The Shoes Sorting Robot

By

Jinghan Guo

Quanhua Huang

Mingxi Zou

Design Document for ECE 445, Senior Design, Spring 2019

TA: Zhen Qin

02 February 2019

Project No. 12

Contents

1 Introduction

1.1 Objective

Nowadays robots are becoming more and more important in our daily lives. During hurricane Katrina, robots like isensys UAV and precisionhawk can collect data needed by rescue team in 7 hours that usually take people 3 days to get any other ways. During the Japan Tsunami, by using the sonar of Sarbot Interface, engineers were able to reopen a fishing port in 4 hours, when the fishing port was told that it was going to be six months before they could get a manual team of divers in. During these six months fisherman in Japan will miss the fishing season, which is the major economy for them. There are many more examples like these two that shows the improvement robots bring to people's life. Robots go places people cannot go and do things that people don't want to spend time doing, robots even do things new by assist us in innovative ways.

Our goal is to bring a house-use robot that helps people organize shoes. Our group comes up with this idea because it is really annoying when people get tripped over a shoe in the doorway. Disorganized shoes can be a mess but sometimes people just don't get time to organize shoes after they take them off. A clear entrance in the doorway can not only make it easy for people to go through but it can make house look nice and clean in the first sight. Organizing shoes is the principal goal of our senior design, but after we successfully build the robot, this application can be extended into organizing all kinds of things like dishes, toys, and clothes.

Therefore, we aim to build a shoe sorting robot to free people from spending time organizing shoes. We will build a mobile robot with chasis and arm, the robot will load a camera for vision and a load cell for sensing the weight. Our robot will initially awaiting next to the shelf, after people take shoes off, robot will come to the shoe, pick it up and place one shoe next to the other on the shelf.

1.2 Background

Our robot is built based on the scenario that shoes are scattered on a 60*60cm entrance mat with a rectangular shoe organizer next to it. A camera will be held above a certain height to capture the whole mat. Our robot is constituted of a 6 degrees of freedom end effector and a mobile chassis. It sits quietly next to the organizer when no shoe appears on the mat, after the camera detects any object, the car moves to the mat and start working.

The robot moves to the shoe that is closest and distinguish each shoe by color filtering and weight filtering. The image captured by camera records the size and color of the object. If that object size lies in a certain range we assign it as a "shoe". Otherwise we just ignore it if the size of the object appears to be too large or too small. And the load cell on the robot records the weight of every object. In this way we can decide if two things are a pair depends on their color, size and weight.

So far there is no such robot on the market, therefore this project will be novel. The hit of mobile vacuum cleaner in the market is an inspiration for us and we believe our shoe sorting robot will be popular if we make it affordable and reliable.

1.3 High-Level Requirements Lists

- The robot must be able to locate a shoe on the 60cm*60cm mat and successfully pick it up in 3 times.
- The robot must be able to recognize two shoes in a pair and place them next to each other on the organizer with accuracy of 60% .
- The robot must be able to sort shoes(at least three pairs) effectively and return to its starting position upon completion within 8 minutes.

2 Design

2.1 Block Diagram

Figure 1 shows that our overall design contains four modules: the control module, the power module, the robot module and the peripheral module.

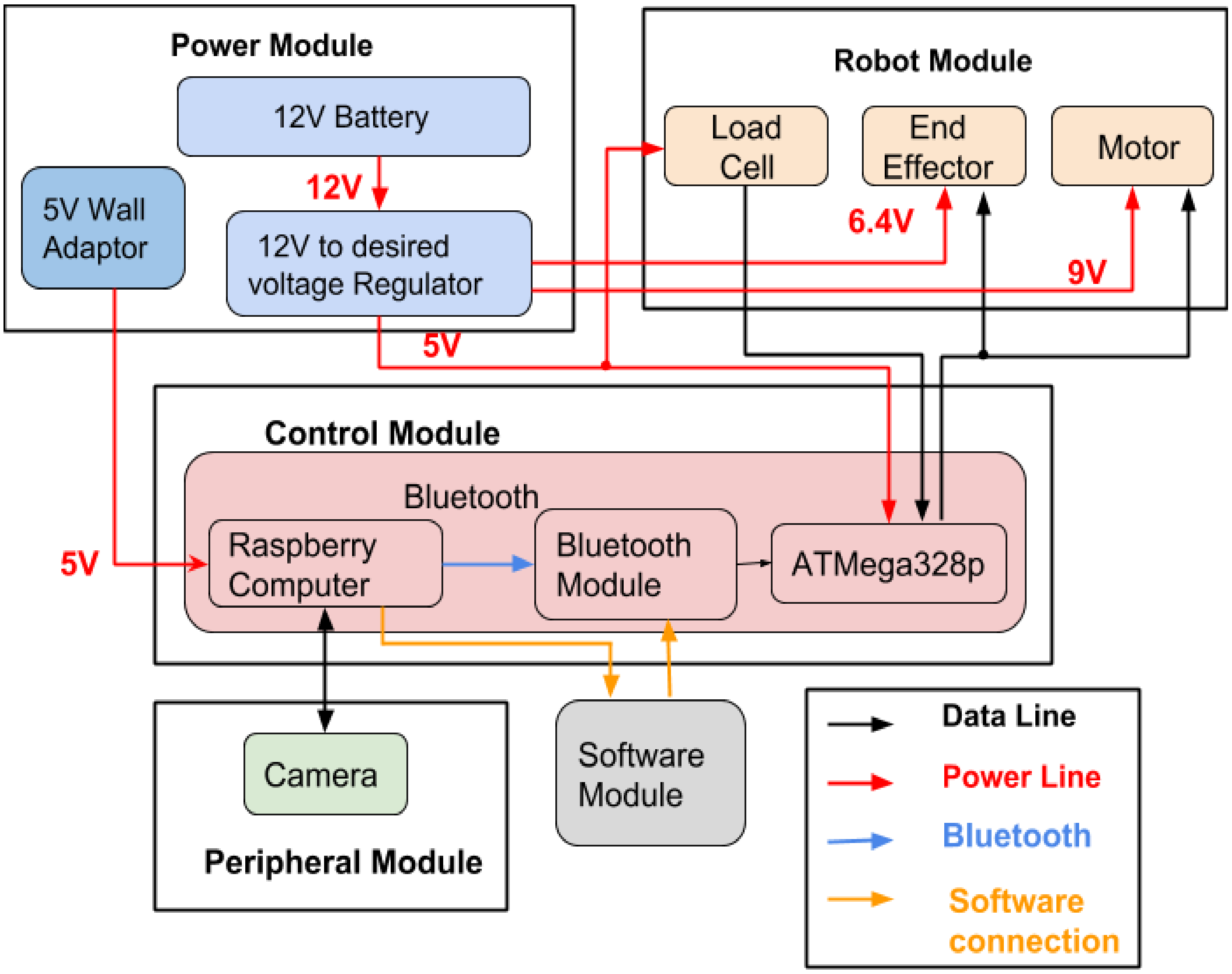


Figure 1. Block Diagram

2.2 Physical Design

The physical design of our project includes a moving chassis with a robotic arm and load sensor on it, as indicated in figure 2. The picture above shows the circumstance when the robot gets the signal of working. The big square in the middle is the shoe mat(60cm*60cm)with three pairs of shoes with different colors. The rectangle above the mat represents the shoe shelf and robot sits on the side will move along the side of the mat to reach the closest shoe when start working.

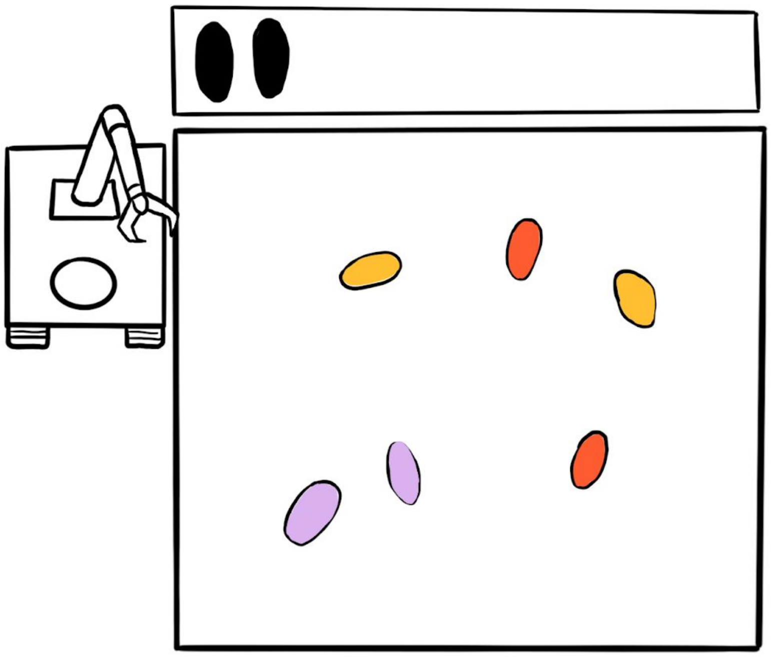


Figure 2. Experiment Setting

Figure 3 is the physical form of our robotic arm and it shows the dimension of the arm is 465*120*120mm. The chassis of our robot has a dimension of 270*255*90mm(L*W*H).

