Mechanical Engineering MECHENG 4M06
Capstone Design Project
Faculty of Engineering – McMaster University – Undergraduate Studies
Winter 2025

Autonomous Debris Collection Robot for City Parks

Project RA01:
Devin Stukel
Valerie Stephens
Salih Al-Tak
Jerry Liu

Project Supervisor:
Dr. Ryan Ahmed

Executive Summary

This project aims to solve the need for collecting trash in public parks by developing an autonomous, garbage-collecting robot, which can locate, collect, and compact trash autonomously. To assess the validity of its development, the objectives for this robot were to move autonomously, operate efficiently, compact debris, operate safely, and locate debris. Multiple designs were proposed and assessed based on their rankings in safety, collection effectiveness, durability, manufacturability, human intervention, compaction ability, cost, and size, and the highest-ranking design was pursued. A final design, which includes a gripper mechanism for collection, a scissor lift for compaction, and an AI vision system for location, was created. The final design was then fabricated in its entirety using a variety of fabrication methods such as laser cutting, 3D printing, and machining metal. Extensive testing was conducted to assess the effectiveness of the design, including its ability to consistently locate, collect, and compact debris. The accuracy of the AI vision system in both its ability to correctly identify and locate debris was also tested rigorously. The conclusions drawn from this project were that the initial objectives were largely met, other than in its ability to operate efficiently. However, this was attributed to budget constraints, thus indicating that this project is a viable solution to replace traditional methods of collection trash manually.

Table of Contents

Project Background/Definition

Currently, trash cleanup in public parks is done manually, relying on individuals to walk around and pick up trash by hand. This is very inefficient, time consuming, and expensive due to the labour cost, and can result in trash being missed or left behind due to sanitary concerns. This project aims to solve this problem by developing an autonomous, mobile robot that can autonomously locate, collect, and compact trash. Currently, there are no solutions on the market that can do this, so the most similar products would be a Roomba or a street sweeper. The Roomba is very small, slow, unable to pick up large or heavy objects, and is limited to indoor use, while a street sweeper, although made for outdoor use, has to be manually operated and thus still requires labour.

To successfully complete this project, the solution should be able to detect, collect, and compact debris through the use of an AI system. This can be done by having a camera connected to a small computer that can detect trash, people, animals, and obstacles, then move towards the detected trash and avoid everything else. Then, the robot would collect the identified trash using a mechanical collection mechanism, deposit the collected trash into a hopper for storage, and finally compact the trash to allow for more trash to be collected.

The objectives that define this project are to move autonomously, operate efficiently, compact debris, operate safely, and locate debris. The robot needs to move autonomously when collecting and compacting to eliminate the need for human involvement, so it should have 0-man hours required for collection and compaction. The robot needs to be able to operate efficiently enough to replace the need for manual collection, so it should be able to collect 85% of the trash in a 100 m^2 area within 1 hour. The robot needs to be able to compact debris to at least 50% of the original volume so that more trash can be collected during a single use of the robot. The robot also needs to be able to operate safely, so the two most critical features of the robot, which are the collection and compaction mechanisms, must have factors of safety of at least 2 so that they do not break. It also needs to be able to detect people and animals so that it can avoid them to ensure it does no harm. Finally, the robot needs to locate an acceptable amount of debris to be considered successful over manual methods, quantified at 85% at a minimum.

Due to the novelty and innovation involved in this project, there are many uncertainties about its design and implementation. These include finding or creating a dataset that meets the needs of the AI system, developing and implementing a successful electrical system, balancing the torques for all the motors, determining the best detection method, finding suitable motor controller, and finding the most efficient compacting mechanism. Each of these uncertainties are crucial to solve if the final product is to be successful. More detail about these uncertainties and the research done to identify them is included in Appendix A.

Concept Generation and Selection

The concept selection for the Autonomous Debris Collection Robot was created from a set of well-defined criteria which included safety, collection effectiveness, durability, manufacturability, human intervention, compaction effectiveness, cost, and size. Each of these criteria were given a weighting out of 5, with 5 being the most important, to identify the priorities of the project. The designs were then rated on a scale of 1-5 for each criteria, then put into a weighted decision matrix to evaluate which designs scored the best given the combination of criteria importance and the design’s scores. The weighted decision matrix can be seen in Table 1 below.

When identifying the criteria for evaluating potential designs, safety was deemed the highest priority, given the robot’s operation in public parks. The proposed designs were assessed for exposure to hazards, with a point deduction for each exposed hazard such as pinch points. A minimum safety rating of 3 was required because more than 2 potential safety hazards was deemed unacceptable by the team. Another priority, collection effectiveness, was deemed essential as robot’s primary function is garbage collection. The proposed designs were rated on how effectively they could collect various types of trash, with a point deduction for each type of garbage they couldn't consistently pick up (eg large garbage). The Autonomous Debris Collection Robot is designed to be used long-term use outdoors, therefore designs lost points for components at high risk of damage from environmental exposure. Likewise, manufacturability is key for creating mass production and easy maintenance, so designs were evaluated based on the ease of assembly, ease of manufacturability, and availability of parts, with a point deduction for each major custom component.

Several other criteria that were considered non-essential but important to consider were also included in the decision matrix. Since one of the major objectives of this project was autonomy of the robot, the amount of human interaction required for each design was evaluated, with a point deduction for each feature requiring frequent maintenance, such as if the garbage would have to be swept out of the hopper or would contact moving parts. However, since the designs were overall made to be autonomous, this was not given a high ranking compared to some of the other criteria. Designs were also rated based on compaction effectiveness which influences the robot’s efficiency in collecting trash, allowing it to collect more per trip. Furthermore, cost was considered to ensure feasibility of mass production, therefore the cost was evaluated based on the number of motors, hoppers, and compaction plates in the design. Finally, it was determined smaller robots are more desirable for their visual impact and space requirements, which was assessed based on the robot’s volume and additional space needed beyond the hopper.

