**ECE 4950 Project 4**

**Team Megafist**

**Group 13**

**Valerie Moran, Brandon Cuttino, Ben Liggett, Sarah Anderson, and Rocky Aho**

**December 11, 2020**

**Executive Summary:**

To begin, Project 4 combines the past three projects to develop an autonomous puzzle-solving robot. To

develop this puzzle-solving robot, we utilized an Arduino, electromagnet, camera sensing system, and a motor. All four of these connected to form a system that would successfully rearrange colored washers on a gameboard. Before combining the four parts, we first read through the customer requirements and developed engineering requirements. We then used these engineering requirements to make a system capable of rearranging washers. Throughout this project, we learned more about combining different electrical components to create a full-scale prototype. In doing so, we learned more about the Arduino Mega, using a camera for background subtraction, electromagnets, and DC motors. Additionally, we learned more about developing a project schedule using a Gantt chart and how to work cohesively in a team setting.

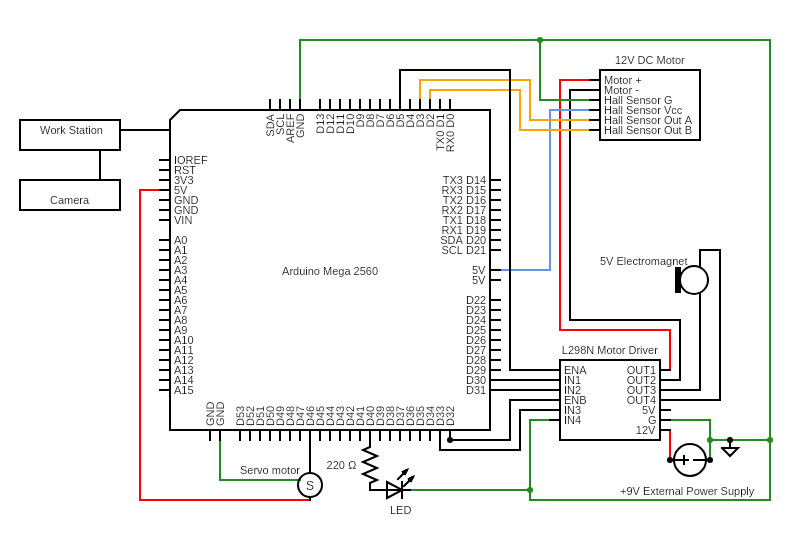
**Engineering Requirements:**

* Inexpensive. Cost less than $300
* System produces consistent results with a smooth motion
* Fast solving time with a solve time under 3.5 minutes.
* Low noise output with no hearing protection required. Less than 60dB
* External power supply used. 9V wall adapter
* The system protects its users with proper safety precautions
* Sturdy system
* Autonomous System
* The design is easily replicated with generic components used
* System uses the provided 5V electromagnet
* System utilizes the provided USB camera for image processing
* System uses the provided DFROBOT DC motor for position control of the gameboard
* System uses an Arduino for closed-loop motor control and to control an electromagnet
* After completion, the system turns on an LED to signal the process is complete
* After completion, the motor turns the game board to a predefined initial position
* User-friendly with easy to use graphical user interface
* The system successfully rearranges a maximum of seven colored washers to match a user-provided configuration.
* System uses background subtraction to identify the position of a maximum of seven washers. Washers can be one of three colors: red, green, or yellow.
* The system moves one washer at a time

