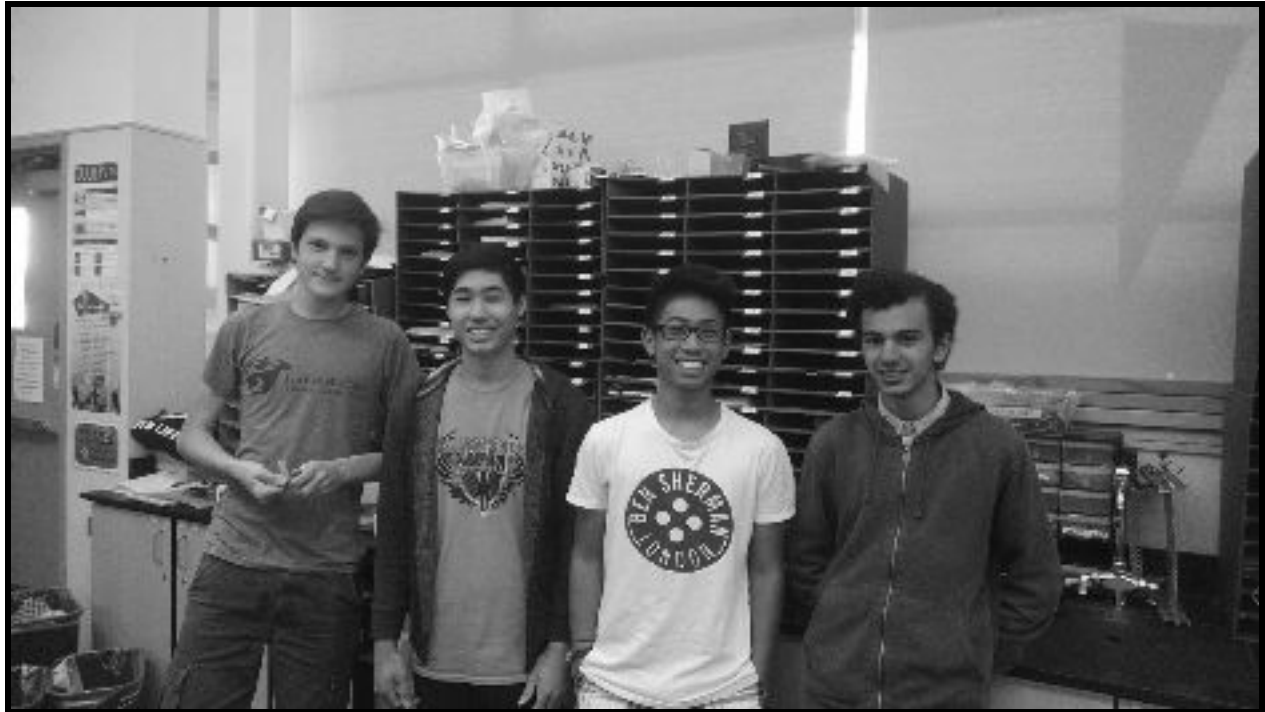


Principles of Engineering 2016/2017



Project 3.2.4 Machine Control Design

Ashir Bhalla-Levine, Noah Boursier, Ethan DeGuzman, Will Lavanakul

Table of Contents

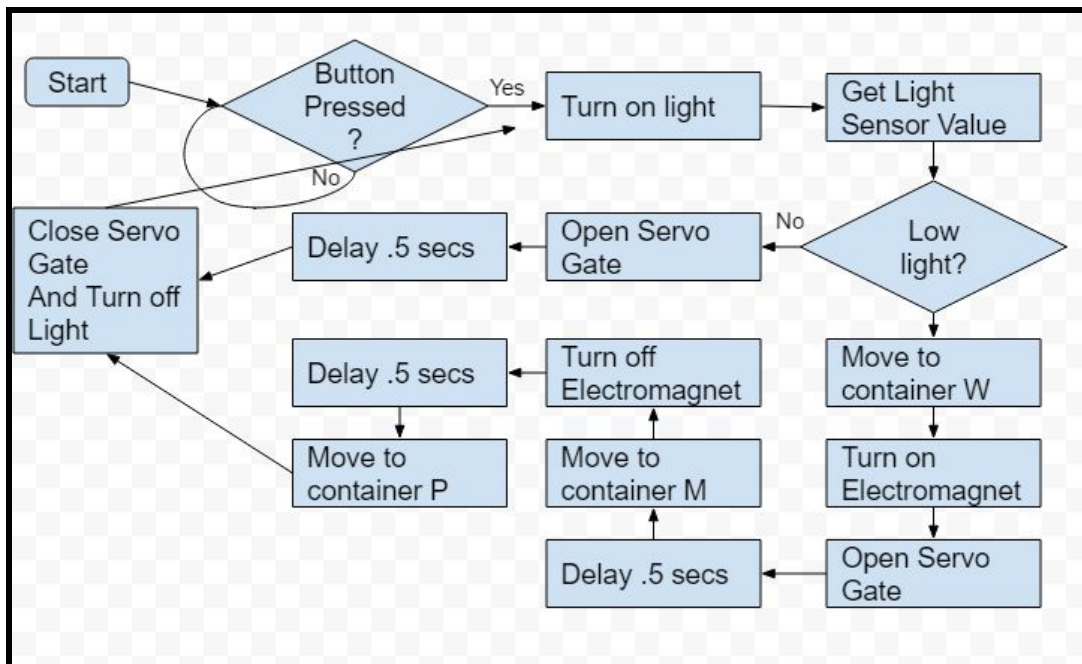
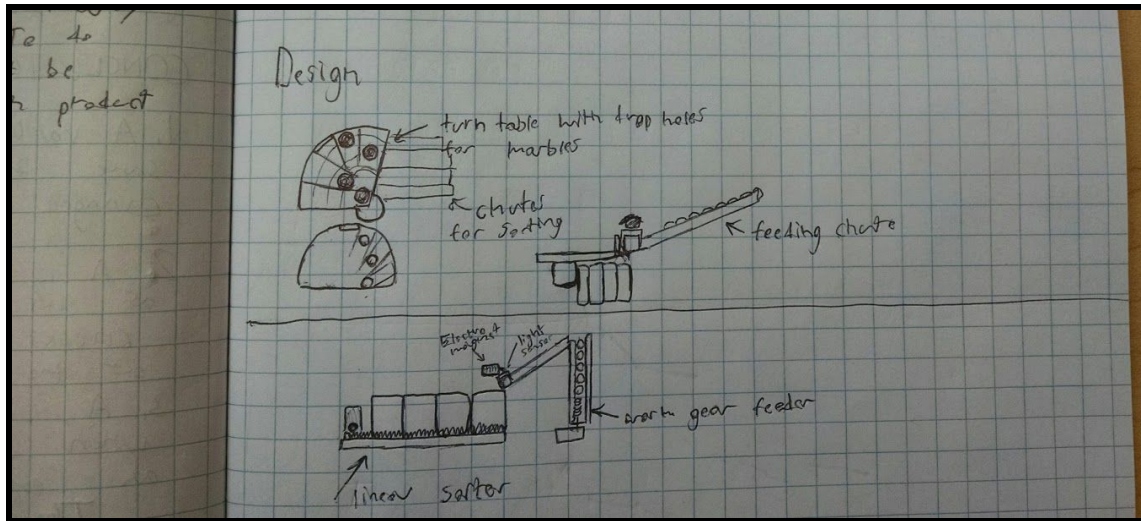
Item	Page Number
Design Brief	2
Brainstorming Sketches & Flowcharts	3-6
Decision Matrix	7-8
Design Modification	9-11
Final Physical Solution Images & Description	12
Final Program Solution & Description	13

