attachment separate from the tripod head would be much simpler to implement and attach.

Lens Attachment	Separate Attachment	Built In
Development Time	Low	Low
Setup time	1 min	30 second

In consideration of how we would control the device remotely, the big concern was power consumption, as well as how cost would factor into the project. Related to our choices of microcontroller, the cost to Wi-Fi or Bluetooth would be included, where RF would not. We decided to implement WIFI to control the device because it was much simpler to set up a communication socket from Android to our Raspberry Pi.

Device Controlled Remotely	Wi-Fi	Bluetooth	RF
Development Time	Low	Low	High
Power Consumption	High	Low	Unknown
Added Cost	None	None	Added antenna

Considering the factors discussed above, for a video feed we will need to go with Wi-Fi, as bandwidth is an issue when it comes to video. Wi-Fi can sustain 40 Mbps on most standards, but Bluetooth only 800 kbps. In this case, we would have to limit the video feed if we wanted to use Bluetooth, which is a case we would not prefer.

Video Feed	WIFI	Bluetooth
Power Consumption	High	Low
Video Quality/ Bandwidth	High	Low

The biggest factors in choosing a microcontroller are the features, cost, and power consumption. Ideally we wanted something that would be efficient enough to handle video processing but low power enough to be sustained. The Intel Edison, while more power efficient is also costlier and doesn't have the necessary processing power to support the SDK. However, it may be useful in testing as we have some in our individual possessions.

Micro- Controller	Raspberry Pi 3	Edison
Wireless Connectivity	WIFI Bluetooth	WIFI Bluetooth
Power Consumption	Medium	Low
Development Time	Low	Medium
Clock Speed requirement	Meets	Does not meet
Cost	$35	$85

Regarding the app to control the device, we decided to implement an Android Application. The reason behind is partially because we had access to a mobile android device, and because of Apple’s developers license fee.

App to control device	Web app	Mobile App
Development Time	High	High

In looking at the sum of all parts on the following table, we can more easily evaluate how our design fits within our objectives and their metrics.

Table 6: This table represents our preferred candidate metrics

Objective	Value
Cost	4 ($200 to $300)
Power Consumption	3-4 (~7 W/ 10 W max)
Video Smoothness	9 (Possible twitchiness)
Video Resolution	15 (1080p)
Setup Time	7 (~3-5 minutes)
Ease of Use	7

Testing

Set Up Testing

- Measure how much time is consumed to set up the device.

Real-World application testing

- Testing the device in a real-world environment. Gain permission to record a sermon at a local church, and test the device in its intended application.

Process recorded video feed

- Facial detection
- Motion tracking
- Does the device accept the video feed?
- What is being sent to I/O process?
- What data we are looking at the end.

Servo Motors Testing

- Degree of motion of the Pan and Tilt
- Degree of motion of the Focus and Zoom
- Test speed and range of motion of the servo motors.
- What are the critical weights to put on and at what angle.

Stability Testing

- Making sure the device and tripod is stable with the weights that are given.

Outside use of untrained user

- Have someone that is unfamiliar with our product set it up and use it. Test how simple our device is to use.

Analyze video feed

- Test latency between received video signal, and processed video.

Consistency of video quality

- Test the video resolution for lost-pixels/distortion.

Evaluation of Success

In terms of quality, observations tend to be objective. However, there is still a noticeable difference between something successful and something not. Most of the tests we intend to perform have to do with monitoring quality of the video, as well as how easy the device is to use. But we also want to take into consideration safety and the range of operation our device will have.

For success, our benchmark is if our device performs stable pan and tilt along with manual focus and zooms in an easy to use manner. In implementing automatic control into our device including tracking and autofocus, they need to be accurate and stable in order to be considered successful additions.

Social and Ethical Considerations

The social ethical issues that may arise from the development of our project include safety, environmental considerations, and job security concerns. The development of our device may raise questions as to whether camera operators risk losing their jobs, due to an automated system replacing them. This is a legitimate social economic issue that questions the overall benefit of our device. Is it worth making a device, which will cause people to lose their jobs? Do the pros outweigh the cons? All these questions must be considered before designing such a device. Another concern is environmental health. What materials will be using in the construction process of our device? Will it emit and harmful radiation, or gases during its operation? Again, these are serious questions that our team addressed when designing our candidate design. And finally, safety is a social consideration. Is our device safe to operate and maintain? Does it place the user, or surrounding people in any danger? Fortunately, our device is safe to use, but we still had to ask ourselves these questions when designing our device.