**Design Details:**

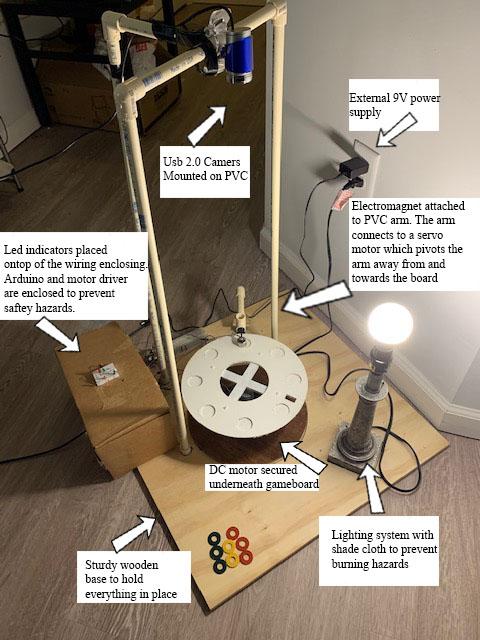
**Hardware Design:**

The complete wiring diagram for the system is shown in **Figure 1a** below. This diagram shows each of the nine components used, including the provided Arduino mega, 12V DC motor, 5V electromagnet, and the USB 2.0 camera. The other components are generic and interchangeable, which makes the system easy to replicate. The other four components are a windows 10 computer with MATLAB and simulink, a 9V external power supply, a servo motor, and LEDs. In total, excluding the computer, the total cost of the build is $115, which is less than the maximum cost of $300.



**Figure 1a: Hardware Wiring Diagram for the Final Design**

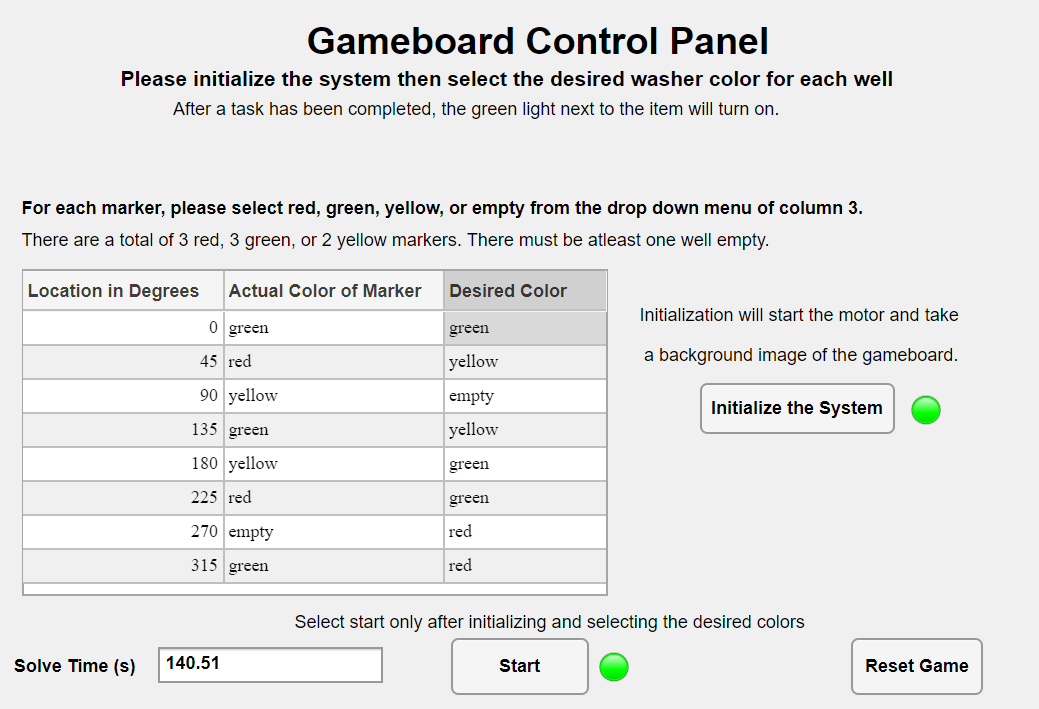
The full design is shown in **Figure 1b**. The frame of the system was built with 2.5-foot sections of ½ inch PVC pipe to make it both sturdy and inexpensive. Each of the PVC joints used was between 10 and 30 cents. The frame was screwed down to a solid wood base to make sure that there are no inconsistencies. The wood base also makes the system sturdy and easy to transport. The DFROBOT DC motor was set inside of a 3d printed base to hold it in place, and this could be replicated by gluing the motor to the base. A small cross was 3d printed to attach the game board to the motor’s shaft. The cross was connected to the game board with bolts, and the middle of the cross was then glued to the motor’s shaft to make sure the game board does not come off. A lazy susan could be used to replicate this connection. The game board and all the visible components were painted the same color, which can help with the image processing portion. Small chips, each with a 16mm radius, were 3d printed and placed inside the wells, and these chips can be replaced with pieces of cardboard. The USB camera was electrically taped to the center of the frame to keep the camera in a consistent location. A servo motor held up an electromagnet using a PVC arm. The electromagnet was powered from the L298N motor driver chip, and the servo motor can operate with 5 volts from the Arduino. This arm was used to take the washers in and out of the camera’s view. The arm was constructed using a 4-inch section of ½ inch PVC pipe. By including this arm, the system has nothing blocking the view of the camera when it initializes and when it is running. By having proper gains, the servo motor and DC motor combined to make a smooth motion for the puzzle solver. Both motors also helped lower the solve time, with the longest trial taking less than three minutes. Additionally, the only noise outputted from the system occurred when the motors turned on, and these sounds were not loud enough to need hearing protection. The other electronics, such as the Arduino and L298N chip, were screwed to the base of the board, such that they do not move around and potentially short circuit. The L298N chip was powered from an external power supply, and this supply was a 9V wall adapter. Bare metal from wires was covered with electrical tape to prevent shock hazards. The area where the electronics are located was covered with cardboard. This can be replicated with any type of structure so that none of it is exposed, and by blocking it, the system is safer and overall more aesthetically pleasing. The only exposed electronics are the two LED’s that indicate whether or not the system is functioning or if the task is complete. Additionally, there is a lamp glued to the base of the system, which makes the background subtraction more consistent. The lightbulb has a cloth shade on it to prevent users from potentially burning themselves from the hot surface.