Design Brief

Client	Aaron Metals, Recology, and Waste Management
Designers	Ashir Bhalla-Levine, Noah Boursier, Ethan DeGuzman, Will Lavanakul
Target Customer	The target customer for a product such as this would be a large scale garbage and recycling corporation to use in daily sorting operations.
Problem Statement	Aaron Metals needs a system which can reliably sort between different types of waste such as trash, recycling, and metals.
Design Statement	We will design and construct a simple mechanism in order to best accomplish the task with the fewest possible errors in sorting. Our machine will be able to differentiate between clear, wooden, and metal test samples before sorting them separately for appropriate disposal.
Constraints	Limited to only Fischer Tech and VEX parts when building.
Deliverables	<p>Team:</p> <ul style="list-style-type: none"> • Group Responsibility Form & Gantt Chart • Electronic document titled P3_2_4_LastNames, containing the following: Title Page/Table of Contents, Design Brief, Brainstorming Sketches/Program, Decision Matrix/Descriptions, Modification Sketches/Descriptions, Final Physical Solution, and Final Program Solution <p>Individual:</p> <ul style="list-style-type: none"> • Complete all items in your notebook: Design Brief Notes, Brainstorming Ideas, and Conclusion Question

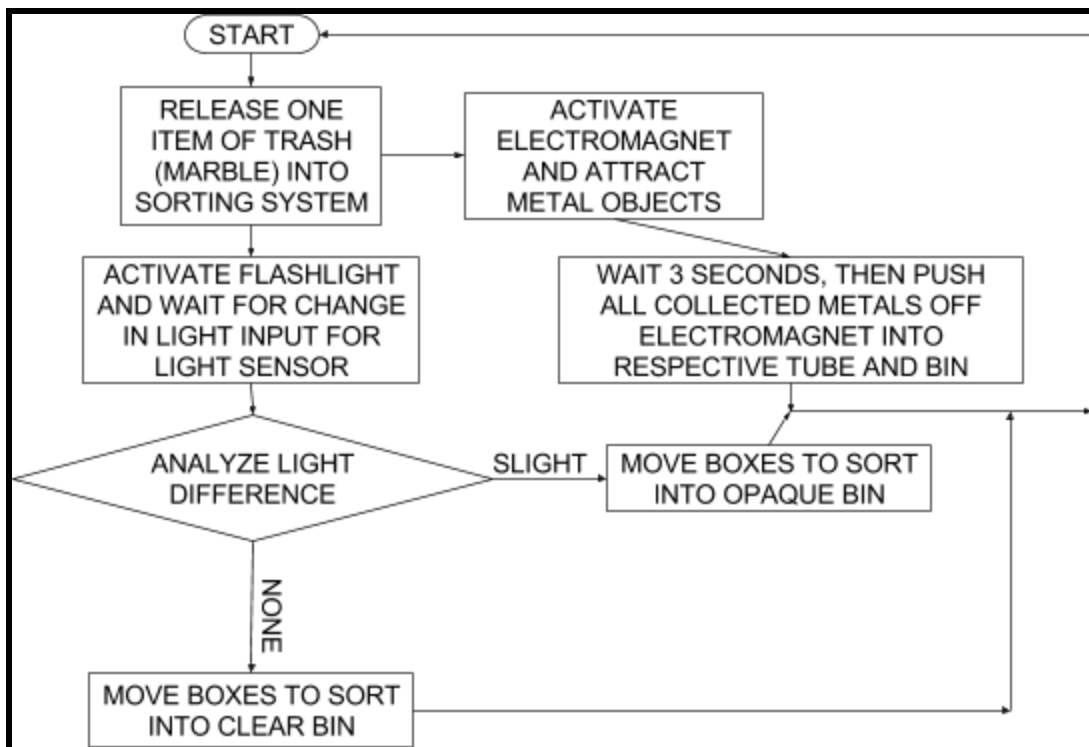
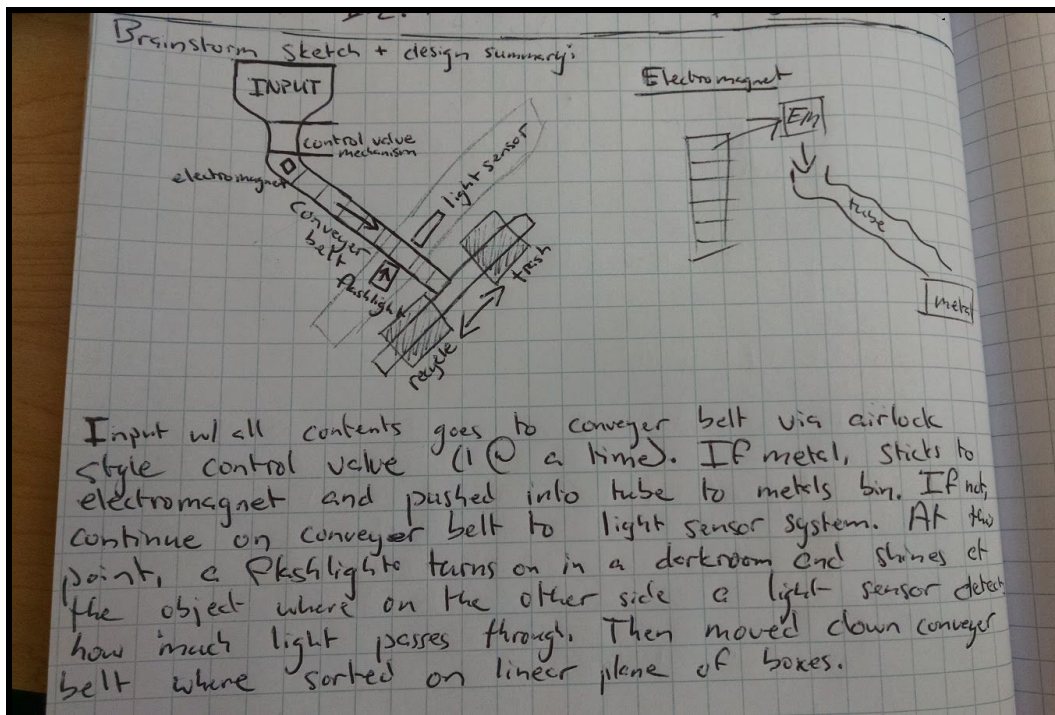
Brainstorming Sketches & Description

