ROBOTIC CARICATURE ARTIST

By

Dylan Huang
Peter Kuimelis
Soumithri Bala

Design Review for ECE 445, Senior Design, Spring 2018
TA: Xinrui Zhu

Project No. 2

Contents

1 Introduction

1.1 Objective

A caricature is a style of portrait that exaggerates certain features of the subject for comedic effect. Modern caricature artists typically work as street vendors or at social events, creating caricatures of patrons for a small fee. But caricature artists have a skill that takes at least a year perfect and train making them inaccessible and rare, especially if you do not live in a city [1]. And since we do not have many hours, we wish to create an automated system that replicates the caricature artist experience as closely as possible. Caricature artists can take a mental snapshot of the subject and draw a simple portrait in a few minutes. Thus, our system should be able to capture an image of a patron, apply effects to the image, and use a motorized system to draw the image onto a sheet of paper.

1.2 Background

Image processing is ubiquitous in entertainment -- applications such as Snapchat allow for one to immediately apply filters to an image, which has proved to be tremendously popular. One feature nearly all apps of this nature share is instant gratification; in some cases, filters can be previewed before the image is captured. The entertainment value of certain art, however, is being able to experience the process of creation as much as the result. Listening to music at a concert, for example, is completely different from listening to a pre-recorded track. This motivated our decision to have our project replicate the excitement and anticipation of watching an artist work rather than just receiving the final product.

Although our system also uses a camera for image capture, the audience will not receive a preview of the applied transformations before the drawing is completed. Next, rather than use a printer to create the drawing, we want to use a v-plotter system mounted to an easel, as this approach has many advantages. First, the illustration process is bared to the user, as the paper is exposed during the process. Further, the toolpath of a v-plotter will more closely mimic that of a human, as the machine will draw each discrete stroke rather than sweep the page along an axis. Finally, the drawing is also done with a pen, which adds a natural variation that a desktop printer lacks.

While the hardware of this project is incredibly important, the software will be responsible for integrating all the individual components into a cohesive product. The image processing component of our project will be done entirely in software; there is prior research in computer-generated caricatures which we are looking to implement [2]. Additionally, lower-level software, such as firmware, will be essential for reliable communication between the computers and microcontrollers in our project.

1.3 High-Level Requirements

- Apply a caricature effect to an image and generate plotter instructions
- Plot geometric shapes and complex lines given a G-code script
- Must perform the entire process autonomously

2 Design

2.1 Block diagram

Our block diagram requires three modules: a power supply, control module, and I/O module. The power supply's purpose is to drive voltage to the control module to the Powered USB Hub, and ATmega328 custom circuit. The power supply ensures that our entire system is reliably powered and can be driven from any wall outlet. The control module consists of a Raspberry Pi Computer, Powered USB Hub, and ATmega328 custom circuit that will be able to accept inputs from a USB camera, caricaturize and vectorize an image, and driver stepper motors that will effectively carry out the vectorized instructions and draw an image. The peripheral module includes the USB camera and may be expanded to improve user experience. The mechanical module consists of all the components that are needed to produce the final drawing: two stepper motors and a servo motor.

