Holcombe Department of Electrical and Computer Engineering
Clemson University

Team 16 Project IV:

Customer Requirements and Final Design Parameters

ECE 4950: Senior Design I
Fall 2025

Submitted by: John Walker Bolding, David Bootle, Camren Khoury, William McGlone, Blake Morgan

GTA: Ms. Precious Eyabi
Instructor: Dr. Kapadia

Executive Summary

In this project, we were tasked with designing a system capable of shooting billiard balls into a designated pocket selected by the user. Given a list of customer requirements and constraints, we used an Arduino microcontroller connected to a Simulink model to control motors, a solenoid, and a camera to meet all criteria. This process required integration of hardware and software components, operating under design limitations, and developing teamwork and documentation skills.

Throughout the semester, subsystems were developed incrementally. Project 1 introduced MATLAB/Simulink and microcontroller control. Project 2 focused on preliminary computer vision for ball and game-state detection. Project 3 prepared us for DC motor design. The final project combined all components into a systems-integration effort. A MATLAB GUI was developed for user interaction and coordination of scripts and hardware drivers. Additional hardware components included a servo motor and solenoid for cue control.

Our hardware platform used an Arduino MEGA 2560 microcontroller and a custom aluminum gantry system with a belt-driven rack-and-pinion mechanism. All subsystems worked together to implement an automated billiard system called *The Pool Shark*.

1. Engineering Requirements

Engineering Requirements	**Customer Requirements**
Use motors for positioning solenoid to shoot pool	Shoots billiard balls into called pocket
Prototype defined by board dimensions	Uses provided game stage
Use 3D printer for customized parts	Cost less than $300
Use motors in kit	Uses provided equipment
Circuit compatible with benchtop equipment	Runs off power supply
Use PID control for desired output	Reliable
Build aluminum gantry	Durable
Run at low motor speeds	Safe
Use bearings for low-noise motion	Low noise
Place barrier around moving parts	Protective guarding
Run on user PC, concise scripts	Fast solving times
Build intuitive GUI	User-friendly
Use off-the-shelf components	Accessible components
Detect changes to billiards from user	Autonomous operation
Use servo, DC, and solenoid motors	One degree of freedom closed-loop

2. Design Details

This project encompassed multiple branches of engineering design, including linear control systems, computer vision, user interface development, and operational design. These are summarized in the sections below.

2.1. Linear Control Systems

Linear control systems are a key involved in the safe and accurate operation of the motor movements in this project. Linear control systems govern how motors and actuators respond to commands by maintaining a predictable and stable manner. Control systems typically use first or second-order differential equations to regulate changes in system states. PID control is one of the most common methods for achieving accurate tracking without extreme overshoot or rebounding. In this project, a closed PID control loop was used to rotate the DC motor to a desired angle. This angle was passed through the system as a distance in mm, but converted to the 2797 possible positions of the motor using a gain block. Hall sensors provided feedback on current position, which enabled the controller to continuously correct errors between the desired angle and the current angle. Careful PID and gain values ensured smooth motion while maintaining a fast settling time. The DC motor model was developed in simulink and can be seen below.

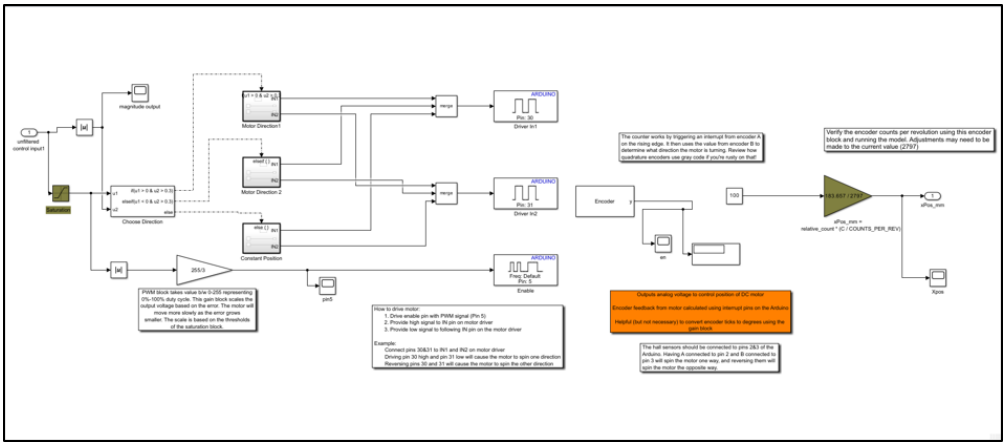

Figure 1: DC Motor Control System (placeholder)