Noah's Design & Description

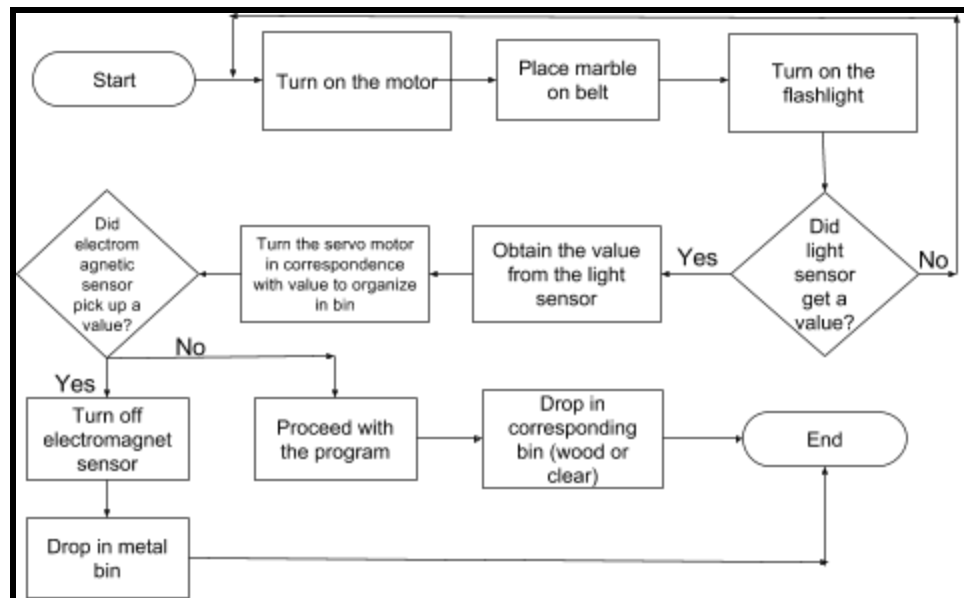
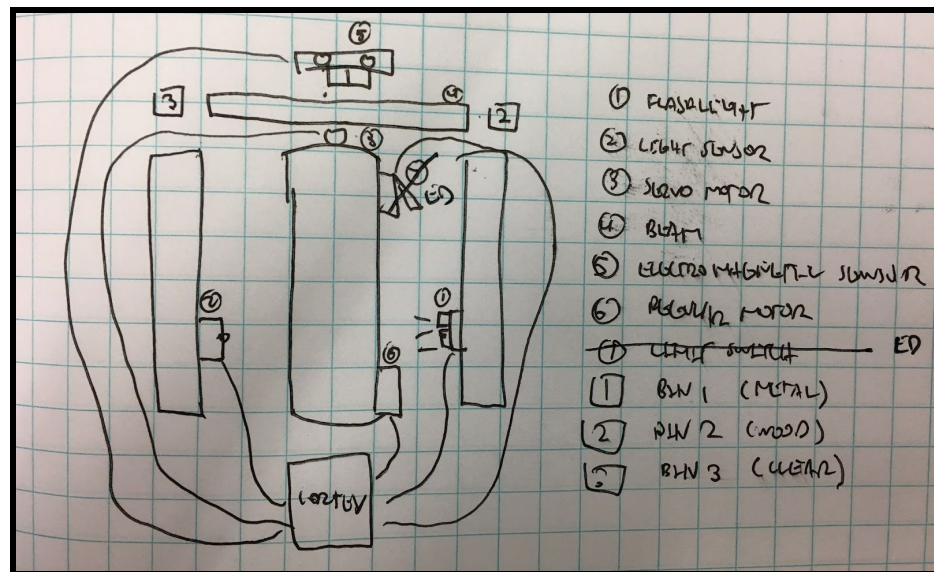


Sensor system utilizes a light sensor and and light that can be interrupted by marble. To separate the steel and wood from each other an electromagnet is used. A chain system that moves using one motor is used to manipulate the containers in which the marbles are dropped.