Project Management

Work Breakdown Structure (WBS)

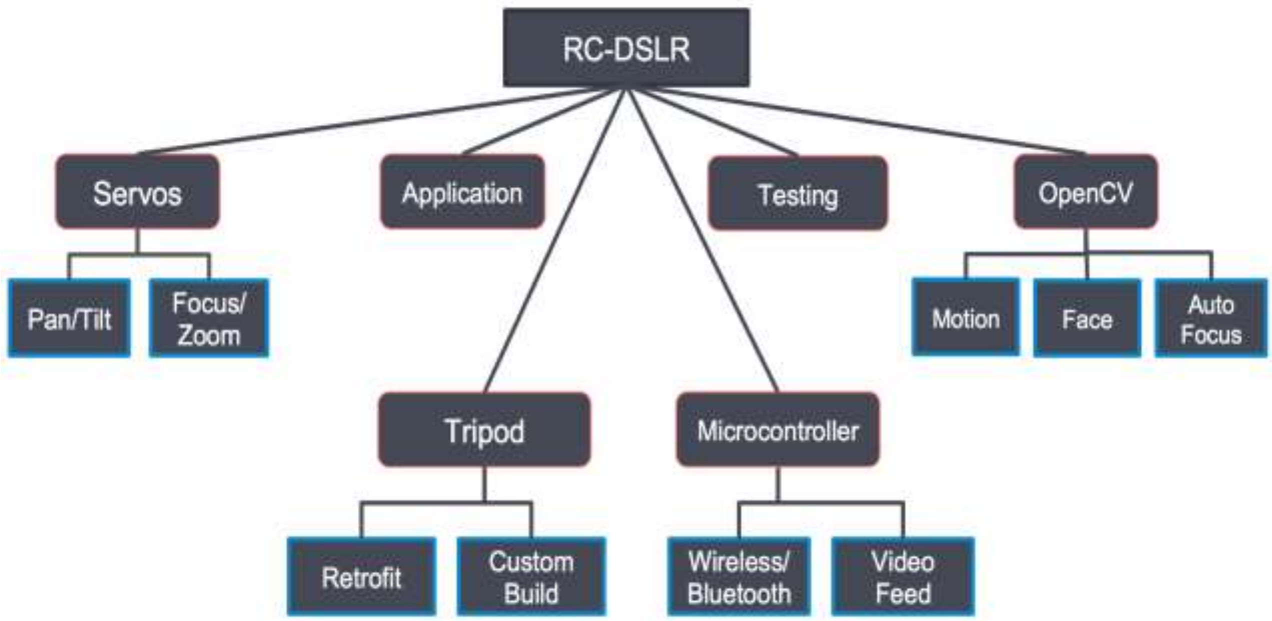


Table 7: This table represents a timeline of our schedule for this project

Date	Milestone
3.23.16	Purchase Raspberry Pi 3
3.25.16	Purchase servos/ gears
3.31.16	Develop servo framework
3.31.16	Access live video feed
4.05.16	Establish remote communication
4.09.16	Basic App/PC program
4.16.16	Prototype head built
5.05.16	Construct final tripod head
5.07.16	Test pan/tilt

5.10.16	Test lens motion (focus/zoom)
5.10.16	Test video feed
5.18.16	Full app/remote program
5.21.16	Compile final device
5.23.16	Test in real world application
5.25.16	Implement basic motion/face/autofocus
5.27.16	Further testing
6.07.16	Poster/project completed, presentation

Budget

For our budget, we've taken various prices and estimates of the assorted parts we'll need to complete the project. We've also considered a margin of extra cost that we may encounter, whether it's getting better parts than what we have or unforeseen costs in taxes and shipping. Overall, our target max budget is $600, but we believe we will keep below that.

