CAMERA POSITIONING SYSTEM

By

Junjiao Tian

Jialu Li

Weicheng Jiang

Final Report for ECE 445, Senior Design, Spring 2017

TA: John Capozzo

May 3, 2017

Project No. 79

Abstract

The camera positioning system is a lab equipement for particle tracking research. Four cameras can be independently controlled to translate in a vertical plane as well as pan and tilt with a high resolution of movement. The four camera system will be controlled by a user on a PC with a customized user interface and will produce four different views, converging on the volume of interest. The complete system is able to translate with 1 mm resolution and pan and tilt with +/-0.5 degree of tolerance. Features such as auto-calibration and close-loop feedback control of servo motors of pan and tilt head are also developed and can be accessed on the user interface.

Contents

1. Introduction

The University of Illinois's Renewable Energy & Turbulent Environment (RETE) research group with sponsorship from John Deere has been tasked with designing a camera positioning system for particle tracking research. The system must be able to operate with the same functionalities as the current particle tracking equipment but with additional scalability, precision, robustness, and ease-of-use in mind. The current equipment includes the use of a high resolution, high speed recording camera and a view splitter which produces four different views, converging them on the volume of interest. It is relatively small and thus limited in experimental range. Manual calibration of the four independent view splitting mirrors is very tedious and requires a minimum of two people to set up. It is also important to note that the total experiment set up time from scratch requires up to four hours.

A proposed solution was to create a four camera system, replacing the view splitter, that can be electronically controlled. The cameras will be able to translate, pan, and tilt with a high resolution of movement using stepper and servo motors. The frame will allow simple adjustability in the system's dimensions to provide a wide experimental range and provide a robust foundation in which future scaling can be possible. A user interface will be provided showing the current position of each of the cameras as well as allowing different modes in which they can be moved. All calibrations will be hands-free from the physical system.

This project has spanned two semesters and will continue to be tested in a lab environment for one year before is transferred to John Deere. The previous mechanical team has finished the physical setup and used an Arduino Mega to control the whole system. This semester, we designed a processing unit on PCB to replace the Arduino Mega, created a user interface in LabVIEW and included more advanced functionalities such as auto-calibration and closed-loop feedback. We also investigated the feasibility of building a power unit.

In section 2, the design of different modules is discussed in detail. In section 3, the requirements and verification of the designs is summarized. In section 4 and 5, overall cost of the project and conclusions are discussed.

1.1 Block diagram

This the high level block diagram of the system. The LAVIEW/PC communicates with the microcontroller through UART. The microcontroller talks to the PCA9685 PWM driver through I2C and controls the stepper motor drivers with digital signals. The power module is responsible for providing power to all electronics in the system.

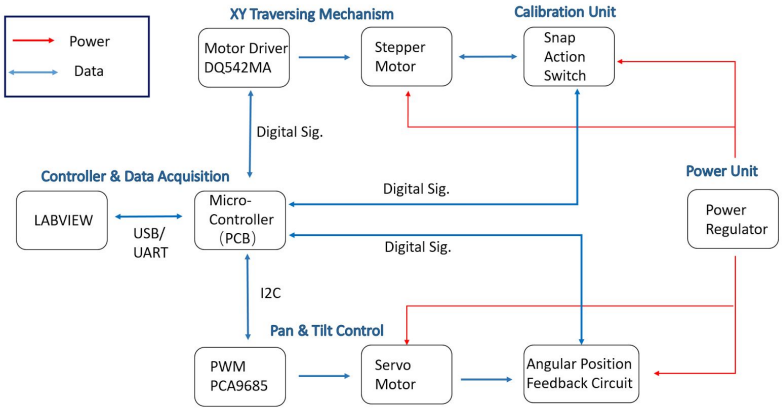

Figure 1, Block Diagram

2 Design

2.1 XY traversing & Calibration

The traversing system allows the camera to move and hold in position in a vertical plane. For the old system built by the MechSE team in last semester, users can only move the traversing unit with the arrow keys on the keyboard, and they had to measure the actual position of the unit relative to the boundaries of the tracks to know where it is. In ECE 445 project this semester, we added calibration to the system, and users are now able to move the traversing unit by specifying the target coordinate in mm or by specifying the direction and distance of the movement. For added safety, the system also allows the users to specify the effective travel on the track after calibration to avoid the traversing unit colliding with the boundaries of the tracks.

2.1.1 Stepper Motor Control & Lead Screw Mechanism

There are 8 high torque NEMA 23 stepper motors for x and y movements. Each motor provides 175 oz./in torque and has a native step size of 200 steps/rev. The load is estimated to be around 24.5 N so the motor torque needed to raise it is about 10.62 oz-in and the torque to lower it is around 2.4 oz.-in. A safety factor of 16.5 can be achieved with this setup. Eight industrial grade motor drivers are used to drive the stepper motors. The DQ542MA is a two-phase hybrid stepper motor driver and is widely used in numerical control devices such as CNC machines. It also provides micro-stepping options to further increase positioning accuracy.