**Figure 1b: Final Design**

**Software Design:**

The graphical user interface, created through MATLAB App Designer, is shown in **Figure 2a**. This interface allows the user to adjust the locations of each washer as desired. As shown, this interface explains all features of the design and how to use it, and by doing so, the user interface is user-friendly. Additionally, the interface connects each section of the design. By selecting the initialization button, the USB 2.0 camera will take an initial game board image. This image will be used for background subtraction when the game starts. The initialization button also starts the Simulink model. This model is discussed further below and is used for closed-loop motor control. Other functionality includes the clear button, which will stop the Simulink model and reset both the user input and background subtraction elements stored in the table. The table on the GUI allows the user to select the desired location for each washer in column three, and it also shows the initial configuration of the game board in column two.



**Figure 2a: Graphical User Interface**

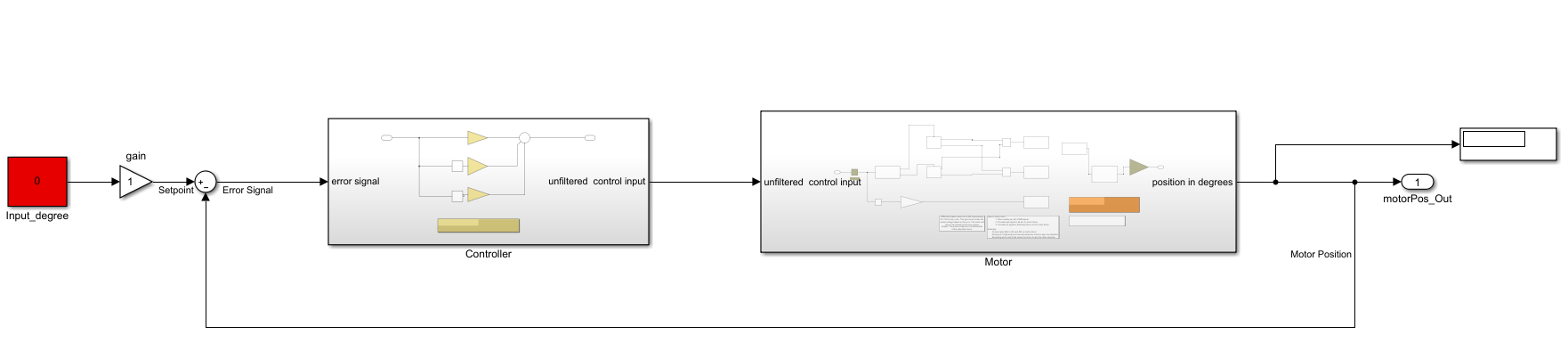
The code, developed in MATLAB 2020a for the final design, is shown below in **Figure 2b**. The code is separated into two steps: Step A and Step B. Step A is the background subtraction stage where the color and position is determined for each washer on the gameboard. Step B is the organization stage where the motor and electromagnet reposition the markers to match the user input. Both of these steps run after the start button is pressed on the GUI, which makes the system autonomous.