3. Computer Vision

3.1. Homography

Homography is a mathematical operation that maps points on one plane to another, allowing a perspective distorted photo to be reshaped to one that has equal proportions. It is a widely used method in computer vision systems, robotics, mapping, and augmented reality. In this project, the camera is mounted on one end of the pool table, and performing homography allowed for the image to be from a birdseye view. Any two views of the same surface can be related via a 3x3 transformation matrix. It requires measurements of angles and distances of the subject and frame. Since we know the dimensions of the pool table, we know that the parts of the frame that appear larger are equal to the parts that appear smaller. In the Matlab code, we use

the “getpts” function to capture the coordinates of the four corners of the pool table. These coordinates are used in the “fitgeotrans” function to solve for the 3x3 matrix. Matlab matches the coordinate pairs, uses the direct linear transformation algorithms, and normalizes the matrix. After obtaining this matrix, Matlab then uses inverse mapping to create the output image via the “imwarp” function. For each output pixel, it finds the input pixel from the inverse matrix. It samples the input image and builds the corrected output image. Correct height-width measurements are crucial because a small inaccuracy of the point selections or pool table measurements can cause distortion. This was a necessary operation because of our camera placement. If we had placed our camera in a birds eye view directly above the pool table, homography would not be necessary. The transformation has little effect on angle calculation errors for shooting a chosen ball into a pocket. In most cases, the inaccuracies are from 2-10