Table 1: Weighted Decision Matrix

Metrics	Weights	Design #1		Design #2		Design #3		Design #4	
Safety	**5**	4	**20**	4	**20**	3	**15**	3	**15**
Collection Effectiveness	**5**	2	**10**	4	**20**	5	**25**	4	**20**
Durability	**4**	3	**12**	3	**12**	3	**12**	5	**20**
Manufacturability	**4**	4	**16**	4	**16**	4	**16**	4	**16**
Compaction Effectiveness	**3**	5	**15**	4	**12**	3	**9**	4	**12**
Human Intervention	**3**	3	**9**	3	**9**	3	**9**	5	**15**
Cost	**2**	3	**6**	4	**8**	4	**8**	4	**8**
Size	**1**	4	**4**	4	**4**	4	**4**	5	**5**
Total			**92**		**101**		**98**		**111**

Concept 1: Rotary Broom with Ramp and Linear Compression Mechanism

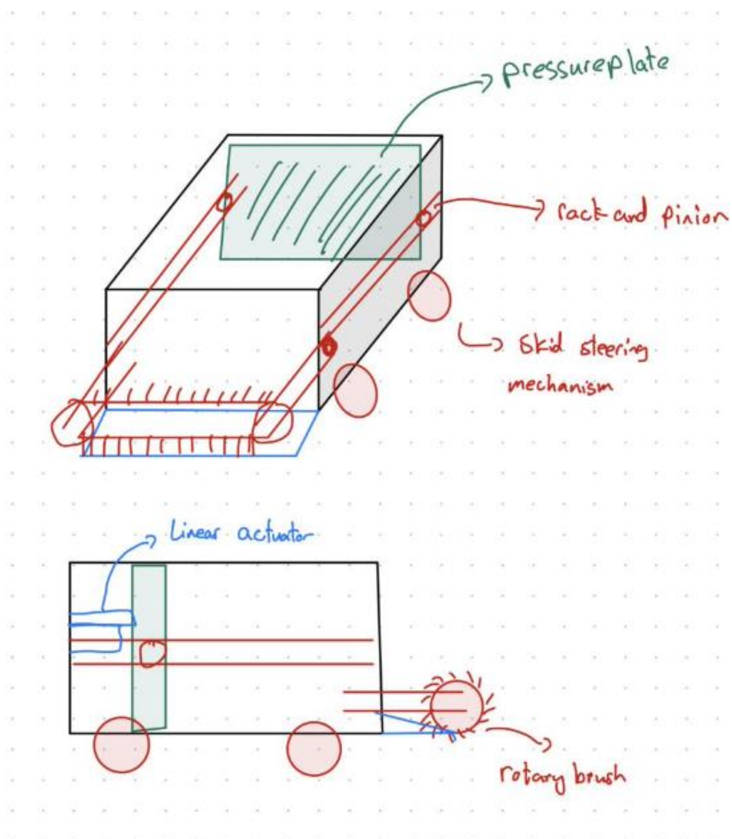


Figure 1: Design 1, Rotary Broom with Ramp and Linear Compression Mechanism

Design 1, shown in Figure 1, uses a front-mounted rotary broom to sweep trash onto a ramp, directing it into a compartment where a linear actuator compresses the waste from back to front. The brush has its own motor, and two additional motors provide skid steering for mobility.

Safety was rated at 4/5 due to the slight risk due to exposed rotating parts. Collection effectiveness was rated at 2/5 due to potential issues with tall, light, or embedded trash. Durability was rated at 3/5 because the brush and compactor may wear from exposure to debris. The manufacturability in this design was rated at 4/5 as it is simple, however the brush design adds complexity. Human intervention in this design was rated at 3/5 as manual emptying was required with and no integrated garbage bag. Compaction for this design was rated at 5/5 because of the effective use of linear actuator. More so, cost was rated at 3/5 because of the extra motor and actuator increase expense. Finally, the size of this robot was rated 4/5 due to its bulky design from its brush protrusion.

Concept 2: Rotary Broom with Dual Compression Plates

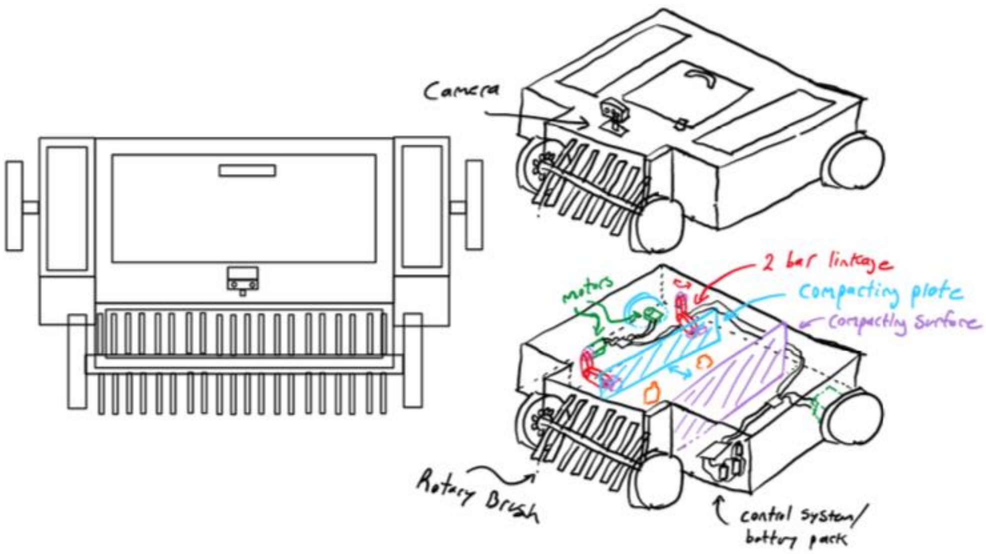


Figure 2: Design 2, Rotary Broom with Dual Compression Plates

Design 2, shown in Figure 2, uses a rotary broom for collection but compacts trash with two side-mounted plates moving inward simultaneously. This ensures balanced compression but increases cost due to the dual motor or actuator requirement.

Safety was rated at 4/5 because exposed rotating parts pose a pinch risk. Collection effectiveness was rated at 4/5 due to minor issues with tall trash. Durability was rated at 3/5 as brush wear and linkage fatigue are concerns. Manufacturability for this design was rated at 4/5 because the custom brush design adds complexity. Human intervention was rated at 3/5 due to manual emptying and absence of a garbage bag. Compaction for this design was rated at 4/5 as it lacks strong mechanical advantage. Cost was rated at 4/5 because the extra plate increases cost. Finally, the size of this robot was rated at 4/5 since the brush extends past the hopper.

Concept 3: Jaw-Based Compression and Disposal Mechanism

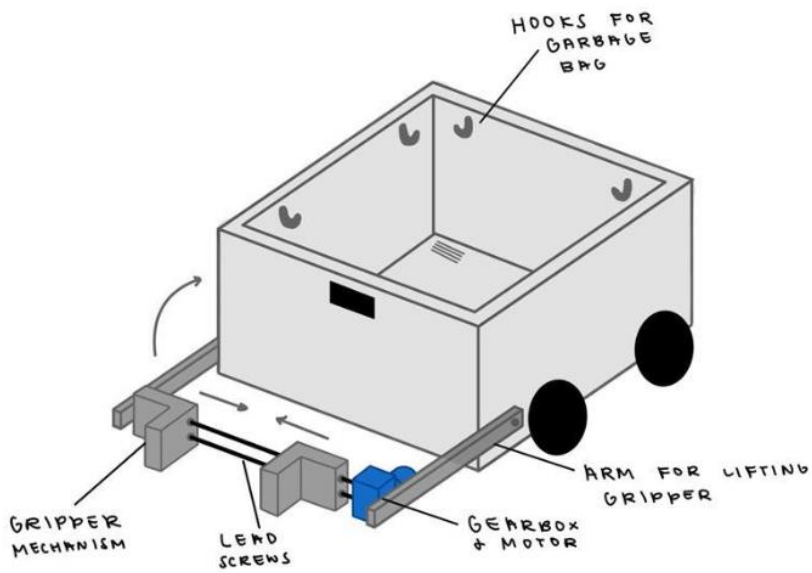


Figure 3: Design 3, Jaw-Based Compression and Disposal Mechanism

Design 3, as shown in Figure 3 uses a jaw mechanism to grab and lightly compress trash in one motion before depositing it into the container. It improves collection efficiency and allows for garbage bag integration, though compression is limited for larger or resilient objects.