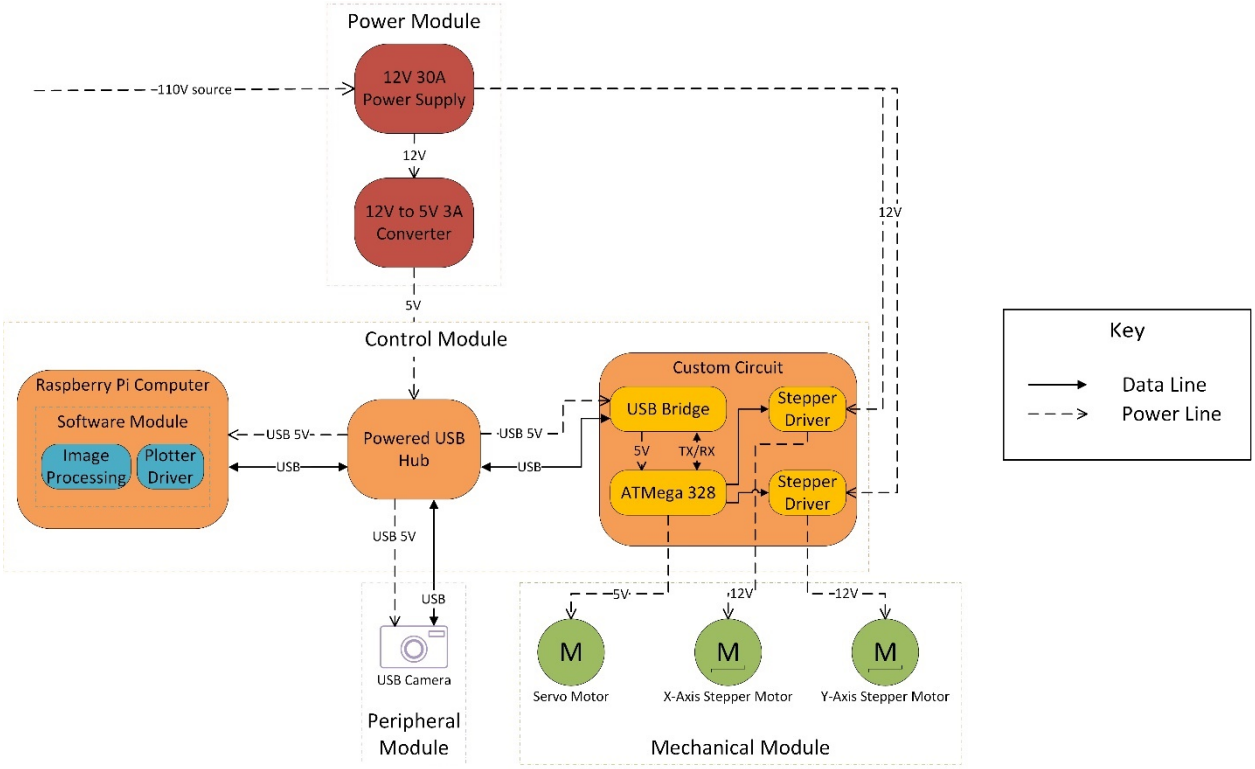

Figure 1 Block Diagram of Electronic Components

2.2 Physical Design

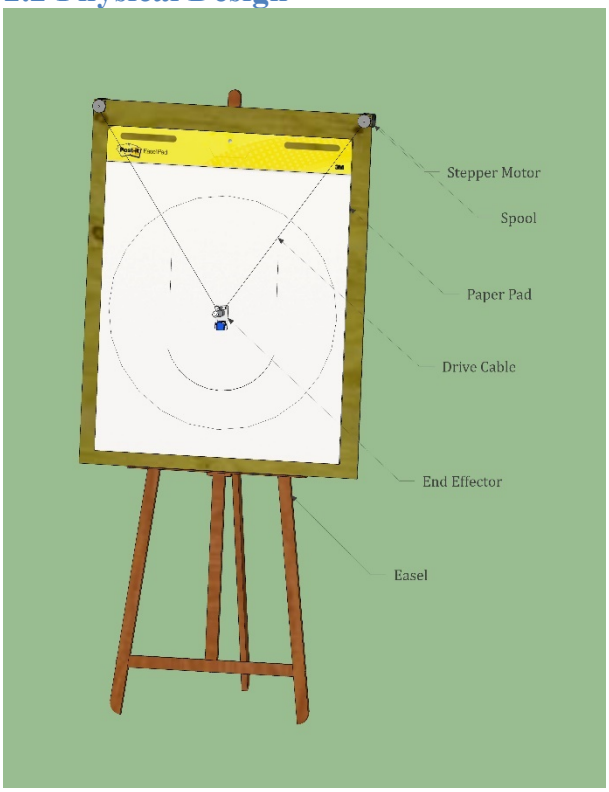

Figure 2 Isometric View of Mechanical Assembly [3] [4]

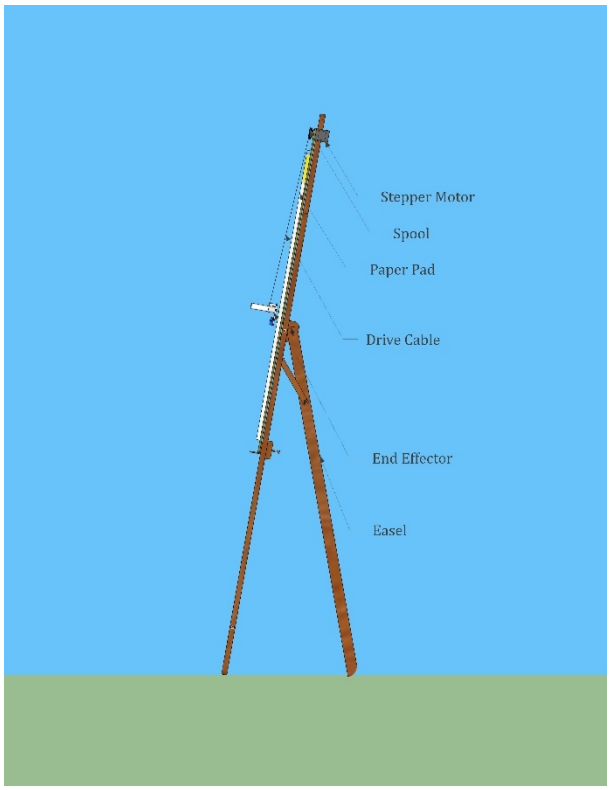

Figure 3 Side View of Mechanical Assembly [5] [6] [7]

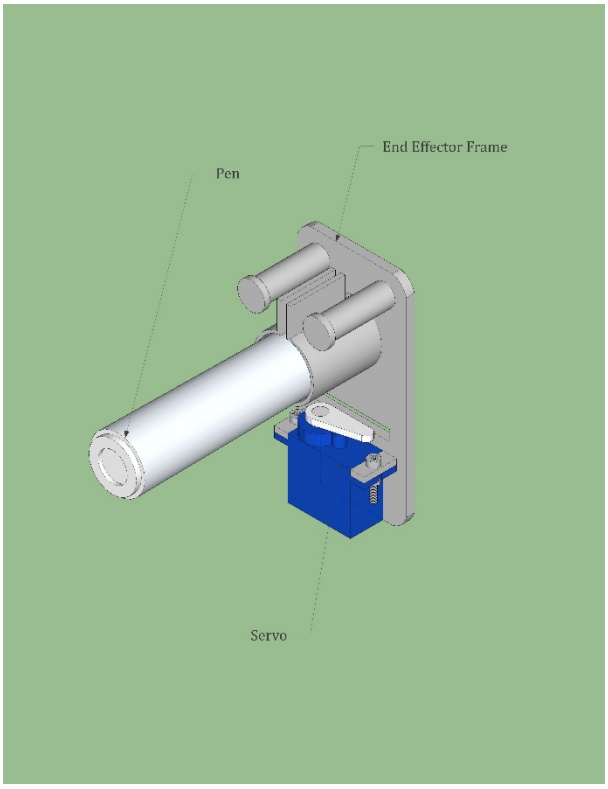

Figure 5 Isometric View of End Effector [8] [9]

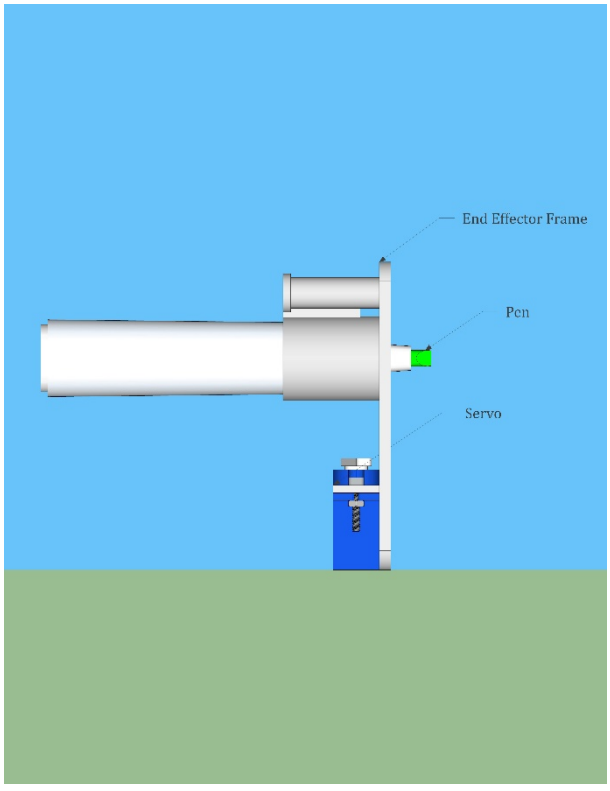

Figure 4 Side View of End Effector [10] [11]