The stepper motor driver DQ542MA provides different stepping options. Currently the 400 steps/rev resolution is used. The lead screw has a 8 mm lead. The equation to relate the number of pulses to move a certain distance is derived.

$$\#\ \text{of pulses} = \text{distance} * 50\ \text{steps/mm} \qquad \text{Eqn.1}$$

2.1.2 Calibration

Calibration is the basis for all other added functions we achieved this semester. When calibration

starts, the traversing unit will first move from wherever it is from last time to the boundaries of the tracks where LabVIEW sets the coordinate display to zero, and then will move to the center of the frame and wait for user commands. After calibration, the LabVIEW front panel displays the actual position of the traversing unit in the plane according to the relation between the stepper motor rotation and lead screw mechanism movement, and updates the coordinate accordingly when the unit moves.

When the calibration button in LabVIEW user interface is pressed, LabVIEW commands PCA9685[1] board to generate short pieces of pulse trains to move the traversing unit to the snap action switches[2] on the boundaries step by step from a possibly unknown initial position. It is important that the unit moves step by step to the boundary from the initial position instead of moving continuously, because functions in LabVIEW only allows for outputting pulse trains with specified number of pulses which moves the unit for a known distance. If we output a pulse train that is too short, the unit will simply not reach the boundary during calibration. On the other hand, if the pulse train is too long, the unit will not stop until it moves a corresponding distance even if it hits the snap action switch on the boundary, which will definitely damage the mechanical structure. However, with the step-by-step movement, once the traversing unit hits the snap action switch on the boundary, the ENABLE pin on the stepper motor driver is immediately pulled HIGH for 2 seconds by LabVIEW so that the motor driver will ignore all the upcoming pulse trains. As a result, the traversing unit will stop at the boundary after finishing the current tiny step of movement without damaging the switch.

It is important that ENABLE is pulled HIGH for 2 seconds because we found through experiment that the ENABLE pin has to be pulled HIGH for at least 1 second to stop the stepper motor from moving. When the traversing unit hits the snap action switch, LabVIEW also sets the coordinate display to zero so that one boundary of the track becomes the origin of the frame. After the ENABLE pin is pulled HIGH for 2 seconds, it is then pulled low by LabVIEW again so that LabVIEW regains control of the stepper motor, which then moves the traversing unit to the center of the frame to wait for user commands.

2.1.3 User Specified Position and Movement & Software Boundary

The user interface allows users to specify a certain distance to move in four directions. Depending on the user's choice, LabVIEW will toggle the corresponding direction output to the motor driver and send pulses to drive corresponding stepper motor.

There is a software boundary limit in LabVIEW. It prevents the traversing system from damaging itself and also avoids finger pinch hazard.

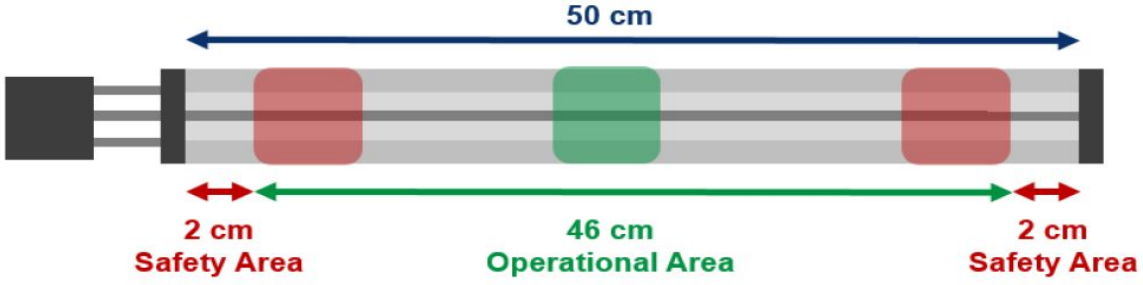

Figure 2, Effective Travel Length

2.2 Pan & Tilt Head and Closed Loop Feedback

2.2.1 Pan & Tilt head and servo motors

The commercially available pan & tilt heads come with two HS-785HB winch servos [3] [4] with approximately 2700 of rotation. There is also a 7:1 gear box between the rotating parts of the pan&tilt heads and the servos. The angle of rotation of the pan&tilt heads is mapped into PWM pulse width (micro-sec) through a linear function. The mathematical function relating the angle to the pulse width is developed through repeated experiments.

2.2.2 Angle VS. Pulse Width Experiment

To relate the angle of rotation of the pan&tilt heads to pulse width input to the servo motors, a series of experiments have been conducted to collect data points. The following plot uses linear regression line to fit the data points and the corresponding equation is displayed. The R-square value is very close to unity indicating that the relationship between the angle and the pulse width is very linear.