To develop the autonomous system, the code first takes an image of the gameboard. This image will contain the locations of all markers on the gameboard and will be used for background subtraction. This image is then subtracted from the blank game board image. The result from this subtraction is then converted to a binary image and is eroded. The regionprops function is called on the eroded image, and these values are stored in a structure called gameState. The centroid values are parsed out of the properties and stored in the same structure.

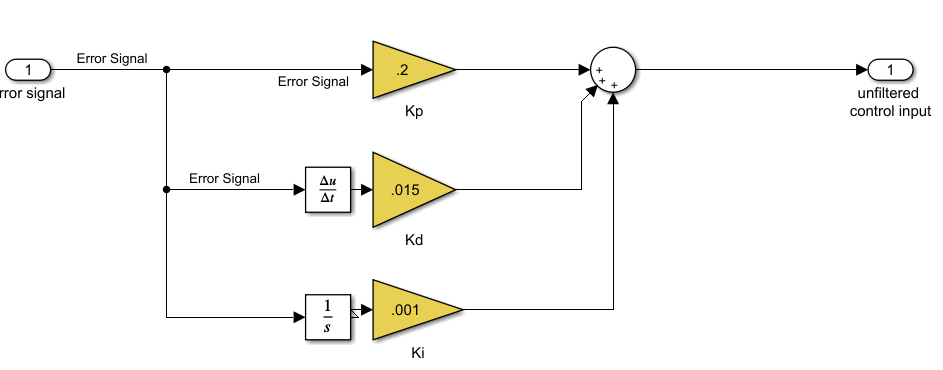
After finding the centroids of each marker, the system then loops through each centroid and sums the RGB values in the bounding box around each marker. If the RGB value is greater than a threshold, then the system determines the color as white and ignores these values. The average RGB values for each bounding box are looped through logic to determine which color is associated with each marker. The color is yellow if the red and green values are greater than 150. The color is red if the red value is greater than 100 and the green value is less than 150. The color is green if the red, green, and yellow values are less than 120. After finding the colors, the positions of each washer is determined by subtracting their current location from the predetermined center of the gameboard. By doing so, the locations and colors of each washer are determined. The color and location found for each washer are stored in the structure and displayed on the GUI as the initial configuration. By displaying the initial configuration, the user can easily determine the functionality of the design.

****

**Figure 2b: Software Flowchart for the Final Design**

Once the initial configuration is determined and displayed, the system starts reconfiguring the washers using closed-loop motor control, as shown in Step B from **Figure 2b**. Both **Figures 2c** and **2d** show the closed-loop motor control model used to control the angular position of the game board. By using this model, the system has at least one degree-of-freedom. The Simulink model is interfaced through the Arduino such that the hardware components can run off the software model. By adjusting the gains, we can increase the speed that the motor turns, which can improve the speed that the system reorganizes the washers. Additionally, adjusting the gains to an optimal setting makes the game board move in a smooth motion.

