Chalk Robot

Design Review
ECE 445: Senior Design
Team #12

Neil Christanto, Enyu Luo, Leonard Lim
TA: Mustafa Mukadam

University of Illinois, Urbana-Champaign
February 28, 2013

Table of Contents

1. Introduction

1.1 Title - Chalk Robot

This is an idea proposed by MIT Lincoln Lab. Currently the only way to draw pictures or words on the floor or sidewalk is to actually draw it by hand. There are a few disadvantages to this method. First of all, it is time consuming and back breaking for a person to crouch down and chalk the floor. Then the end product is often depends largely on the skill and artistic flair of the individual drawing the image. It is also difficult for the average person to replicate images or logos with a great deal of accuracy. The chalk robot aims to automate this process, making chalking the sidewalk a much simpler task.

1.2 Objectives

Goals:

The goal of this project is to develop a robot system that can draw an image on the ground with chalk. This will be a 2-wheel mobile robot with a chalk mechanism. The robot will be able to take in an image file from a USB stick, move accordingly on the floor to draw the image outline.

Features:

- Position tracking relative to the starting point
- Bidirectional motor control allowing for forward, backward motion, turning-on-the-spot
- Control of the chalk mechanism (armature and automatic dispensing)
- Drawing outline of image file
- Up to 2m x 2m range of drawing

Benefits:

- Users can use our robot to easily print chalk advertisement on the sidewalk
- Image outline can be chalked with high level of similarity

2. Design

2.1 Block Diagram
(lower level diagram explained in Block Description)

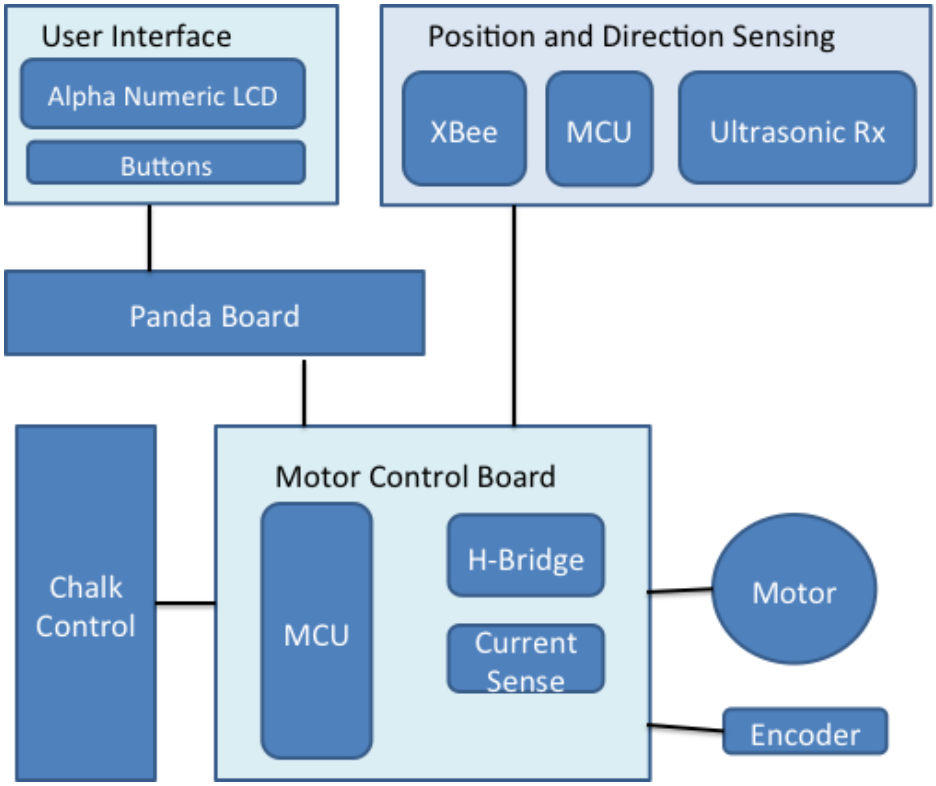


Figure 2.1: Overall robot block diagram

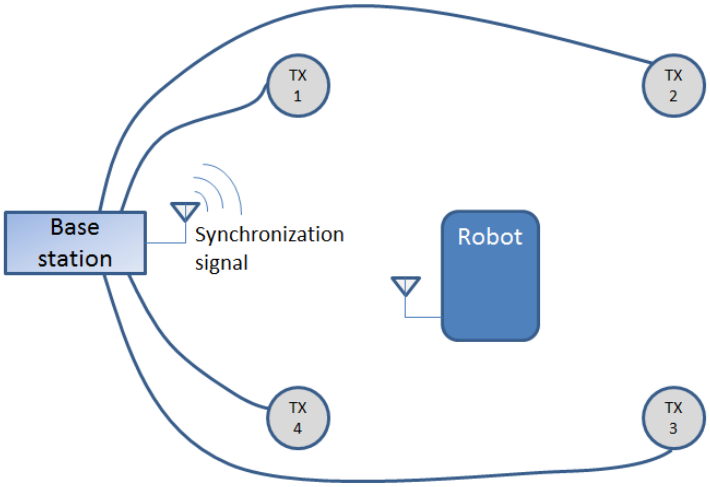


Figure 2.2: Position sensing beacons block diagram

2.2 Block Description

Motor Control Board

This controls the power to the motors to move the robot to the x-y location desired from the panda board. It consists of a microcontroller which sends PWM to the H-bridge driver which determines the voltage applied to the motor. Current sensing is done for either current control or just current limit. Feedback from encoders is used for velocity control of both motors. The microcontroller also receives position and direction information and compensates for any error from the encoders.

1. *H-Bridge*

   The H-Bridge is the driving circuit to the motors. An I.C. LM18200T is used here and can handle 3A continuous current, which is more than what the motors will draw. It contains current sensing and temperature flag integrated into the IC. The H-Bridge draws current from a separate battery that supplies 22.2V.

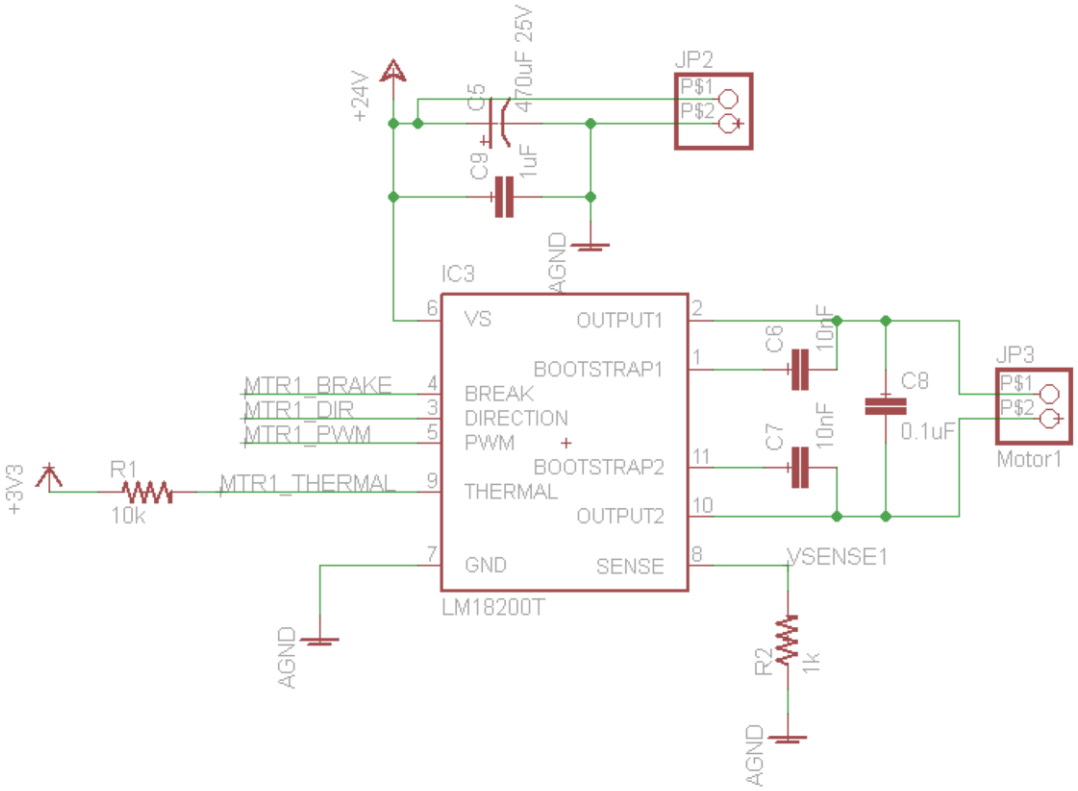


Figure 2.3: Schematic of H-Bridge LM18200T

2. *Current Sensing*

   Current Sensing is done via the current sense pin, which drives 377μA/A current output. A 1kΩ resistor is connected to ground to produce 377mV/A voltage output. Since motor commutation introduces noise to the current sense, analog low pass filter is used before the ADC. ADC samples at 24kHz, thus filter cutoff should be less than 12kHz. A 4th order active LPF with cutoff 6.6kHz is used, which also acts as a buffer to the ADC.

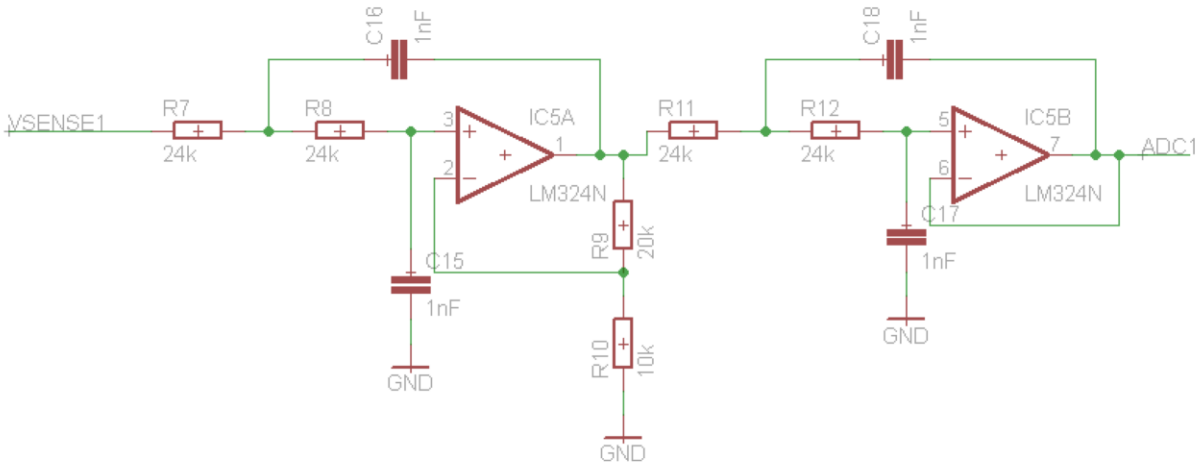


Figure 2.4: Schematic of 4th order low pass filter with cutoff at 6.6kHz

3. Microcontroller

TI Stellaris LM3S9D92 is used. It has 2 Quadrature Encoder Index modules, PWM outputs and ADC channels which is suitable for motor control application. It has a 32bit ARM Cortex M3 core which runs at a maximum of 80MHz.