Safety was rated at 3/5 due to multiple pinch points at the gripper and arm. Collection effectiveness was rated at 5/5 because it can handle varied trash sizes. Durability was rated at 3/5 due to the exposed lead screw and protruding arm. Manufacturability for this design was rated at 4/5 because the custom gripper adds complexity. Human intervention was rated at 3/5 due to manual emptying and no bag mechanism. Compaction for this design was rated at 3/5 as there is no plate and it is limited by the gripper size. Cost was rated at 4/5 because one extra motor is needed. Finally, the size of this robot was rated at 4/5 since the arm extends beyond the hopper.

Concept 4: Jaw Pickup with Internal Compression

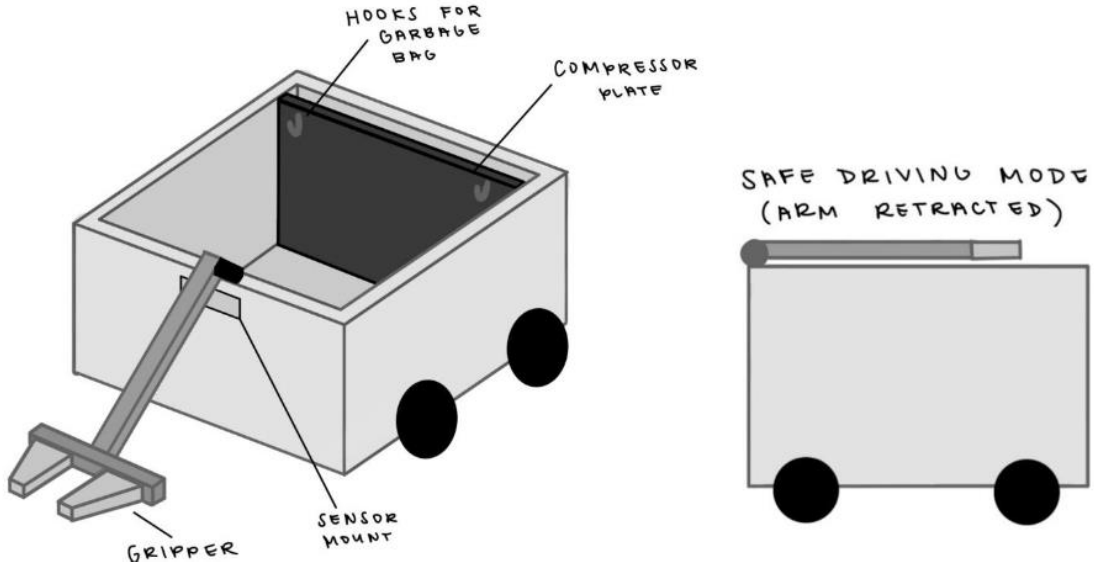


Figure 4: Design 4, Jaw Pickup with Internal Compression

Design 4, shown in Figure 4 uses a jaw to collect trash, which is transferred into a compartment where a plate compacts it. It separates collection and compression for better control and allows the use of a garbage bag that moves as trash accumulates.

Safety was rated at 3/5 due to pinch points from the gripper and compaction plate. Collection effectiveness was rated at 4/5 because it struggles with wide trash items. Durability was rated at 5/5 due to the robust design with a "safe driving mode." Manufacturability for this design was rated at 4/5 because the custom gripper increases complexity. Human intervention was rated at 4/5 as manual emptying was required. Compaction for this design was rated at 5/5 because of the effective scissor mechanism with lead screw. Cost was rated at 4/5 due to the extra motor for the gripper adding cost. Finally, the size of this robot was rated at 5/5 due to its compact design with the arm tucking above the hopper.

As seen in Table 1, Design 4, which features a gripper for collection and compaction plate, received the highest overall score. This was ultimately selected as the final design, as it offered the best overall balance of safety, durability, collection efficiency, compaction effectiveness, and size. The separation of collection, compression mechanisms and improved usability demonstrates strong durability in design and effective mechanical advantage through its scissor-based compression system, making it the most practical and sustainable solution moving forward.

Some of the additional factors that were considered in this process are the social, environmental, and financial implications of each design. To ensure that the robot would be accepted socially, safety was deemed a very high priority in the decision making process, as discussed above. Additionally, the final design includes a camera to detect and avoid people to ensure safe operation. Environmental factors were also discussed such as the sustainability of various materials considered. To mitigate environmental factors, it was decided that sustainable materials such as MDF would be used wherever possible in each design. Financial considerations and limitations were also included in the decision making process by having a criteria for cost, as discussed above.

Design Process

The design of the autonomous trash-collecting robot, was driven by a clear goal: to develop a system that could identify, collect, and compact garbage in parks with minimal human involvement. From the start of the project, it was clear that balancing ambitious functionality with real-world constraints like time, cost, and complexity would play a major role in shaping the final product.

The original project objectives outlined an ambitious set of targets. These included fully autonomous trash collection and compaction, the ability to operate for at least one hour on a single battery charge, and the capacity to detect and collect at least 85 percent of trash in a 100 square meter area. Additionally, the robot was expected to operate safely around people and animals and to compact collected trash to at least 50 percent of its original volume.

While these goals set a strong foundation, the team quickly had to make some decisions to keep the project feasible. For example, the expectation that the robot would detect and collect trash over a 100 m^2 area proved unrealistic given the complexity of path planning and navigation that would be required. Implementing a system capable of covering and searching that much ground autonomously would have required additional sensors, real-time mapping, and a significantly more advanced control system — all of which would have pushed the project beyond the time and budget constraints.

The goal of running for one hour on battery power also had to be eliminated. Although battery options were explored, constant removal and replacement for testing was time-consuming and risked damaging components. Instead, the team opted for a corded power setup during testing to allow uninterrupted development and debugging. Full battery integration remains a planned improvement for future iterations.