Item	Basic Cost Estimate	Upper Bound (including shipping + tax)
Pan/ tilt servos (x2)	2 x $17 = $34	2 x $50 = $100
Zoom servos	$17	$30
Servo Accessories	$30	$50
Raspberry Pi 3	$35	$35
Custom Head	$50	$75
USB video cable	$10 (CVBS)	$200 (HDMI)
Wires	$5	$7
Power supply	$10	$15
Emergency Purchases		$88
Total:	$191	$600

Final Results

Test Results

- Motion test =
  - Pan - OK, Even though it moved pretty good, there is a place to improve, one idea was to add another plate at the bottom and add a bearing type balls between them, that way we could increase the surface area on which the device is standing, improving the stability of the device.
  - Tilt - Active, but issues with weight, since the weight of the device is lighter than the camera itself, camera was overweighting towards the side to which the camera was being tilted. The other problem was that there was space between the rod that connects the gears to the center tilt platform, allowing it to move freely
  - Zoom - OK, this apparatus was build off metal and we had no major issues other than connecting the gear to the aftermarket motor. Other than that we were able to connect it so we had no sharp movements.
  - Focus - OK, Focus worked the best so far and it is because we used already integrated motors within the lens itself.
- Video feed quality
  - Active, but with cropping and visible latency
- Recording quality
  - Pending
- Live Application Test
  - Not performed

Team Dynamics

The team got along very well. The atmosphere was very positive and there were few, if any conflicts. We did have some gaps in communication that we tried to acknowledge later on. Overall we tried to keep our communication on a high level among each other.

Learning Experience

For those who worked on the software side, we gained a lot of experience working in other programming languages, dealing with drivers, and various software in Linux for video. We also gained experience working with I2C and Picture Transfer Protocol along with Bulk USB Transfer. For that, we went well beyond simply implementing software packages, and reverse engineering existing software to implement it on an entirely different device that would have been incompatible with the camera SDK. We may be able to contribute our information to existing open source projects attempting to implement camera control on Linux devices. We also developed an Android app for the first time, and learned more about Bluetooth and its communication protocols.

We were mainly helped by TCES 460. Our project was primarily an embedded systems project, which is why it was so relevant. Having the experience on an embedded system microcontroller using Linux gave us a big advantage in our project, and we didn't realize it would be relevant because we chose our project before we took the class.

For app development, none of us had taken the mobile apps class as an elective. It likely would have helped, but in the end it wasn't necessary.

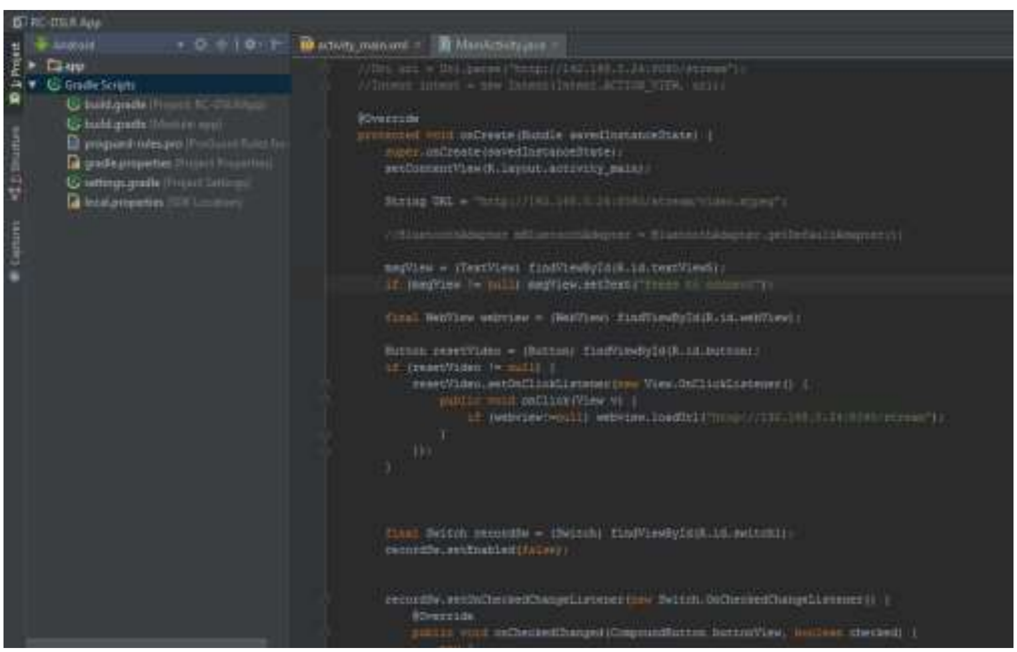


Figure 4 – The Android Studio working environment.