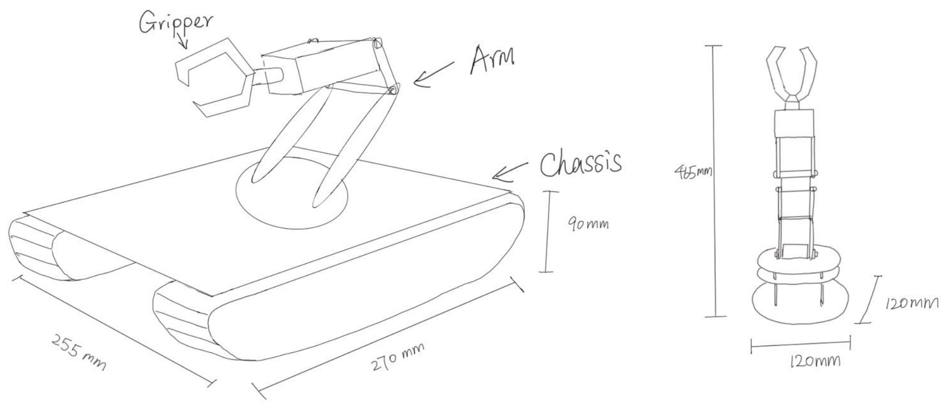


Figure 3. Physical Scale

Figure 4 demonstrates the physical structure of our robot in the side view.

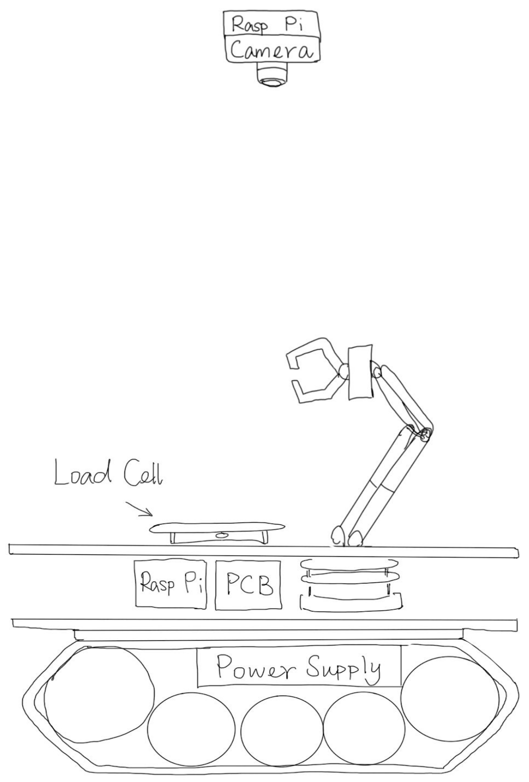


Figure 4. Physical Structure of Robot

2.3 Block Design

2.3.1 Control Module

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

2.3.1.1 Raspberry Pi 3 b

We will use Raspberry Pi Model 3 b for processing the image received from the camera, and find the best path for the robot to follow to pick up shoes. The Raspberry Pi will interface with microcontroller through the HC-05 bluetooth module. The input of Raspberry Pi is the image captured by camera, and it will compute all the data the microcontroller needs and sends it to ATmega328P through bluetooth module, the output of microcontroller will allow our robot to perform the process autonomously.

Requirement	**Verification**
1. Must be able to receive an image from the camera 2. Must be able to apply filters and do raster scan for twice 3. Must be calculate the best path for robot to follow to get to destination 4. Must be able to send commands to ATmega328p through the bluetooth module	1. Ensure images captured by camera must show up in specified location in Raspberry Pi’s file system 2. Ensure the way raster graphics maps shoes matches with how shoes are scattered on the mat in real world setting 3. Calculate the best path for robot by hand and verify if that result matches with Raspberry Pi’s solution 4. Verify the output of microcontroller corresponds to input instructions from Raspberry Pi

2.3.1.2 Bluetooth Module

In our experiment setting, the Raspberry Pi will be placed next to the camera, and the ATmega328p microcontroller will located on the robot. We don't want to use a long cable wire to connect them since the cable will get in the robot's way when it is working, therefore we decide to interface microcontroller and Raspberry Pi through the inexpensive and ubiquitous HC-05 bluetooth module. The communication HC-05 with default baud rate 9600 in communication mode is via serial communication which makes an easy way to interface with microcontroller. This bluetooth module will be connected to ATmega and placed on the robot, and it will pair with Raspberry Pi to receive commands wirelessly, as indicated in figure 5.

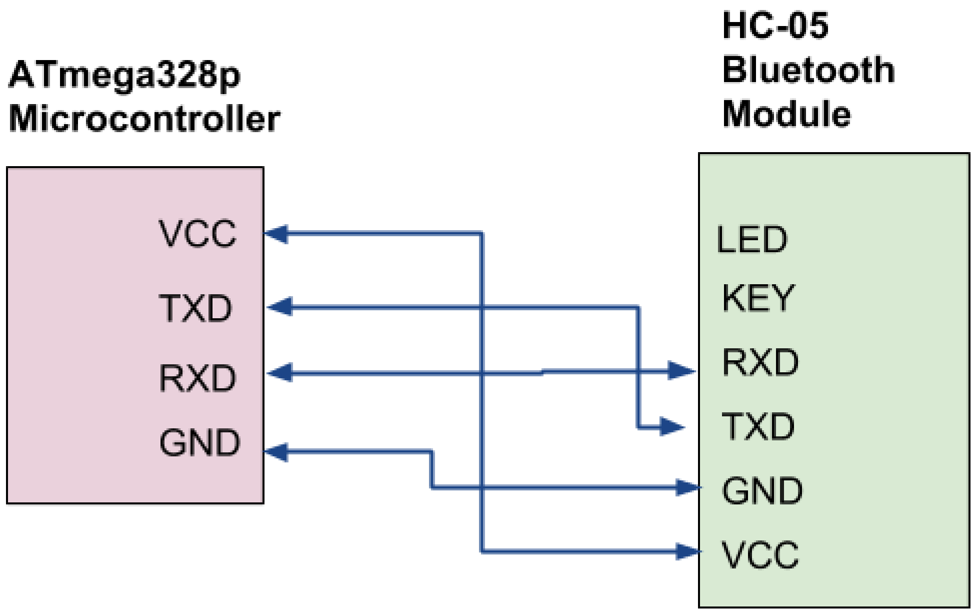


Figure 5. Bluetooth Connection

Requirement	**Verification**
1. Must be able to connect Raspberry Pi wirelessly 2. Must be able to send correct instruction from Raspberry Pi to ATmega328p microcontroller	1. Ensure the acknowledgement signal for pairing shows up in Raspberry Pi's system 2. ATmega328p sends out output signals to motors corresponding to Raspberry Pi's input signal

2.3.1.3 ATMega328p Microcontroller

This component receives instructions from the Raspberry Pi through HC-05 bluetooth module, converts the command to motor rotations, and sends information to the stepper and servo drivers to activate the robot arm and end effector. The supported instructions must include start/stop motors, raise/lower robot arm and open/close end effector.

The interface between microcontroller and Raspberry Pi is handled by HC-05 bluetooth module. In necessary, we will also use a 16MHz oscillator to drive the ATmega328 microcontroller to achieve the maximum clock frequency of the device. The load cell will interface with microcontroller through load cell amplifier HX711. To drive the stepper motors, we use the Allegro A4988 stepper motor controller. The MSX pins on the schematic below have been wired to microcontroller in a way to keep the device in 1/16 microstep mode. Two 100uF decoupling capacitor have been added to two stepper motors to reduce ripple from power supply.

The ATMega328p microcontroller also implements the shoe sorting algorithm, given all the coordinates and properties of each shoe from the Raspberry Pi, and passes commands to end effector. Each shoe has already been labelled and the three coordinates of each shoe's tail, toe and center have been calculated.

Requirement	**Verification**
1. Voltage of power input remains at 5V +/- 10% throughout the operation	1. Use multimeter to measure the power input node and check the maximum deviation

The schematic of ATmega328p microcontroller, HC-05 bluetooth module, HX711 load cell amplifier, stepper and servo motor is shown in figure 6.