Despite the adjustments made to some goals, all other project objectives were successfully met. The robot is able to autonomously collect and compact trash without any human input during operation, fulfilling the vision of hands-free cleanup. Key mechanical components were thoroughly analyzed, with the gripper and compactor mechanisms exceeding safety targets achieving factors of safety of 4 and 15, respectively. On the software side, the AI detection system performed above expectations, reaching over 92% accuracy in identifying garbage. In terms of functionality, the compactor successfully reduced trash volume by at least 50% for lightweight materials, as intended. Additionally, the system was able to detect and avoid people and animals reliably, ensuring safe operation in public environments.

Constraints and Considerations

Throughout the development of GarBot, the team carefully considered a range of practical constraints that shaped the project's design decisions. Time was one of the most critical factors, which led to prioritizing core functionality and scaling back on more complex features such as advanced navigation and full battery integration. These were identified early as time-intensive tasks that could risk delaying the project, so the team focused instead on ensuring reliable object detection, collection, and compaction.

Budget was another key constraint, and the project successfully kept total costs under $500 CAD by using affordable materials like PLA and MDF, along with off-the-shelf components for motors, electronics, and structural hardware. This approach allowed the team to create a functional prototype without compromising on essential features.

Safety was a major consideration, particularly because the robot was intended to operate in public spaces. AI-based object detection was implemented to recognize and avoid people and animals, and ensured that all moving parts followed smooth, non-aggressive motion paths to reduce the risk of accidents.

From an environmental standpoint, sustainable materials were used wherever possible and focused on a secure, modular assembly to prevent parts from becoming dislodged or left behind during outdoor operation. Finally, the social aspect of the design was considered in how the robot interacts with its surroundings—its compact size, quiet operation, and non-disruptive movement patterns were all intended to make it suitable for use in public parks without drawing negative attention or causing inconvenience.

Reference Standards

While this project was not subject to formal regulatory compliance, several established engineering standards were used to guide the design, fabrication, and system integration processes. These references helped ensure that GarBot was developed with safety, reliability, and good engineering practices in mind.

One of the most influential standards referenced was ISO 12100, which outlines the general principles for safe mechanical design. This standard was introduced during machine shop training and was especially relevant during the fabrication of mechanical components. These safety principles were applied when using the drill press to machine holes in metal brackets, and when cutting the steel shaft for the lifting arm and trimming the lead screw to the required length. Following ISO 12100 helped to avoid common hazards such as pinch points and misalignment, and ensured that components were produced to a high safety standard.

For the electrical and software components, IEEE 12207 was considered, which outlines best practices for software lifecycle processes. This was particularly helpful when organizing and validating the workflow for training the YOLOv11 object detection model and preparing it for deployment on the Raspberry Pi. By loosely following this structure, a clear development pipeline was maintained and ensured version control and documentation were in place throughout model training and tuning.

In addition, CSA SPE-1000 was used as a reference for safe wiring and electrical assembly. Although not required to meet full CSA certification, this standard provided guidance on practices such as isolating power connections, protecting exposed terminals, and using appropriate gauge wires for motor and sensor systems. These considerations contributed to a safer and more reliable prototype.

Final Design

Mechanical Design

The final design for the Autonomous Garbage Collecting and Compacting Robot was created based on the preliminary design that scored the highest in the weighted decision matrix, which was Design 4. This design includes a robotic arm with a gripper end-effector to collect trash, a plate and scissor lift to compact trash, and a hopper to store the collected trash. A full drawing package of the final design along with all custom-designed parts is included in Appendix B. Figure 5 below shows a render of the final design.

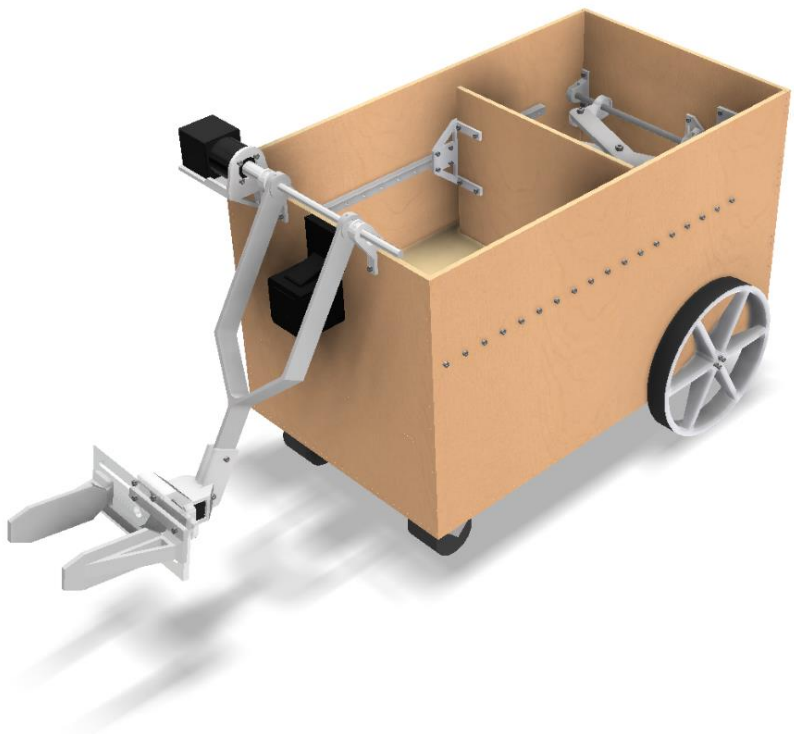


Figure 5: Final Design

Figure 6 below shows a render of the final design with one side removed, so that the false bottom and compaction system can be seen. The cavity under the false bottom houses the electrical systems so that they do not contact the collected trash or the environment around the robot. The design includes L brackets at each corner of the base which connect the hopper to the base. An additional view of the assembly of the hopper and the base is shown in Figure 33 in Appendix B.

Other design features of the base of the robot include large rear, driven wheels, with a rubber exterior to provide more traction for the robot, and non-driven castor wheels at the front of the base. This is to allow for a better ability to turn so that the robot can easily move around when collecting trash. The large wheels make the robot suitable for all terrain types, even icy or slippery conditions, when even manually collecting trash is not feasible. The robot design also contains a large hopper, where collected trash is stored. This allows the robot to collect a large amount of trash at once, which may otherwise be too large or heavy to be collected manually in one trip. The hopper design also allows for a garbage bag to be used where the trash is collected so that less maintenance is required and trash does not contact moving or exposed parts.

Figure 6: Final Design, Transparent Side Wall

To maximize the collection accuracy and versatility of the robot, the collection method included in the final design uses a gripper attached at the end of a long arm, as shown in Figure 7 below. The gripper is controlled by a rack and pinion, connected to a small servo motor. The design also includes a pressure sensor on the inside of the gripper to detect if enough force has been applied to the object being picked up. The lifting arm is connected to a shaft, coupled to a motor, so that the arm can lift up the collected trash and deposit it into the hopper. All components of the collection mechanism were designed by the team, including the motor mounts, bearing mounts, lifting arm, rack and pinion, gripper jaws, and hubs to connect the arm to the shaft. An exploded assembly and detailed views of the gripper and lifting arm assembly can be found in Figure 34 in Appendix B.