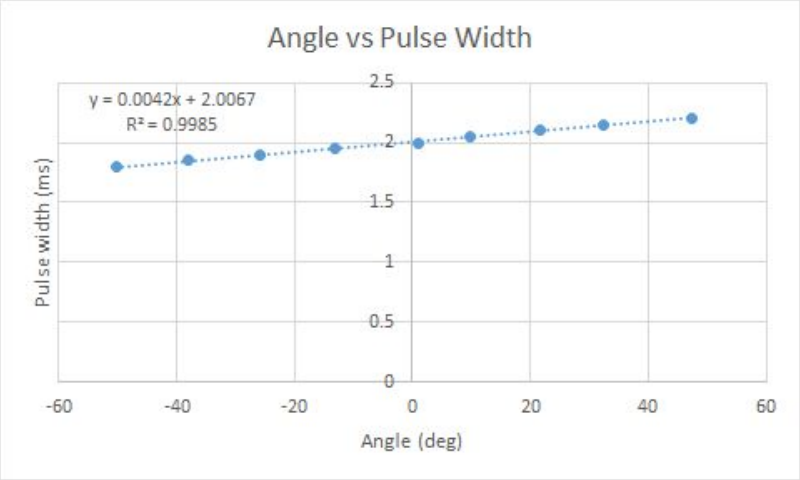

Figure 3, Relationship between angle and pulse width

$$\text{Pulse Width (ms)} = 0.0042 * \text{Angle (deg)} + 2.0067 \qquad \text{Eqn. 2}$$

2.2.3 Closed-loop Control

In last semester when there is no feedback or control implemented for the pan&tilt head, users can never know the actual angular position of the pan&tilt head unless they measure it with tools. The actual angular position may deviate from the user specified angle due to errors in the mechanical parts such as the gears and the way the servo is installed, and also due to noise in the electric signals. Even if the actual angle is known, manually adjusting it to the desired angle is not accurate and very time-consuming. As an improvement in this semester, we installed potentiometer as an encoder to measure the actual angular position of the pan&tilt head and implemented closed-loop control in

LabVIEW to correct deviations. Now with feedback and closed-loop control added to the system, the front panel in LabVIEW displays the actual angular position of the pan&tilt head in real time, correct any deviations to make sure the actual angle is within +/- 0.5 degree of the user specified angle. Now the only thing we need to be careful is to make sure when we physically connect the rotation shaft of the pan&tilt head to the knob of the potentiometer, the neutral position of the potentiometer must match the bottom position of the pan&tilt head.

The closed-loop control algorithm is as follows:

Get **user specified angle**

PWM turns on

While the **actual angle** is outside the **range of tolerance**

If pan&tilt head is not moving

If actual angle is smaller than **specified angle** and outside **range of tolerance**

Increase **PWM** "ON" time by a small amount

If actual angle is larger than **specified angle** and outside **range of tolerance**

Decrease **PWM** "ON" time by a small amount

LabVIEW first gets the specified angle entered by the user and outputs a corresponding PWM signal. The program then checks whether the actual angular position of the pan&tilt head is within the range of tolerance, that is +/- 0.5 degree of the user specified angle. If the pan&tilt head stops at an angle outside the range of tolerance, LabVIEW will make adjustment to the PWM signal one step at a time until the actual angle is within the desired range.

2.3 Power Unit Design

We also measured the power consumption of our system under different cases of operation, since originally we planned to build an integrated power unit on our own to replace the existing power adaptors we bought from the internet.

2.3.1 Power Specifications

Below is the summary of the power specifications for each stepper motor and servo, which are obtained from the data sheets and measurements:

Table 1, Power Consumption

	Acceptable Input Voltage Range	**Chosen Input Voltage with Current Power Adaptor**	**Input Current Range (measured in field tests)**	**Current Power Adaptor Specs**
NEMA 23 Stepper Motor	12V ~ 48V	19.5V	330mA ~ 850mA	**Dell AC Adaptor** Output: 19.5V, 16.9A max, 330W max
HS-785HB Servo	4.8V ~ 6V	6V	12.5mA ~ 375mA	**ServoCity AC Adaptor** Output: 6V, 3A max, 18W max

Note: these values are for one unit. There are 8 stepper motors and 8 servos.

The stepper motor and the servo accept a range of input voltages and we are currently using a 19.5 V, 330 W Dell AC adaptor to power the entire stepper motor module and a 6 V, 18 W Servocity AC adaptor to power the whole servo unit.

2.3.2 AC Adaptor Circuit

A basic AC adaptor circuit consists of four parts: a transformer, a full-wave rectifier, a reservoir capacitor, and a voltage regulator. The transformer steps down the high AC voltage from the wall outlet to the level required by our system load. The full-wave rectifier converts the AC voltage to a pulsating DC voltage. The reservoir capacitor smooths the pulsating DC voltage from the rectifier. However, the DC voltage after the reservoir capacitor still has little ripples, which is then further smoothed out by the voltage regulator. The output voltage of the AC adaptor circuit is approximately a constant DC voltage. In order to achieve a large safety factor, the AC adaptor circuit should be able to supply each stepper motor a maximum current of 1 A and each servo a maximum current of 0.4 A. Therefore, we wanted to design our power unit to be able to supply a maximum total current of 11.2 A (1*8+0.4*8=11.2A).

2.3.3 Sticking With the Existing Power Adaptors

We decided to stick with the existing power adaptors we are using because we could not find all the appropriate parts to build a power circuit that can supply a maximum total current of 11.2 A at 19.5 V DC. Also, building a power unit with very large current consumption can be dangerous to both people and properties. If the power unit we built were not reliable enough, it could destroy the entire system we built. The existing power adaptors have been working quite well and at the same time, we have problems in other parts of our project such as establishing the communication between the chips and LabVIEW and we need more time to solve them.