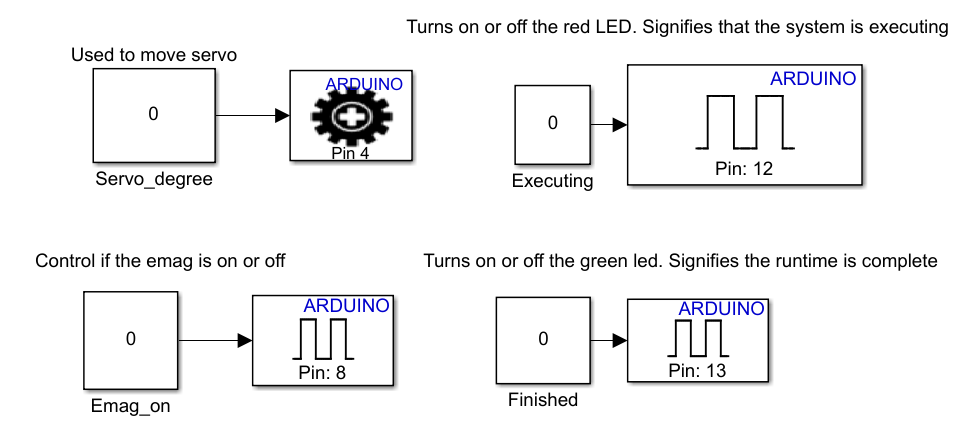
**Figure 2c: Simulink Model for Closed-loop Motor Control**



**Figure 2d: PID Control Gains for Closed-loop Motor Control**

To determine what angular position values to input, the code will loop through each well location and compare the initial configuration to the desired configuration. If the well correctly matches, then the code will skip and go to the next loop. If the well does not match, then the code will check three different logic lines. The first is if the well is not empty, and the desired value for the well does not wish to be empty. If this is true, the system will first move to the current well and pick up the undesired washer. This washer is then moved to the first empty well. Then, the board moves to and picks up the first free washer that matches the desired color. A free washer is a washer that isn’t in the correct position. This washer is then dropped into the initial well. If the statement is false, then it will look to see if the well is empty. If the well is empty, then the board will move to and pick up the first free washer that matches the desired color. This washer is then moved to the initial well. Lastly, if both the previous logic lines are false, then the system looks to see if the well wants to be empty and if so, the system moves to the current well and moves the washer to the first empty well. Only one washer will be moved at any given time.

Each time a washer is moved, there is a pause such that the motor has ample time to turn to the correct position and the system has time to turn on the electromagnet. By pausing, the motion of the board appears smooth. As mentioned, the servo motor pivots to pick up each washer, and by doing so, the accuracy of the system is improved. Additionally, each time a washer is moved, the vector containing the positions of each washer is updated to match the current configuration. Once the code loops through all eight wells, the motor is set to the predefined initial position, and an LED indicator, signifying the system is complete, is turned on. After turning on the LED, the solve time is calculated from the total runtime of the test, and this value is displayed on the GUI in seconds. The LED indicators and total solve time make the system user friendly and easy to use. The model for the servo motor, electromagnet, and both LEDs is shown in **Figure 2e** below.

****

**Figure 2e: Additional Simulink blocks for the Final Design**

**Analysis of Final Prototype Performance:**

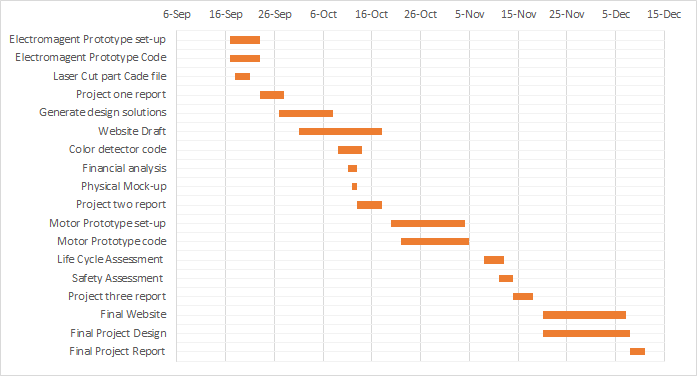
**Table 1: Evaluation of Final Prototype**