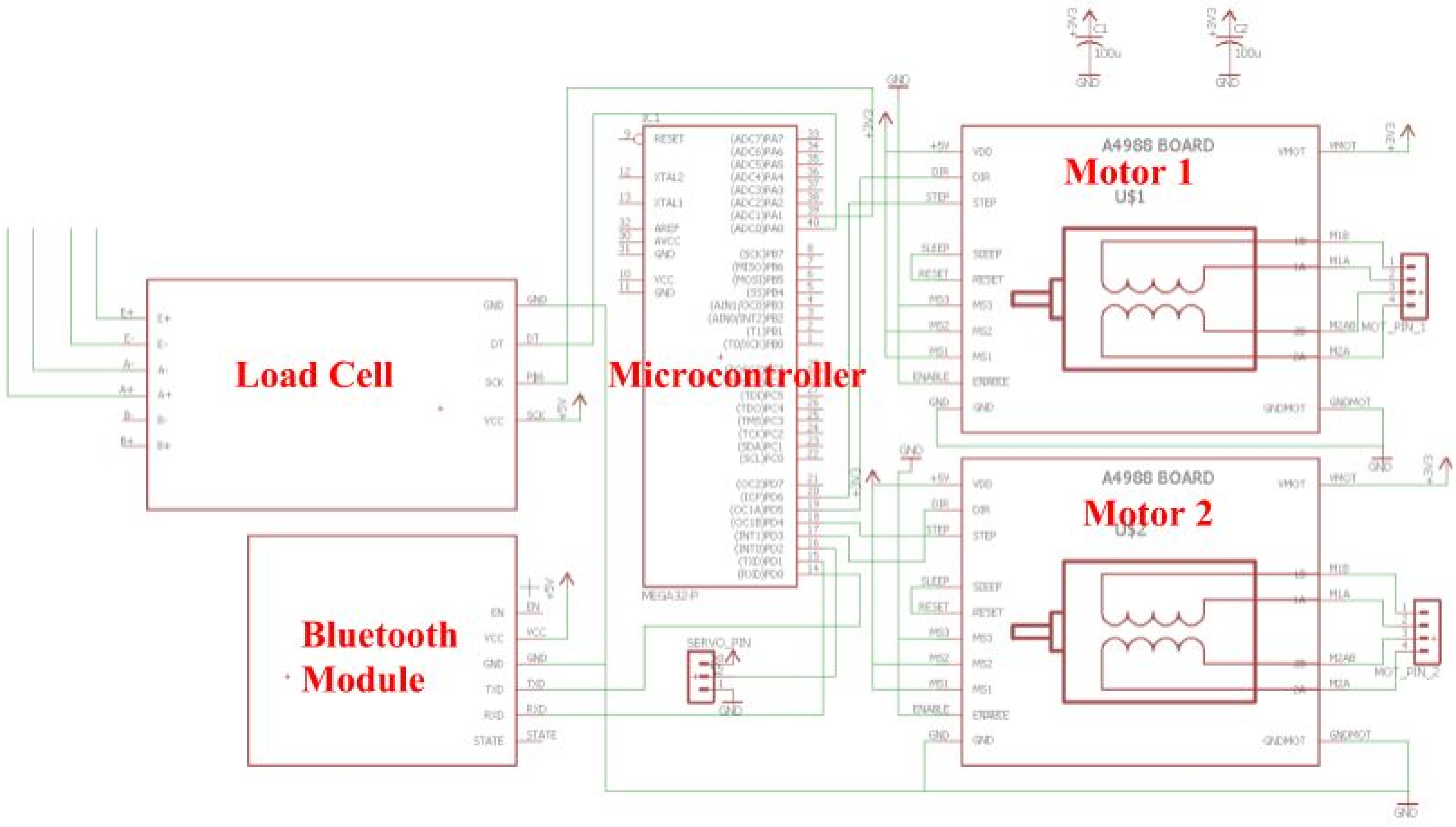


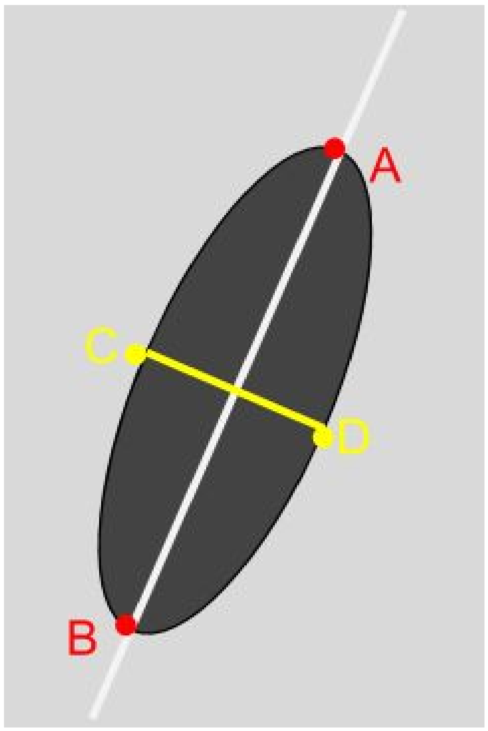
Figure 6. Schematic for Microcontroller

2.3.2 Robot Module

Robot Module receives demands and data from the microcontroller and then controls the chassis movement through two motors and controls the servo on the arm to reach the shoes. We will test robot module by giving input signals to motor and end effector. If the end effector can start grabbing objects and motor starts moving when input signal is high, and end effector can let object go and motor stops when input is low, we think the motor module works fine independent of other modules.

2.3.2.1 Robot Arm-Gripper (End Effector)

We use a gripper that is large enough to grab an object with width at about 10 cm and weight around 400g. Although each shoe is around 300g, we plan to allow some discrepancies between shoes that might occur. The gripper has a maximum stretch up to 14 cm and can catch the object smoothly. The end effector is connected to the microcontroller ATMega328p through the PD pins in ATMega328p.



There is a digital servo to control the direction of the gripper. As shown in the figure 7, the black shadow represent a shoe on the mat, in the process of imaging process we will find two A,B point at the tail and toe of a shoe and connect it as a line(white). After we have the function of the straight line, we will find out another line(yellow) which cross the white line perpendicularly and at the mid point. Then the gripper will turn in the direction of the yellow line and grab the shoe on C,D point. The point yellow line and white line intersect is the center point.

Figure 7 Shoe direction

Requirement	Verification
1. Be able to turn into specific direction. 2. Be able to grab a shoe under 500 g steadily and move to another place without losing it on ground.	1 / 2. Turn the grip in the wanted direction then lower it on the ground to check if the contact points matched those two we calculated.

2.3.2.2 Robot Arm-Servo

Including the one to control gripper, there are in total 6 digital servos on the robotic arm. There are two LFD-06 digital servo with high temperature resistance and two LDX-218 servos with 17kg large torque on the arm. And a 17kg large torque LD-1501 servo on the bottom plate.

Requirement	Verification
1. Be able to receive the commands from ATmega328p microcontroller. 2. Joints are able rotate the angle degrees as Raspberry commands.	After connected to power supply,receive command from computer, check if the arm can move to the destination we want.

2.3.2.3 Robot Chassis-Motor

The robot chassis has two DC motors with output speed 150±10%rpm and working voltage 9V. We will implement A4988 microstepping motor driver chip on our PCB to control the two motor. The figure 8 below explains the connection between motor and driver.

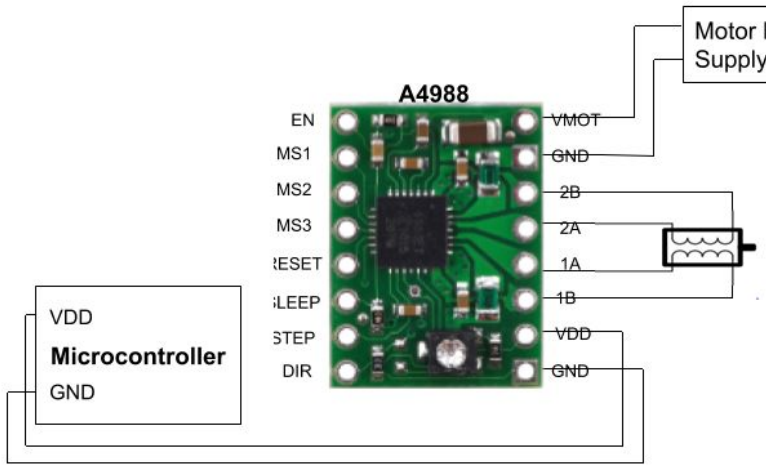


Figure 8. Motor connection

Requirement	Verification
1. Able to change the direction of motor 2. The robot chassis must be steady and must guarantee not to turn over when a shoe(around 300g) is picked up by the gripper.	1. Give a input signal from microcontroller, check if the motor is able to change direction when it is running 2. Put all the components (microcontroller, load cell, batteries) and control the robot arm to pick up a shoe and check if the chassis is steady.