For those who worked on the physical build, we had to learn how to use the AutoCAD Inventor Tool, in order to properly build the apparatus along with software to design 3D printed objects and to laser cut various pieces for the casing and supports.

TCES481 came in handy on how to organize and prepare for the project before we actually started building it. Also, since some of us have a background experience in using power tools we didn’t really had to learn how to use them, but to just apply our experience.

Schedule Results/Team Dynamics

For the last month of the project we fell behind schedule. We attempted at the beginning to create a very strict schedule, but even with a relaxed schedule there were issues that we ran into for the project, along with a legitimate emergency that set one of our group members back a couple weeks during a crucial time in our schedule.

As far as team dynamics, the team got along very well. The atmosphere was very positive and there were few, if any conflicts, despite a lot of stress and tension to complete the project due to setbacks. We did have some gaps in communication that we tried to acknowledge later on. Overall we tried to keep our communication on an open and honest level among each other.

Evaluation of Success

We met our initial project goals. Essentially our final product consisted of panning, tilting, zoom and focus functions along with a full app that could control each of the above, monitor video, and control the camera. Some extended goals we were not able to meet, such as motion tracking or face recognition. But we assumed these would only be possible if we had enough time to implement them. If we choose to continue working on the project after graduation, there’s a wide range of improvements and feature additions we can make.

Schematics and Screenshots

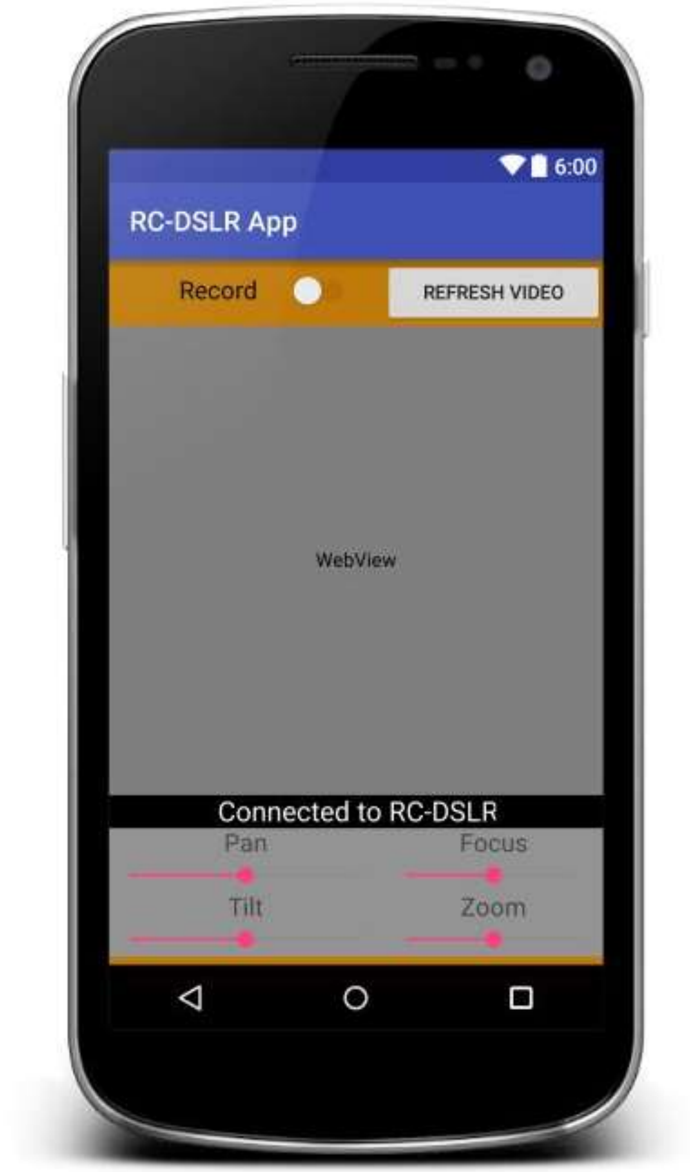


Figure 5 – A concept screenshot of the app UI from Android Studio. The center would be replaced by the video feed in the running version.