|  |  |
| --- | --- |
| **Customer Requirements** | **Performance of System** |
| Plays the game according to the rules | The system follows all rules |
| Uses the provided game board, Arduino, Camera, magnet, and motor | All provided material were used |
| Cost less than $300 | Final cost was 115, which is less than 300 |
| Uses an additional external power supply | Wall adapter used to run the motor and electromagnet |
| Reliable | System consistently solves each puzzle, DC motor can be inconsistent |
| Durable | All components are fastened to a sturdy base such that no components will move or fall apart |
| Safe | Noise output is low. System uses PVC structure to guard users from the board |
| Fast solving times | System consistently solves puzzles in under 3.5 minutes |
| Easy to use/ user friendly | Graphical user interface sufficiently explains the rules and is easy to use. |
| Generic components used | Components are found online and are cheap |
| Runs autonomously | System runs autonomously after receiving user inputs |
| Has one degree-of-freedom that has closed-loop motor control ran from the Arduino for the DC motor | System uses simulink to implement PID control for the DC motor |

The design developed functions as desired by the customer, as it correctly places each washer in their desired locations. Some of the implementations slow down the design significantly. Due to the speed of the closed-loop motor control or the small size of the electromagnet, the washers may be misplaced on the ring of the well. To combat this problem, we implemented a function to wiggle the board back and forth, such that the washer falls into the well after dropping it. Since this function occurs after each washer is dropped, the system takes longer to solve. We also implemented a two-motor system where one motor rotates the board and the other pivots to pick up the washers. This system adds additional delay to the solve time due to the time it takes to move the arm. Another issue that arose was that there were some instances that the Simulink model would not function at all after a test had been completed. Some of these instances were caused by improper communication between the Arduino and the Simulink software. Additionally, during some tests, the Simulink model would remain constant even after an input had been changed. This is the largest issue we faced as it would keep the board stagnant when it should adjust to a new location. When these issues are resolved, the system functions as desired and meets the customer requirements.

**Project Schedule/Gantt Chart:**

Throughout the project, we documented the tasks we needed to accomplish and the time frames in which we needed to accomplish them. **Figure 3** shown below shows the total scheduling of events in the form of a Gantt chart.



**Figure 3: Gantt Chart For Project Schedule**

**ECE 4950 Project 4 – Customer Requirements and Final Design Parameters**

Use the guidelines below to complete your report and add at the end of your report.

Group Member Last Names: Aho, Anderson, Cuttino, Liggett, and Moran

|  |  |  |  |
| --- | --- | --- | --- |
| Score | Pts |  | Performance Indicators |
|  | 5 | **General Format - Professional Looking Document/Preparation (whole document)** a) Fonts, margins (11pt, times new roman, single spaced. 1" margins on all sides). b) Spelling and grammar are correct  c) Layout of pictures – all figures need numbers and captions and must be referenced in the text  d) Follows the page limitations below.  e) References. Use IEEE reference format.  f) This grading sheet is included as the final page. | g.1 |
|  | 0 | **Page 1: Title, Group Name, Group Members, and Date**  **Executive Summary** (1 concise, well-written paragraph)  Provide an overview of this project. Briefly describe what you did and what you learned. | g.1 |
|  | 5 | **Page 2: Engineering Requirements** (<1 page)  Bulleted list of Final Design Engineering Requirements | e.1 |
|  | 10 | **Pages: 3-7: Design Details (<5 pages)**  Describe a system that can be built including System Architecture and System Integration based on the Engineering Requirements. Do not include data sheets or software code. | e.2 |
|  | 10 | **Page 8: Analysis of Final Prototype Performance** (<1 page)  Did it succeed or fail to meet customer requirements? What went wrong and what happened in the design process to allow this problem? Make a table of the customer requirements and address how well your design met these expectations. | e.3  e.4  i.1 |
|  | 5 | **Page 9: Project Schedule/Gantt Chart** (<1 page)  Create a schedule (Gantt chart) that shows the tasks and schedule for your project. Start from the very beginning of your project and extend to the end (completing final report and presentation). | k.2 |
|  |  | **Page 10** This grading sheet is included as the final page. |  |
|  | 50 | Laboratory demonstration of your prototype (evaluated by instructor and TAs). Evaluator will manipulate the interface and evaluate how well the system provides the timing and display functions (i.e. how well does the closed loop control work). Is it well built? Neat wiring? (.6 \* the prototype evaluation score) | g.2 |
|  | 15 | Rating by reviewers during competition | g.2 |