Ashir's Design & Description

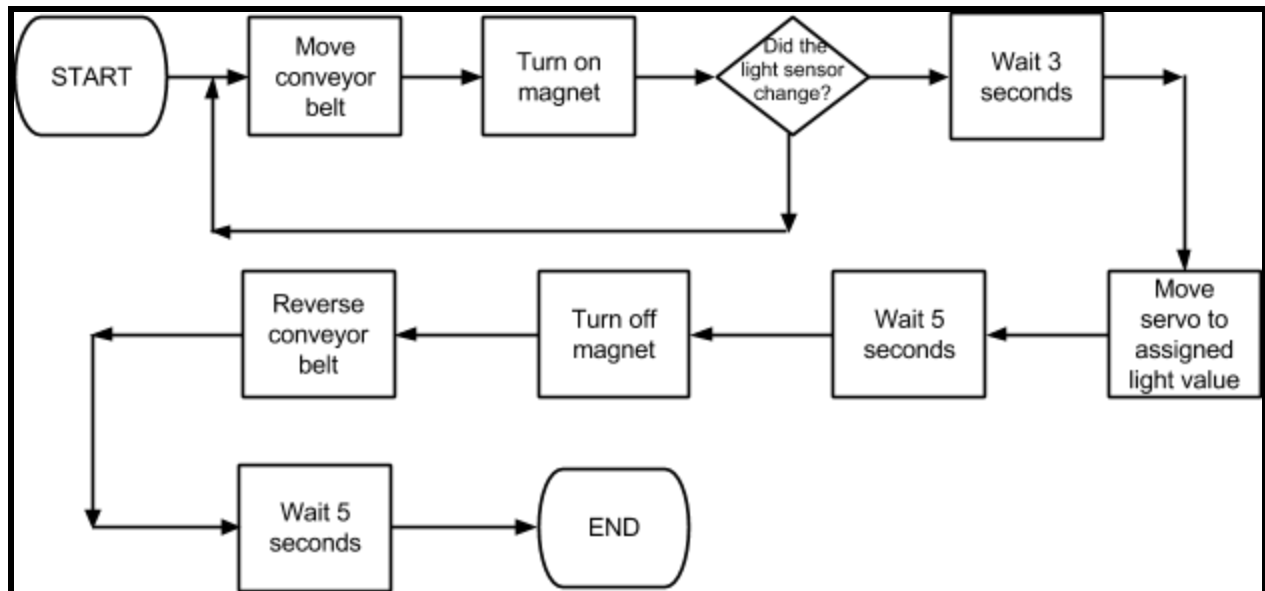
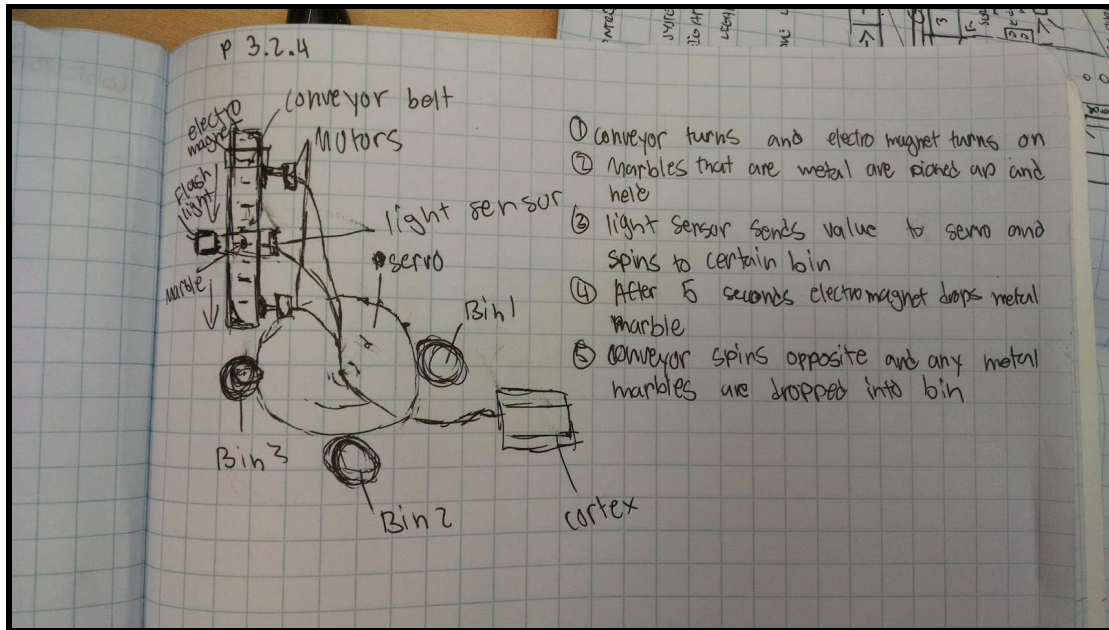


Ethan's Design & Description



Description: The program starts when you turn on the motor. After placing a marble on the conveyor belt, it will be taken through until it reaches the middle of a flashlight and a light sensor. The flashlight will turn on and a value will get received by the light sensor. The value from the light sensor will turn the servo motor which is attached to a beam. Depending on the value received from the light sensor, it will turn the servo motor based off of the range values for each marble bin. If the marble is metal, before it drops to the beam attached to the servo motor, it will stick to the electromagnetic sensor. This overrides any movements of the servo motor since it is attached to the electromagnet. When attached to the electromagnet, it will then turn off which drops the marble, sending it to the metal bin. If the marble is not metal, it will proceed with the program.

Will's Design & Description



This machine starts off with a conveyor belt that transports marbles to a flashlight and light sensor. The light sensor picks up a certain value and moves a servo to 2 designated bins. Before the marbles reach the flashlight, an electromagnet picks up any metal marbles passing through. After the servo turns to the designated bins, the conveyor belt then transports the wood or plastic marbles to the bin. Shortly after, the electromagnet turns off and reverses the conveyor belt. This drops the metal marbles into the last bin behind the machine sorting all the marbles into 3 bins.

Design Matrix

*scored 1-5, worst-best, respectively

	Simplicity	Parts	Plausibility	Aesthetics	Speed	Total
Noah	4	4	5	3	3	19
Ashir	3	3	5	3	3	17
Ethan	4	4	5	2	3	18
Will	3	3	5	4	3	18

X Eugene Chou 04/27/2017

Assignment of Values:

Simplicity: We scored each brainstorm idea on the amount of moving parts or mechanisms required. The less amount of parts used with a good functionality was scored highly on the matrix.

Parts: In correlation to the simplicity aspect of the design, the parts were graded by the number and type of parts we thought were needed for the design to be created. A low score would be barely using any of the VEX parts used for programming, and a high score would be using the correct amount of VEX parts for programming. This is of course without any excess, or unneeded parts in the design.

Plausibility: This was scored based on the plausibility of the system working for the given task, which was to create a design to organize marbles. A low score would be having a design that completely does not follow the minimum requirements of the system, and a high score showing signs of functionality and ability to carry out the needed tasks.