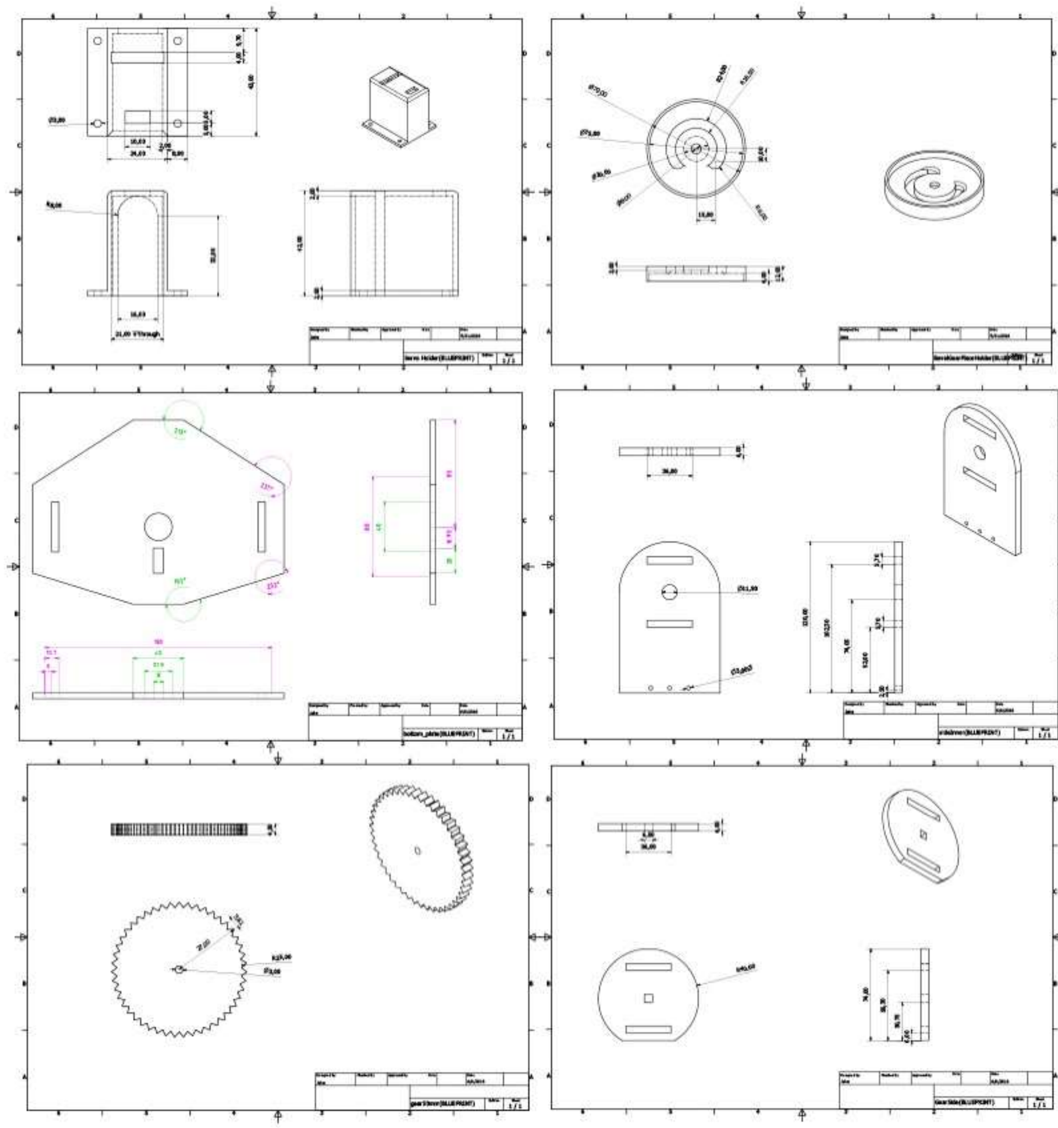