2.4 User Interface In LabVIEW

LabVIEW is a widely used graphical programming language. It has a library to support communication with ATmega328. The main reason that LabVIEW is used is that it is easy to create a user interface in the LabVIEW environment. The interface splits into different blocks. There are settings blocks and function blocks.

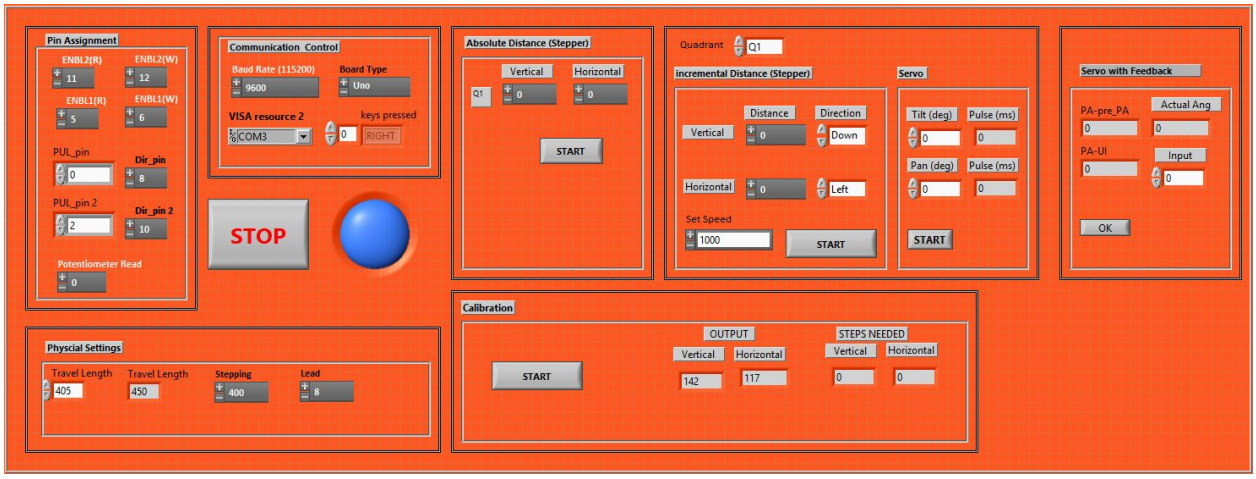

Figure 4, LabVIEW User Interface Front Panel

2.4 Printed Circuit Board(PCB) Design

2.4.1 Descriptions of PCB design

The PCB we designed need to replace with the original data acquisition device Arduino Mega in order to avoid overkill issues. The most three important chips we used in our design included FT230X, ATmega328, and PCA9685. The FT230X chip is the most basic chip that converts the USB signal to UART signal, which is responsible for the communications between the LabVIEW software and the main microprocessor ATmega328. We choose ATmega328 as our main microprocessor because we can then reuse the LabVIEW Interface for Arduino (LIFA) software code that was previously written from last semester. The PCA9685 is a 16-channel LED driver chip, which communicates with the main microprocessor via I2C and output PWM signals to drive 8 servo motors in our project.

Other components in the PCB design include three 3-to-8 decoders 74AC11138, which is used to output the control signals, enable, direction, and pulse, for 8 stepper motors in all the quadrants. One 3-to-8 mux is used to receive potentiometer signals for 8 servo motors in all the quadrants. Sixteen snap action switches are put in parallel, and one digital pin is used for all these switches signals. In addition, eight lines from PWM channels in PCA9685 are connected to 8 servo motors with 220 ohms resistors protected.

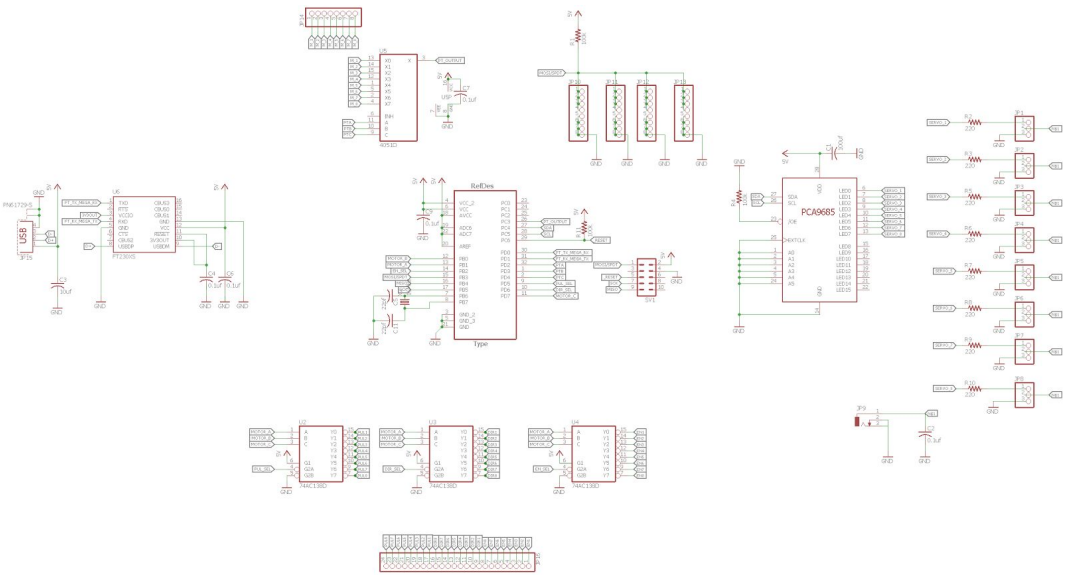

Figure 5, PCB Design Schematics

The above figure is the PCB schematic we designed.