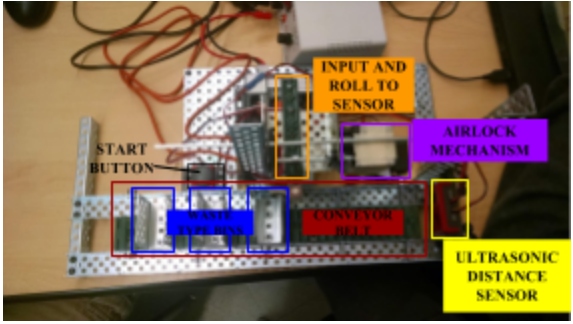
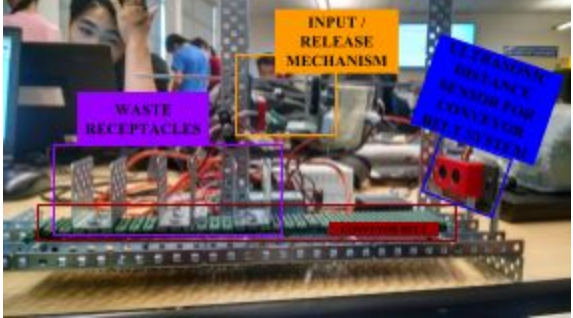
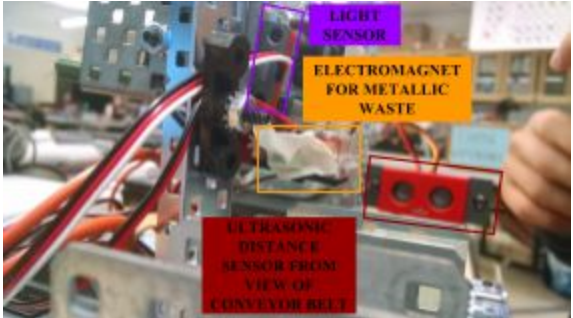
Aesthetics: The aesthetics of each design were graded depending on how we believe it would look when it was complete. We also looked for uniqueness in each design and how it stood out from the others which made it a top score. Having a low score meant that it would look very plain with no unique features, or ideas.

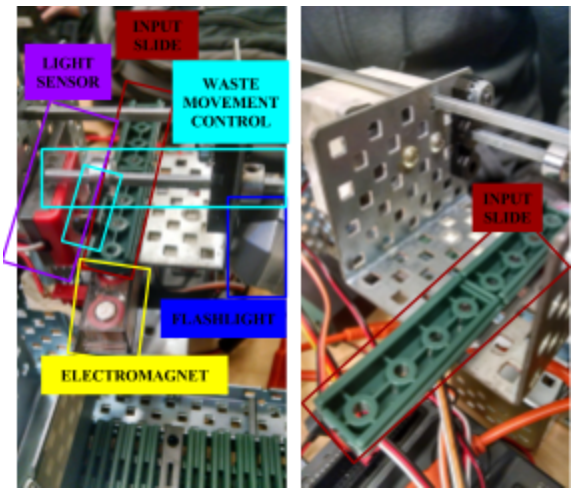
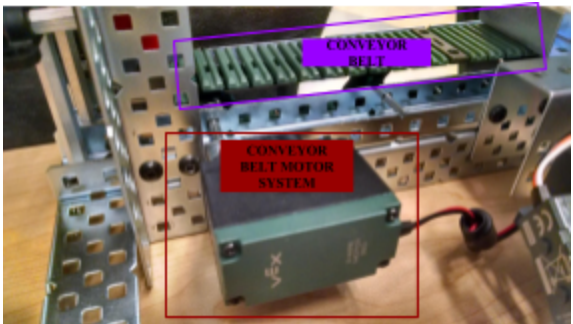
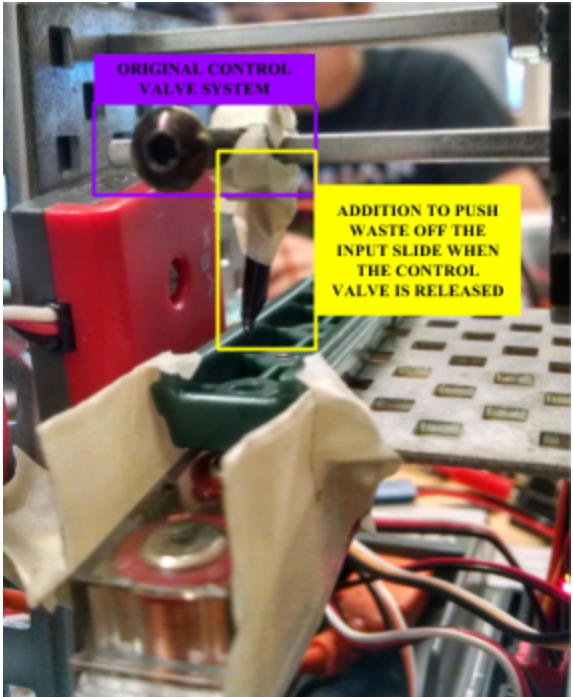
Speed: This was scored by how fast each design would be able to run through its tasks. We wanted a design that would be able to sort each and every marble efficiently with the right amount of speed. We weren't looking for something that we felt would be too slow or fast of a process. A high score would be a perfect run time speed, and a low score would be either too fast or slow if a run time.