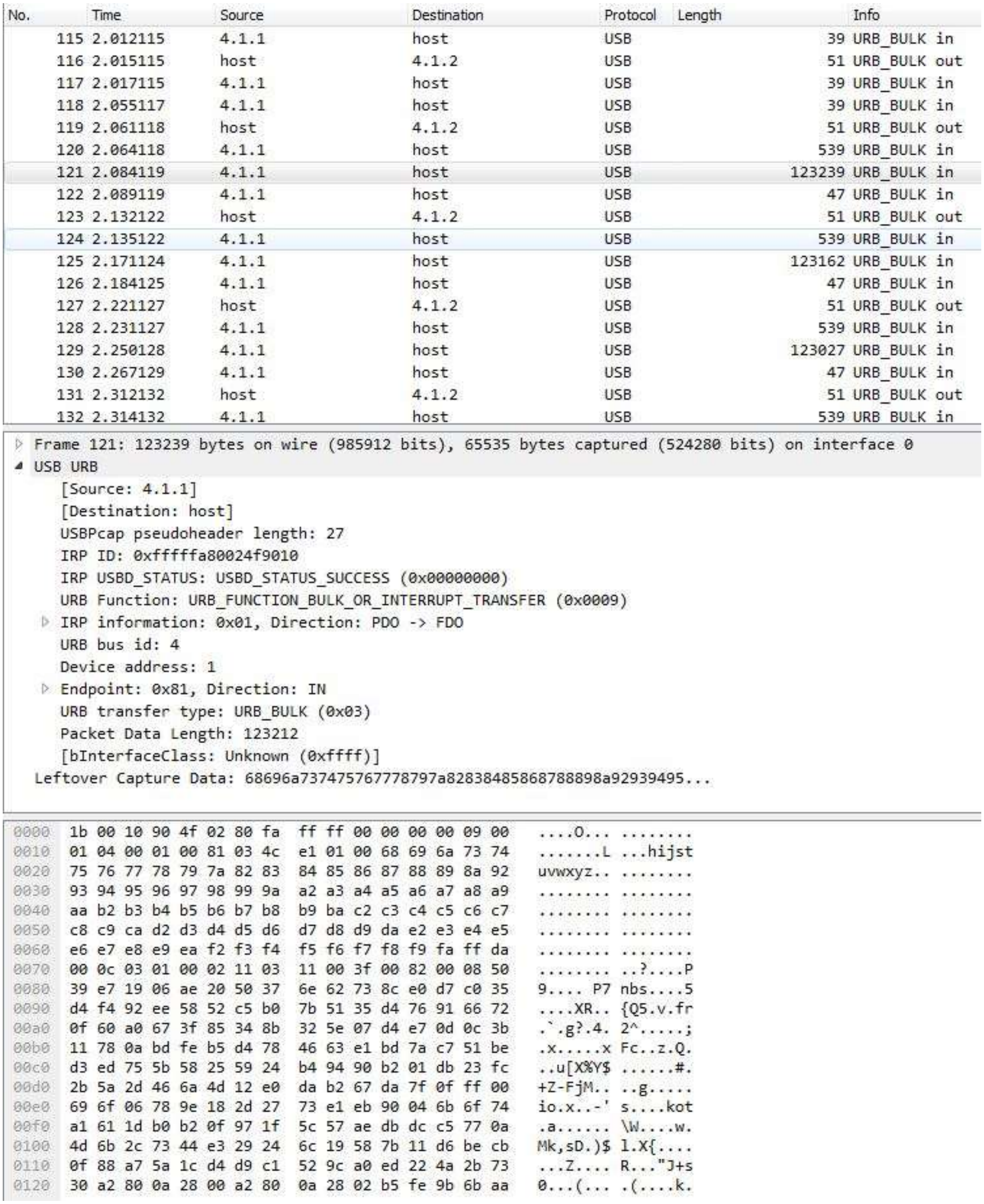