2.3.2.4 Load Cell(NX711 weight sensor)

The load cell will be implemented on the robot. When the robot gripper put shoe on it, the load cell send the digital value of the shoe's weight to microcontroller.

Requirement	**Verification**
1. Must be able to connect to Raspberry Pi ang transfer the digital out value 2. Must be able output a stable weight value for a same shoe	1. Connect the HX711 sensor to microcontroller. 2. Put each shoe on it to read out the weight value from computer for several times to verify if the output is same.

2.3.3 Power Module

The power module will power the major circuit with the 12V Energizer A23 Alkaline Battery. The battery holder we are going to use is the corresponding A23 battery 12V Clip Holder Box. With two such holder boxes in parallel along with 2 A23 batteries together, the power module will be able to provide sufficient power to other parts, considering the voltage decay during and the power consumption at the same time. Besides, to fulfill different voltage requirements, the power supply will be connected to a 7.5V step down regulator to activate the end effector, a 5V step down regulator to power the load cell and PCB, and a 9V step down regulator for the motor. In addition to using the batteries, there will also be a wall plug adaptor powering the Raspberry Pi computer that is next to the camera, and the camera connected to the Raspberry Pi will then get powered automatically.

2.3.3.1 Battery

We will use two 12V Energizer A23 Alkaline Battery as the major power supply in our design. Each A23 Alkaline Battery is 12V, and we use two batteries in parallel to increase the capacity,n as indicated in figure 9. The battery and the battery holder box have the same size so they should be compatible.

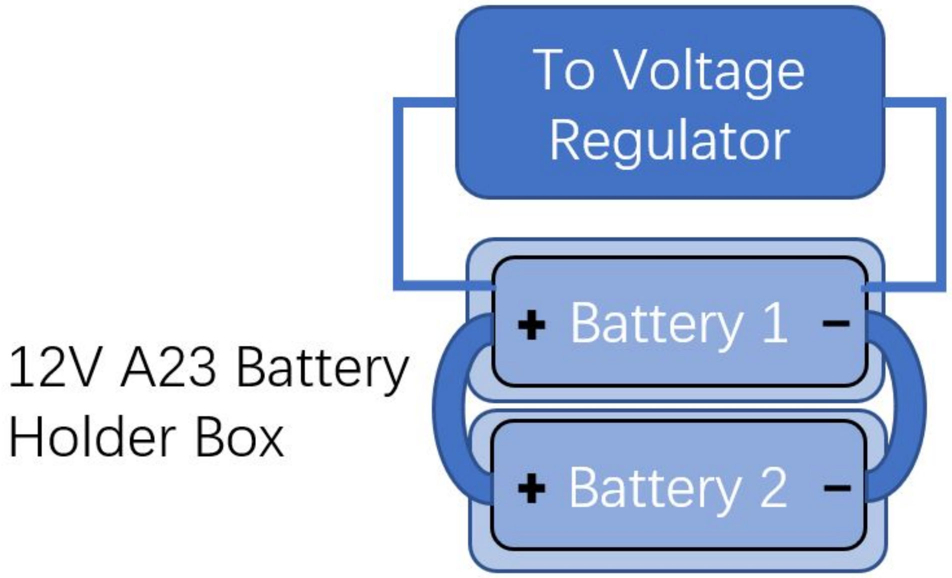


Figure 9. Battery layout

In addition, the 12V A23 Battery should meet the requirement in the following chart.

Requirement	Verification
1. Must be able to provide steady power to other parts. 2. The output voltage is at least 9V.	1. Test the current with this battery set and check whether the power is steady. 2. Use a voltmeter to measure the voltage.

2.3.3.2 Power Supply for Raspberry and Camera

To power the camera, first we use the 5V wall adaptor to power the Raspberry computer and then we connect the camera through Raspberry Pi with the CSI-2 Camera Connector. The Raspberry Pi operates at 5V so there is no need to use voltage regulator here. Besides, the current through the Raspberry Pi is 2A, which is sufficient to supply the camera. The connections between camera and the Raspberry computer is illustrated in the following figure 10. The camera module and the raspberry pi is wired through a 15-pin ribbon cable.

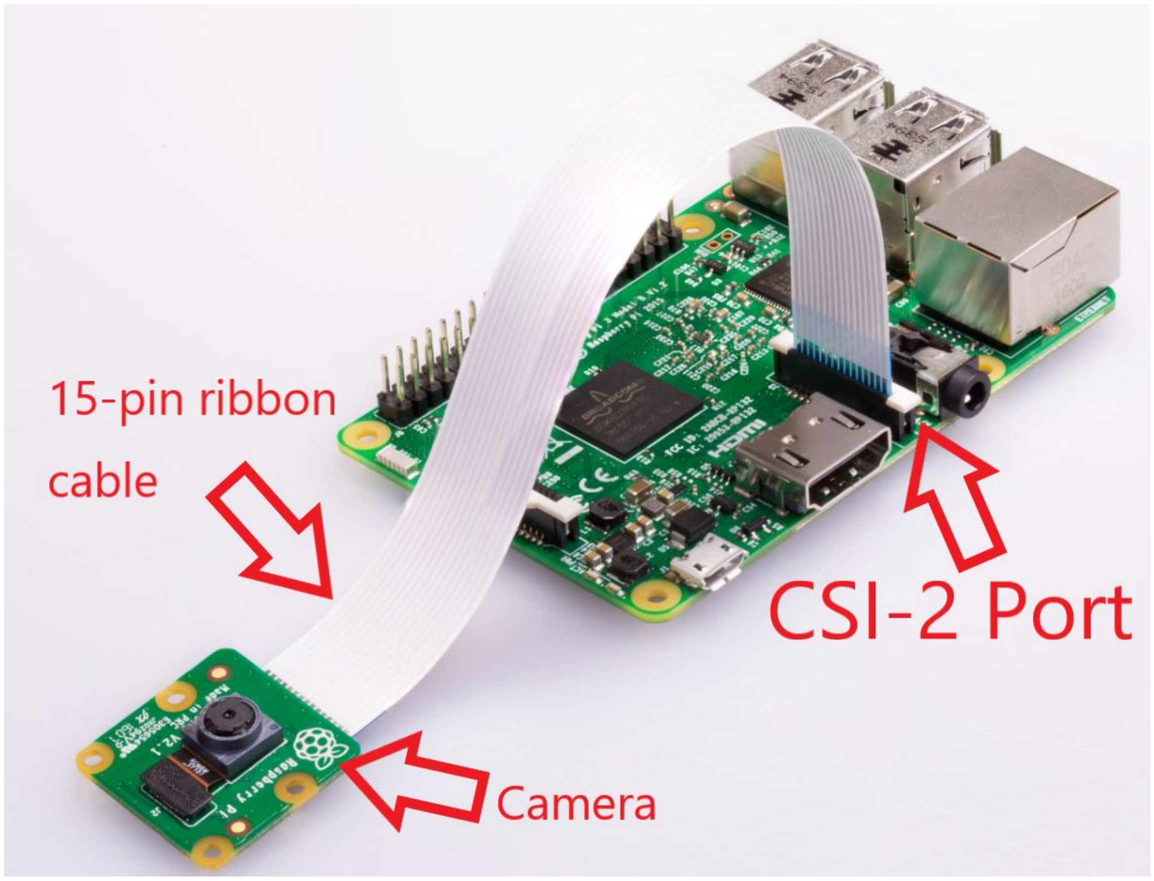


Figure 10. Raspberry Pi(right) and camera(left)

Requirement	Verification
1. Camera can be turned on when connected to the Raspberry Pi	1. By observation. Check the indication light on the camera.

2.3.3.3 Voltage Regulator

Different parts of our design require distinct voltage so we include three types of voltage regulators. The robot motor needs exactly 9V so it is directly wired a 9V regulator. For other components, such as the microcontroller and the load cell, they both need a 5V voltage regulator. For the end effector, it will use a 7.5V voltage regulator.

9V Voltage Regulator

This voltage regulator will convert a higher voltage into 9V, which powers the robot motor and avoid damaging the motor module because of high voltage. The chip LM7809 accepts 11.5-26V as input and then outputs 9V with a maximum current 2.2A. The working current for the robot motor is 1.2A so this voltage regulator satisfies this requirement.

Requirement	Verification
1. The regulator should output 9V +/-5% 2. The regulator accepts 12V as input voltage	1. Connect the regulator output to a voltmeter and measure the voltage. 2. The same method as verification 1.

6.4V Voltage Regulator

With a 6.4V voltage regulator, we can make sure that the end effector works properly in the range of 6V to 7.4V. This R1524x04x voltage regulator has a input voltage range of 3.5-36V and the output voltage is 6.4V and 200mA.