The gripper design is novel compared to other debris-collecting robots, such as a Roomba or street sweeper, as these designs rely on the use of vacuums and sweeping mechanisms to collect debris, which limits the size of debris to the size of the vacuum intake and can get clogged. The gripper collection mechanism is novel because it allows the robot to collect a larger range of trash both in size and shape.

Figure 7: Collection Method Design

Another novel feature of this design is the ability for the robot to compact the trash that is has collected. This maximizes the amount of trash that it can collect at a time, and reduces the frequency of required cleaning, thus minimizing the amount of human intervention needed for its operation. Compacting trash using a compactor as opposed to doing it manually is also more sanitary than someone collecting trash manually and using their foot to compact the trash. The compactor mechanism in this design, as shown in Figure 8 below, features a scissor lift mechanism, connected to a lead screw, so that an inexpensive DC motor can be used to control the linear motion of the compactor plate. The use of a scissor lift also provides mechanical advantage over connecting the compactor plate directly to the lead screw, thus increasing the compaction ability of the robot. The scissor lift was designed to be constructed from 1/8" aluminum flat bar, which is inexpensive and easy to machine, to reduce the number of custom components and manufacturing costs of the design. 3D printed brackets were designed to attach the scissor lift to the compactor plate and to the lead screw, as well as to attach the compaction plate to the linear rails along the sides of the hopper. These rails are essential for holding the plate straight when compaction is occurring, and the tall bracket design helps to prevent any torsion or bending from occurring on the compaction plate. Drawings of the compaction assembly can be found in Figure 35 in Appendix B.

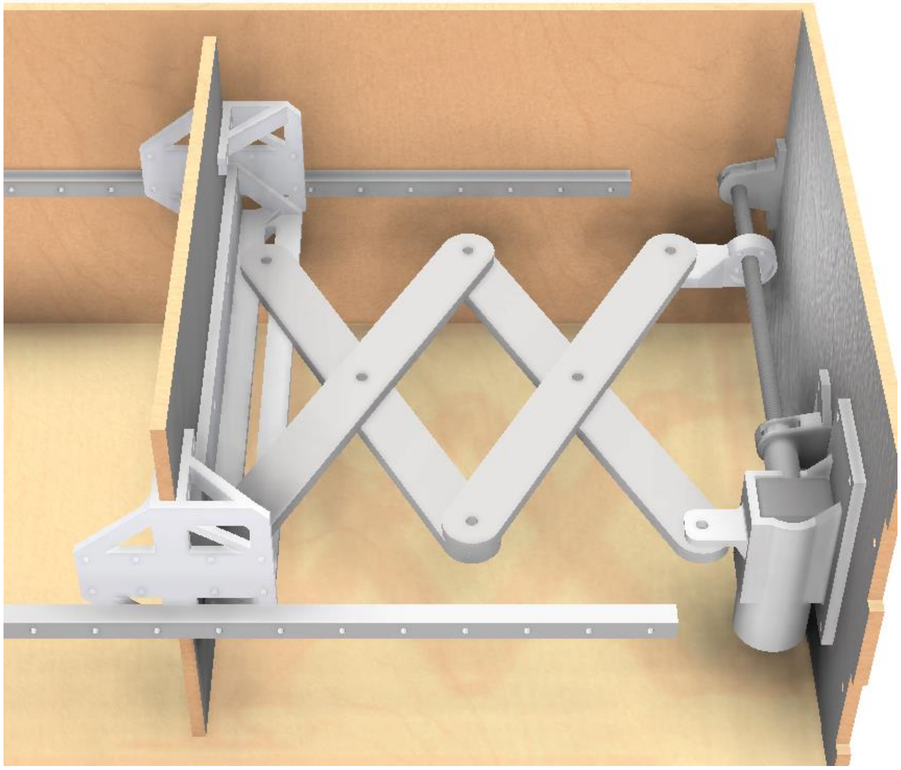


Figure 8: Compaction Method Design

Finally, the final design also included a friendly, brightly-colored and informative exterior, designed by the team, to help the robot to blend into its environment to increase the public's perception of it. This design is shown in Figure 9 below. The exterior design provides information on the project, including a QR code that links to a website explaining more about how the robot was developed and what it does. This will help the robot socially, as it may help park patrons understand not to disturb the robot's operation and can help them feel safer, knowing about how it works.

Figure 9: Exterior Image Design

A render of the final design with the exterior image is shown in Figure 10 below.

Figure 10: Final Design with Exterior Imagery

A full parts list, including fasteners, can be found in the table below. Assembly drawings with a labelled parts list can be found in Figure 31 and Figure 32 in Appendix B.

Table 2: Design Parts List

ITEM	QTY	PART NUMBER	PURPOSE
1	1	Hopper Back Piece	Hopper
2	1	Hopper Front Piece	Hopper
3	2	Hopper Side Piece	Hopper
4	1	Hopper False Bottom Piece	Hopper
5	1	Hopper Bottom Piece	Hopper
6	1	Compactor Plate	Compaction Mechanism
7	1	Gripper Motor Mount	Collection Mechanism
8	1	Lead Screw Motor Mount	Compaction Mechanism
9	2	Wheel Motor Mount	Base
10	1	Gripper Base	Collection Mechanism
11	1	Lead Screw Bearing Mount Small	Compaction Mechanism
12	1	Lead Screw Bearing Mount Large	Compaction Mechanism
13	2	Linear Ball Bearing	Compaction Mechanism
14	1	Lead Screw	Compaction Mechanism
15	1	Scissor Lift Lead Screw Connection	Compaction Mechanism

16	1	Lifting Arm Motor Mount	Collection Mechanism
17	2	Lifting Arm Bearing Mount	Collection Mechanism
18	1	Lifting Arm	Collection Mechanism
19	2	Motor Coupling	Miscellaneous
20	1	8 mm Shaft	Collection Mechanism
21	1	MG996R Servo Motor	Collection Mechanism
22	1	Gripper Gear	Collection Mechanism
23	2	Gripper Rack	Collection Mechanism
24	2	Gripper	Collection Mechanism
25	8	L Bracket	Base
26	2	Wheel	Base
27	1	Compactor Plate Rail Connection	Compaction Mechanism
28	1	Compactor Plate Rail Connection	Compaction Mechanism
29	1	Compactor Plate Scissor Lift Connection	Compaction Mechanism
30	4	Scissor Lift	Compaction Mechanism
31	1	Camera	AI Detection System
32	1	NEMA 17 Motor	Collection Mechanism
33	3	DC Motor	Compaction Mechanism
34	4	Bearing	Miscellaneous
35	124	AS 1420 - 1973 - M3 x 6	Miscellaneous
36	10	AS 1420 - 1973 - M5 x 10	Miscellaneous
37	2	Castor Wheel	Base

AI System Design

The AI system implemented in GarBot was developed using the Ultralytics YOLO object detection framework [1], with deployment specifically tailored for real-time edge computing on the Raspberry Pi. A Raspberry Pi 4 running the 64-bit Raspberry Pi OS (Bookworm) [2], [3] was selected as the hardware platform, and a dedicated Python virtual environment was configured to manage dependencies and avoid library conflicts. The YOLOv11n model, pretrained on the COCO dataset, was chosen for its lightweight architecture and strong balance between speed and detection accuracy, making it well-suited for embedded applications with limited compute power, as shown in Figure 12: Performance comparison of YOLOv11 model variants [4].