The diagrams on the previous page illustrate the mechanical component of our project. The frame consists of a plywood board mounted to an off-the-shelf easel; the plywood board will have mounting points for a 23” x 20” easel pad and two stepper motors. The end effector connects the pen to the stepper motors through the drive cable and contains a servo to lift the pen off the page between strokes.

2.3 Block Design

2.3.1 Power Module

The power supply will power the circuit from a standard wall outlet. It will provide a constant 12V voltage to the custom circuit so that the stepper drivers can reliable receive voltage and pulse the motors. And it will provide a constant 3A 5V to the USB powered hub that will go on to power the ATmega328 and the Raspberry Pi Computer.

2.3.1.1 12V 30A Power Supply

The 12V power supply will power our control module through the stepper drivers in the custom circuit and the powered USB hub. The converter and stepper drivers will both receive a 12V input from this component. The power supply will help contribute to all three of the high-level requirements by being the bedrock of voltage source for all our components.

Table 1 12V 30A Power Supply R&V

Requirement	Verification
Converts wall voltage to (110V) to 12V 30A DC ± 5%	• Use a multimeter to make sure that 12V and 30A is coming out of the power supply

2.3.1.2 12V to 5V Converter

The 12V to 5V converter is wired to the USB hub, which supplies a 5V conversion of voltage to the Raspberry Pi, ATmega328, and any other components that need a 5V source. This 12V to 5V converter is important to our design because there are many components that accept a 5V power source.

Table 2 12V to 5V Converter R&V

Requirement	Verification
Supply a steady 5V at 3A ± 5% to the Powered USB hub	• Measure the open-circuit voltage of the input and output of the 12V to 5V 3A converter using a multimeter • Measure the 3A output of the 12V to 5V 3A converter using an ammeter in series

2.3.2 Control Module

The control module will control all of the logic of the system from capturing the image, generating the G-code instructions, accepting the instructions on the PCB, and carrying out the stepper motor and servo motor instructions.

2.3.2.1 Raspberry Pi

The Raspberry Pi is our image computing and data facilitation unit for our entire system. It will allow us to receive images from the USB camera, filter the image, vectorize, and convert it to consumable G-code for our custom circuit to read and drive our necessary motors. This component is especially necessary to achieve the "caricature" effect for our drawing robot. The Raspberry Pi will interface, through our USB hub, with the ATmega328 and the USB camera. This component is necessary to generate consumable G-code and allow our robot to perform the entire process autonomously.

Table 3 Raspberry Pi R&V

Requirement	Verification
1. Receive an image from the camera 2. Apply filters to the image, vectorize the image and save in SVG format 3. Convert final vector drawing into G-code 4. Send commands to the ATmega328 over a USB serial interface.	1. Ensure images taken on camera appear in specified location in the Raspberry Pi's file system 2. Open the SVG file in a vector manipulation program (Inkscape) to empirically verify that the vectorized image approximates the raster image 3. Run the G-code through a toolpath simulator that checks for syntax errors, and empirically verify that the visual output of the G-code simulation matches the vector file 4. Send G-code instructions to ATmega328 circuit and check for acknowledgement signal

2.3.2.2 Powered USB Hub

The USB hub accepts power from the 12V to 5V converter and supplies 5 volts to all USB devices, such as the Raspberry Pi, Camera, and the custom circuit. Additionally, the USB hub allows for all USB devices to communicate with the Raspberry Pi. The hub also modularizes our system, allowing us to conveniently connect additional peripherals and interchange our computing component (Raspberry Pi). Furthermore, the Powered USB Hub ensures our components reach their required current draw; this could be beneficial if we decide to add more peripherals to the design and the Raspberry Pi's 1.2A maximum total USB current draw is exceeded [12].

Table 4 Powered USB Hub R&V

Requirements	Verification
5V to 3A ± 5%	• Measure the open-circuit voltage across the VCC and GND pins of the USB female connectors and ensure 5V 3A ± 5%

2.3.2.3 ATmega Custom Circuit (PCB)

This component receives machine code in the form of G-code instructions from the Raspberry Pi, converts the commands to motor rotations, and sends information to the stepper and servo drivers. The instructions supported must include start, stop, set feed rate (to control the speed of drawing), raise/lower drawing implement and movement between coordinates on an X-Y plane.

The interface between USB and the microprocessor is handled by a Silicon Labs CP2102, an inexpensive and ubiquitous USB bridge IC [13]. The ATmega328 microprocessor is driven with a 16MHz oscillator to achieve the maximum clock frequency of the device, to ensure that we accommodate our requirements for instruction throughput [14]. The in-chip serial programming (ICSP) pins are made accessible with pin headers to allow the bootloader to be burned on to the EEPROM of the ATmega328.

We use the Pololu carrier board for the Allegro A4988 stepper motor controller to drive the stepper motors [15]. The MSX pins on the IC have been wired to keep the device in 1/16 microstep mode, and the digital control inputs are wired to digital I/O pins on the ATmega. We have also included a momentary tactile switch which allows the user to reset both the ATmega and USB bridge IC devices for debugging purposes.

We add one 100uF decoupling capacitor per stepper motor driver to reduce ripple from the power supply, as specified in the reference circuit in the A4988 documentation.

The full schematic is included on the next page.