Requirement	Verification
1. The regulator should output 6.4V +/-5% 2. The regulator accepts 12V as input voltage	1. Connect the regulator output to a voltmeter and measure the voltage. 2. The same method as verification 1.

5V Voltage Regulator

This 5V voltage regulator reduce the voltage from 12V to 5V to meet the voltage requirement for the load cell and the microcontroller ATMega328p. The load cell requires a voltage input in the range of 3.3 to 5V so this voltage regulator works well. The microcontroller requires 5V as input so this voltage regulator also meet the requirements. Besides, this R1524x050x voltage regulator has a input voltage range of 3.5-36V and the output voltage is 5V and 200mA, and the current is enough to power the microcontroller.

Requirement	Verification
1. The regulator should output 5V +/-5% 2. The regulator accepts 12V as input voltage	1. Connect the regulator output to a voltmeter and measure the voltage. 2. The same method as verification 1.

2.3.4 Peripheral Module

2.3.4.1 Raspberry Pi Camera Module V2

The Raspberry Pi Camera V2 is placed 1m above the top left corner of the mat and takes image of the whole mat upon instruction and then sends the image back to the Raspberry Pi. After implementing image processing in the Raspberry Pi, all the objects will be distinguished from the background. The still picture resolution of this camera is 3280 x 2464, which is suffice to record the color of the objects on the mat. The connection between the Raspberry Pi Camera Module V2 and the Raspberry Pi computer is through a CSI-2 port, which is extremely fast with 60fps for 720p. In addition, the dimension of this camera is only 23.86 x 25 x 9mm, which is small enough and meets our requirements.

Requirement	Verification
1. The camera has a high resolution and can record objects with different colors clearly 2. The camera should be turned on after being connected to the CSI-2 port in the Raspberry Pi 3. The camera must be able to capture the image of the whole mat	1. By observation. Use the camera to capture an image and see if the image display different colors clearly. 2. Check the LED light in the corner of this camera module. When the camera is activated, the light is on. 3. Place the camera 1m above the top left corner of the mat to check if the camera captures the image of the whole mat.

2.3.5 Software module

The software module in our design takes the image captured by the camera as input from Raspberry Pi, and outputs commands to the robot arm and chassis through ATMega328p. The Raspberry Pi implements the image processing and then sends the property of each shoe such as color, coordinates, object number and paired condition to ATMega328p. In the next step, ATMega328p will run the sorting algorithm and start picking up shoes and sorting them into the shelf. In the meanwhile, every position that the end effector reaches is calculated using inverse kinematics inside the ATMega328p as well. The following figure 11 illustrates the image processing in Raspberry Pi.

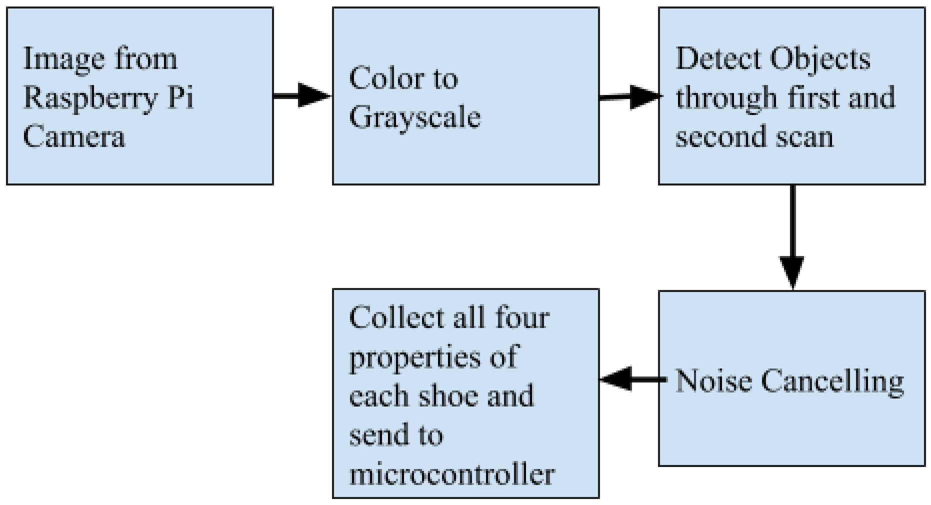


Image from Raspberry Pi Camera
Color to Grayscale
Detect Objects through first and second scan
Noise Cancelling
Collect all four properties of each shoe and send to microcontroller

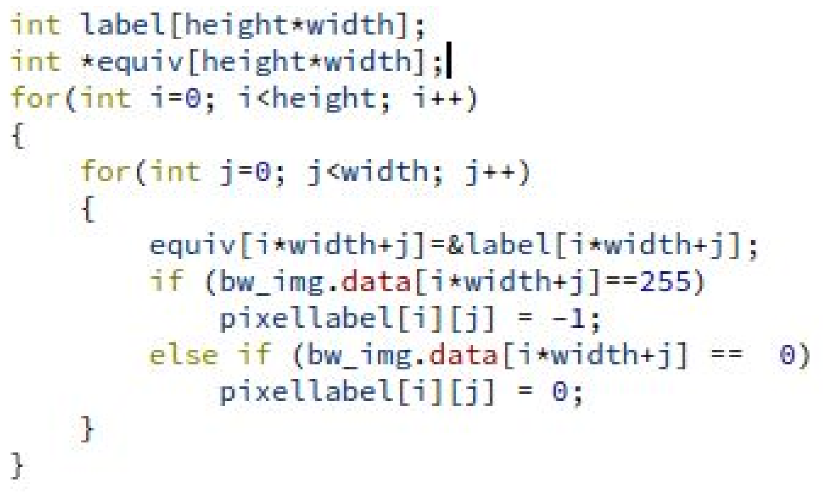
Figure 11. Software Module Flowchart

Color to Grayscale

Since camera provides us colorful images, we want to convert it to grayscale first. That is, each pixel in the image has associated grayscale value 0-255, where 0 is black and 255 is white. We will separate the image into background(light region) and objects(black regions) by surveying the image and selecting a grayscale value "GRAY" that best distinguishes between background and objects: all pixels with values greater than GRAY(lighter than the threshold) will be assigned as background while all pixels with values smaller than GRAY(darker than the threshold) will be considered as an object. After the grayscale filtering, every pixel in the image will be replaced with a black pixel if it is background or a white pixel if it is an object, and this grayscale image is represented as "bw_img" in our code.

First and second raster scan[13]

We will associate our object in the image using two raster scans. To begin with, we create two array label[] and *equiv[] to help us. The first array label[] represents the integer labels and the second array *equiv[] is an array of pointers for noting the equivalence relations. Although this method consumes more memory, it turns out to be faster than using only one raster scan. In the first step, we initialize these two arrays and associate the array pointer "equiv" with the address of the integer label. Then we assign all the elements in pixellabel, which is a 2D array containing the RGB value of each pixel, to be -1 or 0 depending on whether the pixel is white or black. In the following content, we use pixel label to represent the RGB value of each pixel. Besides, height and width refers to the pixel size of the image in the code. The initialization process is shown in the following figure 12.