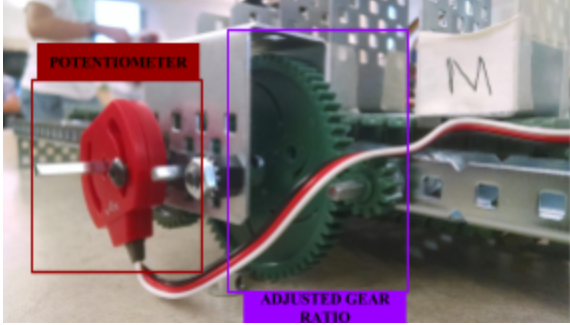

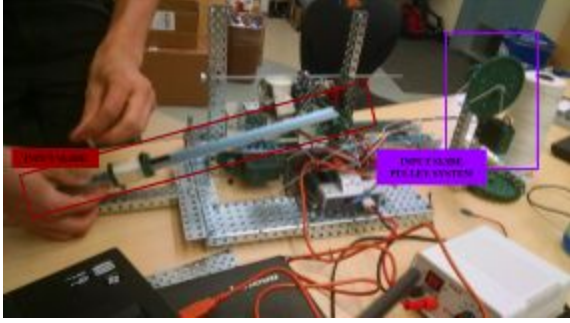
Explanation:

After all aspects of each brainstorm design were taken into consideration and graded using our decision matrix based off of our assignment of values, we came to the conclusion that Noah's brainstorm design was the best one. We ranked the simplicity as a 4 because of how simple the design was, and how there weren't that many parts used with a good functionality. The parts were graded as a 4 because the right amount of VEX programming parts were in use. There were not too many parts in use which contributed to its simple design. His design mainly consisted of the following programming parts: flashlight, light sensor, servo motor, and an electromagnetic sensor. The aesthetics were graded as a 3 because in comparison to other design in the group it was not as aesthetic. However, it still contained unique qualities on its own, such as the use of a feeding shoot for the marbles to drop into the design one at a time. Finally the speed was ranked at a 3, because just like the grading of the aesthetics, in comparison to other designs, it was not as fast, due to the fact that it encountered multiple delays in time and numerous processes.

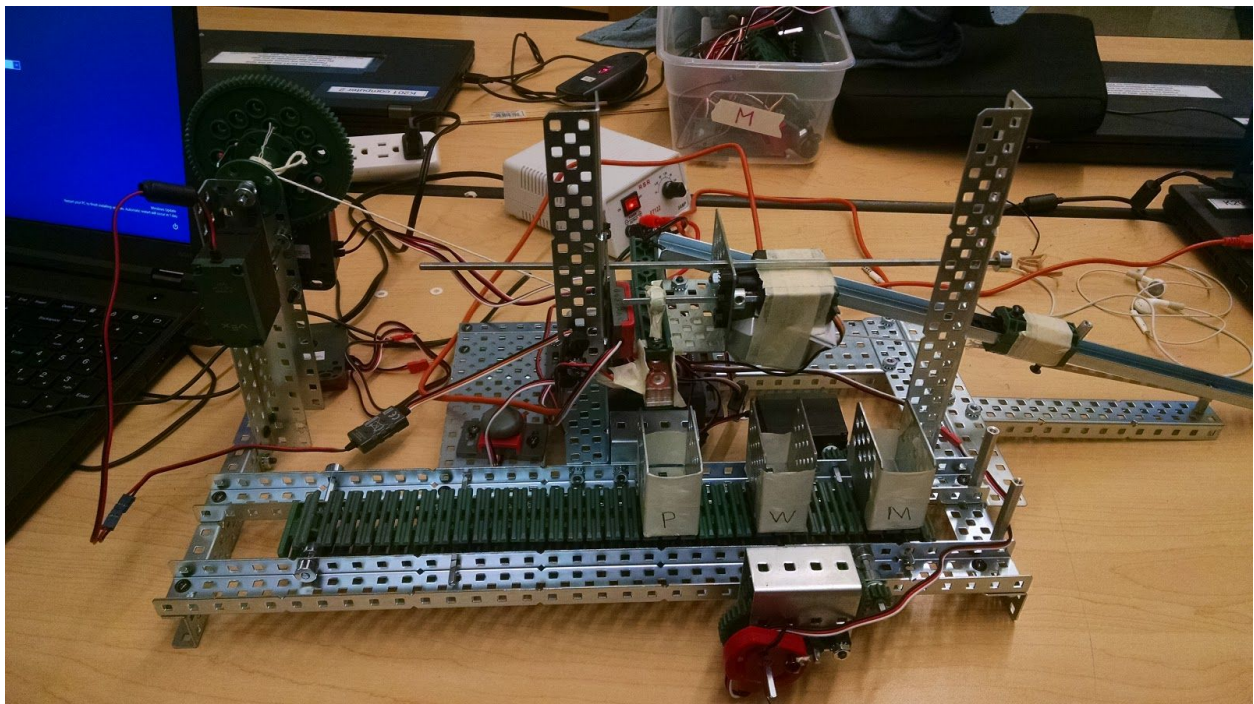
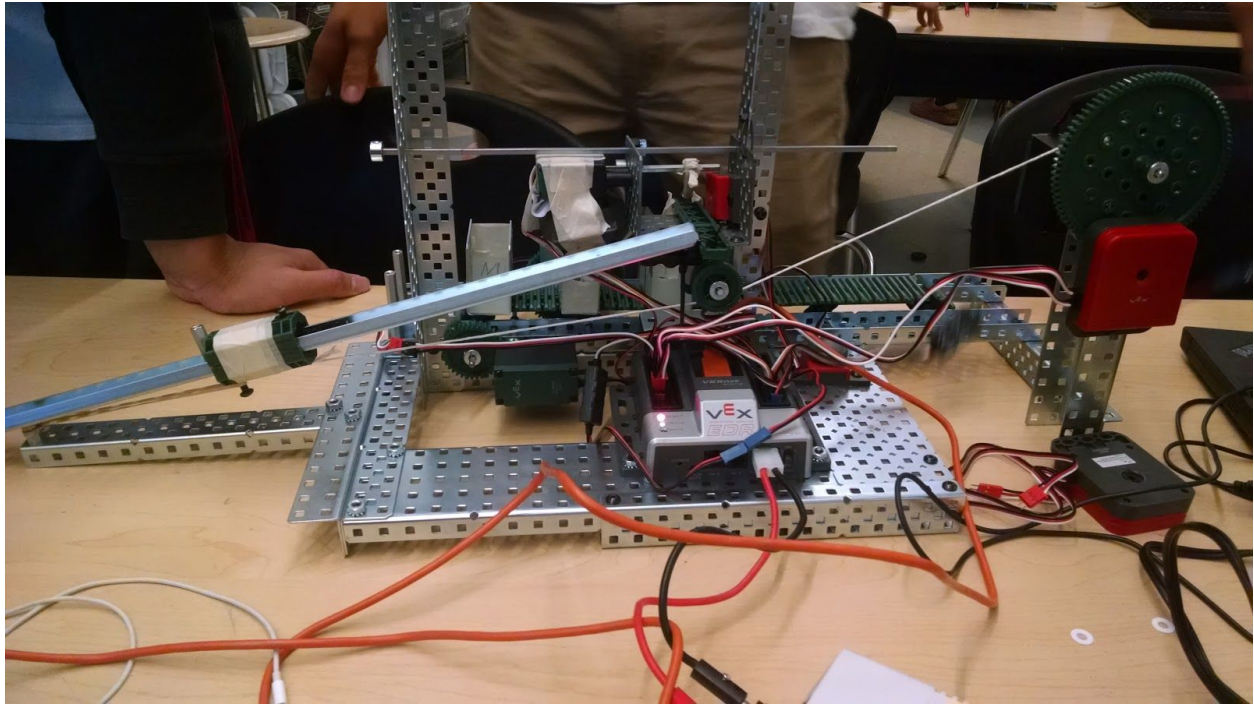
Design Modifications