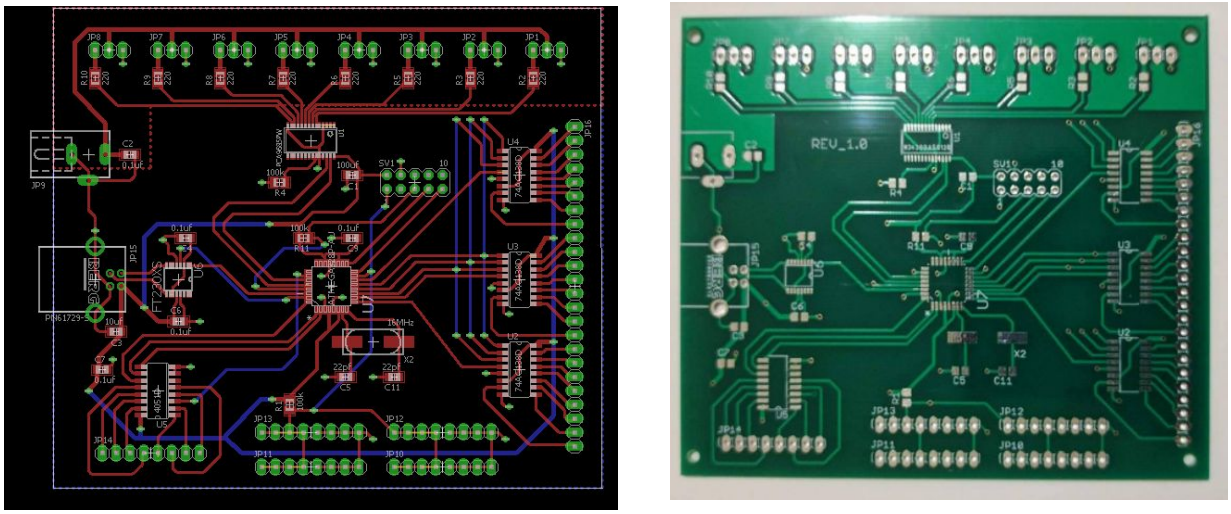

Figure 6, PCB board layout(left) and actual board front view(right)

The PCB board is a 2-layer board. The bottom plane is the GND plane, and one-fourth area on top plane is the power plane for servo motors.

2.4.2 Problems and errors of PCB

We eventually gave up our PCB design due to several reasons. The first round of PCB order was delayed, and our actual first round of PCB was the second round order of PCB through ECE department. Some big mistakes included that we received wrong packages of our decoders 74AC11138, and we didn't

have time to reorder them. Because we ran out of time, we only accomplished all functionalities and features that we proposed for one quadrant in the system. The decoders were no longer needed. Some minor mistakes we made were that we need an extra logic OR gate to receive correct digital signals from snap action switches, and we forgot to pull HIGH SDA and SCL pins, which were critical for I2C communications between PCA9685 and our main microprocessor.

2.4.3 Final circuit on perfboard

Instead, we kept our most important three chips, FT230X, ATmega328, and PCA9685.

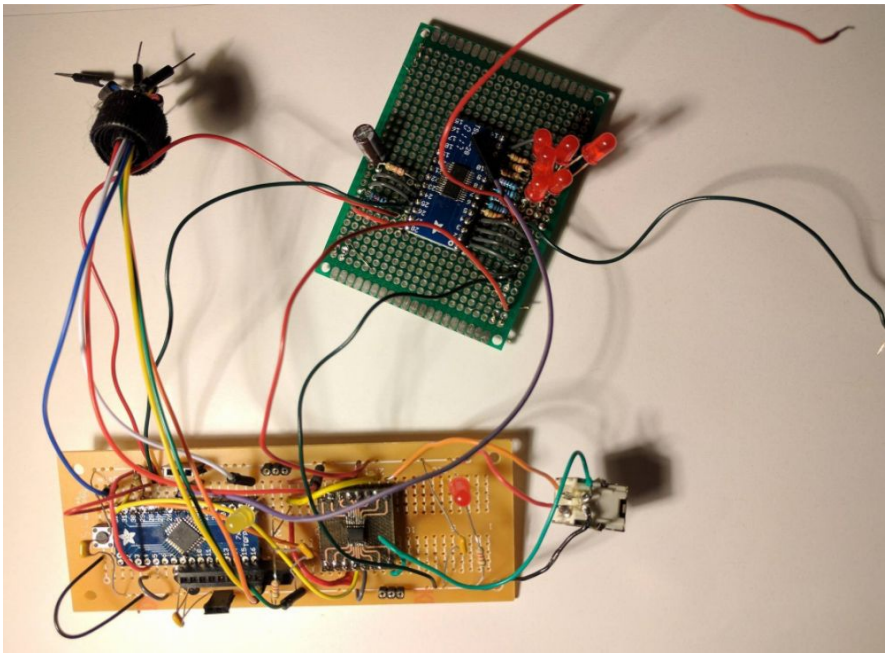

Figure 5 Final circuit on perf board

3. Design Verification

The requirements and verifications for all the modules are included in Appendix A Requirements and Verifications table.