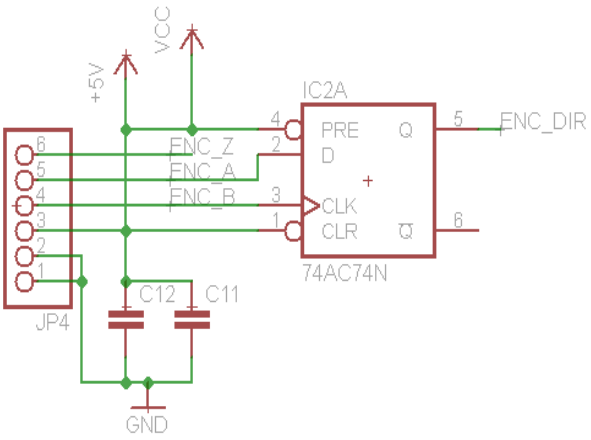


Figure 2.5: Schematic of Encoder direction sensing

The microcontroller program will implement the control algorithm as shown in the figure below. The inner loop is the current controller that controls the torque output of the motor, and has friction compensation to it. The outer control loops are the PID controls for velocity and position. Current control loop will run at the PWM frequency. Velocity control loop gets velocity from the encoder and runs every 5ms. Position control loop runs at every 50ms since that is the rate that the position sensor provides information.

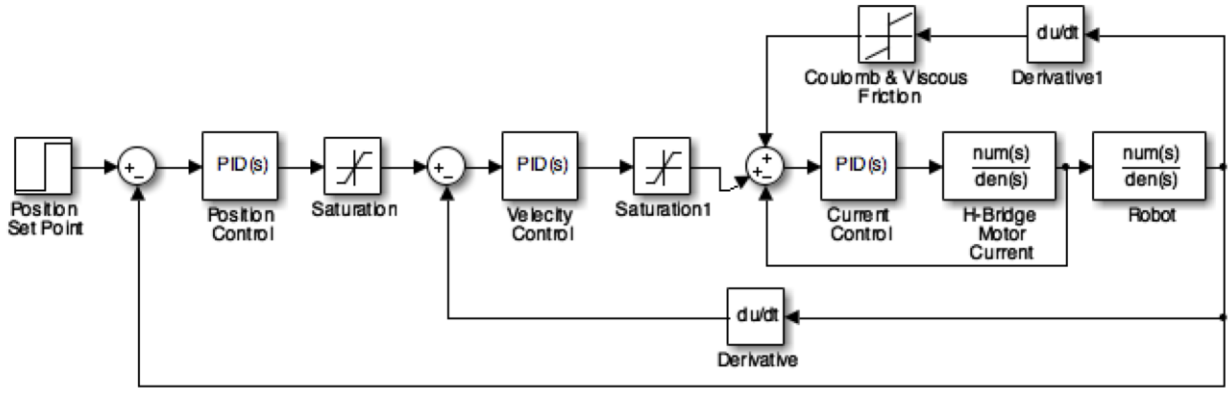


Figure 2.6: Control algorithm to be implemented by the microcontroller

4. Voltage Regulators

This block supplies +3.3V to the digital circuits of the motor control board. There is also a +5V output for the op-amp of the motor control board and also to supply power to the Pandaboard. Both regulators use LM2576 switching regulator IC. The input voltage is 11.1V from the battery.

- 5V regulator – configured to supply 2A to panda board. Catch diode 1N5820 is used as it is a fast switching Schottky diode with 3A and 20Vrrm rating.
    - Inductor selection: $ET = (Vin - Vout)\frac{Vout}{Vin}\frac{1000}{52kHz} =$ 56.1, from the datasheet, for 2A I_{out}, inductor value of 100µH should be used.
    - Capacitor selection: $Cout \geq 13300\frac{Vin}{Vout \times L} = 319\mu F$. For a smaller output ripple, use higher capacitance. For electrolytic capacitors, higher capacitance has lower ESR, which is good. 2200µf, 10V electrolytic capacitor is chosen.
- 3.3V regulator – configured to supply 0.5A for the digital logic of motor control board. Catch diode 1N5817 is used. It is the 1A equivalent of the 1N5820.
    - Using same calculations as above, inductor of 330µH and capacitor of 2200µF, 6.3V electrolytic is chosen.

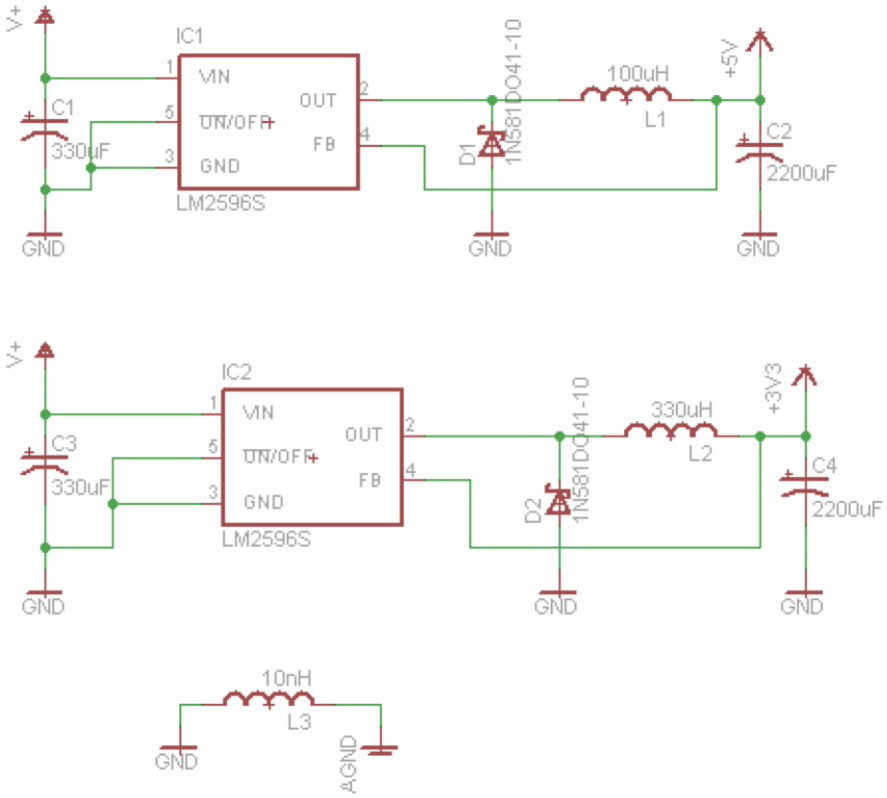


Figure 2.7: Voltage Regulators

Chalk Control

The chalk control mechanism consists of a first-class-lever (a pivot in the middle, load on one end, and force applied to the other end) and a chalk gripper on the load end.

1. *Normal Operation*
   The force applied to the lever is achieved by the operation of a servo. When we do not want to write, the servo will push the lever up until the chalk is about one inch off the ground. When we want to write, the servo will let go of the arm and allow gravity to push the arm downward.

2. *Reloading Mechanism*
   We will have a contact switch that will be activated when the arm is too low, indicating that the chalk is too short. Once the switch is activated, the gripper will let go of the short chalk and the arm will be adjusted to the desired writing position. A new chalk will then be dispensed and the gripper grips the new chalk. Normal operation then continues. This whole operation will be controlled by the microcontroller in the Motor Control Board since the chalk mechanism needs to coordinate with the robot movement.

Position Sensing

The job of the position sensing is to tell the robot its position relative to the placement of the transmitters. The goal is to send the position (and direction) every 50ms. The time-of-flight approach is going to be used heavily in the distance calculations, so every delay produced by the module itself (microcontroller IO delay for example) needs to be taken into consideration.

1. *Base Station*
   The base station's job is to do the synchronization for all the transmitters and the ultrasonic receiver module. The main components consist of a microcontroller and an Xbee. An interrupt will trigger periodically to send a reference pulse to the RX module via Xbee. When the reference pulse is sent, four timers with different expiration will start (each timer corresponds to each transmitter). There are four outputs from the microcontroller corresponding to the four timers, and those outputs are hooked to an interrupt input at each transmitter's microcontroller. When each timer expires, it means that it is time for a particular transmitter to send a pulse.

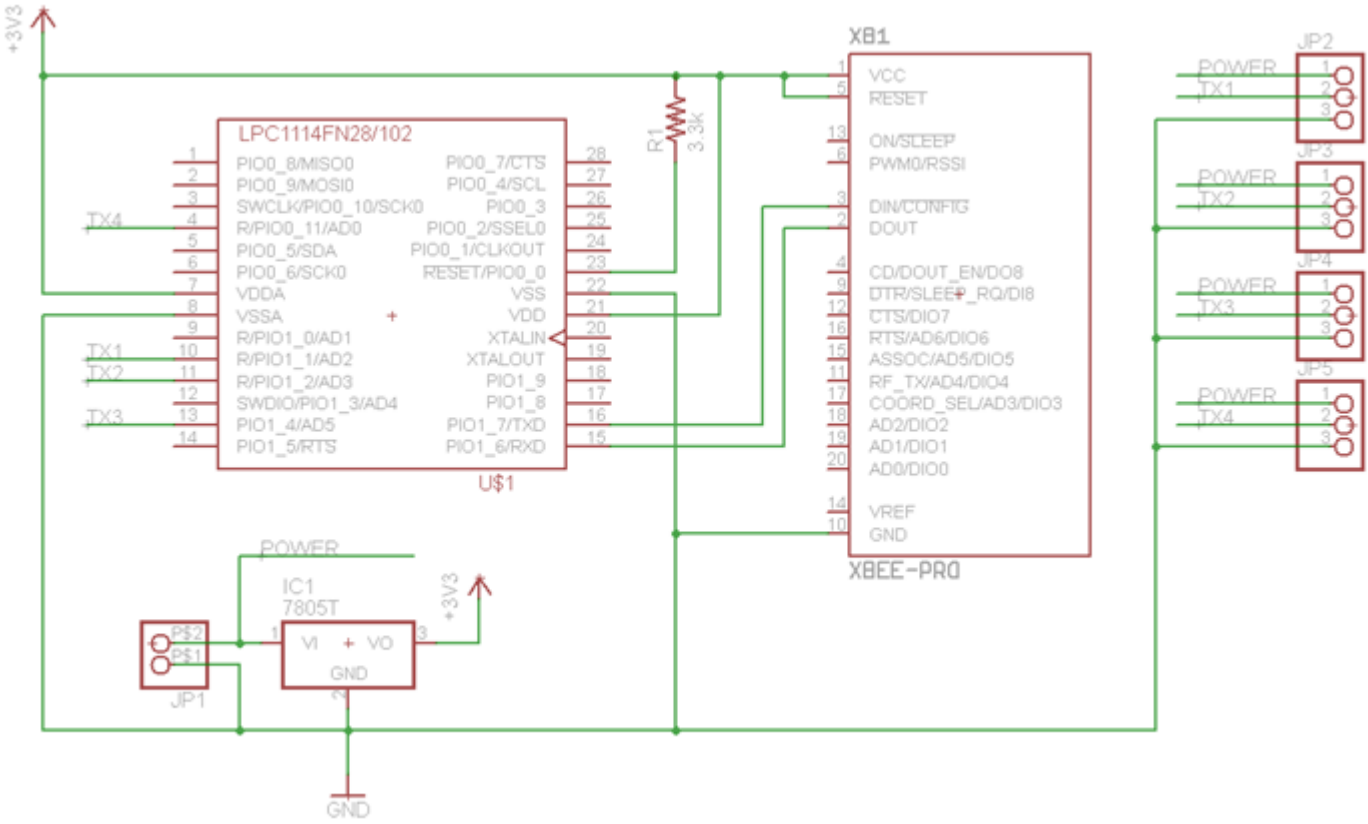


Figure 2.8: Schematic of the base station (microcontroller and Xbee)

The four timers are configured to have different expiration, but the time between consecutive timers is constant. It is configured this way so that the receiver can determine which timer is sending the pulse. That being said, there is a limit of how short the interval between two consecutive pulses can be. The following equations are used to determine the lower limit of the interval between two consecutive pulses.

$$\text{Speed of sound } (\text{m.s}^{-1}) = 331.3 \times \sqrt{1 + \frac{T}{273.15}};$$

T is the surrounding temperature in Celsius.