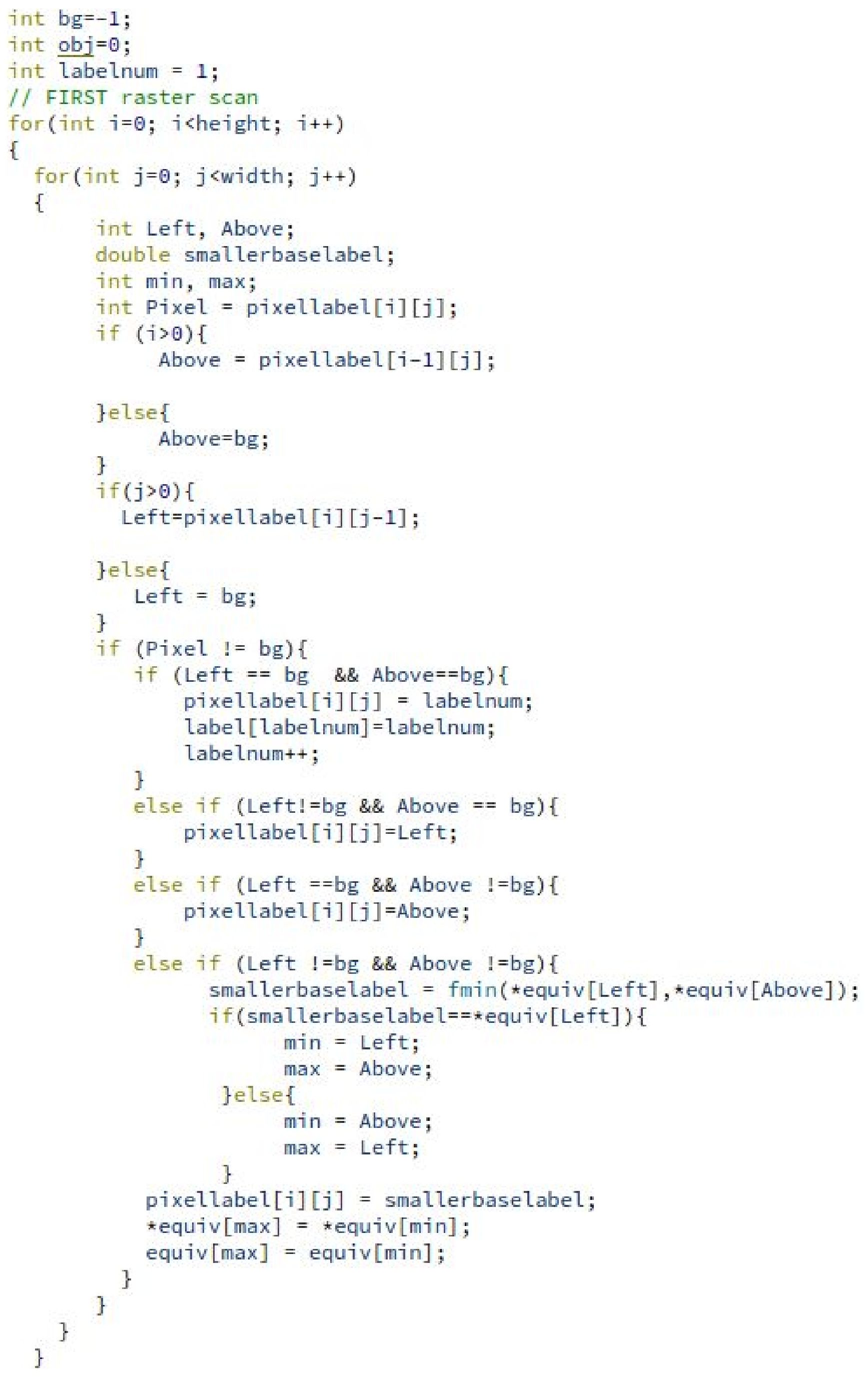


```
int label[height*width];
int *equiv[height*width];
for(int i=0; i<height; i++)
{
    for(int j=0; j<width; j++)
    {
        equiv[i*width+j]=&label[i*width+j];
        if (bw_img.data[i*width+j]==255)
            pixellabel[i][j] = -1;
        else if (bw_img.data[i*width+j] ==  0)
            pixellabel[i][j] = 0;
    }
}
```

Figure 12. Initialization for Raster Scan

In the first raster scan, we create three elements for each pixel label: Pixel, Left and Above. Pixel represents the pixel label for this cell. Left indicates the pixel label for the cell that is on the left to the current cell and Above indicates the pixel label for the cell that is on top of the current cell. There are two special conditions during the assignments of Left and Above so we assign Above to be the background color "bg" in row 0 and Left to be the background color "bg" in column 0.

Next, we implement the algorithm where the current pixel cell is not in background color. There are four cases when we loop through the cells and we deal with these four cases differently. When Left and Above is in background color, then the current pixel label gets a new number "labelnum" and then increments labelnum. When we only have Above that is in the background color, we assign the current pixel label to be the value in Left. When we only have Left that is in the background color, we assign the current pixel label to be the value in Above. In the last case, when Left and Above are not in the background color, we choose the smaller base label by comparing "*equiv[Left]" and "*equiv[Above]" to be the current pixel label. Additionally, we also update the equivalence pointers to point to the smaller base label. After running the first raster scan, we assign all the objects with some labels, as illustrated in the figure 13.



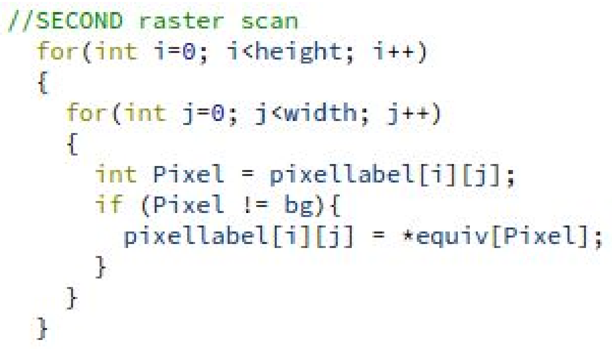
```
int bg=-1;
int obj=0;
int labelnum = 1;
// FIRST raster scan
for(int i=0; i<height; i++)
{
  for(int j=0; j<width; j++)
  {
       int Left, Above;
       double smallerbaselabel;
       int min, max;
       int Pixel = pixellabel[i][j];
       if (i>0){
            Above = pixellabel[i-1][j];

       }else{
            Above=bg;
       }
       if(j>0){
         Left=pixellabel[i][j-1];

       }else{
          Left = bg;
       }
       if (Pixel != bg){
          if (Left == bg  && Above==bg){
              pixellabel[i][j] = labelnum;
              label[labelnum]=labelnum;
              labelnum++;
          }
          else if (Left!=bg && Above == bg){
              pixellabel[i][j]=Left;
          }
          else if (Left ==bg && Above !=bg){
              pixellabel[i][j]=Above;
          }
          else if (Left !=bg && Above !=bg){
                smallerbaselabel = fmin(*equiv[Left],*equiv[Above]);
                if(smallerbaselabel==*equiv[Left]){
                      min = Left;
                      max = Above;
                 }else{
                      min = Above;
                      max = Left;
                 }
           pixellabel[i][j] = smallerbaselabel;
           *equiv[max] = *equiv[min];
           equiv[max] = equiv[min];
         }
       }
    }
  }
```

Figure 13. First Raster Scan

In the second raster scan, we assign the same label to all pixels in the same objects using the equivalences pointers “equiv”. In this way, all the pixels inside the same object share the same label. The process of the second raster scan is demonstrated in the figure 14.



```
//SECOND raster scan
  for(int i=0; i<height; i++)
  {
    for(int j=0; j<width; j++)
    {
      int Pixel = pixellabel[i][j];
      if (Pixel != bg){
        pixellabel[i][j] = *equiv[Pixel];
      }
    }
  }
```

Figure 14. Second Raster Scan

After running two raster scans, we are able to distinguish objects from the background.

Noise Cancelling

After we have identifies all the objects with unique labels, we will perform “noise elimination” and discard objects that are too big or too small. To accomplish this, we will compute the number of pixels in each object. Again we will choose a range of number of pixels that a shoe usually contains and use this threshold number of pixels to distinguish legitimate shoes from noise objects. For objects lie outside of range, we change its color into white and thereby forcing the objects into background. After noise elimination, the number of legitimate objects and their corresponding properties will be reported to microcontroller. And figure 15 is a simulation of software module after complete objects association and noise cancelling, each square represents an individual wooden block and we assign each block a different color:

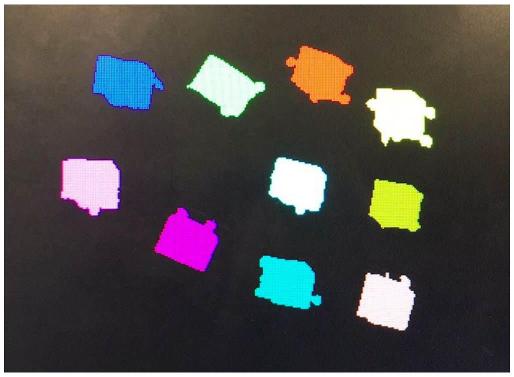
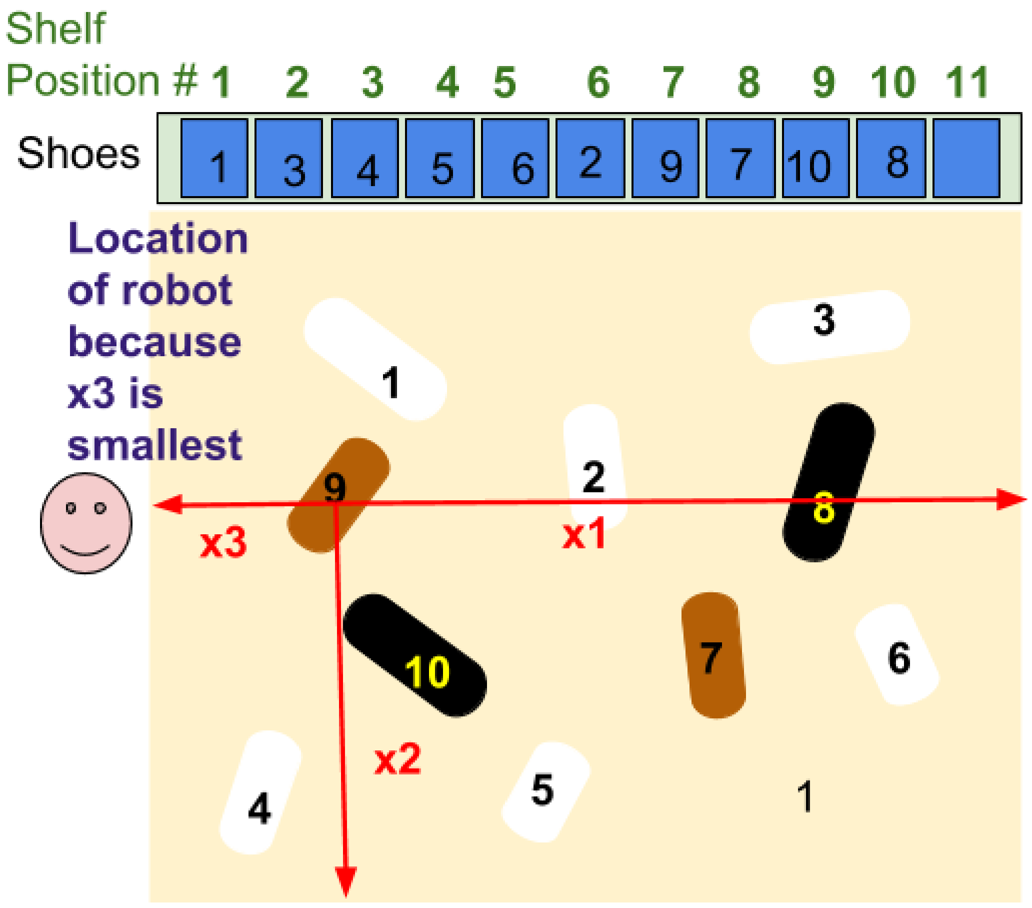