4. Costs

4.1 Parts

Table 2, Parts List 2016

	Item Name	Supplier	Quantity	Cost Per	Total	Notes
XY Positioning						
1	C-Beam Linear Actuator	OpenBuild	8	139.75	1006.2	500mm

	Bundles					
Pan/Tilt						
2	PT785-S Pan & Tilt System	Servocity	4	349.99	1399.96	7:1 ratio option
3	Aluminum Plate	McMaster Carr	1	31.24	31.24	
Frame						
4	Caster Corner Bracke	OpenBuild	8	1.5	12	
5	Corner Bracket	McMaster Carr	16	5.39	86.24	
6	Frame Tubing 60mm (10 ft)	McMaster Carr	9	109.89	989.01	30ft. Total
7	Frame Brace	McMaster Carr	4	27.36	109.44	
8	Frame Bracket	McMaster Carr	16	8.96	143.36	
9	Extended Frame Bracket	McMaster Carr	4	12.06	48.24	
Controls						
10	Arduino Mega 2560 R3	adafruit	1	45.95	45.95	
11	8-Channel PWM	adafruit	1	9.95	9.95	
12	DQ542MA Stepper Motor Driver	OpenBuild	8	39.95	287.64	
13	2-Axis Joystick	adafruit	1	6.95	6.95	
14	Toggle Switch	adafruit	1	2.95	2.95	
15	6V, 3A Power Supply	Servocity	1	24.99	24.99	
16	Laptop AC Adapter	LaptopPartStore	1	79.99	79.99	
17	2-axis Joystick	Adafruit	1	6.95	6.95	
18	Power Adapater	Adafruit	1	6.95	6.95	
19	On-Off Button	Adafruit	5	1.95	9.75	
20	Plastic Case	Amazon	1	9.48	9.48	
21	Plastic NEMA Box	Amazon	1	25.39	25.39	
Tool + Hardware						
22	Wire Cable	OpenBuild	8 ft	2.5	20	4 Conductor Option
23	Square Nut (94855A216)	McMaster Carr	1	2.75	2.75	
24	Socket Cap Screw (92220A153)	McMaster Carr	1	11.79	11.79	
25	Low Profile Cap Screw	McMaster Carr	1	6.25	6.25	
26	No.8 Washer	McMaster Carr	1	3.87	3.87	
27	Wire Sleeve	McMaster Carr	3	3	30	50ft. Option
28	WD40	McMaster Carr	1	1	6.77	

29	Twisted Wire	Servocity	1	15.95	15.95	
30	Male/Female Set	Servocity	8	2.99	23.92	
31	Security Clips	Servocity	3	3.95	11.85	
32	4-Strand Cable	Amazon	1	37.99	37.99	
33	Electrical Tape	Amazon	1	4.42	4.42	Single Roll
34	Prototype Board	Amazon	1	5.17	5.17	
35	Heat Shrink	Amazon	1	11.16	11.16	3/8 Diameter, 48 Inch
				total	4534.5	

Table 3, Parts List 2017

Part	**Item Descriptions**	**Retail Cost ($)**	**Unit Purchased**	**Actual Cost ($)**
74HC4051N	MUX/DEMUX 8X1	0.60	2	1.20
74AC11138	3-8 Decoders	2.78	4	11.12
535-10226-6-ND	16 MHz Crystal	0.50	4	2.00
PCA9685	LED-driver	2.31	2	4.62
609-3657-ND	USB B Connector	0.58	2	1.16
FT230X	USB to BASIC UART IC	2.04	4	8.16
ATmega328	Processor	1.96	2	3.92
N/A	USBASP programmer	6.49	1	6.49
Total				**38.67**

4.2 Labor

The minimal stipend for a graduate student who has 50% teaching assistantship (TA) or research assistantship (RA) is about $1817.87 per month [5]. Considering the standard working time 40 hours per week, the hourly salary for a graduate student can be safely estimated around $11.36.

Our team consists of three members, and we estimate the hours need to complete this project is about 40 hours in total. Thus the total cost of labor in this project is estimated around (graduate student minimal salary/hour) x (2.5) x (hours to complete) x (number of members) = $11.36 x 2.5 x 40 x 3= $ 3408.00.

5. Conclusion

5.1 Accomplishments

We successfully implemented our circuit that consisted of three most important chip FT230X, ATmega328, and PCA9685 on perf board and thoroughly tested it. The circuit performed as expected, and the requirements of all these chips were carefully verified. The calibration and servo motor close-loop feedback control work properly on one quadrant of the camera system.