But for the following equations, we are using the worst case (slowest) speed of sound (at sea level) which is 340m/s. The transmitters are put forming a square of 2.4m × 2.4m because the beam angle is only 80 degree wide (it is configured this way so that the transmitters can cover an area of 2m by 2m). So the worst case travel distance for the ultrasound is from the transmitter in one corner to the other corner where the robot is the

furthest. Then the worst case travel time is the worst case distance divided by the worst case speed of sound.

$$\text{Worst case travel time} = \frac{0.2\sqrt{2} + 2\sqrt{2}}{340} = 9.16\text{ms}$$

So we assign a 10ms window for each transmitter to send a pulse. In that window, only one assigned transmitter will send a pulse. Since there are four transmitters, the time required for all transmitters to send a pulse is 40ms. Then we can assign a 10ms time for the receiver module to process all the given data, so the total time allocated for the entire process, from the first transmitter sending a pulse to the receiver module finishes processing, is 50ms. Therefore, the reference pulse can be sent every 50ms.

2. *Ultrasonic Transmitters*

   The transmitters will be connected to the base station using a long wire. They just wait for an interrupt triggered by a signal from the base station that indicates when to start sending pulse. To deliver a pulse with the strongest power, it has to be a 40kHz signal with 20V peak-to-peak. To realize this, each transmitter has a microcontroller to output a square wave generated by the PWM module. Since the microcontroller can only support a digital logic output between 0V and 3.3V, an H-bridge is used, supplied at 10V to enable a switching between +10V and -10V at the output of the bridge, producing a 20V peak-to-peak wave.

   The square wave is sent only for a limited time to emulate a pulse. In this case, the microcontroller will only send 5 periods of square wave to the transmitter. So beside the PWM, the microcontroller also make use of another timer to count up to what is equivalent to 5 periods of 40kHz square wave, and after that timer expires, the microcontroller stops sending the wave.

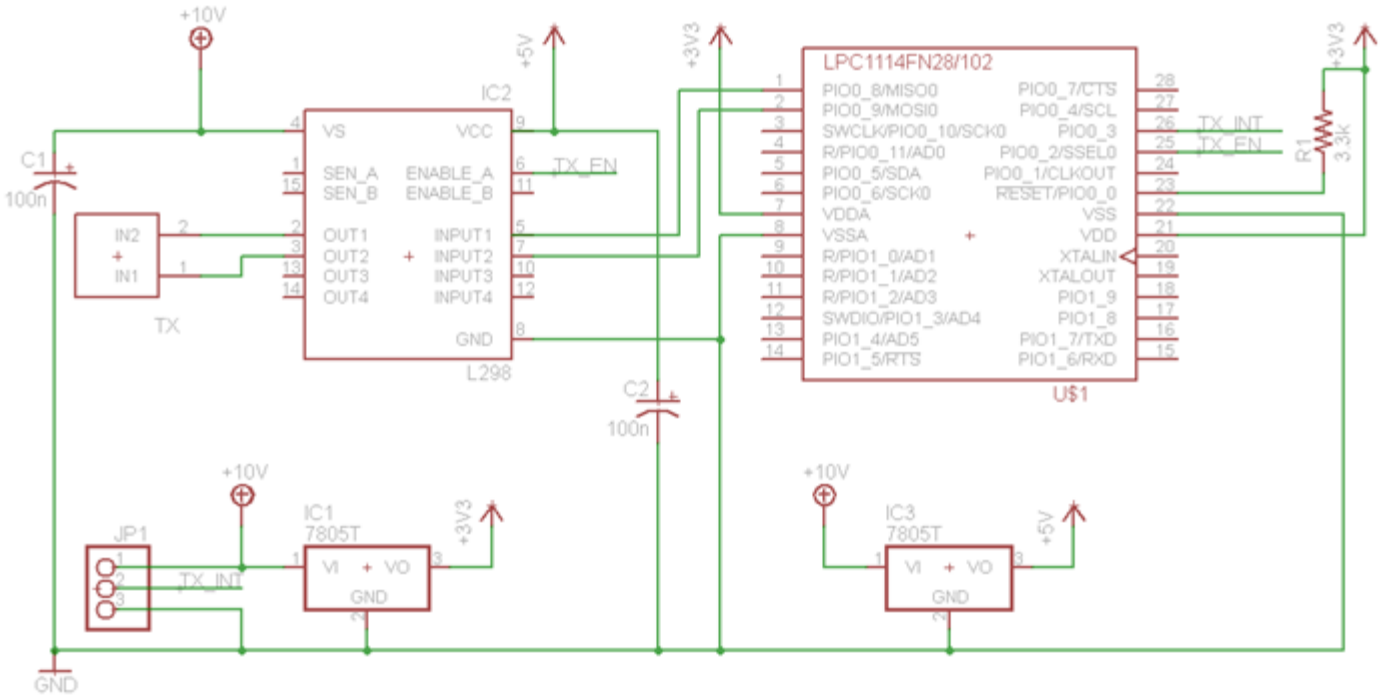


Figure 2.9: Schematic of the transmitter

3. *Ultrasonic Receiver Module*

   This module has the toughest job in the position sensing mechanism. The main components consist of a microcontroller, an Xbee, and two ultrasonic receivers. For the basic position sensing, we just need one ultrasonic receiver. But we're doing two receivers so that we can also do direction sensing. The algorithm to do direction sensing will be explained in the section "direction sensing." The arrangement of the two receivers is shown below.

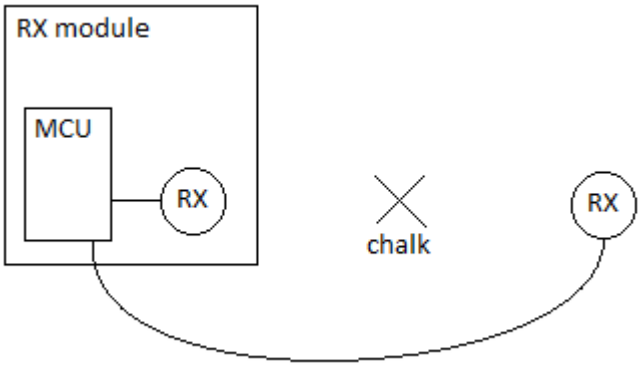
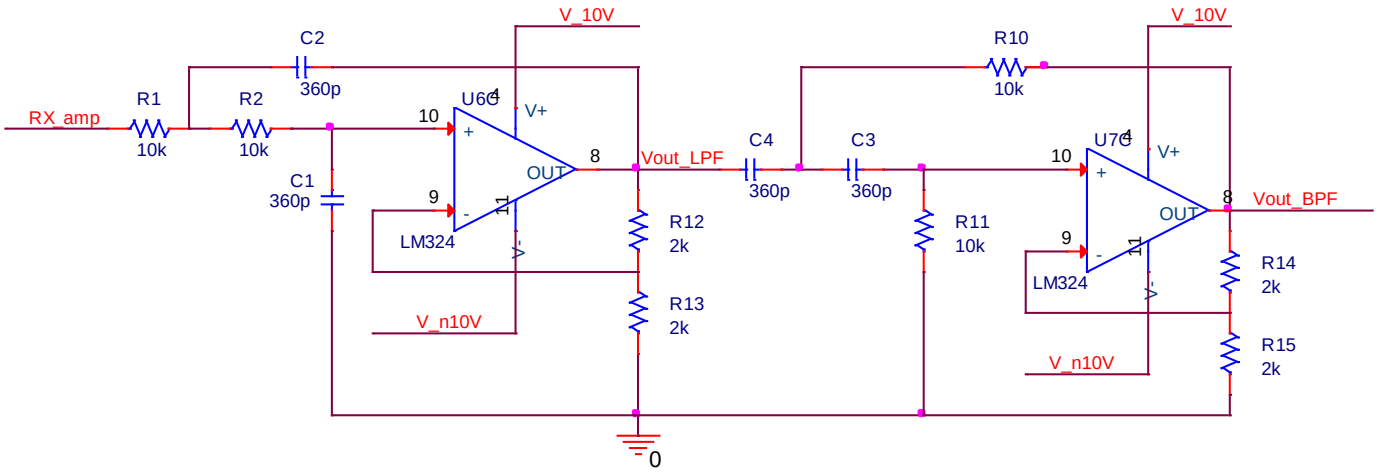


Figure 2.10: Chalk and receivers arrangement

The receiver is most sensitive at 40kHz, so a second-order band pass filter with a gain of four is made by cascading a second-order low pass filter and a second-order high pass filter with cutoff frequency at 40kHz, each with a gain of two. The schematic is as follow



RX_amp = input from the RX, amplified
V_10V = +10V supply
V_n10V = -10V supply
Vout_BPF = Band pass filter output, connected to another amplifier

Figure 2.11: Schematic of 2nd order band pass filter

According to figure 2.12, right after the receiver, there is a coupling capacitor to block DC offset that the receiver might introduce. Then there are two high gain amplification stages, before and after the bandpass filter. The total gain of the two amplifiers is chosen to be high enough so that in the very end, a very weak signal will be at least 1V peak-to-peak (500mV amplitude), and the noise is less than 500mV peak-to-peak (250mV amplitude).