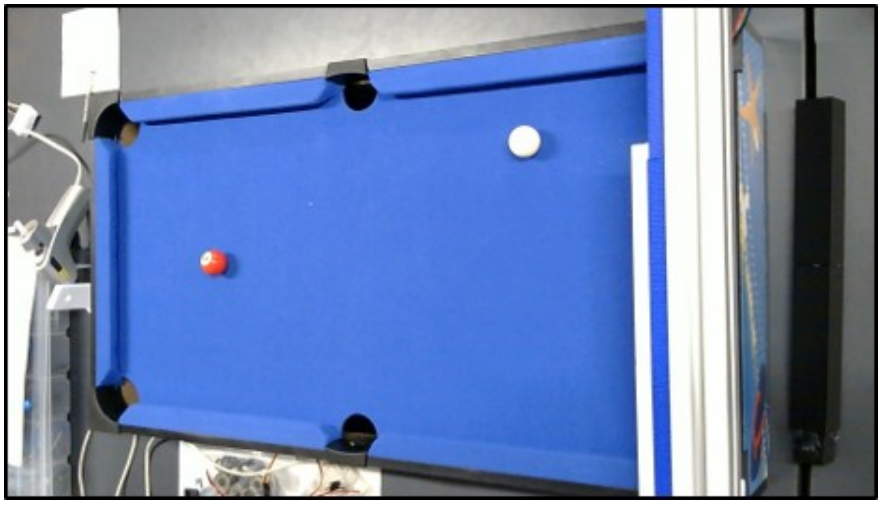

Figure 2: Raw Webcam Frame

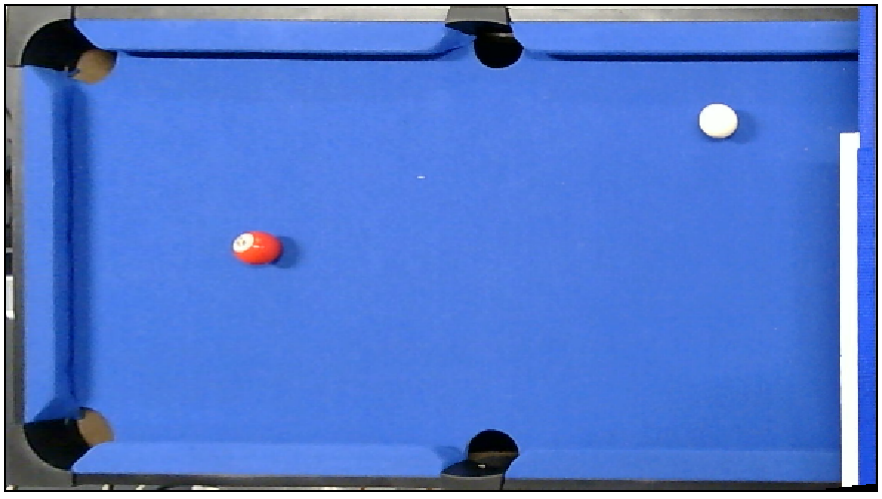

Figure 3: Post-Homography Transformation

3.2. Image and Color Detection

Image and color detection was a huge element of the vision system designed and was responsible for identifying billiard balls and their locations on the pool table. After homography correctly occurred, the image and color detection used an HSV detection code to develop raw masks of the photo for each color ball. These images were then passed through a morphology pipeline where the raw mask had to pass a circularity test and an isolation test to eventually get passed through as a valid mask. The valid mask would then be modified to refill the ball to a uniform circular shape and size. This rebuilding of the pool balls was necessary due to factors like glare, reflections, detection inaccuracies, edge detections, and the numbers on the balls causing a gap in color detection. An example of this morphology operation pipeline can be seen below.

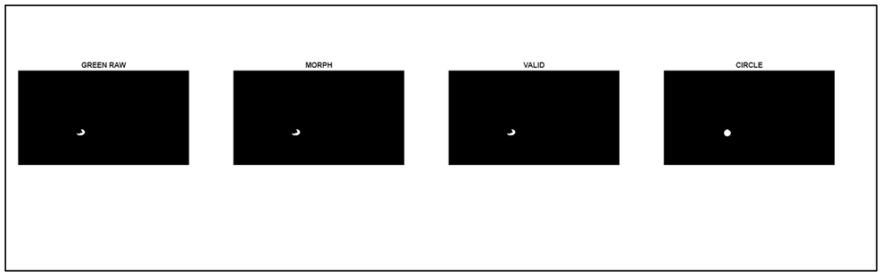

Figure 4: Morphological processing example for green ball detection.

The image detection code then took the centroid of these balls after the morphology was formed and passed those coordinates onto the computeShotGeometry.m function which was responsible for finding the angle and distance the motors needed to move to make the requested shot.

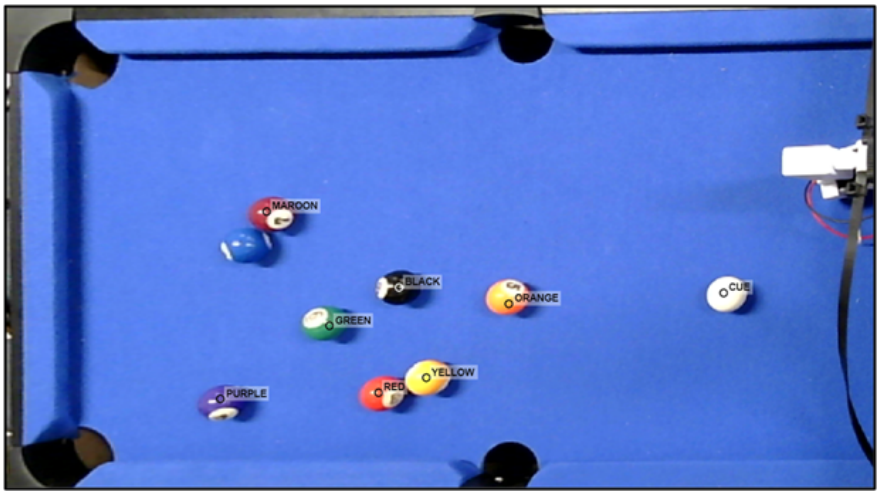

Figure 5: Final centroid placement and color-based ball classification.