5.2 Uncertainties

The demo LabVIEW interface is optimized for one quadrant. To control all the quadrants, the LabVIEW needs to be modified. Also, the installment of the potentiometer on the pan & tilt head and the snap action switches on the traversing system is temporary and fragile. To make a more solid attachment, holders or housing units can be designed and 3D printed.

5.3 Ethical considerations

The whole system consumes a significant amount of power in operation. Especially, the 8 stepper motors require a relatively large amount of current supply. The datasheet of the specific model of NEMA 23 stepper motor indicates that the maximum current draw of the motor is 2.8 A/Phase. The current required to power the servo motors scales up with the load applied. The stepper motors draw current even when they are idle. We are aware of the fact that the power regulator not only needs to supply appropriate voltage but also tolerates the theoretically maximum current draw. Therefore, the power unit will be designed with the maximum current and voltage settings in mind. This is to prevent the system from overheating and damaging the power unit.

In accordance with the IEEE Code of Ethics, #6 “to undertake technological tasks for others only if qualified by training or experience, [6]” all group members will be trained and certified to use standard lab tools. Only members experienced with and trained to use prototyping machines such as laser cutter, milling machines, etc. will be in charge of manufacturing parts if necessary.

We will adhere to and honor the IEEE Code of Ethics, #7 “to seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, and to credit properly the contributions of others [6].” The team will keep in contact with the users and listen to their feedback. We will accept honest criticism from our sponsors and try to improve upon current work. We appreciate all participants of this projects for their valuable suggestions and feedback.

5.4 Future work

We want to expand our calibration feature and servo motor close-loop feedback controls to all quadrants, and we want to integrate all necessary circuit parts into one PCB unit to replace with Arduino Mega and avoid overkill issues. In the future, we also want to make advanced version of LabVIEW software to control four quadrants at the same time.

References

[1] Adafruit.com, ‘PCA9685’, 2017, [online]. Available: https://cdn-shop.adafruit.com/datasheets/PCA9685.pdf

[2] Omron.com, ‘Snap Action Switch’, 2017, [online]. Available: https://www.components.omron.com/components/web/PDFLIB.nsf/0/FE0F8E8EEB5D725485257201007DD573/$file/V_1110.pdf

[3] Phidgets.com, ‘3201_0 - Hitec HS-785HB Winch Servo’, 2017, [online]. Available: http://www.phidgets.com/products.php?product_id=3201_0

[4] Phidgets.com, ‘GENERAL SPECIFICATION OF HS-785HB WINCH SERVO’, 2003, [online]. Available:http://www.phidgets.com/documentation/Phidgets/3201_0_HS-785HB.pdf

[5] Illinois Human Resources,"Graduate Minimum Salaries," 2017. [Aanlyn]. Available: http://www.ahr.illinois.edu/grads/grad1617ratesB.pdf.

[6] IEEE,”7.8 IEEE Code of Ethics.”2017, [Online]. Available: http://www.ieee.org/about/corporate/governance/p7-8.html

Appendix A Requirement and Verification Table

Table 4, System Requirements and Verifications

Requirement	Verification	Verification status (Y or N)
FT230X usage 1. The USB cable should provide a voltage from 5V+/-0.5V to the circuit board. 2. The FT230X should be able to convert the USB signal to UART signal and communicate with ATmega328 3. The FT230X has data transmission (baud) rate 9600, packet size 15 bytes	1. Plug in a USB cable to a host computer and connect it to a type B connector connected to the circuit 2. Turn on digital multimeter and use banana pin cables to measure the voltage across the chip 3. Open LabVIEW interface on computer and press the push button to pull low RESET pin on ATmega328 4. Observe if the "Ready" signal on the LabVIEW interface is turned on, which means the communications are successful	1.Y 2.Y 3.Y 4.Y
PCA9685 usage 1. The PCA9685 chip should output 60Hz signals on channel 0 to 7. 2. A user should be able to change the angle of a simple servo motor by changing the pulse width of the PWM signals. 3. The I2C communication should operate in full-speed data transfer rate around 100 kHz. 4. The PCA9685 chip normally operates from 24 Hz to 1526 Hz with adjustable duty cycles of PWM. 5. The board should output a PWM with pulse width varying from 600 to 2400	1. Plug in a USB cable to a host computer and connect it to a type B connector connected to the circuit 2. Open LabVIEW interface on computer and press the push button to pull low RESET pin on ATmega328 3. Probe the PWM channels on the chip with an oscilloscope and observe square waveform on the screen. 4. Light up the LEDs attached to channel 2 to 7 in sequence. 5. Observe the pulse width displayed on the interface.	1.Y 2.Y 3.Y 4.Y 5.Y
ATmega328 usage 1. The RESET pin on the ATmega328 chip works properly 2. The ATmega328 chip should be	1. Plug in a USB cable to a host computer and connect it to a type B connector connected to the circuit 2. Turn on digital multimeter and use banana pin cables to measure the voltage	1.Y 2.Y 3.Y 4.Y