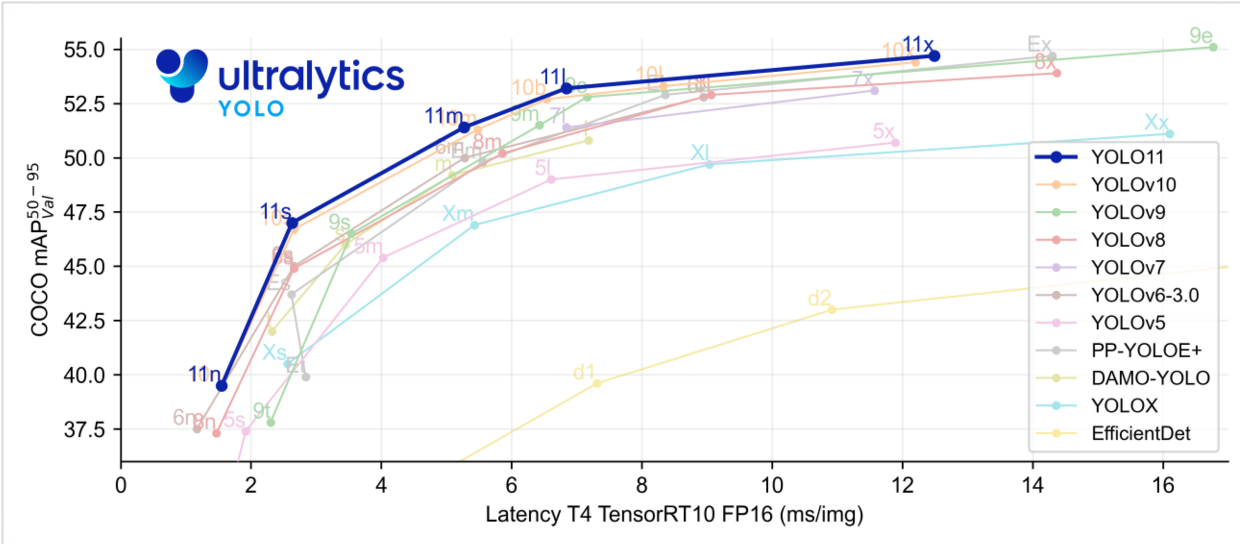


Figure 11: YOLO Model Family Benchmark – mAP vs. Latency (TensorRT10, FP16) [4]

Among the YOLO model variants, YOLOv11 has demonstrated leading performance in both speed and precision benchmarks. The nano version (YOLOv11n) was specifically selected for its ability to deliver real-time inference at high efficiency, as demonstrated in benchmarking results provided by Ultralytics [4], [5]. This model offered a clear trade-off: slightly lower accuracy compared to larger variants like YOLOv11s or YOLOv11x, but significantly faster inference speeds. As shown in Figure Y, YOLOv11n achieves the lowest latency while maintaining acceptable accuracy, making it ideal for use in GarBot, where low-latency performance was critical to enabling responsive and autonomous behavior on a resource-constrained Raspberry Pi.

Performance

Detection (COCO) | Segmentation (COCO) | Classification (ImageNet) | Pose (COCO) | OBB (DOTAv1)

See Detection Docs for usage examples with these models trained on COCO, which include 80 pre-trained classes.

Model	size (pixels)	mAP val 50-95	Speed CPU ONNX (ms)	Speed T4 TensorRT10 (ms)	params (M)	FLOPs (B)
YOLO11n	640	39.5	56.1 ± 0.8	1.5 ± 0.0	2.6	6.5
YOLO11s	640	47.0	90.0 ± 1.2	2.5 ± 0.0	9.4	21.5
YOLO11m	640	51.5	183.2 ± 2.0	4.7 ± 0.1	20.1	68.0
YOLO11l	640	53.4	238.6 ± 1.4	6.2 ± 0.1	25.3	86.9
YOLO11x	640	54.7	462.8 ± 6.7	11.3 ± 0.2	56.9	194.9

Figure 12: Performance comparison of YOLOv11 model variants [4].

To further enhance runtime performance, the model was exported to NCNN format, a lightweight, high-performance inference framework optimized for ARM-based systems [6]. NCNN allowed the model to run directly on the Raspberry Pi's CPU without the need for external accelerators, significantly improving inference speed by reducing computational overhead. As illustrated in Figure z, YOLOv11n in NCNN format achieved the lowest inference time among the tested formats, outperforming larger models such as YOLOv11s in every format category [5]. This performance advantage was critical for maintaining real-time responsiveness during robot operation.

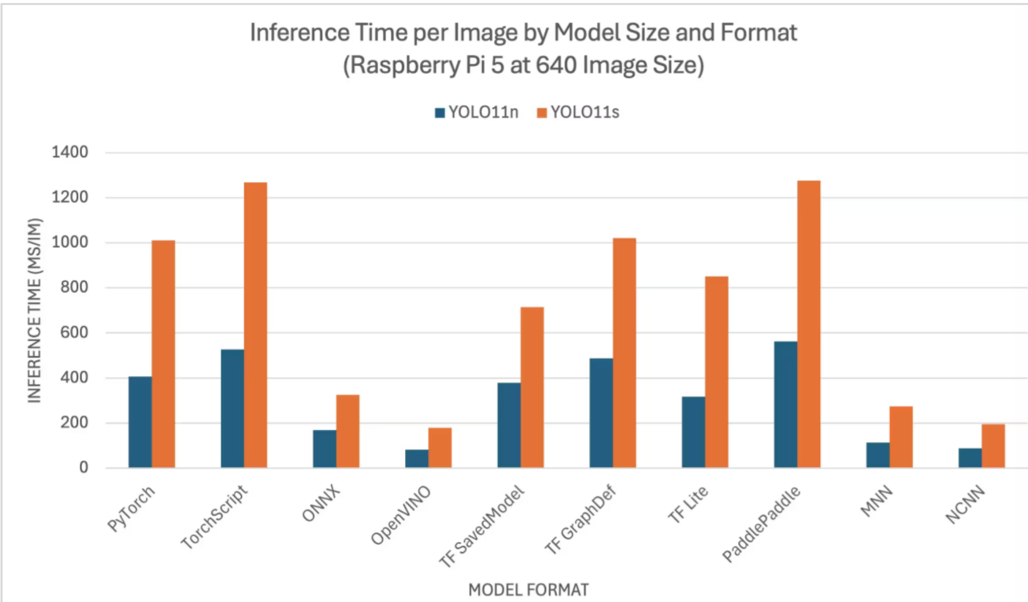


Figure 13: Inference Time per Image by Model Size and Format (Raspberry Pi 5, 640x640) [5]

A custom Python script was developed to run the model, capture video input from a USB camera, and perform real-time object detection. The script was versatile and supported various input types, including static images, video files, and live camera feeds. Detected objects were highlighted using bounding boxes drawn over the live video feed, enabling the robot to react to visual information in real time.