4. Shot Geometry Calculations

Computing the shot geometry presents a significant challenge because, while the process is intuitive from a human perspective, translating it into a structured software protocol requires a series of explicit geometric

steps. The shot geometry calculation was based on coordinates extracted from the image: the cue ball, the desired object ball, and the target pocket. A preliminary sketch is shown below.

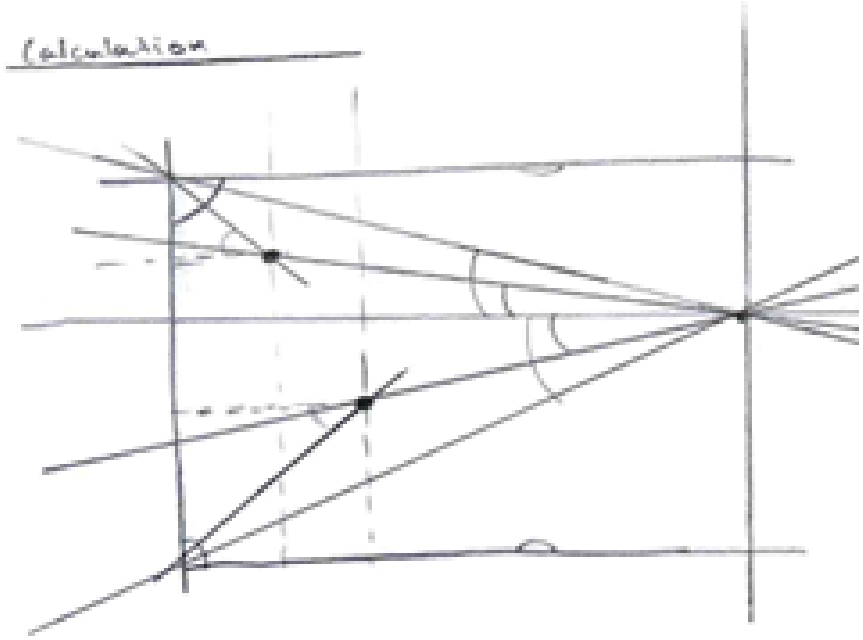

Figure 6: Shot Geometry Concept Sketch

The shot geometry not only involves the angles from ball to ball or ball to pocket, but also the necessary angle to strike the object ball at the correct point on its circumference. This required a combination of polar and Cartesian coordinate calculations. We used the `calculate_cue_angle` function to compute the initial orientation for the cue ball to face the desired object ball. Additional processing translated this angle into a usable servo rotation command.

Our method followed the ghost-ball aiming system. First, the algorithm computes the target vector between the object ball and the center of the target pocket. This vector is inverted by 180 degrees to obtain the collision normal—the direction in which the cue ball must apply force to send the object ball into the pocket. The system then projects a point outward from the object ball's center along this normal by a distance equal to the cue ball radius. This point, known as the ghost-ball center, marks where the cue ball's center must be at impact. Finally, the launch angle is computed by determining the vector between the cue ball's current position and this ghost-ball center. This ensures the shot geometry accounts for the physical size of the balls and yields an accurate aiming angle.

5. User Interface

The user interface was developed to appear visually pleasing but still easy to use for the customer. The interface uses several self-made graphics from the team, as well as consistent coloring and text formatting styles. The entrance page can be seen below.

Figure 7: App Entrance Page

After the entrance page is opened for visual appeal, the user can press the play game button to initiate the control page. The initial phase of the control page includes a calibration step for the homography to gather the coordinates of the pool table borders before the operation, to accommodate for minimal user interruption. The control page contained all of the code and design necessary for the user to input their desired ball and pocket based on the current image given. This control page was responsible for calling all of the other subfunctions in the MATLAB files. The control page can be seen below.