Table 5 ATmega PCB R&V

Requirements	**Verifications**
V5 (name of power input line) remains at 5V ± %10 throughout operation of machine	1. Use multimeter to measure V5 node and measure the maximum deviation

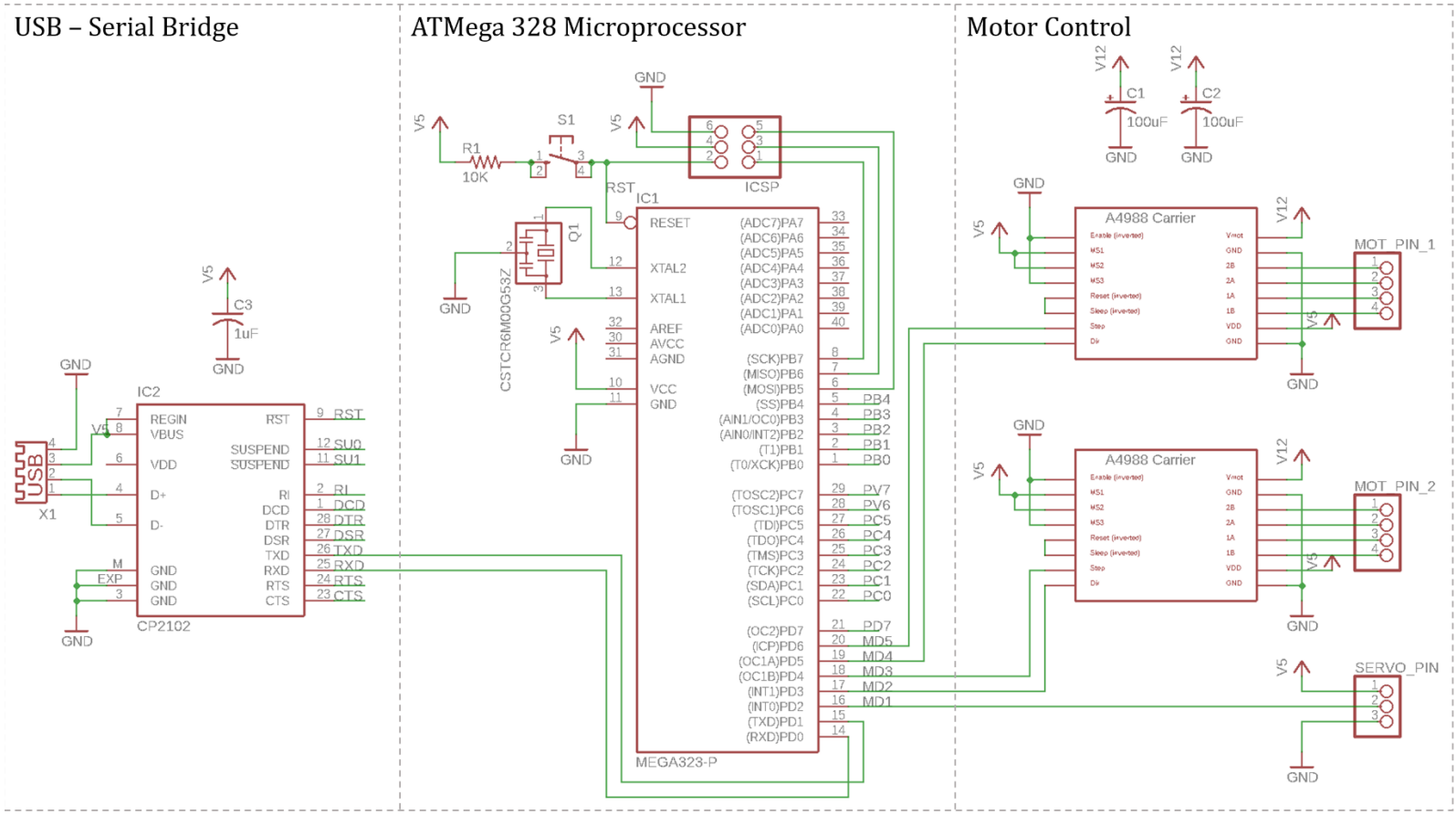

Figure 6 Schematic for PCB

2.3.3 Peripheral Module

2.3.3.1 Camera

The Raspberry Pi will capture an image of our subject using the camera. The camera must be able to receive all its power from the USB hub, must support autofocusing, and must be able to capture still photos with a minimum of 8 megapixels. Finally, the physical design of the camera should have mounting hardware to easily interface with the rest of our design.

Table 6 Camera R&V

Requirements	Verifications
1. Minimum of 8-megapixel still photos 2. Automatic focusing in range of lighting conditions	1. Check dimensions of captured image 2. Empirically verify the photos appear focused and properly lit

2.3.4 Mechanical Module

2.3.4.1 Stepper Motors

We are using two Kysan Nema 17 1.8°-step stepper motors in our project, with each motor corresponding to one of the plotter's axes. A spool will be mounted onto each stepper, which will control the radius of each axis.

Table 7 Stepper Motors R&V

Requirements	Verifications
1. 1.8° rotation per step, ± 5% 2. Drawing precision for 5 units of distance at less than 1° of angle error 3. Temperature cannot reach over 40°C when operating	1. Use an encoder, microcontroller, and developed firmware to verify the rotation per step falls within our tolerance range. To measure the angle are going to use the simulation outlined in our tolerance analysis in section 2.4 2. Direct the system to draw a straight line of 5 units of distance and measure with dial calibers the actual Y and X coordinates. Compare these measurements with the expected X and Y coordinates and calculate θ_{error} . Ensure that θ_{error} is less than 1° 3. Execute the entire pipeline for a filtered image and measure the temperature during operation with an IR thermometer. Ensure that the temperature does not exceed 40°C

2.3.4.2 Servo Motor

The servo motor is going to be fitted onto the end effector. Its function is to lift the end effector off the easel to allow the pen to glide to another location without dragging. The servo motor will be directly connected to our custom circuit and be driven by the ATmega328.