Figure 15. simulation after object recognition and noise cancelling

2.4 Tolerance Analysis

We will assign four properties to each shoe: **color, object number, weight,** and **check condition {if Paired}**. In our experiment setting(figure 16), we are going to have five pairs of shoes, 3 pairs of white, 1 pair of black shoes, and 1 pair of brown shoes. The color will be recorded based on its RGB value, thus the color and object number will be assigned before the robot starts working.



Shelf
Position # 1 2 3 4 5 6 7 8 9 10 11
Shoes
1 3 4 5 6 2 9 7 10 8
Location of robot because x3 is smallest
1
3
9
2
8
x3
x1
10
7
6
x2
4
5
1

Figure 16. Experiment Setting

Each time the robot picks up a shoes, it will record the weight and mark the shoe as PAIRED or NOT PAIRED. The robot will organize shoes based on their color and pairs of same color will be placed next to each other.

The flowchart for the entire shoe organizing process is shown in figure 17. In our project, we believe that the most critical requirement is the ability for the robot to get to the shoe. We will calculate the most effective path our robot should follow by using Inverse Kinematics. Inverse Kinematics makes use of the kinematics equations to determine the joint parameters that provide a desired position for each of robot end-effectors. That is, given location of the shoe on the mat, we will provide three angles theta1,2,3 for robot arm's three joints so that robot can pick up the shoe.

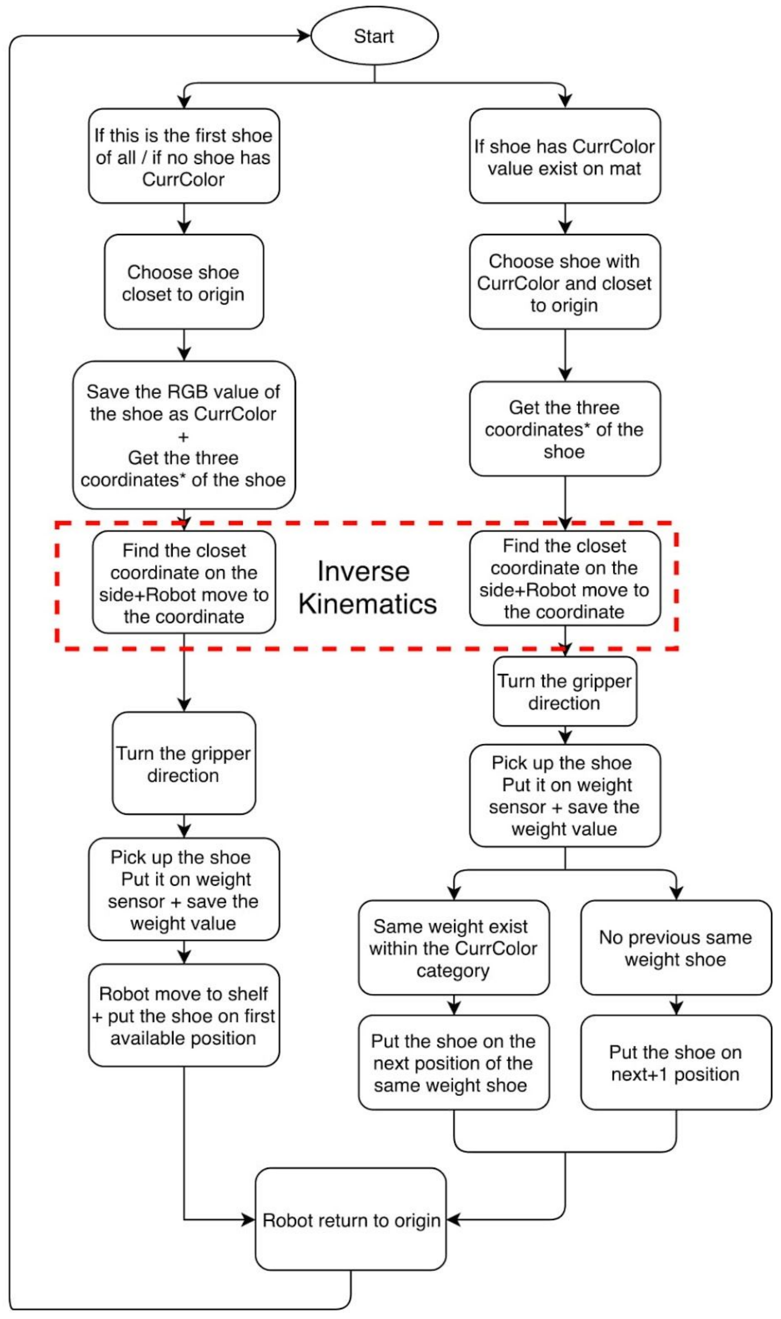

Start
If this is the first shoe of all / if no shoe has CurrColor
If shoe has CurrColor value exist on mat
Choose shoe closet to origin
Choose shoe with CurrColor and closet to origin
Save the RGB value of the shoe as CurrColor + Get the three coordinates* of the shoe
Get the three coordinates* of the shoe
Find the closet coordinate on the side+Robot move to the coordinate
Inverse Kinematics
Find the closet coordinate on the side+Robot move to the coordinate
Turn the gripper direction
Turn the gripper direction
Pick up the shoe Put it on weight sensor + save the weight value
Pick up the shoe Put it on weight sensor + save the weight value
Same weight exist within the CurrColor category
No previous same weight shoe
Robot move to shelf + put the shoe on first available position
Put the shoe on the next position of the same weight shoe
Put the shoe on next+1 position
Robot return to origin

Figure 17. Algorithm flowchart

Map of shoes with four properties

Color	**Object Number**	**Weight**	**Paired**
A	1	360	NO
A	2	420	NO
A	3	300	NO
A	4	420	YES
A	5	360	YES
A	6	300	YES
B	7	250	NO
C	8	450	NO
C	9	450	YES
B	10	250	YES

In the sorting algorithm, the shoes coordinates and the properties are calculated by Raspberry Pi first and are transmitted to ATMega328p through the Bluetooth module. Then, the algorithm choose a shoe to pick up. Before the robot picks up the shoe, it will move to the point closest to the shoe, and the distance between our robot and the shoe is **d** as marked on the figure below. L1 and L2 represent the fixed robot length in the side view. In addition, everytime the gripper tries to pick up a shoe, it lowers to 5cm directly above the mat and then pick up the shoe. Theta1, theta 2 and theta 3 are the joint angles as illustrated in the figure 18.