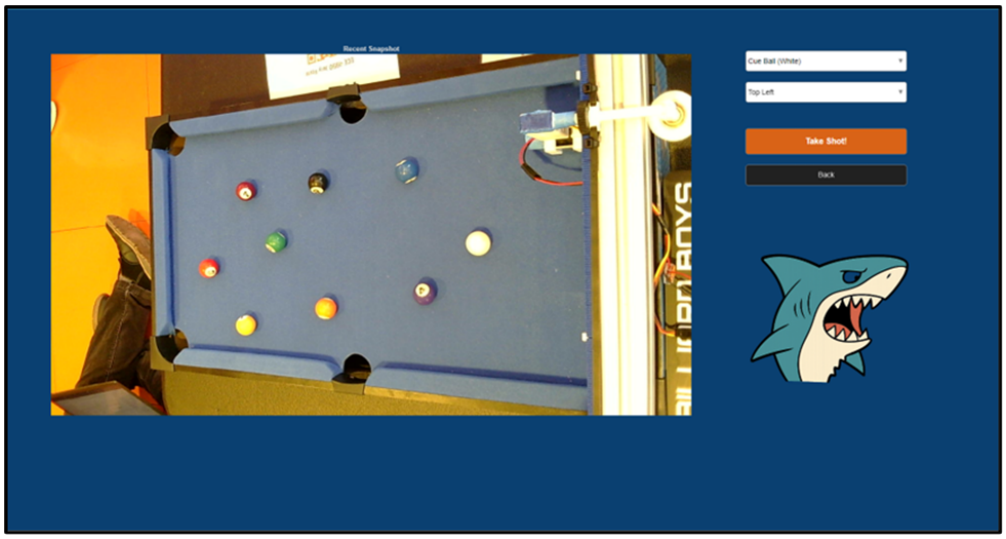

Figure 8: Pool Shark Control Page

6. Operational Design

6.1. Circuit Design

The circuit design for the project included seven main components. These components included: the DC motor, servo motor, solenoid, motor driver(s), Arduino Mega 2650, MOSFET, and Power supply. We initially started working with the servo motor for testing to be configured so that the servo would point to the location of a selected billiard ball. This device used three inputs: power (5V), ground, and a signal input. The signal was connected to the microcontroller and relied on pulse signals to orient itself correctly. In a MATLAB script, we wrote a test function that allowed the user to enter a degree and it would respond to that variable. The other two parts included more complicated configurations with motor drivers. For the DC motor, the driver was given to us and after analysis of the component we were able to wire it to the DC motor's six inputs as a medium between the microcontroller and motor. This allowed for sufficient power, as we applied 12 volts and 5 volts across the actual motor. A and B controlled the hall sensors, which were important for the direction of motion. Pin 5 connecting to the enable pin of the motor driver controlled the speed which was important to control for the design's safety. After this was completed, the belt system could be tested. The last major component of this circuit was the solenoid and its respective driver. It was necessary to build with common parts as mentioned in the design requirements. This would require a MOSFET, commonly used in power applications as an electronic switch. We grounded the source, and the gate was connected to ground and received a pulse signal from the microcontroller. The voltage across the diode attached to the drain would be the voltage applied to the solenoid. To keep the circuit from malfunctioning, resistors were used to prevent overcurrent.

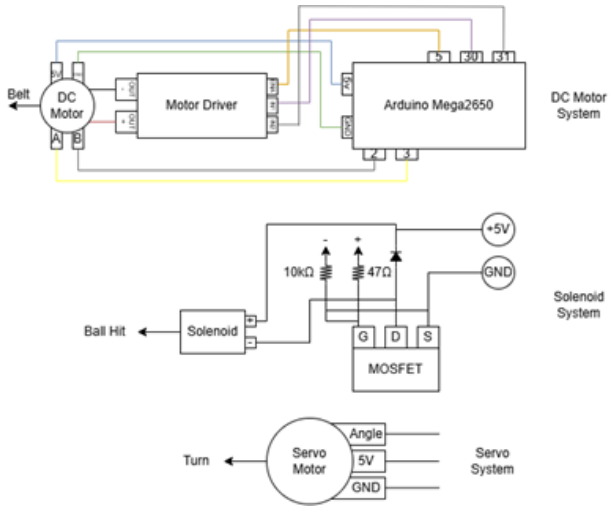

Figure 9: Circuit diagram showing connections for the DC motor driver, solenoid driver, servo motor, and Arduino MEGA 2560.