Table 8 Servo Motor R&V

Requirements	Verifications
1. Have enough torque to lift the end effector off the easel such that the pen is not touching the paper	1. Place the servo motor under the end effector unit. Use a function generator to supply a PWM wave with a frequency of 50Hz and a duty cycle of 7.5% for position 0. Then change the duty cycle to 10% to move the to the 90° position. Ensure that the servo motor can lift the end effector off the ground.

2.3.5 Software Module

2.3.5.1 Image Processing

The image processing software module converts a PNG image to grayscale, identifies a face in the image, crops the image with a small margin, applies an edge detection filter, vectorizes the image to produce an intermediate SVG file, and generates a sequence of G-code instructions from the SVG file. The high-level flow of operations is illustrated below, with simulated outputs:

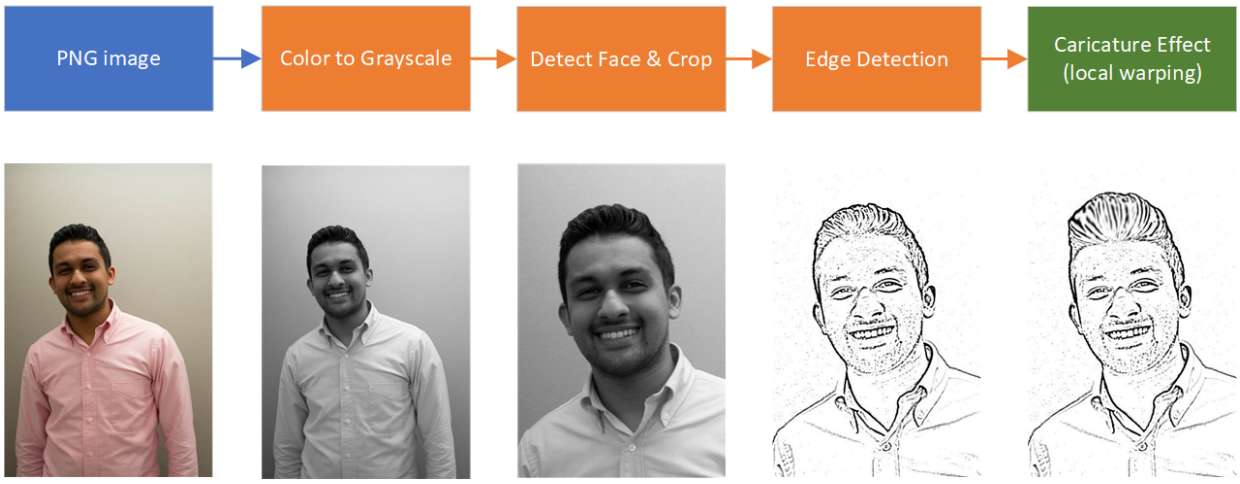

Figure 7 Flow diagram for image processing

The grayscale operation is performed by simply averaging the values of the RGB color channels in the original image. Face detection will be implemented with Haar Cascades in OpenCV. Edge detection will be performed by convolution with a Laplacian of Gaussian (LoG) filter kernel, which combines smoothing and differentiation into one step. The 2-D Laplacian of Gaussian is:

$$LoG(x,y) = -\frac{1}{\pi\sigma^4}\left[1 - \frac{x^2+y^2}{2\,\sigma^2}\right] e^{-\frac{x^2+y^2}{2\,\sigma^2}} \quad (1)$$

Which can be approximated by a discrete kernel [16]. There is one parameter σ (to be determined later) which can be varied to produce an aesthetically pleasing output. We experimented with other edge detection schemes and found this approach to be both effective and computationally efficient.

For the conversing from raster to vectorized format, we will implement a modification of the algorithm described in the Potrace paper [17]. Since our firmware only supports G-code instructions to move from one coordinate to another, we will only include straight line segments in the intermediate SVG file. Thus, we only need to implement the "optimal polygon" portion of the Potrace algorithm, because our G-code does not support Bezier curves or circles. This design decision reduces the computational requirements of the firmware.

We will generate G-code instructions by using an off-the-shelf TSP solver to find an ordering of line segments from the SVG file that minimizes the total distance the end effector must travel [18].

Table 9 Image Processing R&V

Requirements	Verifications
1. Face detection accuracy 90% or better under favorable lighting conditions 2. All processing takes less than 1 minute.	1. Place subject 5 feet in front of camera in a uniformly lit indoor space. Capture image; repeat 100 times. Software must correctly identify and crop face at least 90/100 times. 2. Use a timer library to measure the wall-clock time from beginning to end process (raw image to G-code).

2.3.5.2 Firmware

The firmware for the ATmega328 converts a sequence of G-code instructions to incremental motor movements. A state machine of the operations is shown below:

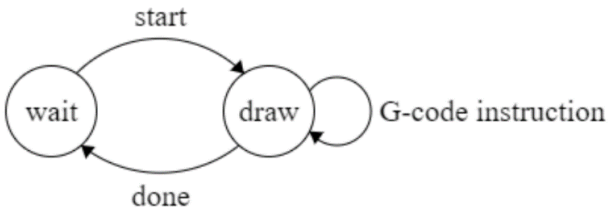

Figure 8 Firmware FSM Model

Our firmware achieves this functionality by receiving G-code instructions over the USB bridge, queuing instructions in the microprocessor's RAM, converting each G-code instruction into a series of small delta movements, and send commands to the stepper drivers accordingly. Ideally, the driving software on the Raspberry Pi should not know of the firmware's coordinate system and should be able to supply Cartesian coordinates through G-code to the circuit. The firmware makes inverse kinematic calculations using the model below:

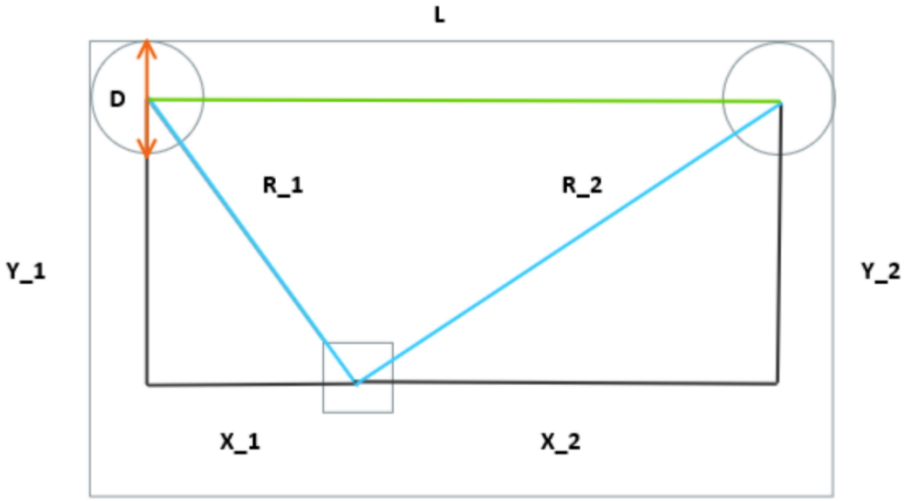

Figure 9 Spatial representation used in kinematic calculations

Using the spatial representation above, we can derive the following equations for calculating the number of motor steps needed to move the end effector from current point $(X_{1,inital}, Y_{1,initial})$ to destination point $(X_{1,final}, Y_{1,final})$.

$$\Delta R_1 = \sqrt{X_{1,final}^2 + Y_{1,final}^2} - \sqrt{X_{1,initial}^2 + Y_{1,initial}^2} \quad (2)$$

$$\Delta R_2 = \sqrt{X_{2,final}^2 + Y_{2,final}^2} - \sqrt{X_{2,initial}^2 + Y_{2,initial}^2} \quad (3)$$

$$MSteps_1 = 3200 * \frac{\Delta R_1}{D\pi} \quad (4)$$

$$MSteps_2 = 3200 * \frac{\Delta R_2}{D\pi} \quad (5)$$

Table 10 Firmware R&V

Requirements	Verifications
1. Receive G-code commands from computer 2. Queue G-code in RAM 3. Convert G-code into multiple delta movements 4. Send commands to stepper drivers	1. Add log statements to print received commands; verify received command is the same as sent command 2. Print contents of storage data structure every second to verify G-code commands have not been lost 3. Capture output of G-code to segment conversion, and use simulation software to verify delta segments correspond to G-code instruction 4. Execute test instruction, measure rotation of stepper motors and verify with inverse kinematic equation.

2.4 Tolerance Analysis

Because drawing requires a certain amount of precision, we decided that a critical requirement is the ability to draw a straight line. We define a suitable stepper motor to be one that can draw a straight line of 5 units of distance at less than 1° of angle error. We can test the performance of our plotter by drawing a straight line and measuring the error in the actual vs. expected ending coordinates as follows:

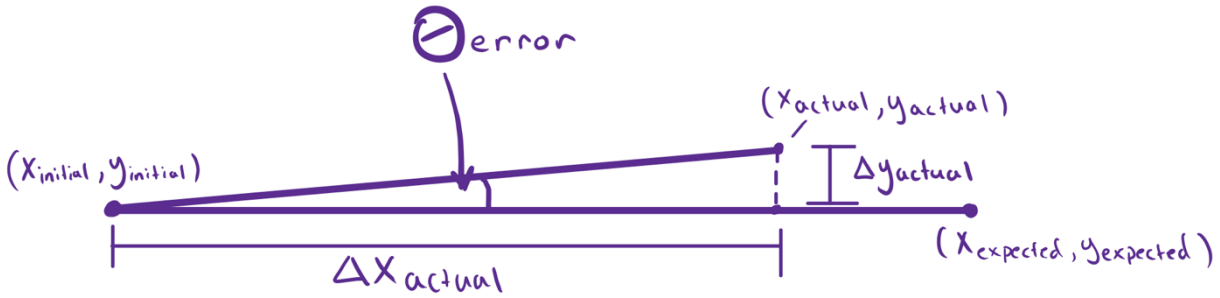

Figure 10 Tolerance analysis illustration showing the effect of stepper motor positional accuracy on degree error while drawing a straight line

We calculate θ_{error} by using the formula:

$$\theta_{\text{error}} = \tan^{-1}\left(\frac{\left|y_{actual} - y_{expected}\right|}{\Delta x_{actual}}\right) \quad (6)$$

Our stepper motors have a 1.8° step with ± 5% accuracy [19]. To calculate our expected error, we can use the following equations based on Figure 9 to calculate the number of steps for each motor:

$$\Delta R_{1,expected} = \sqrt{X_{1,final}^2 + Y_{1,final}^2} - \sqrt{X_{1,initial}^2 + Y_{1,initial}^2} \quad (7)$$

$$\Delta R_{2,expected} = \sqrt{X_{2,final}^2 + Y_{2,final}^2} - \sqrt{X_{2,initial}^2 + Y_{2,initial}^2} \quad (8)$$

$$MSteps_1 = \left(\frac{200\frac{step}{rev}}{\frac{1\ step}{16microstep}}\right) * \frac{\Delta R_{1,expected}}{D\pi} \quad (9)$$

$$MSteps_2 = \left(\frac{200\frac{step}{rev}}{\frac{1\ step}{16microstep}}\right) * \frac{\Delta R_{2,expected}}{D\pi} \quad (10)$$

We then plug $MSteps_1$ and $MSteps_2$ back into the above equations with a degree error of $1.8° * .95 = 1.71°$ per step to yield $\left(\frac{210.53\frac{step}{rev}}{\frac{1\ step}{16microstep}}\right)\frac{microstep}{revolution}$ and find $\Delta R_{1,actual}$ and $\Delta R_{2,actual}$:

$$\Delta R_{1,actual} = \frac{MSteps_1 * D\pi}{\left(\frac{210.53\frac{step}{rev}}{\frac{1\ step}{16microstep}}\right)} = \sqrt{X_{1,final}^2 + Y_{1,final}^2} - \sqrt{X_{initial}^2 + Y_{initial}^2} \quad (11)$$

$$\Delta R_{2,actual} = \frac{MSteps_2 * D\pi}{\left(\frac{210.53\frac{step}{rev}}{\frac{1\ step}{16microstep}}\right)} = \sqrt{X_{2,final}^2 + Y_{2,final}^2} - \sqrt{X_{initial}^2 + Y_{initial}^2} \quad (12)$$