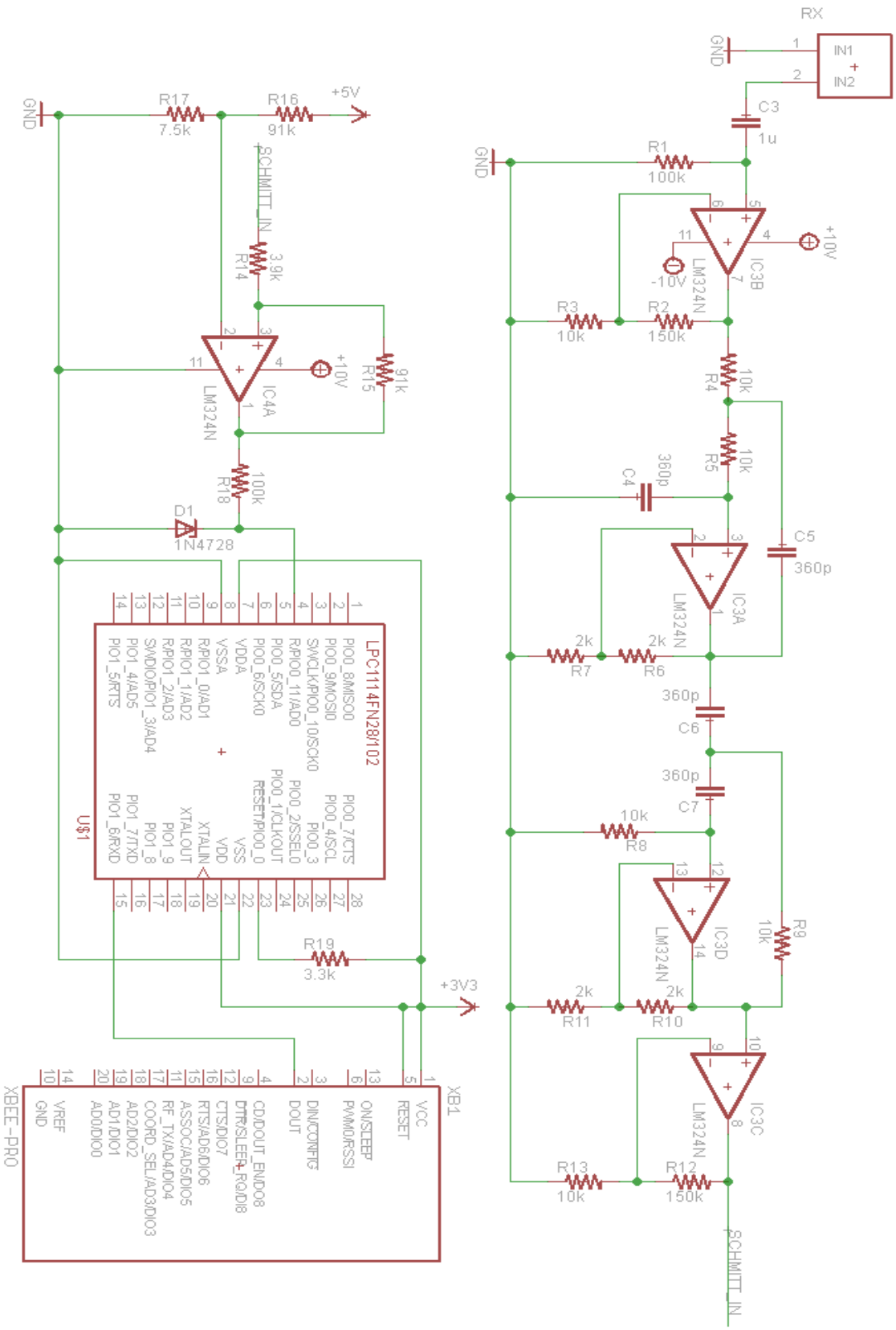


Figure 2.12: Schematic of the whole receiver module

The last stage is a Schmitt trigger that has a band of 400mV with 400mV upper threshold and 0V lower threshold. The op-amp is supplied at 10V for Vcc and ground for -Vcc so that the output range is between 0V and 10V (less than 10V actually because of the saturation). The following is the calculation to determine the resistor values needed to implement such Schmitt trigger.

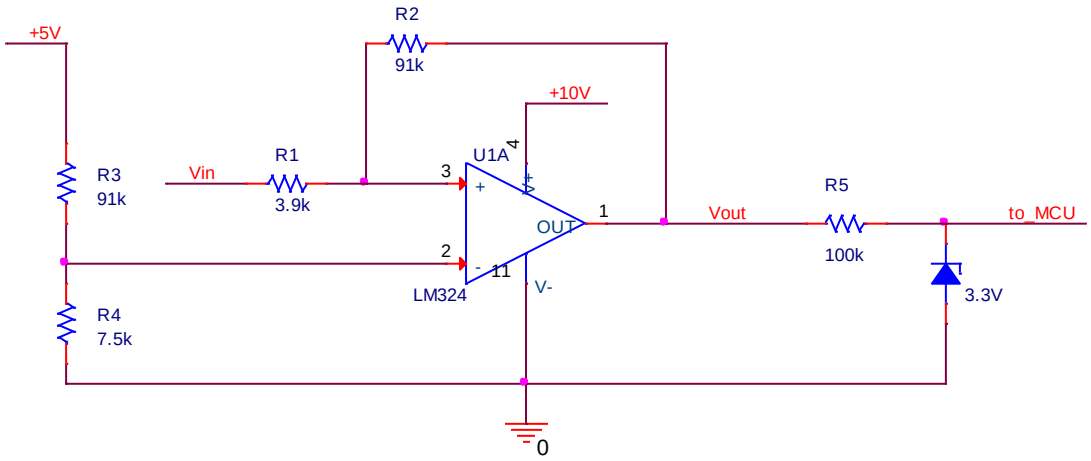


Figure 2.13: Schmitt trigger schematics
upper threshold = 400mV; lower threshold = 0V

Range = 0.4V

At V_{out} = 0V:

$$V^+ = \frac{R_2}{R_1+R_2} V_{in} > V^- \text{ to make } V_{out} \text{ goes high.}$$

$$V_{in} > \left(1 + \frac{R_1}{R_2}\right) V^-$$

At $V_{out} = V_{max}$ (saturation)

$$V^+ = \frac{R_2}{R_1+R_2} V_{in} + \frac{R_1}{R_1+R_2} V_{max} < V^- \text{ to make } V_{out} \text{ goes low.}$$

$$V_{in} < \left(1 + \frac{R_1}{R_2}\right) V^- - \frac{R_1}{R_2} V_{max}$$

We found out that the highest the output can achieve is 9V, so V_{max} is 9V.

The range becomes $\frac{R_1}{R_2} \times 9 = 0.4$ so $\frac{R_1}{R_2} = 0.044$. We choose R_1 to be 3.9kΩ and R_2 to be 91kΩ.

We want the upper threshold of the Schmitt trigger to be 400mV; if V_{in} is greater than 400mV, we want V_{out} to go high. That means $V^- = \frac{0.4}{1+ {}^{3.9}/_{91}} = 0.38V$.

To get 0.38V, we use a voltage divider. We choose R_3 to be 91kΩ and R_4 to be 7.5kΩ.

Finally the output of the Schmitt trigger is regulated by a 3.3V Zener diode so that the final output that is hooked to the input of the microcontroller to be processed is either 0V or 3.3V.

After the reference pulse is received by the Xbee, a timer in the microcontroller will start counting. The microcontroller expects four incoming pulses before it receives the next reference pulse. Since we know the time-of-flight of the pulses and the order of which the beacons transmit the pulse, we can determine the position of the robot relative to each beacon. The speed of sound depends on the surrounding temperature, but variation is minimal.

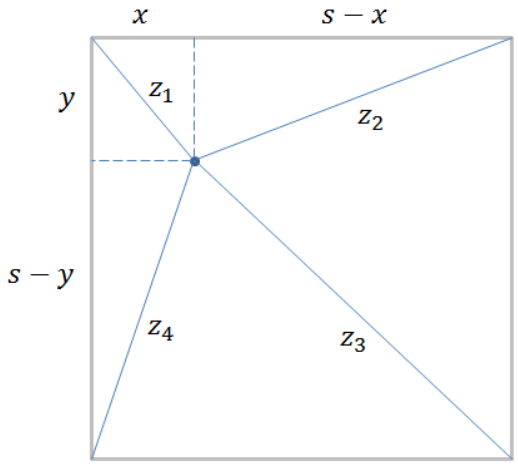


Figure 2.14: Robot position inside the square

$$x^2 + y^2 = {z_1}^2 \qquad \text{eq1}$$

$$(s-x)^2 + y^2 = {z_2}^2 \rightarrow s^2 - 2sx + s^2 + y^2 = {z_2}^2 \qquad \text{eq2}$$

$$x^2 + (s-y)^2 = {z_4}^2 \rightarrow x^2 + s^2 - 2sx + y^2 = {z_4}^2 \qquad \text{eq3}$$

$$(s-x)^2 + (s-y)^2 = {z_3}^2 \rightarrow 2s^2 - 2sx - 2sy + x^2 + y^2 = {z_3}^2 \qquad \text{eq4}$$

Subtracting eq1 from eq2 gives us $\quad {z_2}^2 - {z_1}^2 = s^2 - 2sx \rightarrow x = \frac{{z_1}^2 - {z_2}^2 + s^2}{2s}$

Subtracting eq1 from eq3 gives us $\quad {z_4}^2 - {z_1}^2 = s^2 - 2sy \rightarrow y = \frac{{z_1}^2 - {z_4}^2 + s^2}{2s}$

Subtracting eq3 from eq4 gives us $\quad {z_3}^2 - {z_4}^2 = s^2 - 2sx \rightarrow x = \frac{{z_4}^2 - {z_3}^2 + s^2}{2s}$

Subtracting eq2 from eq4 gives us $\quad {z_3}^2 - {z_2}^2 = s^2 - 2sx \rightarrow y = \frac{{z_2}^2 - {z_3}^2 + s^2}{2s}$

The previous equations shows that two x and y values can be calculated in one period of the pulses transmission. The average value can be taken and the result would certainly be more reliable. This can also reduce the error since the two values that we get are from opposite transmitters (note that the first x coordinate depends on $z_1$and $z_2$while the second x coordinate depends on z_3 and z_4, and the pair of transmitters are opposite from each other). Hence, even when the speed of sound varies and the two readings of each x and y coordinate will be further apart, the average will not vary that much.

Direction Sensing

This block's job is only to give information about where the robot is facing. It is actually integrated within the position sensing block since it is using the same ultrasonic receivers in the position sensing block, and the performance of this block relies on the accuracy of the position sensing mechanism.

After the positions of the two ultrasonic receivers are known, the direction of the robot can be determined using a simple arc tangent formula. This is possible only if we know which sensor is placed where (to the left or to the right of the robot), which can be done by simply assigning the receivers to particular input pins on the MCU. The DSP based approximation for arc tangent is used to make the computation faster. The error for this approximation is also really small, which is 0.01 radian, or equivalent to 0.7 degree [2].

Pseudo code:

```
float arctan2(float y, float x)
{
    coeff_1 = pi/4;
    coeff_2 = 3*coeff_1;
    abs_y = fabs(y)+1e-10   // kludge to prevent 0/0 condition

    if (x>=0)
    {
        r = (x - abs_y) / (x + abs_y);
        angle = 0.1963 * r^3 - 0.9817 * r + pi/4;
    }
    else
    {
        r = (x + abs_y) / (abs_y - x);
        angle = 0.1963 * r^3 - 0.9817 * r + 3*pi/4;
    }
    if (y < 0)
        return(-angle);     // negate if in quad III or IV
    else
        return(angle);
}
```

Finally, the data can be sent to the MCU in the motor control board together with the position of the chalk. Therefore, every time the receiver module communicates with the motor control board, it gives both position of the chalk and the direction of the robot. And since the UART pins of this MCU is already used to do the RF communication via Xbee, this MCU will communicate with the motor control board's MCU using I^2C protocol.

Image processing

The image processing for our robot is done using programs installed in a Pandaboard running Ubuntu operating system. The Pandaboard will be powered via the 5V output from the voltage regulator described previously. The input to the image processing block is a simple image in bitmap format loaded into the Pandaboard by a USB flash drive. By going through a few steps outlined in figure 2.13 below, the image processing would produce a list of ordered vectors to be sent to the motor control board for use.

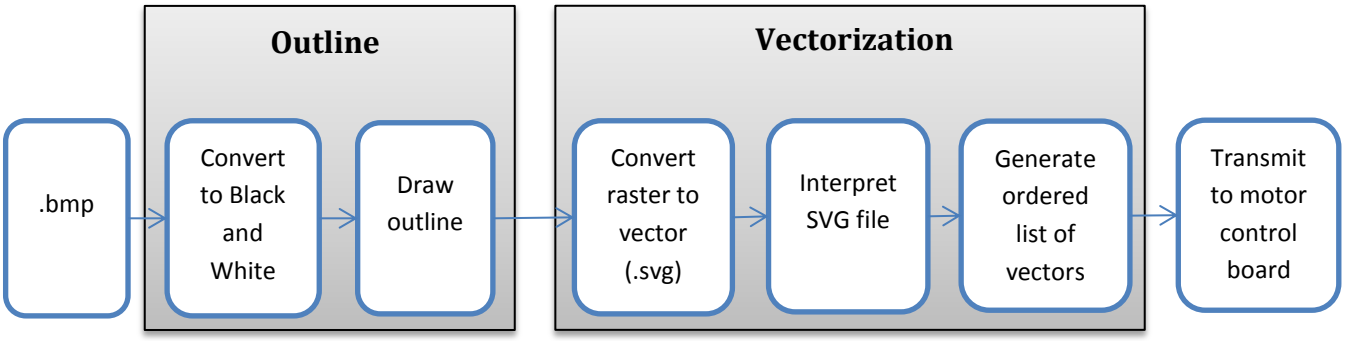


Figure 2.15: Flowchart for Image Processing

1. *Outline*

   The simple bitmap image is first read and then processed in its raster form. Firstly, we would convert the color image into a grayscale image by averaging their R, G and B channels to obtain a single grayscale value between 0 and 255 (0 corresponds to black and 255 corresponds to white). Then, we would use K-means to determine the threshold to which we would split the grayscale pixels into black and white. The K-means flowchart is shown in Figure 2.16. With the black and white image, we would then proceed to generate the outline by comparing each pixel to its neighbor. An outline would then be formed on the edge of the black-white space. The outlining flowchart is given in Figure 2.17.