When testing, we connected the benchtop wave generator to create a 1 Hz signal with a high of 5V and low of 0V and sent it through the 47 ohm resistor. When we controlled it using the microcontroller, we could control the duty cycle and frequency at which the solenoid fired at. Lastly, we considered adding a variable

drive to control the amount of power the solenoid came in contact with the billiard, however the power we tested was sufficient and decided it was not viable, by adding cost and unnecessary complexity.

6.2. Mechanical Design

The mechanical design was implemented using an aluminum extrusion chassis that supported the computer vision system, gantry structure, and firing assembly.

Our team chose to custom-design all fittings, plates, and mounts for the system using the cloud-based CAD platform Onshape. The shared modeling workspace allowed every team member to access, edit, and iterate on designs in real time. Each part went through several refinement stages to optimize geometry, printability, and structural performance.

Unlike many other approaches, our computer vision system was not mounted above the center of the pool table. Instead, it was positioned at the end of the table, capturing the game from a bird's-eye perspective. This created both software challenges—such as the need for homography correction—and mechanical constraints related to camera positioning. To achieve the correct angle and field of view, we designed a vertical extension inspired by the adjustable mechanism of a selfie stick, providing both rigidity and variable height.

The aluminum chassis also supported the belt-driven gantry system. Due to the extruded frame geometry, designing the gantry required custom brackets and mounting plates to integrate the DC motor, timing belt, and pulleys. This provided smooth, controllable lateral motion needed to position the firing assembly.

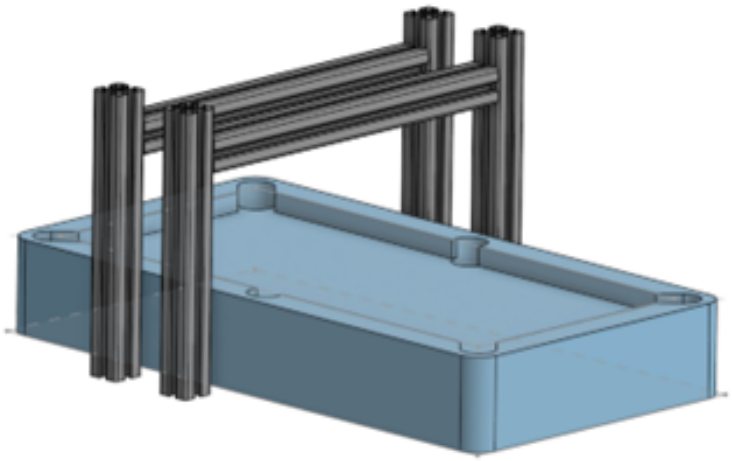

Figure 10: Gantry CAD design illustrating the belt-driven lateral motion system.

The firing assembly traveled laterally using a hybrid rack-and-pinion mechanism influenced by automotive steering assemblies. A gear attached to the motor acted as a linear actuator, driving the firing module along the width of the table with consistent force and direction control.

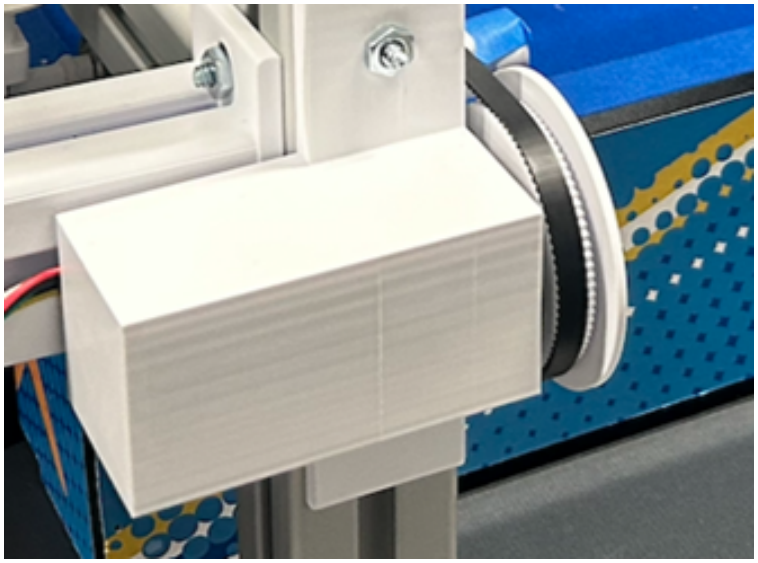

Figure 11: Motor encasing used to mount and protect the DC motor within the gantry.