Which simplify to:

$$\Delta R_{1,actual} = \frac{200}{210.53} * \Delta R_{1,expected} = \sqrt{X_{1,final}^2 + Y_{1,final}^2} - \sqrt{X_{initial}^2 + Y_{initial}^2} \quad (13)$$

$$\Delta R_{2,actual} = \frac{200}{210.53} * \Delta R_{2,expected} = \sqrt{X_{2,final+}^{} + Y_{2,final}^2} - \sqrt{X_{initial}^2 + Y_{initial}^2} \quad (14)$$

And since the resulting Y distances are the same and the X distances can be represented as a sum of the ΔX:

$$\Delta R_{1,actual} = \frac{200}{210.53} * \Delta R_{1,expected} = \sqrt{(X_{initial} + \Delta X_{actual})^2 + Y_{actual}^2} - \sqrt{X_{initial}^2 + Y_{initial}^2} \quad (15)$$

$$\Delta R_{2,actual} = \frac{200}{210.53} * \Delta R_{2,expected} = \sqrt{(X_{initial} - \Delta X_{actual})^2 + Y_{actual}^2} - \sqrt{X_{initial}^2 + Y_{initial}^2} \quad (16)$$

To simulate a straight line of 5 units of distance to the right, I assume that our unit of distance is centimeters and that our stepper motors are at a resting position of 20cm. By plugging in the initial X and Y values of (20,20) for ($X_{1,initial}, Y_{1,initial}$) and ($X_{2,initial}, Y_{2,initial}$) and final X and Y values of (25,20) and (15,20), respectively. We find that $Y_{actual} = 20.0155$ and $\Delta X_{actual} = 4.74805$. Plugging this into our error equation (6), we get:

$$\theta_{\text{error}} = \tan^{-1}\left(\frac{\left|y_{actual} - y_{expected}\right|}{\Delta x_{actual}}\right) = 0.187° \quad (17)$$

Which is acceptable under our tolerance of 1° considering that this assumes every step reaches its maximum possible error. Thus, we conclude that the stepper drivers we are using with a 1.8° step and ± 5% accuracy meet the requirement of our design.

3 Cost and Schedule

3.1 Cost Analysis

3.1.1 Bill of Materials Cost

Part	QTY	Unit Price	Cost
A4988 Stepper Motor Driver Carrier	2	5.95	11.90
Omron B3F-1022 Tactile Switch	1	0.21	0.21
CSTCE16M0V53-R0 16MHz Ceramic Oscillator	1	0.50	0.50
Breakaway Headers (40 pins, straight)	1	1.50	1.50
SG90 Mini Servo	1	3.79	3.79
100 uF Electrolytic Capacitor	2	0.35	0.70
1 uF Ceramic Capacitor	1	0.15	0.15
10 kOhm Resistor	1	0.01	0.01
ATmega328-P	1	2.01	2.01
Silicon Labs CP2012 USB Bridge	1	3.17	3.17
Raspbery Pi 3 Model B	1	35.80	35.80
Sketch Pad	1	18.98	18.98
Sabrent 4-Port USB 3.0 Hub	1	14.39	14.39
Kysan 1124090 Nema 17 Stepper Motor	2	14.50	29.00
Mangolian Car LED Display Power Supply 12V to 5V 3A DC/DC Converter	1	7.99	7.99
BMOUO 12V 30A DC Universal Regulated Switched Power Supply 360W	1	18.56	18.56
Total			148.66

3.1.2 Labor Cost

We have chosen an hourly wage from a salary info sheet provided by engineering at Illinois for a Computer Engineering graduate [20].

Team Member	Hourly Rate	Hours	Cost x 2.5
Dylan Huang Peter	39.30 $/hr	180 hrs	$17,685
Peter Kuimelis	39.30 $/hr	180 hrs	$17,685
Soumithri Bala	39.30 $/hr	180 hrs	$17,685
Total			$53,055

Our total cost analysis comes out to be $53,055 + $148.66, which is **$53,203.66**.

3.2 Schedule

Week	Peter	Soumithri	Dylan
2/26/2018	Prepare for Design Review.	Prepare for Design Review.	Prepare for Design Review.
3/5/2018	After receiving the components, begin verification process. Begin the framework for the software suite.	Purchase necessary components and begin verifying that components work properly.	Verify components and work on the beginnings of the software suite with Peter.
3/12/2018	Complete software to receive USB image, filter, and vectorize images.	Complete verification process of every component.	Work on software to receive vectors and generate G-code instructions for our PCB.
3/19/2018 (SPRING BREAK)	Make sure the code is well commented and modularized. Do any additional unit testing for each module and process with a real image.	Collaborate with Dylan to write the firmware for the PCB and begin assembling design.	Start writing the firmware on the PCB to accept the G-code and drive the stepper motors.
3/26/2018	Work with Soumithri on assembling the physical design and hooking up the electrical components.	Assemble the physical design and continually aiding Dylan with the firmware production.	Start testing the firmware with fabricated G-code instructions.
4/2/2018	Complete physical and electrical design.	Complete physical and electrical design.	Complete the firmware for the PCB and complete assembly of the physical and electrical design.
4/9/2018	Start final testing of the complete system and focus on any issues related to image manipulation and data transfer from the Raspberry Pi.	Start final testing of the complete system and focus on any issues related to the physical and electrical design.	Start final testing of the complete system and focus on any issues related to firmware.
4/16/2018	Iron out any software application performance/edge case issues.	Stabilize physical design and make sure that all circuitry and physical components are safely secured and demo-ready.	Iron out any firmware issues and do numerous complete executions for testing.
4/23/2018	Complete entire system and begin final report.	Complete entire system and begin final report.	Complete entire system and begin final presentation.
4/30/2018	Finish Final Report	Finish Final Report	Finish Final Presentation

4 Discussion of Ethics and Safety

We have some safety concerns with this project. The power supply requires some wiring to receive a 110V input; these connections need to be robust and correct to ensure safe operability. The stepper motors also receive a high amount of power during operation, so poor circuit design and wiring could also cause a fire or electrocution hazard. The stepper motors we plan to use can produce high torque figures, which could potentially injure the audience were they to get too close. Before trying to assemble our design or drive any motors with our circuitry, we will ensure that the circuitry has no shorts and each unit is modularly tested for expected voltage and current outputs and inputs. It is also imperative that we use insulated wires such that any external contact with the wire will not cause a circuit malfunction.