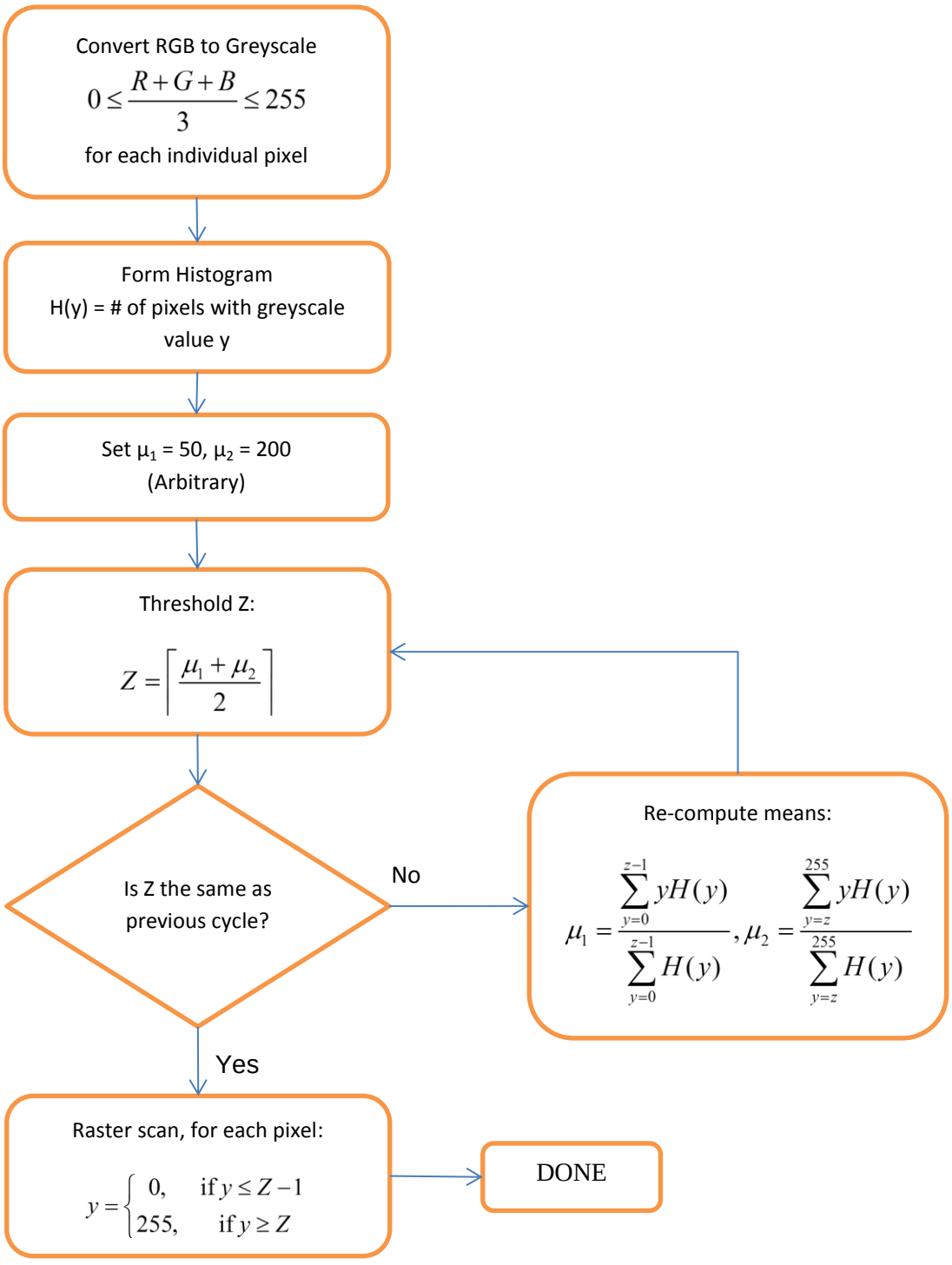


Figure 2.16:Flowchart for K-means thresholding algorithm

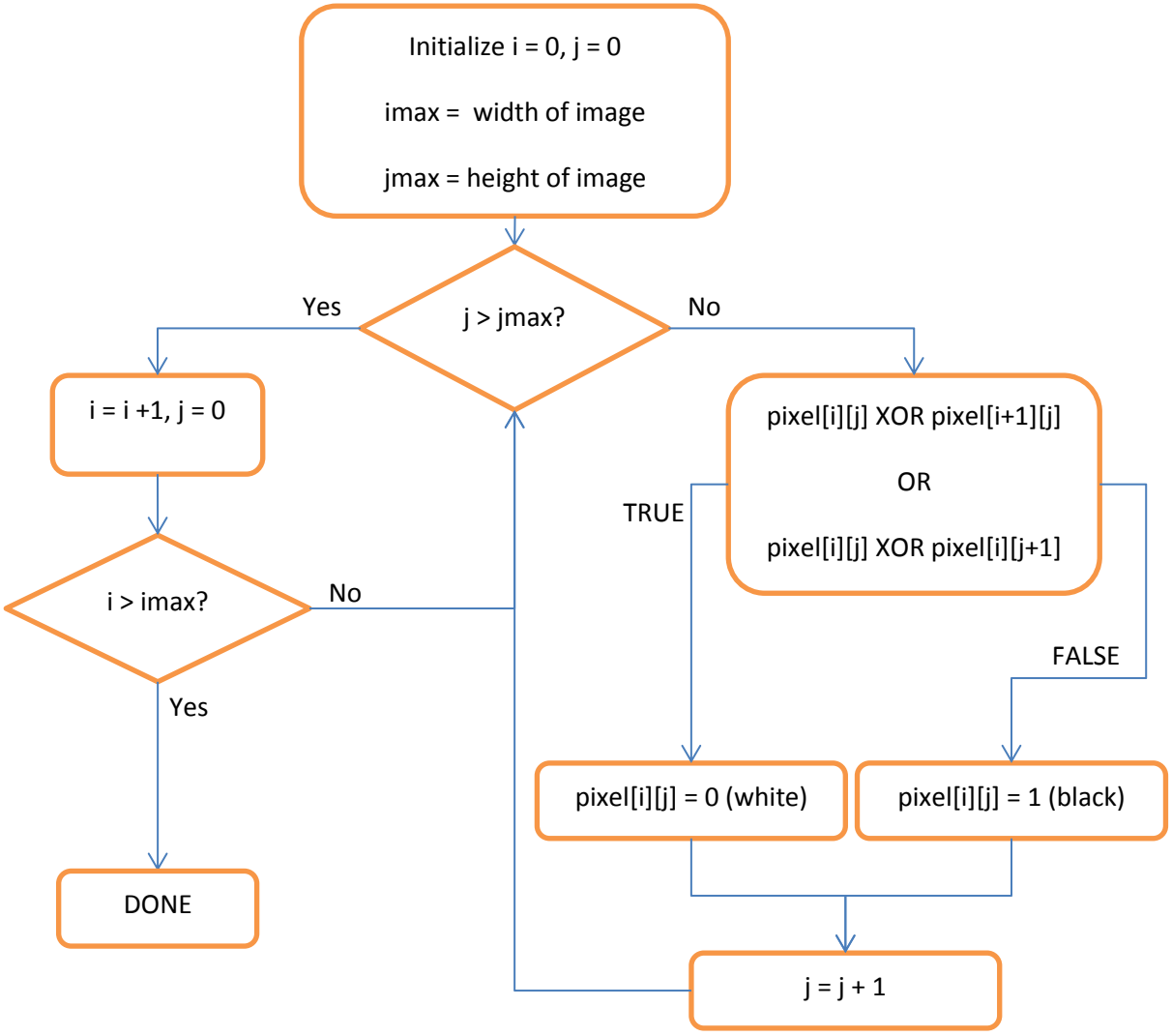


Figure 2.17: Flowchart for Outline algorithm

2. *Vectorization*

   The raster black outline image would then be put into another program, where it would be converted from a bitmap to a Scalable Vector Graphics (SVG) format. We would use a free online software to accomplish this. The individual strokes and vectors in the .svg file are then read and interpreted. (Multiple vectors may make up a stroke) The strokes will then be reordered to minimize the distance between the end of one stroke to the start of the next.

3. *Interaction with Motor Control Board*

   Pandaboard will communicate with the microcontroller on the motor control board via UART. The vectors will be sent from the pandaboard to the microcontroller grouped based on their strokes. Only one stroke is sent per cycle. Handshaking will be used to determine if the microcontroller is done with the previous stroke and ready to receive the next stroke.

User Interface

The user interface consists of a simple alpha numeric LCD screen and a USB numeric keypad. This unit will be connected to the Pandaboard and will aid the user in determining the image to draw (in case we have multiple images in the USB flash drive).

3. Requirements and Verification

3.1 Requirements and Verifications

Motor Control

Requirement	Verification
H-Bridge	
1. Transistors in the H-bridge turn on with a supply voltage between 12V to 24V.	1. Set the PWM pin to high and vary the supply voltage from 12V to 24V with a 50Ω. Use a digital multimeter to ensure that the output voltage is more than supplied voltage minus 2V.
2. Maximum steady state current of 2A per motor channel.	2. Set the PWM pin to high, and use 10Ω load across the output of the H-bridge. Increase the supply voltage until the voltage across the load reaches 20V. Use a digital multimeter to confirm this voltage drop.
3. Peak instantaneous current of 3A per motor channel.	3. Connect 5Ω load across the output of the H-bridge. Use a function generator to generate pulses of 16V for 100μs every 10ms (square wave of 100Hz with 1% duty cycle). Use the oscilloscope to measure the voltage across the load. The voltage reading should be 15±0.2V.
Encoder	
1. Encoder output channel A and B are 90±10 degrees out of phase from each other. For clockwise rotation, one channel should lead while for counter-clockwise rotation, the other channel should lead.	1. Connect both output channels to an oscilloscope. Rotate the encoder in a clockwise direction. Measure the phase difference between the two channels. Repeat for counter-clockwise rotation. Make sure the leading of the channel is correct according to the rotation of the encoder.
DC Motor	
1. Motor can turn in both directions.	1. Connect the motor input pins to a DC power supply. Output 10V from the power supply. The wheel connected to the shaft should rotate in one direction. Now switch the connection of Vcc and ground. The wheel should rotate in the opposite direction.