The firing assembly integrated a servo motor and solenoid using a 3D-printed holder. The servo mount connected directly to the pinion axle for rotation control, while the solenoid was positioned beneath it to deliver the impact required to strike the cue ball. The assembly was compact and rigid, ensuring minimal mechanical delay and consistent firing behavior.

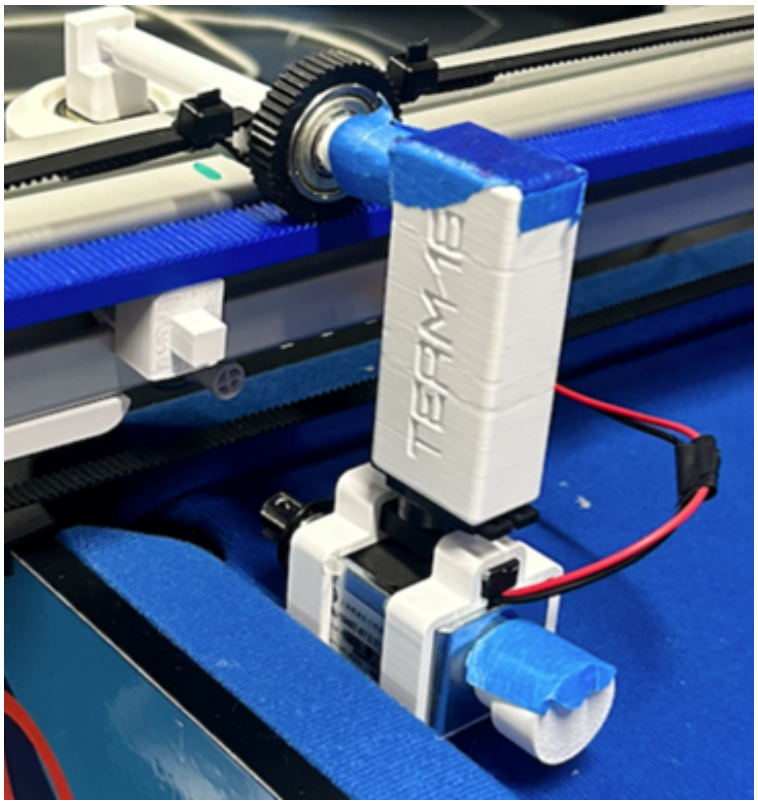

Figure 12: Servo and solenoid holder assembly used to aim and strike the cue ball.

7. Analysis of Final Prototype Performance

The final demonstration successfully met the required performance criteria. The system reliably detected all billiard ball colors, computed table geometry, and executed accurate motor movements. The GUI facilitated autonomous operation with minimal user input. The total system cost stayed under $300 while using only accessible off-the-shelf components.

A minor shortcoming was incomplete implementation of protective guarding. For a higher-power or production-level system, guarding around belts and moving parts would be essential. Overall, the final prototype demonstrated accuracy, safety, and successful integration of all subsystems.

8. Project Schedule / Gantt Chart

The Gantt chart is divided into four parts. Each unique project task is color coded (1-blue, 2-green, 3-orange, 4-purple). The chart describes the general timeline and the major topics addressed during each phase of development. The purple segments represent tasks originating in earlier projects that were later implemented in the final system. The final portion of the schedule highlights the testing phase, which included modifications to scripts, mechanical components, and circuitry.