Another safety concern is that our design requires two moving parts controlled by a stepper motor. If the stepper motors were to malfunction or be incorrectly driven due to circuitry or faulty software, it could lead to the pen-holder to fall or the easel to violently move which could potentially injure bystanders or spectators. To avoid this, we will first perform circuit tests and then individually test our control module components by attempting incremental movement milestones before attaching our contraption to the easel. We will also perform software validation to confirm that we are correctly producing the expected results. To mitigate any possible malfunction or physical system failure, we have designed our robot to be remotely controllable through the interface of the Raspberry Pi. These precautions will ensure the safety of our machine operator; once our design is fully operational, we will fine-tune the speed and voltage of our stepper motors to reduce vibrations and prevent any movement of the entire unit itself.

One ethical consideration for our project are the negative externalities associated with automating the job of a caricature artist. If our machine were made available for sale, human caricature artists could suffer from reduced job opportunities. This could be a possibly be conflict of interest with any caricature artist who feels that their livelihood is threatened which would violate #2 of the IEEE Code of Ethics [21]. However, we believe that customers would still be willing to pay a premium for the personal touch of a human caricature artist. Additionally, we must be mindful of the ethics surrounding digital manipulation of portraits; it is important that we do not offend patrons with overly grotesque manipulations of their image which would violate #9 of the IEEE Code of Ethics [21]. We currently do not have a solution to this since recognizing such intentions requires someone with a sentient understanding of bullying or malicious behavior. But we imagine that it is still possible to do some sub-optimal software detection to shut down the robot if we detect malicious images being drawn.

5 References

1. The Art Career Project. *How to Become a Caricaturist*. [Online]. Available: https://www.theartcareerproject.com/become/caricaturist/. [Accessed: 17-Feb-2018]
2. Brennan, Susan. *The Caricature Generator*. MIT Media Lab master's thesis, 1982. https://www.jstor.org/stable/1578048. [Accessed: 17-Feb-2018]
3. *easel, support.* Sketchup Warehouse, 2014, [Online]. Available: https://3dwarehouse.sketchup.com/model/ud12546de-5beb-4f56-9cd7-197e3e4edf16/easel-support
4. *Post-It Easel Pads – White & Yellow*, 2015, Sketchup Warehouse, [Online]. Available: https://3dwarehouse.sketchup.com/model/ud12546de-5beb-4f56-9cd7-197e3e4edf16/easel-support
5. *Nema 17 stepper motor.* Sketchup Warehouse, 2013, [Online]. Available: https://3dwarehouse.sketchup.com/model/ad4e266bd910a82ea8bf2263f57d91e2/Nema-17-stepper-motor
6. *Fulie 21 T5 40*. Sketchup Warehouse, 2012, [Online]. Available: https://3dwarehouse.sketchup.com/model/4d91082a0dd495ba11f5f333e878d705/Fulie-21-T5-40
7. *M3 x 0,5 Hex Socket Cap Thumb Screw, 6mm*. Sketchup Warehouse, 2013, [Online]. Available: https://3dwarehouse.sketchup.com/model/54faa4362bc8385eead5604ea0ec0346/M3-x-05-Hex-Socket-Cap-Thumb-Screw-6mm
8. *Board Marker (Open).* Sketchup Warehouse, 2009, [Online]. Available: https://3dwarehouse.sketchup.com/model/6bd7180193a8f310729e474769d21582/Board-Marker-Open
9. *TG9E*. Sketchup Warehouse, 2012, [Online]. Available: https://3dwarehouse.sketchup.com/model/32251201447a3dc9eddb005d3e1563b1/TG9E
10. *M2 Screw – 12mm – Pozidriv Head*, 2014, [Online]. Available: https://3dwarehouse.sketchup.com/model/u5f65936e-aade-4155-b9df-14052a894f27/M2-Screw-12mm-Pozidriv-Head
11. *Mutter, Nut, M2, DIN934*, 2009, [Online]. Available: https://3dwarehouse.sketchup.com/model/dbeb169dcb66434a7773449d0e3d556a/Mutter-Nut-M2-DIN934
12. *Raspberry Pi FAQs – Frequently Asked Questions*. https://www.raspberrypi.org/help/faqs/. [Accessed: 18-Feb-2018]
13. *CP-2012 datasheet.* https://www.silabs.com/documents/public/data-sheets/CP2102-9.pdf. [Accessed: 18-Feb-2018]
14. ATmega328 Datasheet. http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-42735-8-bit-AVR-Microcontroller-ATmega328-328P_Datasheet.pdf. [Accessed: 18-Feb-2018]
15. Pololu. *A4988.* 2018, [Online]. Available: https://www.pololu.com/file/0J450/A4988.pdf. [Accessed: 18-Feb-2018]
16. *Laplacian of Gaussian.* https://homepages.inf.ed.ac.uk/rbf/HIPR2/log.html [Accessed: 21-Feb-2018]
17. Selinger, Peter. *Potrace: a polygon-based tracing algorithm.* SourceForge, 2003.

18. *Concorde TSP Solver*. http://www.math.uwaterloo.ca/tsp/concorde.html. [Accessed: 18-Feb-2018]
19. *Kysan Nema 17 Stepper Motor*. https://ultimachine.com/products/kysan-1124090-nema-17-stepper-motor. [Accessed 22-Feb-2018]
20. Salary.Info.Sheet.pdf. https://engineering.illinois.edu/documents/Salary.Info.Sheet.pdf. [Accessed 22-Feb-2018]
21. IEEE. *IEEE - IEEE Code of Ethics*. 2018. [Online]. Available: https://www.ieee.org/about/corporate/governance/p7-8.html. [Accessed: 18-Feb-2018]