Following deployment, the camera system was calibrated for distance estimation. A known object was placed at a fixed distance, and the focal length of the camera was calculated using the pinhole camera model formula:

$$Distance = \frac{Real\ Object\ Width \times Focal\ Length}{Perceived\ Width\ in\ Pixels}$$

After calibration, the system was tested using objects at various distances. It consistently achieved a distance estimation accuracy of ±1 cm, which was essential for allowing the robot to approach and grip objects within the effective range of its lifting arm. The YOLOv11n model also achieved a 92% detection confidence rate, indicating robust and consistent trash identification performance under controlled conditions. Together, these features enabled reliable and efficient autonomous trash collection.

In the integrated system, the Raspberry Pi serves as the high-level controller, running the object detection model and analyzing visual data from the camera. Once an object is detected, the Pi calculates its position on the screen and estimates the distance using the calibrated focal length. This spatial information is then transmitted to the Arduino, which acts as the low-level controller responsible for actuating the motors. Based on the received data, the Arduino determines the necessary motor movements to align the robot and activate the gripper, enabling accurate and efficient object collection. This division of responsibilities ensures smooth coordination between vision processing and mechanical execution.

Electrical System Design

The electrical system consists of two main cycles, the power cycle and the control cycle. The power cycle starts with a 12V power supply plugged into the wall which then provides power to the three L298N motor controllers connected in parallel. The motor controllers then power the Arduino uno through a stepped down 5V voltage output which is also supplemented by power from the Raspberry Pi. On the control side, the Raspberry Pi 4, which is also plugged into the wall, has full control over the camera and computation related to it as well as the AI classification model. The code for the control of the 26:1 stepper, the mg996r servo, and the 3 DC motors is stored on the Arduino but is influenced by the camera output on the Raspberry Pi. When an object from one of the specified groups is detected by the code, it will cause the wheels to turn towards it and drive to collect it. This relationship also determines the distance of said object, so the robot knows how far it needs to drive and once it reaches its destination, the collection and compaction sequence starts. Finally, the 26:1 stepper motor was chosen due to its output torque and positional accuracy, the mg996r was chosen due to its size weight and torque output and the 3 dc motors were chosen due both their high torque and high speed. A labelled electrical schematic can be seen in the figure below.

1. 3 x DC motors
2. 3 x L298N motor drivers
3. Nema 17 gearbox stepper motor
4. MG996R servo motor
5. 12v Power Supply
6. Raspberry Pi 4
7. USB Camera
8. Arduino Uno

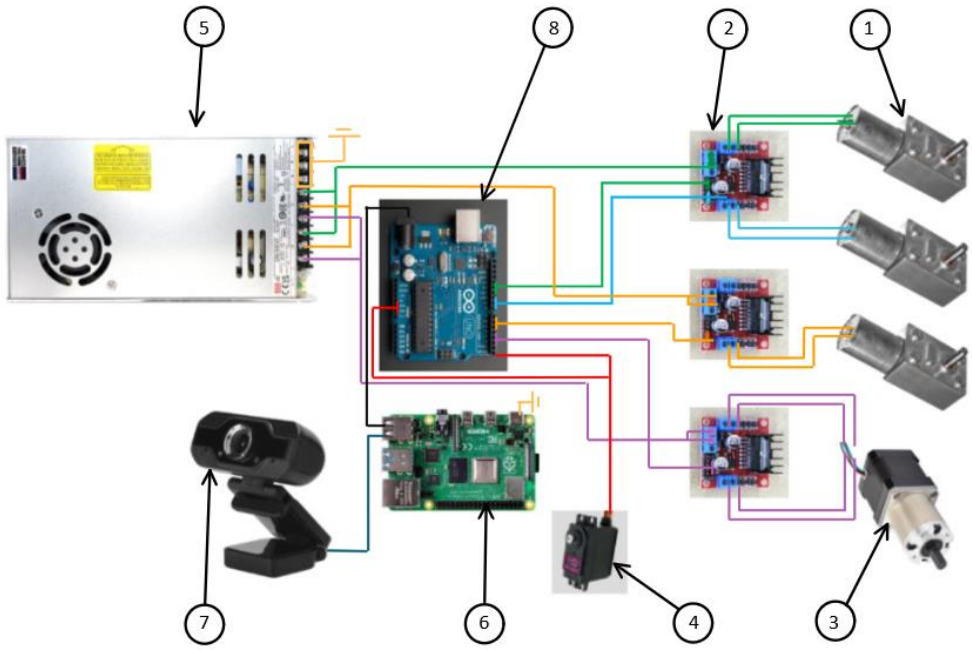


Figure 14: Electrical Schematic

Detailed Design Analysis

The selected design was met the functional, safety, and performance targets through a rigorous analysis of the entire system which involved modeling key mechanical subsystems in CAD, verifying design feasibility through calculations and performing finite element optimizations to enhance feasibility of this project. The analysis was supported by core mechanical engineering principles, with references to various mechanical engineering theories and applying design standards.

Gripper System Analysis

The gripper system for collecting garbage uses a rack and pinion mechanism. The design choice allows for reliable linear motion with good amount of mechanical advantage, which is ideal for picking up debris in various conditions. In this design, the pinion gear is driven by a motor meshing with a rack and pinion which runs along the gripper system. Powered by the motor, the motion of the pinion gear translates to the linear motion of the gripper, providing grip for trash collection.

The force required to operate the rack and pinion system for garbage collection was calculated using principles of static equilibrium, accounting for the weight of the garbage, the coefficient of friction of the trash against the gripper, radius of the pinion gear, and the required gripping force.

To ensure effective garbage collection using the rack and pinion gripper system, a force and torque analysis was conducted. The gripping mechanism depends on friction between the surface of the garbage and the gripper, which must equal or exceed the weight of the garbage to prevent slipping. The worst-case scenario assumed a friction coefficient of μ = 0.1, for metal on metal. To find this force using a rack and pinion setup, the required torque needed at the pinion was determined as approximately 0.16 Nm. This analysis ensures that the gripping system can securely hold garbage without slippage, ensuring this system works.