Figure 6 – Blueprint schematics of the acrylic and 3d-printed pieces used in the physical build of the project.

Figure 7

Figure 8 – The final build of the project.



No.	Time	Source	Destination	Protocol	Length	Info
115	2.012115	4.1.1	host	USB	39	URB_BULK in
116	2.015115	host	4.1.2	USB	51	URB_BULK out
117	2.017115	4.1.1	host	USB	39	URB_BULK in
118	2.055117	4.1.1	host	USB	39	URB_BULK in
119	2.061118	host	4.1.2	USB	51	URB_BULK out
120	2.064118	4.1.1	host	USB	539	URB_BULK in
121	2.084119	4.1.1	host	USB	123239	URB_BULK in
122	2.089119	4.1.1	host	USB	47	URB_BULK in
123	2.132122	host	4.1.2	USB	51	URB_BULK out
124	2.135122	4.1.1	host	USB	539	URB_BULK in
125	2.171124	4.1.1	host	USB	123162	URB_BULK in
126	2.184125	4.1.1	host	USB	47	URB_BULK in
127	2.221127	host	4.1.2	USB	51	URB_BULK out
128	2.231127	4.1.1	host	USB	539	URB_BULK in
129	2.250128	4.1.1	host	USB	123027	URB_BULK in
130	2.267129	4.1.1	host	USB	47	URB_BULK in
131	2.312132	host	4.1.2	USB	51	URB_BULK out
132	2.314132	4.1.1	host	USB	539	URB_BULK in

```
Frame 121: 123239 bytes on wire (985912 bits), 65535 bytes captured (524280 bits) on interface 0
USB URB
    [Source: 4.1.1]
    [Destination: host]
    USBPcap pseudoheader length: 27
    IRP ID: 0xfffffa80024f9010
    IRP USBD_STATUS: USBD_STATUS_SUCCESS (0x00000000)
    URB Function: URB_FUNCTION_BULK_OR_INTERRUPT_TRANSFER (0x0009)
    IRP information: 0x01, Direction: PDO -> FDO
    URB bus id: 4
    Device address: 1
    Endpoint: 0x81, Direction: IN
    URB transfer type: URB_BULK (0x03)
    Packet Data Length: 123212
    [bInterfaceClass: Unknown (0xffff)]
Leftover Capture Data: 68696a737475767778797a82838485868788898a92939495...
```


```
0000  1b 00 10 90 4f 02 80 fa  ff ff 00 00 00 00 09 00   ....O... ........
0010  01 04 00 01 00 81 03 4c  e1 01 00 68 69 6a 73 74   .......L ...hijst
0020  75 76 77 78 79 7a 82 83  84 85 86 87 88 89 8a 92   uvwxyz.. ........
0030  93 94 95 96 97 98 99 9a  a2 a3 a4 a5 a6 a7 a8 a9   ........ ........
0040  aa b2 b3 b4 b5 b6 b7 b8  b9 ba c2 c3 c4 c5 c6 c7   ........ ........
0050  c8 c9 ca d2 d3 d4 d5 d6  d7 d8 d9 da e2 e3 e4 e5   ........ ........
0060  e6 e7 e8 e9 ea f2 f3 f4  f5 f6 f7 f8 f9 fa ff da   ........ ........
0070  00 0c 03 01 00 02 11 03  11 00 3f 00 82 00 08 50   ........ ..?....P
0080  39 e7 19 06 ae 20 50 37  6e 62 73 8c e0 d7 c0 35   9.... P7 nbs....5
0090  d4 f4 92 ee 58 52 c5 b0  7b 51 35 d4 76 91 66 72   ....XR.. {Q5.v.fr
00a0  0f 60 a0 67 3f 85 34 8b  32 5e 07 d4 e7 0d 0c 3b   .`.g?.4. 2^.....;
00b0  11 78 0a bd fe b5 d4 78  46 63 e1 bd 7a c7 51 be   .x.....x Fc..z.Q.
00c0  d3 ed 75 5b 58 25 59 24  b4 94 90 b2 01 db 23 fc   ..u[X%Y$ ......#.
00d0  2b 5a 2d 46 6a 4d 12 e0  da b2 67 da 7f 0f ff 00   +Z-FjM.. ..g.....
00e0  69 6f 06 78 9e 18 2d 27  73 e1 eb 90 04 6b 6f 74   io.x..-' s....kot
00f0  a1 61 1d b0 b2 0f 97 1f  5c 57 ae db dc c5 77 0a   .a...... \W....w.
0100  4d 6b 2c 73 44 e3 29 24  6c 19 58 7b 11 d6 be cb   Mk,sD.)$ l.X{....
0110  0f 88 a7 5a 1c d4 d9 c1  52 9c a0 ed 22 4a 2b 73   ...Z.... R..."J+s
0120  30 a2 80 0a 28 00 a2 80  0a 28 02 b5 fe 9b 6b aa   0...(... .(....k.
```


Figure 9 – The Wireshark environment after running the Canon SDK and capturing the USB data.