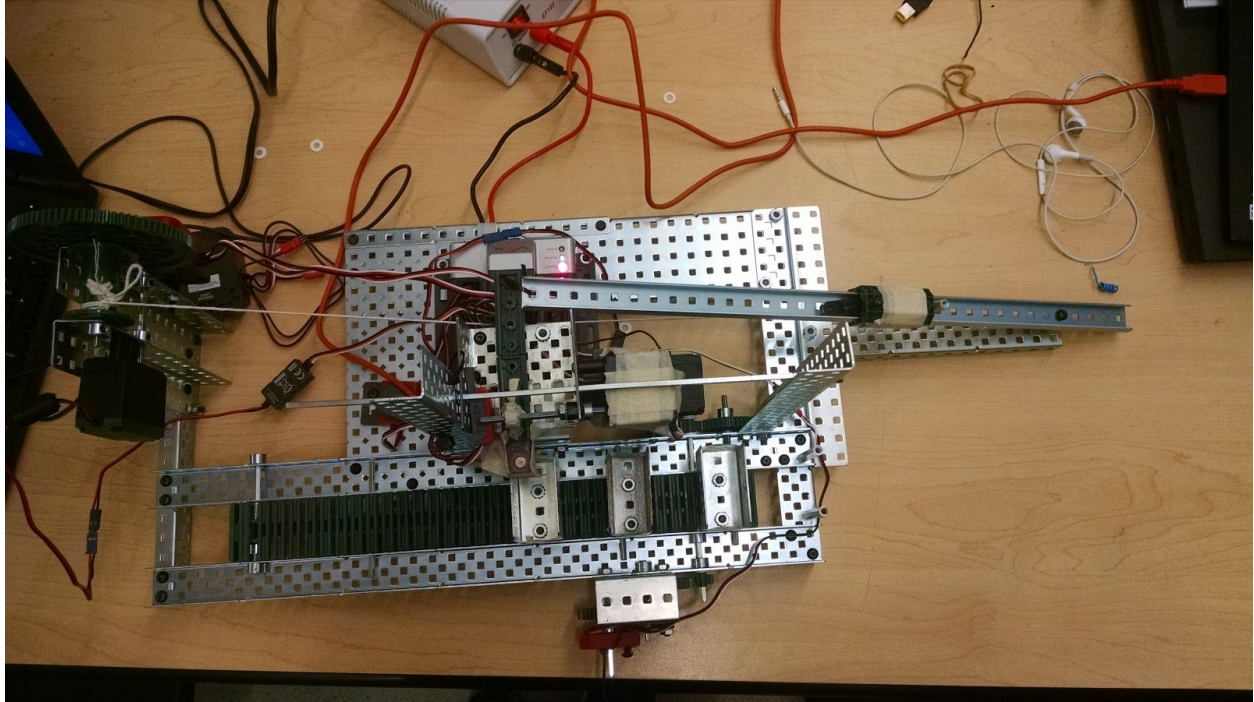
Image	Description / Changes	Initials
	Initial design overhead with no changes.	ABL ED 05/01/17
	Initial design from side with no changes.	ABL NB 05/01/17
	Closeup of the sorting mechanism from the original design using light sensors and electromagnets to differentiate between different types of waste material. When light sensor picks up a certain value, spin conveyor belt to certain position and spin servo release marble.	ABL WL ED 05/01/17 5/4/17

	<p>Closeups of input and sorting mechanism displaying the route of travel of waste in the system to the final receptacles. Shows the use of the VEX flashlight, light sensor, electromagnet, and servo motor. Marbles go through input slide to servo.</p>	<p>ABL ED WL 05/01/17</p>
	<p>Original design conveyor belt system with motor and melt itself.</p>	<p>ABL ED 05/01/17</p>
	<p>Updated control valve system with piece to help push the waste items off the input slide and into their respective bins. We noticed that not all the types of waste had the momentum to roll off the input slide after the control valve is released, so we added this piece in order to give it an extra push along with the control valve being released.</p>	<p>ABL ED 05/01/17</p>

<pre> wait1Msec(500); } else //if { motor[mot] = -127; //tut waitUntil(SensorValue[pot] < 3150-tweak); //tut motor[mot] = 0; //tut wait1Msec(500); //wai motor[electromagnet] = 127; //tut motor[servo] = 80; //c wait1Msec(750); //wai motor[mot] = -127; //tut waitUntil(SensorValue[pot] < 1850-tweak); motor[mot] = 0; //stc wait1Msec(500); //wai motor[electromagnet] = 0; //tut motor[mot] = 127; //tut waitUntil(SensorValue[pot] > 3950-tweak); motor[mot] = 0; //stc wait1Msec(500); //wai </pre> 	<p>Switched from ultrasonic sensor to potentiometer due to the inaccuracy of the ultrasonic sensor in comparison. Also using the potentiometer with a decreased gear ratio in order to stop the conveyor belt at the end of the track. Included tape to indicate which bins have which marbles. We also changed from ultrasonic sensor to the potentiometer in our program as well.</p>	<p>ABL ED WL 05/02/17 5/4/17</p>
	<p>Adjusted the conveyor belt gear ratio for torque in order to move more smoothly.</p>	<p>ABL ED 05/02/17</p>
	<p>Added an extended input slide with servo motor pulley system in order to allow for more than one marble/piece inputted at a time.</p>	<p>ABL NB 05/04/17</p>