Control Algorithm	
1.a) Able to control current to within ±0.1A of desired current.	1.a) Use the microcontroller to output current at 0.3A, 0.5A, 0.8A, 1A, 1.3A, 1.5A and ensure that the steady state current is within ±0.1A with a 12Ω load
b) Rise time of the current control needs to be less than 5ms.	b) Using the microcontroller, alternate the current output between 0.3A and 1A every 10ms. Probe on the oscilloscope to make sure that the rise time is less than 5ms.
2.a) Able to control velocity from 0 to 512 encoder pulses per second.	2.a) Using microcontroller, set a fixed velocity to make the wheel rotate. Then by reading from the encoder, make the microcontroller output the instantaneous velocity every 10ms via UART (check using hyperterminal) or DAC (check using oscilloscope).
b) Velocity control within ±10% of desired velocity.	b) Using microcontroller, send fixed velocities of 150, 175, 200, 225, 250, 275, 300 pulses per second and measure the actual velocity. The measurement is done using the same method: read from encoder and give the output via UART or DAC.
3. Position Control to an error of ±1cm.	3. Use microcontroller to tell the robot to go to a certain position (preprogram the coordinate). The final position has to be within ±1cm of the exact coordinate. We can also tell the robot to move on a 0.5m radius circle and the chalk mark has to be within 1cm of the start and end point. Repeat for a 1x1m square path.

Chalk Control

Requirement	Verification
1. When writing, the part of the chalk touching the ground should be exactly in the middle of the two wheels.	1. Put some kind of mark on the wheels (ink is possible) so that they will leave mark as the robot moves. Send commands to draw some squiggly lines over 1 to 2 meters. The chalk line should be exactly in between the two ink line.
2. Servo that controls the chalk armature can be directed to 3 different positions (let go of the lever completely, lift the lever completely, and lift the lever to 1±0.2 inch off the ground).	2. Using function generator, output a PWM to control the servo, directed to the three different positions. When the lever is let go completely, the chalk should be in contact with the ground. When the lever is lifted all the way up, the last contact switch is activated. For the last command, measure the distance of the lever from the ground with a ruler. It should be at 1±0.2 inch off the ground. After the tests are completed, output a PWM to the servo using the microcontroller. Then repeat the same test.
3. During chalk dispensing mechanism, the robot should not move until the chalk has been dispensed appropriately. After that, the robot should continue drawing.	3. Make the robot start drawing with a very short chalk (short enough to make the lever just about to make contact with the switch). Send command to the robot to draw squiggly lines until the chalk needs to be replaced. At this point, the robot should stop and the chalk dispensing mechanism will start. Once it is ready to continue the drawing, the robot will start moving again.

Position Sensing

Requirement	Verification
Base Station	
1. Interrupt is triggered every 50±0.1ms (use MCU timer) to send a reference signal via Xbee.	1. The communication between MCU and Xbee is done using UART interface. Check the TXD pin of the MCU on the oscilloscope, make sure that it sends out data every 50±0.1 ms. According to the UART interface, TXD will stay high when there's no data to send, and the start bit of every data is 0. Therefore, check the very first zero after a long high signal on the oscilloscope, and make sure this happens every 50±0.1 ms.
2. The four timers have expiration at 2ms, 12ms, 22ms, and 32ms after the reference signal is sent (with ±0.01ms tolerance)	2. Program the microcontroller so that the timer expiration causes a toggle in the matched output pin of the MCU. Probe each matched output pin corresponding to each timer. Matched output 1 should toggle 2ms after the beginning of the UART transmission to the Xbee. Output 2 should toggle 10ms after output 1 toggles, output 3 should toggle 10ms after output 2 toggle, and output 4 should toggle 10ms after output 3 toggles.
3. After the last timer expires, none of the timers should expire again. The timers are restarted at the next reference pulse.	3. After matched output 4 toggles, nothing should happen. Probe in the scope for all of the MCU's matched output to make sure that they don't toggle again after output 4 toggles.

Ultrasonic Transmitter	
1.a) Generate a 40±1 kHz square wave right after a signal from the base station is received.	1.a) The PWM timer with 40kHz and 50% duty cycle should start right after the MCU receives a signal from the base station as an interrupt (take the toggling of the matched output from the base station as the source of interrupt). Probe the output of the MCU PWM on the oscilloscope and make sure that it is a square wave of 40kHz and 50% duty cycle.
b) The 40kHz square wave needs to be at most 20V.	b) Using 10V as a supply voltage, drive the H-bridge with the PWM output from the MCU and probe the output at the oscilloscope. The output should alternate between at most +10V and -10V, producing a 20V peak-to-peak voltage. Make sure the frequency is still 40±1 kHz.
2. Stop sending the wave after 5 periods.	2. Have a second timer that starts at the same time the PWM timer starts. Make this timer's expiration equivalent to 5 periods of 40kHz wave. Once this timer expires, disable the H-bridge. Use oscilloscope to probe the output of the H-bridge and make sure that the wave stops after 5 periods.
3. Transmitter works when driven by the MCU.	3. Hook the transmitter to the output of the H-bridge. Drive the transmitter with 40kHz 20V peak-to-peak square wave. Place the receiver very close to the transmitter and probe the receiver using oscilloscope. Make sure the receiver produces a noticeable steady sine wave at around 40kHz.

Receiver Module - Bandpass Filter	
1. Bandpass filter with a high gain centered at 40±1 kHz to filter out noise and amplified the received signal at 40kHz.	1. Make a signal generator produce a low voltage to simulate the receiver (around 20mV peak-to-peak) and probe the output of the filter on the oscilloscope. Vary the frequency from 20kHz to 60kHz, the output should be the strongest at around 40kHz.
2. After the bandpass filter, the signal that comes out needs to be at least 1V peak-to-peak.	2. Make the transmitter transmit a 20V peak-to-peak signal at 40kHz driven by the function generator and place the receiver far from the transmitter. To test the worst case, place them 4 meter apart, then probe the output of the bandpass filter. The output should be at least 1V peak-to-peak.
3. When there is no ultrasound transmission (the transmitters are not active), the output of the bandpass filter needs to be at most 500mV peak-to-peak.	3. Leave the receiver powered up with no transmitters transmitting anything. Probe the output of the bandpass filter on the oscilloscope, the noise needs to be at most 500mV peak-to-peak.
Receiver Module - Schmitt Trigger	
1.a) Weakest signal should produce an alternating GND to Vcc output. b) Noise should produce a GND output.	1.a) The weakest signal has at least 500mV amplitude, this should be enough to trigger a Vcc output. The lower threshold is 0V, so when the signal goes below zero the output will be driven to -Vcc supply of the op amp (we choose a single supply op amp, so -Vcc supply is connected to ground). Probe the output using oscilloscope. b) The output of the bandpass filter when there is nothing being transmitted has at most 250mV amplitude, so this should not make the output of the Schmitt trigger to go to Vcc; the ouput should stay at GND level. Probe the output using oscilloscope.
2. Input to the microcontroller needs to be in a range of -0.5V to +5.5V. Input is considered logic high if the voltage is at least 0.7Vdd and the input is considered logic low if the voltage is at most 0.3Vdd, with Vdd = 3.3V.	2. Put a zener diode with a break down voltage between 3.3V and 5.5V after the Schmitt trigger. Probe that terminal using an oscilloscope, the peak voltage should be between 2.31V (0.7Vdd) and 5.5V, the low signal should be between -0.5V to 0.99V (0.3Vdd).

Receiver Module - MCU	
1. Have one timer that starts after the MCU received the reference signal from the base station via Xbee. This timer should be able to count up to 50ms equivalent amount of time to avoid timer overflow. The time-of-flight calculation will be based on this timer.	1. Taking the fact that the clock frequency of the MCU is 48MHz, a 32-bit timer will be able to count up to 50ms equivalent amount of time. The signal from the Xbee is received by the MCU via UART, so start the timer at the UART interrupt handler. For testing, program the timer to toggle a particular output pin (pick anything) right after it starts. Probe this output pin and the RXD pin of the MCU, the start bit of the UART data and the toggling of the output pin should line up.
2.a) Trigger an interrupt every time there is a pulse coming in. The interrupt is triggered only once per pulse. b) The interrupt needs to be triggered only once per pulse.	2.a) Connect the clamped output of the Schmitt trigger to a GPIO pin on the MCU and configure that pin to be a source of interrupt. The interrupt handler is invoked on the rising edge input of the GPIO pin. Make the interrupt handler toggle a particular output pin and probe this output pin on the oscilloscope. Every time there is an incoming pulse, this output should toggle. b) To make sure that the interrupt is only triggered once per pulse, have a timer that starts counting when the first interrupt is received, and make the expiration equivalent to 7 periods of a 40kHz (note that the transmitter sends 5 periods, we put extra 2 periods as a safeguard). This interrupt handler then disables the reception of the next interrupts. The interrupt will be re-enabled and the timer will stop counting when the timer expires. For testing, program the interrupt handler to toggle a particular output pin. Make the transmitter sends a steady signal and probe the assigned output pin using oscilloscope. It should toggle every 8 periods.

3. Calculate the position of the robot with a maximum error of .	3. Using oscilloscope, set the trigger to 'single'. Probe at the ultrasonic receiver and the GPIO pin on the MCU that is configured to be the interrupt source, then turn on the transmitter. Measure the delay between the first abrupt change at the receiver and the time when the voltage at the GPIO pin reach 2.31V. This delay should be taken into account when we calculate the time-of-flight. Using the time-of-flight information from each transmitter to calculate the x and y coordinate. The end result should be within 2cm of the actual measured position.

Direction Sensing

Requirement	**Verification**
1. Calculate the direction of the robot relative to the placement of all four beacons (give each beacon an ID number for example) within 10 degree. Full rotation is 360 degree.	1. We have two ultrasonic receivers with the chalk in the middle. Given the reading from the two receivers, determine the angle using trigonometric (arc tangent) knowing which receiver is which (give each receiver an ID number as well). To calculate the arc tangent fast, use a DSP based approximation for the arc tangent implementation. Move the robot around and see the direction this function outputs. The result should be within 10 degree of the actual result (measured with a real compass).

Image Processing

Requirement	Verification
Outlining	
1. Black and white image generated from the thresholding should retain the general shape of the original image.	1. Using a simple test image (basic shapes), compare the print out of the black and white image to a print out of the original image. This is going to be a subjective judgment, but at least the outline should be noticeably matched.
2. Outline should trace the thresholded black and white image.	2. Using a simple test image, compare the print out of the outline to the black and white image. Again, this is going to be a subjective judgment, but the outline should trace the black and white image well. Ensure there are no bumps on the outline.
Vectorization	
1. Vectored image should look like the outline.	1. Using a simple test image, compare the vectorized outline with the raster outline. For a simple square or circle, the vectorized outline should match exactly with the raster outline.
Interface with Motor Control Board	
1. Pandaboard receives signal from microcontroller correctly.	1. Manually set the microcontroller to 'busy' and then 'ready'. Check via software that Pandaboard receives the signal of this handshaking mechanism.
2. Program successfully sends vectors to microcontroller	2. Send a stroke to the microcontroller. The Pandaboard should shortly receive the handshaking signal and display on its LCD.

User Interface

Requirement	Verification
LCD	
1. LCD connected properly to Pandaboard	1. When the Pandaboard is on, LCD will draw power from the Pandaboard and displays appropriate status.

2. LCD can display multiple files presented to the Pandaboard.	2. Have multiple files stored in a USB flash drive and plug it to the Pandaboard (the LCD is already connected to the Pandaboard). The LCD should then display at least two files at once (depends on the size of the LCD screen).
Numeric Keypad	
1. Numeric Keypad connected properly to Pandaboard	1. With the keypad connected to the Pandaboard via USB, toggle the 'Num Lock' key and the indicator light should toggle on and off.
2. Numeric Keypad can provide correct input to Pandaboard.	2. Connect the Pandaboard to a regular screen and open a word processor application. With both 'Num Lock' on and off, check that the inputs to the word processor application are consistent to what is being pressed.
3. When integrated with the LCD, the numeric keypad can scroll through files.	3. Connect a USB flash drive with multiple files. With 'Num Lock' off, scroll up and down with the arrow keys. The LCD should display the current selected file name and change as we scroll with the keypad.