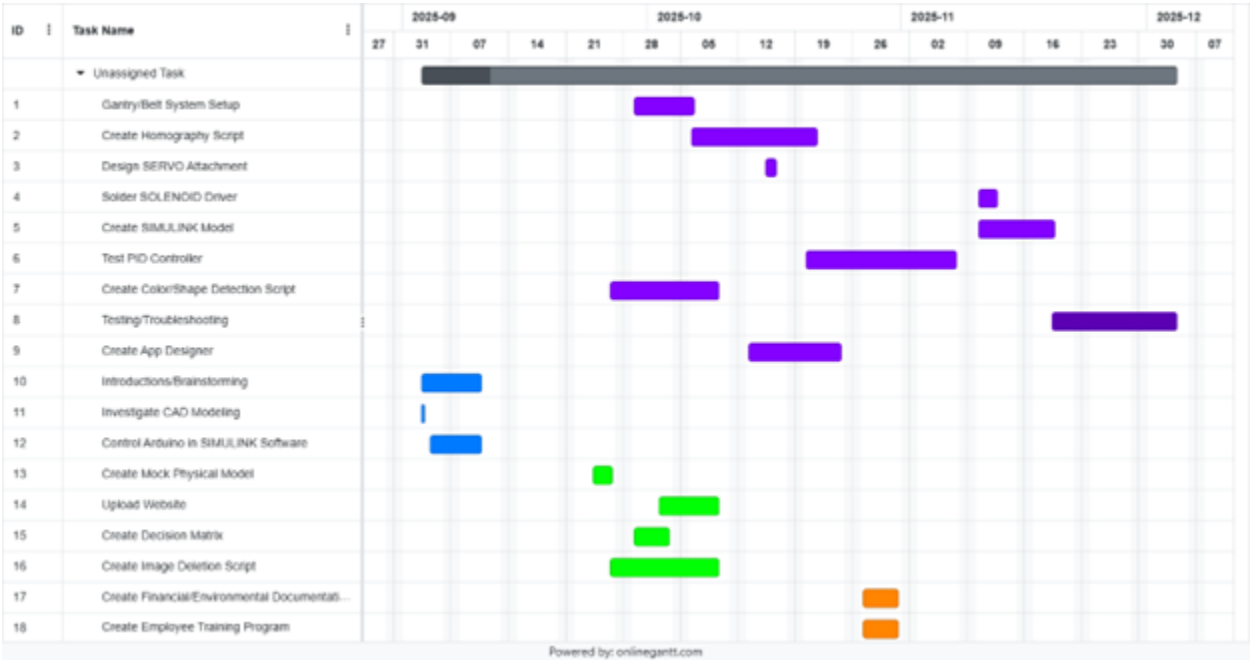

Figure 13: Project schedule and task progression throughout the semester.

References

References

[1] M. Kozovsky and P. Blaha, "Simulink generated control algorithm for nine-phase PMS motor," 2017 7th IEEE International Conference on Control System, Computing and Engineering (ICCSCE), Penang, Malaysia, 2017, pp. 59–64, doi: 10.1109/ICCSCE.2017.8284380.

[2] Munadi, M. A. Akbar, T. Naniwa and Y. Taniai, ”Model Reference Adaptive Control for DC motor based on Simulink,” 2016 6th International Annual Engineering Seminar (InAES), Yogyakarta, Indonesia, 2016, pp. 101–106, doi: 10.1109/INAES.2016.7821915.

[3] C. H. Houpis and S. N. Sheldon, *Linear Control System Analysis and Design with MATLAB*, 6th ed. Boca Raton, FL, USA: CRC Press, 2013.

[4] NCEES PE Control Systems Engineering Exam, National Council of Examiners for Engineering and Surveying, 2024. [Online]. Available: https://ncees.org/engineering/cse

[5] ”Hardware-in-the-loop simulation,” Wikipedia. [Online]. Available: https://en.wikipedia.org/wiki/Hardware-in-the-loop$_simulation$.[*Accessed* : 10/30/2025]

[6] National Institute of Standards and Technology (NIST), Building for Environmental and Economic Sustainability (BEES) Online Tool, U.S. Department of Commerce, 2025. [Online]. Available: https://www.nist.gov/services resources/software/bees

Rubric:

Score	Pts	
	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.
	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.
	5	**Page 2: Engineering Requirements** (<1 page) Bulleted list of Final Design Engineering Requirements
	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.
	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.
	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).
		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)
	15	Rating by reviewers after competition
	15	Slideshow for Demo containing the following information: • Executive Summary • Demonstration Video • Customer and Engineering Requirements • Brief Software Explanation • Performance Analysis

website link: https://sites.google.com/g.clemson.edu/ece4950group16f25/home