Final Physical Solution Images & Description





Description:

Our final design was intended to include the ability to sort as many as 20 or more marbles, without human interaction, however, due to issues that needed to be solved before the presentation date that were more pressing, we were unable to finish the feeding trough. Our design uses a button, light sensor, flashlight, servo motor, 393 motor, and an Fisher Tech electromagnet. To operate our machine, a user must simply place a marble on the intake ramp, and press the button to begin the process. Our machine sorts the marbles, differentiating between wood, plastic, and metal by first testing to see if they interrupt light. To do this, we have a light sensor that is opposite the flashlight, and when the marble is stopped by the servo gate in between the two, it will either interrupt the light and give our sensor a higher value or, if it is plastic, give us a lower value, as the light is not interrupted as much. Then, depending on whether the marble is opaque or not, we can tell if it is plastic, or something else, if it is plastic, it is dropped into the first bin. To differentiate between metal and wood, we first have the bin conveyor belt, which utilizes a 393 motor, move to the second (wood) bin, the servo gate will release the marble which is either wood or metal at this point. If the marble is wood, it will simply fall through to the wood bin, however if it is metal, the electromagnet will be on, and catch it, holding as the bin conveyor positions the metal bin below. Once the metal bin is below the electromagnet will turn off, and release the metal marble into the respective bin. After this process, the bins will return to starting position where once the start button is pressed, the process can be repeated an infinite number of times.

Final Program Solution & Description

```
#pragma config(Sensor, in1,    lightsensor,    sensorReflection)
#pragma config(Sensor, in2,    pot,              sensorPotentiometer)
#pragma config(Sensor, in3,    pot2,           sensorPotentiometer)
#pragma config(Sensor, dgt11,  startbutton,   sensorTouch)
#pragma config(Motor,  port1,    light,         tmotorVexFlashlight, openLoop, reversed)
#pragma config(Motor,  port2,    feeder,        tmotorServoStandard, openLoop)
#pragma config(Motor,  port3,    servo,         tmotorServoStandard, openLoop)
#pragma config(Motor,  port4,    mot,          tmotorVex393_MC29, openLoop)
#pragma config(Motor,  port10,   electromagnet, tmotorVexFlashlight, openLoop, reversed)
/**!!Code automatically generated by 'ROBOTC' configuration wizard !!*/
//Start Potentiometer at maximum (~4090)
int tweak = 400; //variable for potentiometer tweaks
int feederpos = -127; //set initial position for feeder servo
task main()
{
  while(true)
  {
    while(SensorValue[startbutton] == 0) //stop doing this loop when button is pressed
    {
      feederpos = ((SensorValue[pot2] * 254) / 4090); //ratio calculation to get a value.
      if(feederpos > 127){ //converting value from ratio calculation into a value the servo can interpret.
        feederpos = ((feederpos - 127) * (-1)); //converting value from ratio calculation into a value the servo can interpret.
      }
      else{ //converting value from ratio calculation into a value the servo can interpret.
        feederpos = (127 - feederpos); //converting value from ratio calculation into a value the servo can interpret.
      }
      motor[feeder] = feederpos; //sets the feeder servo to the value that was calculated and converted.
    }

    while (feederpos < 121) //while the position of the feeder is non-compromising
    {
      motor[servo] = -160; //closes the servo, stoppig marble
      motor[light] = 0; //turns off the light if it is on.

      motor[feeder] = (feederpos + 6); //increment feeder servo to add one marble.
      wait1Msec(1000); //wait one second

      if(SensorValue[startbutton] == 1) //emergency loop stopper
      { break; } //stops the loop if button is pressed.

      motor[light] = 127; //turns on the light so that the light sensor has something to look at.
      wait1Msec(750); //wait for things to settle.

      if(SensorValue[lightsensor] < 200) //if the light sensor detects a value less than 200, which means it is seeing the light through a plastic marble..
      {
        motor[servo] = 110; //open the servo gate and let the marble drop into the plastic marble bin.
        wait1Msec(750);
      }
      else //if the light sensor value is not less than 200 which means the marble is wood or steel...
      {
        motor[mot] = -127; //turn on the bin motor, move the bins to the left.
        waitUntil(SensorValue[pot] < 3150-tweak); //continue moving left until the potentiometer reads a value less than 3150, meaning the wood bin is positioned below the marble.
        motor[mot] = 0; //turn off the motor.
        wait1Msec(500); //wait half a second.
        motor[electromagnet] = 127; //turn on the electromagnet to catch the marble if it is metal.
        motor[servo] = 80; //open the servo gate to let the marble either fall into the wood bin or be stuck to the electromagnet if it is steel, which is magnetic.
        wait1Msec(750); //wait 3/4 of a second for the marble to drop.
        motor[mot] = -127; //turn on the bin motor, move the bins to the left.
        waitUntil(SensorValue[pot] < 1850-tweak); //continue moving left until the potentiometer reads a value less than 1850, meaning the steel bin is positioned below the marble.
        motor[mot] = 0; //stop the motor.
        wait1Msec(500); //wait half a second.
        motor[electromagnet] = 0; //turn off the electromagnet, allowing the marble, steel, to fall into the steel bin.
        motor[mot] = 127; //turn on the bin motor, move the bins back to the right.
        waitUntil(SensorValue[pot] > 3950-tweak); //continue moving left until the potentiometer reads a value greater than 4070, meaning the plastic bin is positioned below the marble.
        motor[mot] = 0; //stop the motor.
        wait1Msec(500); //wait .5 seconds
      }
    }
  }
}
```

Our program's first process is the calibration of the marble feeder, which is controlled with a secondary potentiometer so that varying amounts of marbles can be placed into the machine. The second process is the actual sorting, it starts off by making sure the light is off while it dispenses a marble into the sensor bay, where a flashlight will shine at the marble and a light sensor at the other side will test for interruption of light. If it detects less light, meaning the marble is opaque and is either wood or steel, the program will go to the else statement and execute a process that utilizes an electromagnet to differentiate and sort the marbles between the steel in wood containers. If the light sensor detects light then the servo gate will simply release the marble into the plastic bin, as only plastic marbles (that we are sorting) are not opaque. [DOWNLOAD HERE](#)