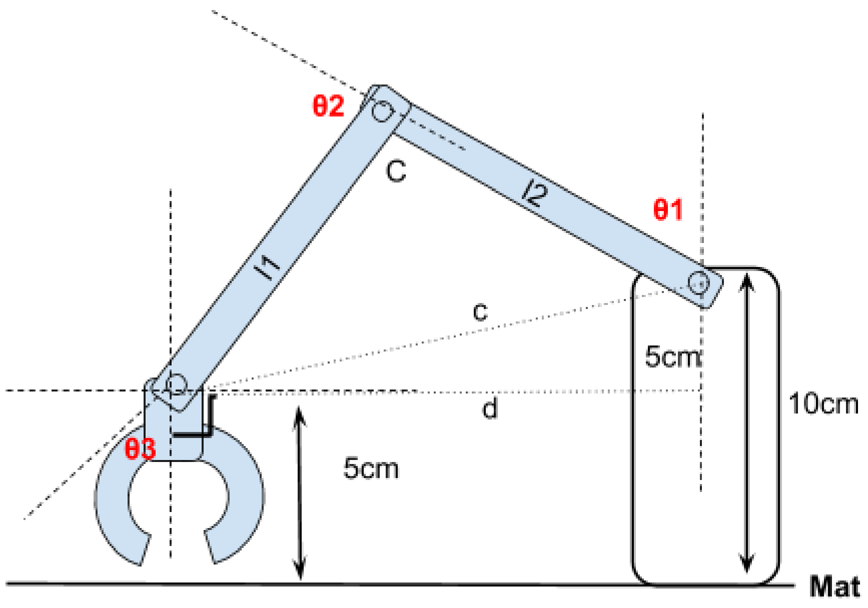


Figure 18. Robot arm side view

$$c = \sqrt{5^2 + d^2} \tag{1}$$

$$c^2 = l_1^{\,2} + l_2^{\,2} \tag{2}$$

$$\angle C = cos^{-1}\left(\frac{-c^2+l_1^{\,2}+l_2^{\,2}}{2l_1 l_2}\right) \tag{3}$$

$$\theta 2 = 180° - \angle C \tag{4}$$

$$\theta 1 = \square - [tan^{-1}(\tfrac{d}{5}) + cos^{-1}(\frac{-l_1^{\,2}+l_2^{\,2}+c^2}{2\,c\,l_2})] \tag{5}$$

$$\theta 3 = 180° - \theta 2 - \theta 1 \tag{6}$$

Using equations (1) to (6) and doing inverse kinematics calculations, we can solve the joint parameters and pass these parameters to the end effector.

3. Cost and Schedule

3.1 Cost Analysis

3.1.1 Parts and Components Cost

Description	**Quantity**	**Unit Price**	**Cost**
A4988 Stepper Motor Driver Carrier	2	5.95	11.9
CSTCE16M0V53-R0 16MHz Ceramic Oscillator	1	0.5	0.5
100 uF Electrolytic Capacitor	2	0.35	0.7
ATmega328-P	1	2.01	2.01
Raspbery Pi 3 Model B	1	35.8	35.8
Energizer A23 Battery	4	9.71	38.84
Energizer A23 Battery Holder(2 pc)	1	3.18	3.18
Raspberry Pi Camera Module V2	1	29.95	29.95
Raspberry Pi 3b Power Supply	1	10.99	10.99
9V Voltage regulator	1	0.15	0.15
5V Voltage regulator	1	1.59	1.59
6.4V Voltage regulator	1	1.21	1.21
Robot arm+shipping	1	163.35	163.35
Gripper	3	22.5	67.6
Digital load cell	1	11.29	11.29
Robot chassis	1	98.99	98.99
Total			478.08

3.1.2 Labor Cost

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

Team Member	**Hourly Rate**	**Hours**	**Cost ✕ 2.5**
Jinghan Guo	39 $/hr	180 hrs	17550
Quanhua Huang	39 $/hr	180 hrs	17550
Mingxi Zou	39 $/hr	180 hrs	17550
Total			52650

3.1.3 Grand Total

Grand Total = Parts and Components Cost + Labor Cost = 478.08 + 52650 = 53128.08 dollars

3.2 Schedule

Week	Jinghan	Quanhua	Mingxi
2/25	Prepare for Design Review	Prepare for Design Review	Prepare for Design Review
3/4	Help verify all the components and begin the software module with Quanhua.	Begin the framework for software module.	Purchase needed components and begin verifying that all components work properly.
3/11	Complete software module to receive image, filter and raster scan images.	Work on software and bluetooth module to generate instructions to microcontroller.	Complete verification process of all needed components.
3/18	Work with Quanhua and Mingxi on assembling the physical design and hooking up the electrical components.	Start assembling the physical design .	Start hooking up the electrical components.
3/25	Complete physical and	Complete physical and	Complete physical and

	electrical design	electrical design	electrical design
4/1	Start final testing of the complete system and focus on any issues related to image processing	Start final testing of the complete system and focus on any issues related to physical design of robot.	Start final testing of the complete system and focus on any issues related to data transfer from Raspberry Pi
4/8	Iron out any software application and data transfer performance issues	Stabilize physical design by making sure that all physical components and circuitry are safely secured and demo-ready.	Iron out any performance issues related to physical components and circuitry
4/15	Do numerous executions for testing	Do numerous executions for testing	Do numerous executions for testing
4/22	Complete entire system check and begin final report	Complete entire system check and begin final report and final presentation	Complete entire system check and begin final presentation
4/29	Finish final report	Finish final report and final presentation	Finish final presentation.

4. Ethics and Safety

4.1 Ethics

After reading the IEEE Code of Ethics[1] and the ACM Code of Ethics and Professional Conduct[2], I think our project is compliance with all the Ethics requirements mentioned above. In our design, the shoes sorting robot is small and does not take much space. The maximum height of the robot is about 55.5cm but usually it only takes around 30cm. In addition, the robot will only pick shoes and will not harm people. The speed of the car chassis is slow and the voltage that needed for this robot is only 5 Volts, which is acceptable for human body and will not hurt others. The IEEE Code of Ethics #9, “to avoid injuring others, their property, reputation, or employment by false or malicious action”, also support this idea. In addition, we will try our best to reduce any possible risk when using this robot, as according to IEEE Code of Ethics #1, “to hold paramount the safety, health, and welfare of the public, to strive to comply with ethical design and sustainable development practices, and to disclose promptly factors that might endanger the public or the environment”[1].

4.2 Safety

There are two underlying safety issues that we will face. The first one is that in our robot design, we need to convert 12V power to 5V 3A by a regulator. The 3A current might be relatively high with respect to human being and people will get hurt easily by improper touching. The maximum current an average man can grasp and let go is 16 mA and 2A current can cause cardiac standstill and internal organ damage.[3] In addition, the Lithium Battery with a poor wiring and bad connection will lead to safety issues as well. Therefore, we must make sure that the connections between elements are stable and are in good conditions.

The second issue is the risk of soldering. Soldering can be dangerous if we inappropriately manipulate the soldering tools. Also we need to be careful of the high temperature during soldering and remember to turn the tools off when we are done.

5. References

[1] “IEEE Code of Ethics”, Ieee.org. (2019).[online]. Available:
https://www.ieee.org/about/corporate/governance/p7-8.html [Accessed 5 Feb. 2019].

[2] Acm.org. (2019). The Code affirms an obligation of computing professionals to use their skills for the benefit of society. [online] Available:
https://www.acm.org/code-of-ethics [Accessed 5 Feb. 2019].

[3] Raymond M. Fish, L. (2019). Conduction of Electrical Current to and Through the Human Body: A Review. [online] PubMed Central (PMC). Available at:
https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2763825/ [Accessed 7 Feb. 2019].

[4] Raspberry Pi 3b
https://www.terraelectronica.ru/pdf/show?pdf_file=%252Fds%252Fpdf%252FT%252FTechicRP3.pdf

[5] Raspberry Pi
https://static.raspberrypi.org/files/product-briefs/Raspberry-Pi-Model-Bplus-Product-Brief.pdf

[6] Energizer A23
http://data.energizer.com/pdfs/a23.pdf

[7] Energizer A23 Battery Holder
https://www.amazon.com/Featurestop-Environmentally-Friendly-Battery-Holder/dp/B073QL1GP2/ref=sr_1_11?keywords=a23+battery+holder&qid=1550706818&s=gateway&sr=8-11

[8] Camera
https://cdn.sparkfun.com/datasheets/Dev/RaspberryPi/RPiCamMod2.pdf

[9] Raspberry pi 3b power supply
https://www.amazon.com/Miuzei-Raspberry-Heatsinks-Supply-Compatible/dp/B07BTHNW9W/ref=sr_1_7?keywords=raspberry+pi+3b+5V&qid=1550715692&s=gateway&sr=8-7

[10] 9V Voltage regulator
https://static1.squarespace.com/static/5416a926e4b09de8832655bc/t/54427037e4b03de3b67b895a/1413640247188/lm7809.pdf

[11] 6.4V voltage regulator / 5V voltage regulator
https://www.mouser.com/datasheet/2/792/r1524-ec-1099548.pdf

[12] salary info
https://engineering.illinois.edu/documents/Salary.Info.Sheet.pdf

[13] ECE 470 raster scan reference
http://coecsl.ece.illinois.edu/ece470/lab.pdf