3.2 Tolerance Analysis

The most sensitive part of the entire system is the accuracy of the robot position. We will have ultrasonic transmitters put on fixed locations, and the robot will have an ultrasound receiver. According to the block descriptions above, the overall setup itself will be a challenge: to put the beacons accurately. Another difficult task is to make all the transmitter-receiver and wireless communication part operate in a total synchronous fashion. All these will affect the accuracy of the robot position.

Rigorous testing needs to be done to ensure that the position is within ±5mm of the exact known position. The general test is to mark the initial position of the robot, instruct the robot to go over a long distance and come back to the initial position. The final position of the robot should be within the error tolerance relative to the initial position. To test corner cases, deliberately send commands to the microcontroller to make the robot moves around (close to) the border of operation.

4. Simulation Results

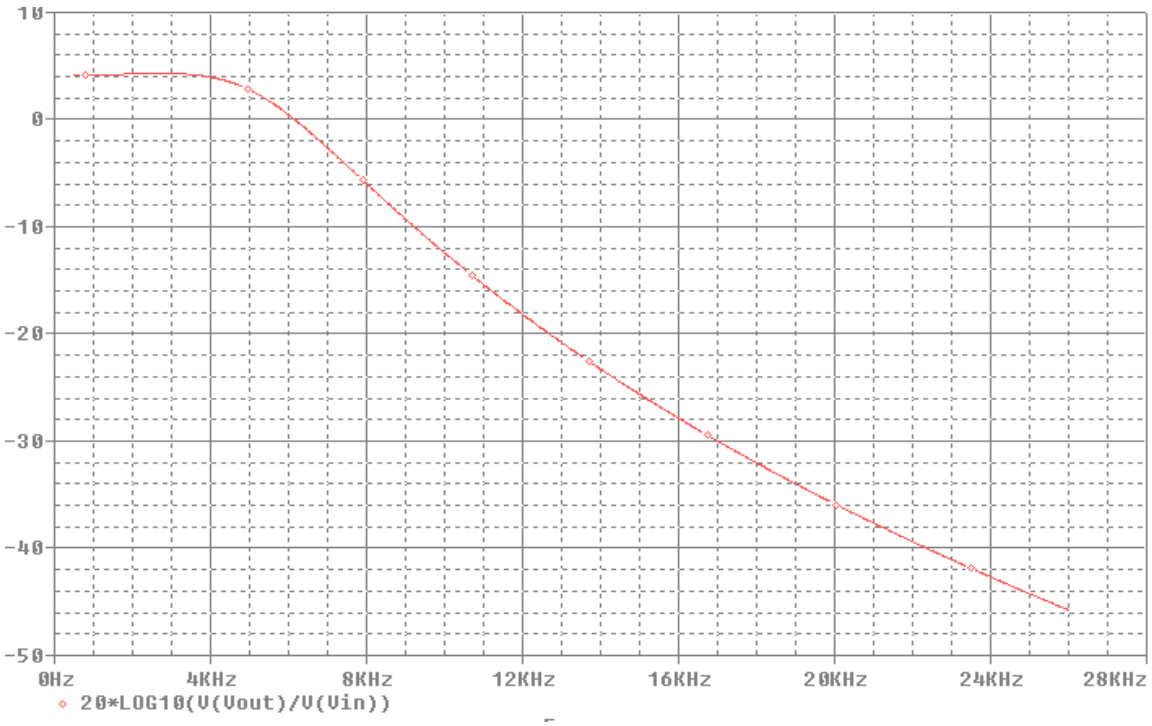


Figure 4.1: Low pas filter frequency response for current sensing

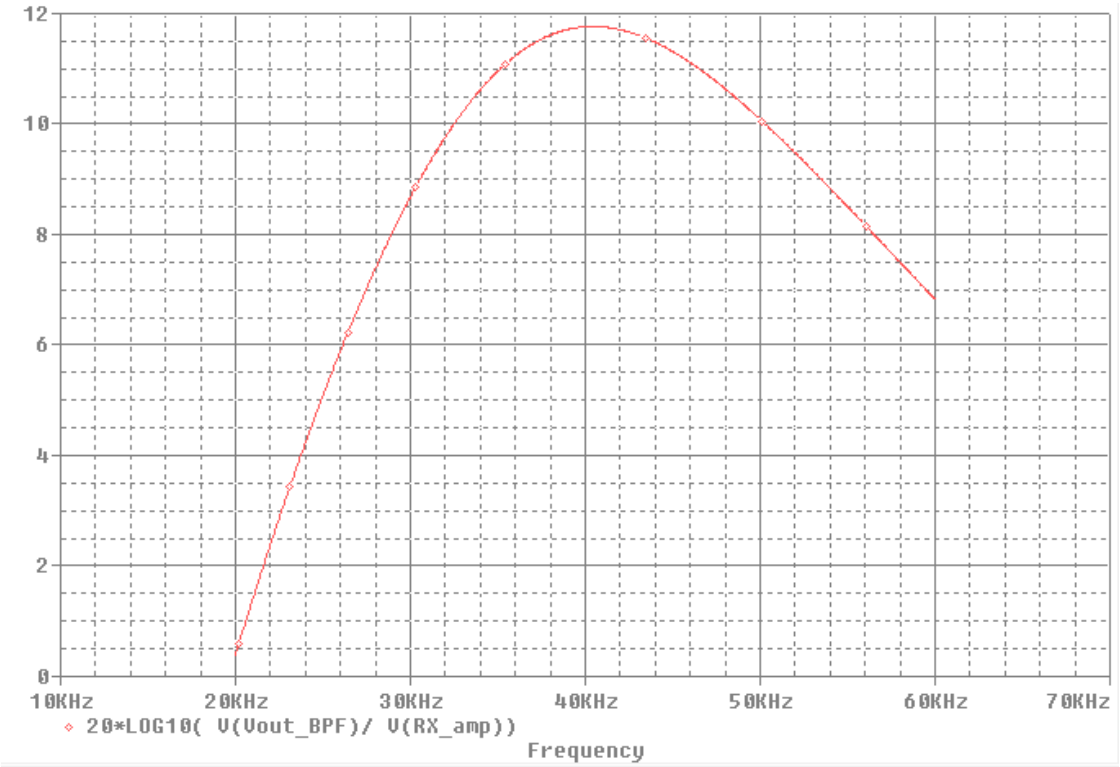
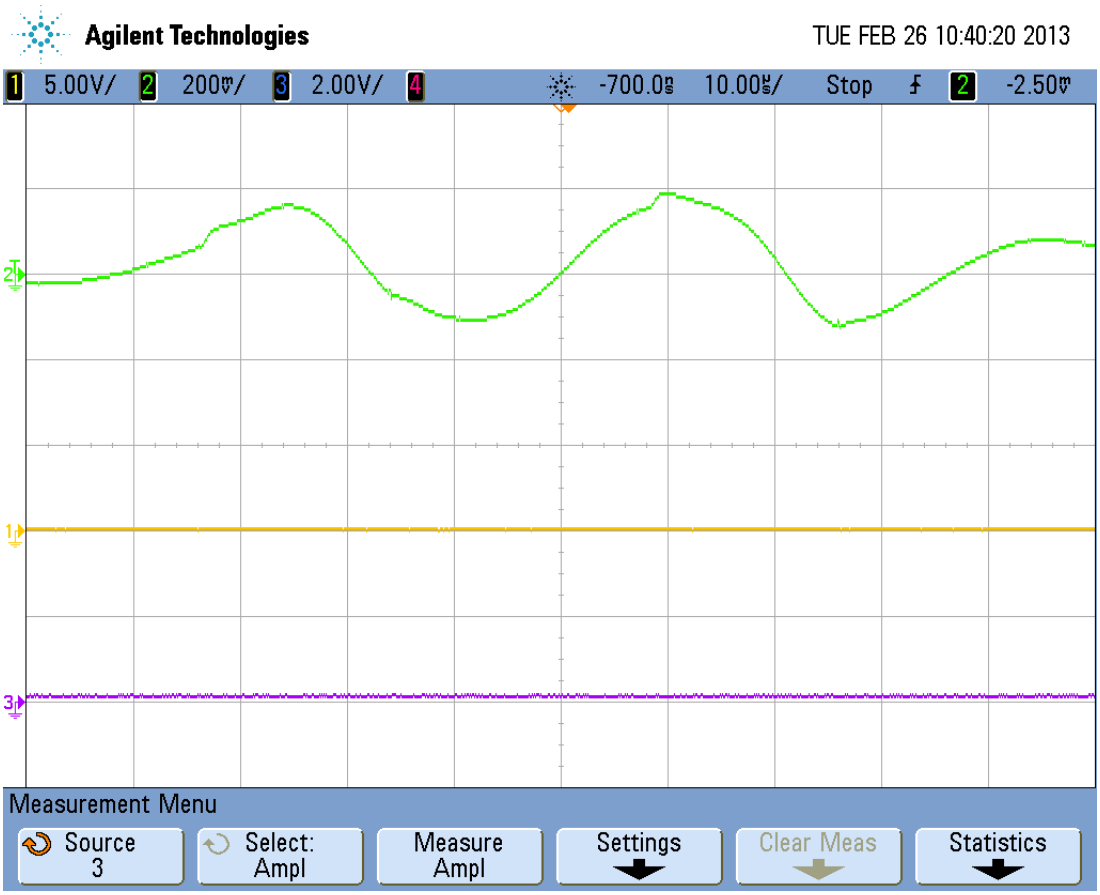
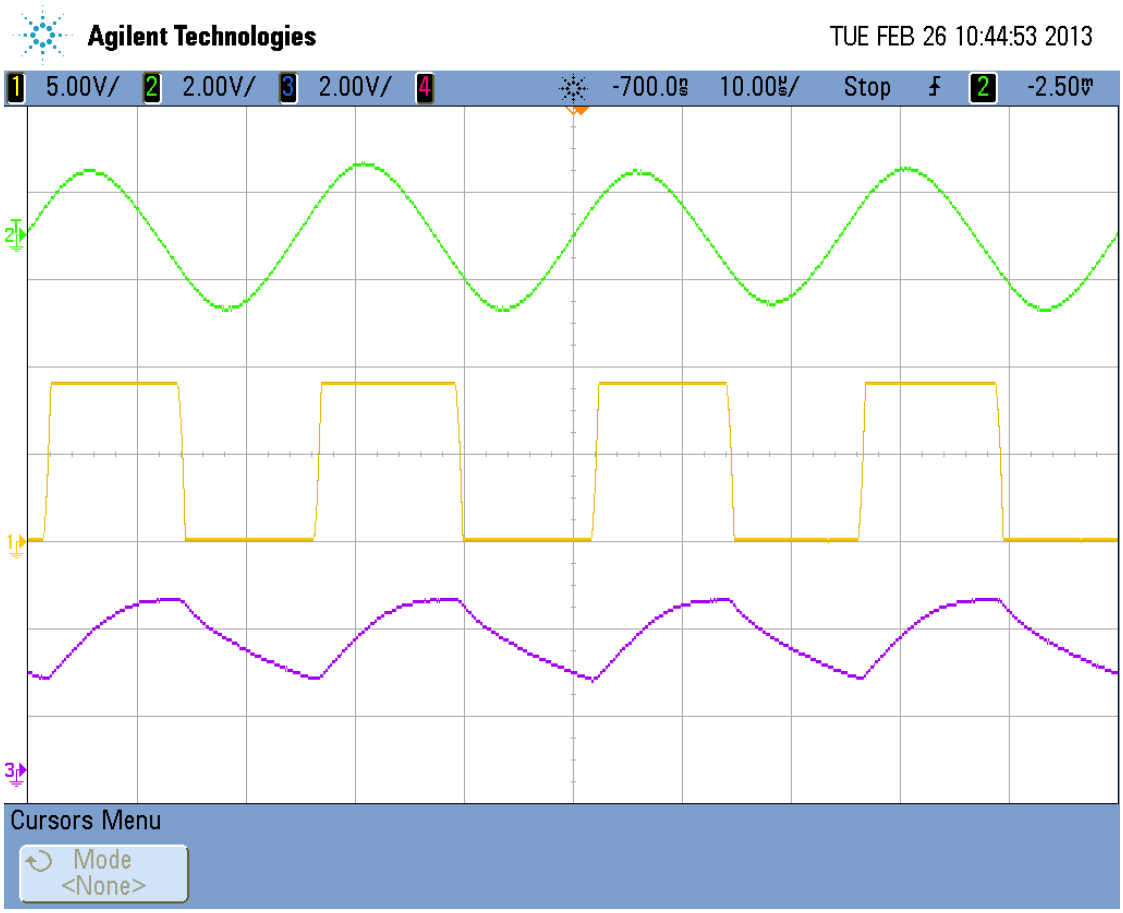