programmable using external crystal of 16M Hz 3. The ATmega328 chip should be able to communicate with FT230X chip via UART signal and PCA9685 chip via I2C 4. The ATmega328 chip should be able to output correct direction, enable, and pulse signals to stepper motors for one quadrant 5. The ATmega328 chip should be able to receive correct analog signal from a potentiometer. 6. The ATmega328 chip should be able to receive correct digital signal from snap action switches	across the chip which should be 5V +/- 0.5 3. Observe whether a 16 MHz crystal is connected between Pin 7 and 8 4. Open LabVIEW interface on the computer and press the push button to pull low RESET pin on ATmega328. The LED that is connected to SCK pin should blink three times to indicate RESET. 5. Observe if the "Ready" signal on the LabVIEW interface is turned on, which means the FT230X communications are successful 6. Test all functionalities of LabVIEW interface including calibration process, stepper motor controls, and close-loop feedback to see if it passes all the tests. (The procedures of testing the softwares are described in next sections.) 7. To test ATmega328 chip correctly received analog signal, probe the analog pin that potentiometer connected using oscilloscope. Observe the voltage changes from potentiometer. If the desired angle of servo motor is less than actual angle, the analog signal should increase. Vice versa. 8. To test ATmega328 chip correctly received digital signal, connect Banana cable pin to the digital pins where snap action switches are used. The LOW state should output voltage 0+/-0.5V, and the HIGH state should output voltage 5V+/-0.5V.	5.Y 6.Y 7.Y 8.Y
Stepper motor control via LABVIEW 1. The LabVIEW interface should allow a user to move the traversing system in two different ways. First, whenever an arrow key is pressed, the camera should move along the input direction for 1 mm. Any other keys presses should be ignored. 2. The stepper motor steps 400 times per	1. Plug in a USB cable to a host computer and connect it to a type B connector connected to the circuit 2. Open LabVIEW interface on computer and press the push button to pull low RESET pin on ATmega328 3. Click the "Start" button inside the "stepper move" block on the LabVIEW interface	1.Y 2.Y 3.Y 4.Y 5.Y 6.Y

revolution and an object travels 1 mm along the setscrew per revolution. 3. The user can also input a specific travel distance and direction from the current position and expect the camera platform to move to that location. 4. A soft travel limit should be imposed on the traversing system. The moving platform should stop 1cm-+1mm short of hitting the snap action switches during normal operation.	4. To test the arrow key function, press each arrow key on the computer and observe whether the camera moves to the desired location. Meanwhile, observe if the horizontal or vertical position displayed on LabVIEW interface is incremented or decremented by 1 mm depending on the key pressed. Also notice the steps indicator on the interface. 5. To test the input command function, select the travel direction and input desired travel distance. Then observe whether the camera moves to the desired location. Meanwhile, observe if the horizontal and vertical position displayed on LabVIEW interface is incremented or decremented by the input distances 6. To test the safe zone protection feature, input a travel distance that is outside the travel limit. Verify that the camera stops 1cm +/- 1 mm before it hits the limit switches.	
Calibration Unit 1. The LabVIEW interface should allow a user to start a calibration sequence 2. The calibration function should move the camera platform to the motor ends of each beams until it hits the snap action switches. 3. Once the snap action switches are hit, and the digital signal received in ATmega328 will change from LOW (0V+/-0.5V) to HIGH(5V+/-0.5V). The horizontal and vertical positions displayed on LabVIEW interface should be reset to zero. 4. After both directions are calibrated, the camera should move to the middle point of the quadrant (180mm+/-1mm, 180mm+/-1mm).	1. Plug in a USB cable to a host computer and connect it to a type B connector connected to the circuit 2. Open LabVIEW interface on computer and press the push button to pull low RESET pin on ATmega328 3. Click the “Start” button inside the “calibration” block on the LABVIEW interface 4. Observe whether the camera moves to the motor ends of each beam and stops when it hits the snap action switches. 5. Verify whether the horizontal and vertical positions displayed on the LabVIEW interface are reset to zero after it hits the snap action switches. 6. Observe whether the camera moves to the middle point of the quadrant within the tolerance.	1.Y 2.Y 3.Y 4.Y 5.Y 6.Y
Close-loop control feedback	1. Plug in a USB cable to a host computer	1.Y

1. The LabVIEW testing program should able to adjust the angles of servo motors to match the angle measurement from a potentiometer. 2. The testing software should be able to accept desired angles as input and output both angles with and without close-loop feedback control 3. The angle with the close-loop feedback control has a tolerance within +/-0.5 degrees	and connect it to a type B connector connected to the circuit 2. Open LabVIEW interface on computer and press the push button to pull low RESET pin on ATmega328 3. Connect a servo motor to one of the PWM channel on PCA9685 and a potentiometer to an analog pin on ATmega328 4. Input desired angle and observe whether the servo motor rotates to the desired angle 5. Observe if the angle adjusts from the initial position to the desired angle with close-loop feedback 6. Compare the differences between the angles between open-loop control and close-loop control to prove the concept	2.Y 3.Y 4.Y 5.Y 6.Y