Figure 4.2: Bandpass filter frequency response in the receiver module



Top signal = amplified RX signal (weak signal)
Middle signal = output of the Schmitt trigger
Bottom signal = zener-diode-clamped output from the Schmitt trigger

Figure 4.3: noise (no ultrasonic transmission) probed on oscilloscope



Top signal = amplified RX signal (weak signal)
Middle signal = output of the Schmitt trigger
Bottom signal = zener-diode-clamped output from the Schmitt trigger

Figure 4.4: weak signal probed on oscilloscope

5. Safety

Building our robot:

Building the circuits and testing would require us to work with complex lab equipment and in a somewhat hazardous environment. Our own safety, as well as the others in the lab, would depend on the choices we make.

Firstly, all team members have completed and passed the required online safety training course. While working in the lab, we will closely follow all lab rules and be responsible for our safety. We will look out for potential hazards and take precautions to reduce or avoid the risks.

Operating our robot:

As it is an autonomous robot running around a workspace, it has the potential to accidentally injure people. Hence, we will take care to restrict the robot's movement to within the confines of the workspace (the rectangle defined by our positional beacons). Also, we will put up tape and flags around the perimeter to warn other people of the presence of the robot.

While this robot is intended for autonomous drawing, it is not to be operated without human supervision. The operator must keep a close watch on the robot, and be prepared to shut it down if it puts itself or others at risk.

6. Cost and Schedule

6.1 Cost Analysis

Parts

Motor and Chalk Control

Part name	Part number	Qty	Unit Cost	Total Cost	Ordered?
Microcontroller Evaluation Kit	EKS-LM3S9D92	1	$107.66	$107.66	No
Voltage Regulator	LM2576	2	$2.91	$5.82	No
Geared DC Motor	GM8224S014	2	$250.00	$500.00	Yes
Encoder	E6B2-CWZ3E	2	$39.95	$79.90	Yes
Op amp	LM324	1	$0.41	$0.41	Yes
H-bridge	LMD18200	2	$8.10	$16.20	Yes
Servo	HS-311	1	$7.99	$7.99	Yes
Micro Servo	HXT900	1	$2.69	$2.69	Yes
11.1V 2100mAH Battery	VNR1577	2	$19.99	$39.98	No
LiPo Battery Charger	---	1	$49.99	$49.99	No
			Total:	$810.64	

Position and Direction Sensing

Part name	Part number	Qty	Unit Cost	Total Cost	Ordered?
Ultrasonic Transmitter	MA40S4S	4	$7.17	$28.68	Yes
Ultrasonic Receiver	MA40S4R	2	$7.38	$14.76	Yes
Xbee s1 Pro	XBP24-AWI-001	2	$37.95	$75.90	Yes
Microcontroller	LPC1114fn28	7	$2.55	$17.85	No
Op Amp	LM348	2	$0.23	$0.46	Yes
Op Amp	MC34074	1	$0.60	$0.60	Yes
5V to ±10V Converter	MAX680	2	$4.82	$9.64	Yes
5V Linear Regulator	UA7805	2	$0.27	$0.54	Yes
3.3V Linear Regulator	UA78M33	2	$0.27	$0.54	Yes
3.3V Zener Diode	1N4728A	2	$0.20	$0.40	Yes
Compass Module	HMC6352	1	$34.95	$34.95	No
H-bridge	LMD18200	4	$8.10	$32.40	Yes
11.1V 2100mAH Battery	VNR1577	2	$19.99	$39.98	No
			Total:	$256.70	

Image Processing and User Interface

Part name	Part number	Qty	Unit Cost	Total Cost	Ordered?
Alphanumeric LCD	HD44780	1	$17.95	$17.95	No
Numeric Keypad	---	1	$6.47	$6.47	Yes
Pandaboard A3	750-2152-021(A)	1	$180.00	$180.00	Yes
SD Card 8GB	B000OF2F36	1	$4.98	$4.98	Yes
			Total:	$209.40	

Labor

Name	Hourly Rate	Hours/week	Weeks	Total*
Neil Christanto	$40	14	12	$16,800
Enyu Luo	$40	14	12	$16,800
Leonard Lim	$40	14	12	$16,800
Total				**$50,400**

*Total Labor (per person) = Hourly Rate X Total Hours X 2.5

Grand Total

Total Labor	Total Parts	Grand Total
$50,400	$1,276.74	**$51,676.74**

6.2 Schedule

Wk	Enyu Luo	Neil Christanto	Leonard Lim
2/4	Parts list, Block diagram	Requirements and Verifications	Compile Proposal
2/11	Choose microcontroller and other parts for motor control	Configure the Xbees to do RF communication correctly. Program the MCU for base station	Algorithm for conversion of simple bmp images to outline.
2/18	Read datasheets and draft schematic for motor control	Simulation and schematics of RF communication, direction sensing, and ultrasonic sensing	Image Processing - Vectorization
2/25	Finalize Schematics	Test the transmitters using PWM module on the MCU.	Compile Design Review.
3/4	PCB Design for motor control board	Integrate the RF comms and ultrasonic sensing using several breadboards	Integrate code into Panda Board and test.
3/11	Individual Progress. Programming of microcontroller for motor control board (communications with other sub components of robot)	Individual Progress PCB design for the base station, beacons, and the receiver module.	Individual Progress. Program the user interface to interact with Pandaboard.
3/18	Spring Break	Spring Break	Spring Break
3/25	Soldering PCB and test program	Solder PCB, test functionality of everything with the PCB	Integrate Pandaboard with motor control board.
4/1	Test requirements for motor control Integrate with ultrasonic sensing	Test requirements for position sensing and verify that everything works within tolerance limit. Integrate with motor control board	Test requirements of chalk control.
4/8	Optimize Control Coefficients	Fine tuning, debugging	Complete test run.
4/15	Final paper: write motor control section. Integration	Final paper: write position sensing section. Rigorous testing, corner case	Begin writing Final Paper: General Outline. Testing unlikely cases
4/22	Demo: Functionality	Demo: Scenario and Setup	Presentation
4/29	Lab Notebooks	Parts Checkout	Compile Final Paper

7. Ethical Issues

In line with the IEEE Code of Ethics,

1. With the building of our robot, we will be mindful to take responsibility and make decisions to ensure the safety and welfare of the public. We will disclose promptly factors that, when operating the robot, might endanger the public or the environment. At no point will we allow the robot to operate without human supervision.
    1. *to accept responsibility in making decisions consistent with the safety, health, and welfare of the public, and to disclose promptly factors that might endanger the public or the environment;*
    9. *to avoid injuring others, their property, reputation, or employment by false or malicious action;*

2. We will be honest and realistic in reporting our work and will not falsify or fabricate results.
    3. *to be honest and realistic in stating claims or estimates based on available data;*

3. Through this project, we would learn various real-world applications and technologies. These skills serve to maintain and improve our technical competence.
    6. *to maintain and improve our technical competence and to undertake technological tasks for others only if qualified by training or experience, or after full disclosure of pertinent limitations;*

4. We will be critical of technical work, both of our own and of others. Also, we will always cite all work that we use, giving credit where credit is due.
    7. *to seek, accept, and offer honest criticism of technical work, to acknowledge and correct errors, and to credit properly the contributions of others;*

5. We will assist teammates and fellow students in their professional development through knowledge sharing and support them in following the IEEE code of ethics.
    10. *to assist colleagues and co-workers in their professional development and to support them in following this code of ethics.*

8. References

[1] Bjerknes, Jan D., Wenguo Liu, Alan FT Winfield, and Chris Melhuish. "Low Cost Ultrasonic Positioning System for Mobile Robots." (n.d.): n. pag. Web. <http://www.ias.uwe.ac.uk/~a-winfie/Bjerknes_etal_TAROS07.pdf>.

[2] Shima, Jim. "DSP Trick: Fixed-Point Atan2." *DspGuru.com*. N.p., 23 Apr. 1999. Web. 26 Feb. 2013. <http://www.dspguru.com/dsp/tricks/fixed-point-atan2-with-self-normalization>.

[3] "NXP LPC111X Datasheet." N.p., 20 Feb. 2013. Web. <http://www.nxp.com/documents/data_sheet/LPC111X.pdf>.

[4] *NXP LPC111X User Manual*. N.p.: n.p., 24 Sept. 2012. PDF.

[5] "Scalable Vector Graphics (SVG) 1.1 (Second Edition)." W3C. N.p., 16 Aug. 2011. Web. <http://www.w3.org/TR/SVG>.

[6] "LM2576/LM2576HV Series SIMPLE SWITCHER® 3A Step-Down Voltage Regulator." Texas Instruments, Nov. 2004. Web. 27 Feb. 2013. <http://www.ti.com/lit/ds/symlink/lm2576.pdf>.

[7] "LMD18200 3A, 55V H-Bridge." Texas Instruments, May 2004. Web. 27 Feb. 2013. <http://www.ti.com/lit/ds/symlink/lmd18200.pdf>.

[8] "Stellaris® LM3S9D92 Evaluation Kit User's Manual." Texas Instruments, 5 July 2011. Web. 27 Feb. 2013. <http://www.ti.com/lit/ug/spmu174/spmu174.pdf>.

[9] "Stellaris® LM3S9D92 Microcontroller." Texas Instruments, 22 Jan. 2012. Web. 27 Feb. 2013. <http://www.ti.com/lit/ds/symlink/lm3s9d92.pdf>.

[10] "LM124/LM224/LM324/LM2902 Low Power Quad Operational Amplifiers." Texas Instruments, May 2004. Web. 27 Feb. 2013. <http://www.ti.com/lit/ds/symlink/lm324-n.pdf>.

[11] "OMAP™ 4 PandaBoard System Reference Manual." Pandaboard.org. N.p., 29 Nov. 2010. Web. <http://pandaboard.org/sites/default/files/board_reference/pandaboard-a/panda-a-manual.pdf>.

[12] "HD44780U (LCD-II)(Dot Matrix Liquid Crystal Display Controller/Driver)." N.p., n.d. Web. 15 Feb. 2013. <http://www.sparkfun.com/datasheets/LCD/HD44780.pdf>.

[13] "IEEE Code of Ethics." IEEE. N.p., n.d. Web. 27 Feb. 2013. <http://www.ieee.org/about/corporate/governance/p7-8.html>.