To validate the structural integrity of the gripper, a finite element analysis was performed on the design. A worst-case applied force was modeled, simulating an impact from a heavy object, determined to be a 3 kg branch falling onto the gripper. Therefore 30 N force was applied at the end of the gripper mechanism to represent this loading condition, and a constraint was applied at the rear of the gripper to represent the connection between the gripper and the rack and pinion mechanism. The initial FEA showed that the gripper would fail at the base (shown in Figure 15), so support was added in iterations until a final design that met the factor of safety was created (shown in Figure). A factor of safety of 4 was desired for the gripper to account for any additional, unexpected stresses, so more supports were added until this factor was reached. The updated results show maximum von Mises stress occurs at the support connection point of the gripper, with a maximum stress value of 5.50 MPa. The results of this analysis are shown in Figure 17 below.

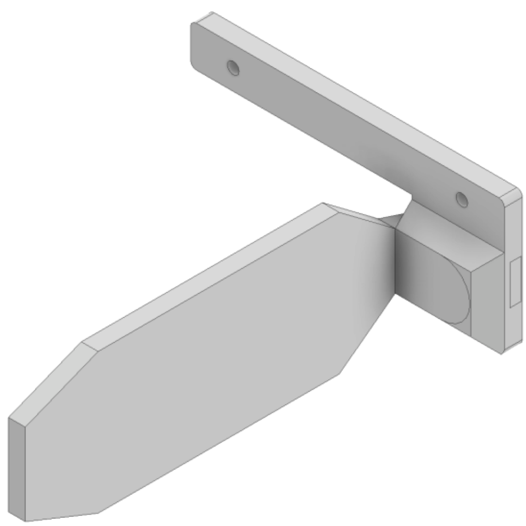


Figure 15: Initial Gripper Design

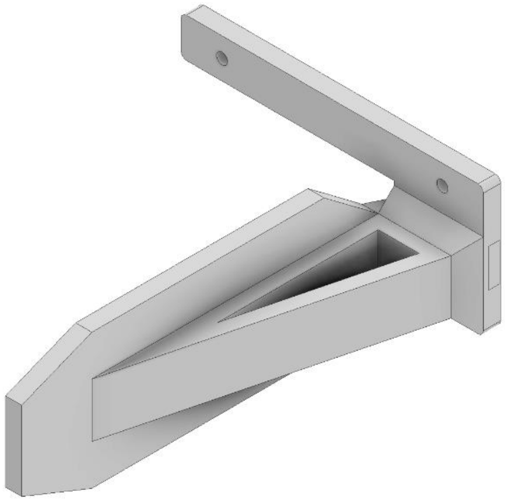


Figure 16: Updated Gripper Design, Added Support

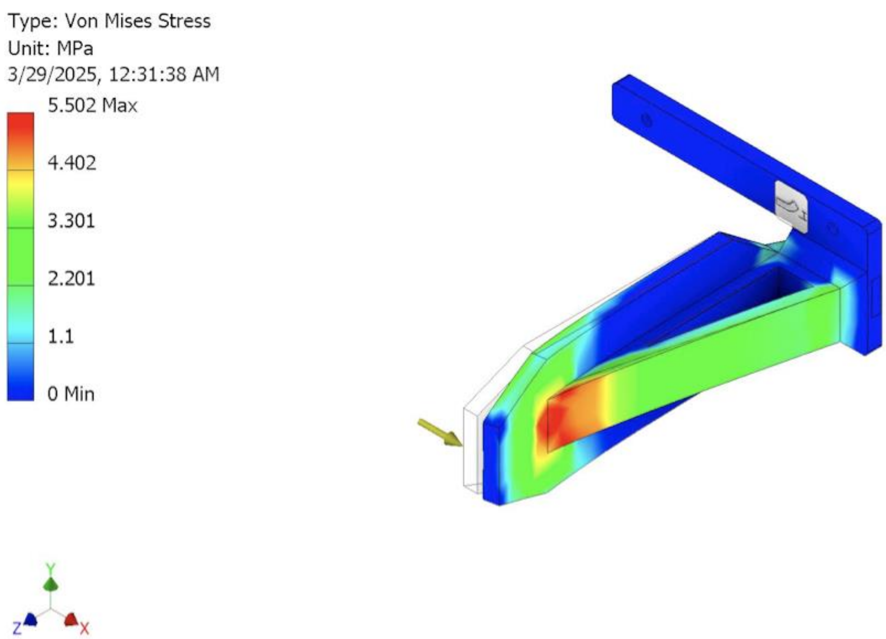


Figure 17: FEA, Von Mises Stress on Gripper. Applied Force of 30 N

The FEA was based on several assumptions, which carry limitations. This includes linear elastic material behavior, static loading conditions, perfect fixed constraints at the gripper's rear, and uniform isotropic material properties. More so, environmental effects such as temperature or humidity were neglected during this analysis. Limitations include inaccuracies due to mesh sensitivity and a simplified geometry that does not capture possible imperfections. Additionally, the applied load was treated as a concentrated point force instead of a larger distributed load which all may affect the results. Detailed calculations are provided in Appendix D.

Lifting Arm Analysis

The lifting arm mechanism responsible for raising and lowering the gripper system, was designed to be fabricated from PLA 3D-printing and mounted onto a rotating shaft powered by a motor. As the motor applies torque to the shaft, raising or lowering occurs, depending on the motor direction.

A force and torque analysis were completed to ensure the lifting arm can reliably operate under full load conditions. The key factors that were considered were the weights of the gripper, garbage, and arm structure, as well as the moment of the arm and torque. Using static equilibrium analysis, the torque required at the shaft was determined by calculating the moment generated by the lifted mass. To determine the moment, the relationship of length × cos(θ) × mass is used, accounting for the effective component of the load contributing to rotation. Likewise, torque simplifies to length × mass × gravity, and through applying these equations, our calculations ensures that the lifting arm is properly sized to overcome the weight of the gripper system and garbage without failure.

The required torque to operate this system was calculated assuming a worst-case garbage load and the known geometry of the lifting arm, the required torque was calculated to ensure the selected motor and shaft could handle the load with an appropriate safety factor of 2.5.

For the arm mechanism, a 30 N load was applied at the connection point between the arm and the gripper, representing the same worst-case scenario discussed in the gripper analysis. Constraints were applied at the connection with the shaft, using a pin constraint and a regular constraint along the x-axis. The results show the maximum von Mises stress occurs at the split section of the arm, with a maximum stress value of 8.09 MPa. The arm was designed with a factor of safety of 2.5 to ensure safe operation while maintaining the lightweight properties of the design. The variables explored to reach this factor of safety included the arm thickness and infill density, to alter its weight and size. Results of the analysis are shown in Figure 18 below. Detailed calculations are provided in Appendix D.

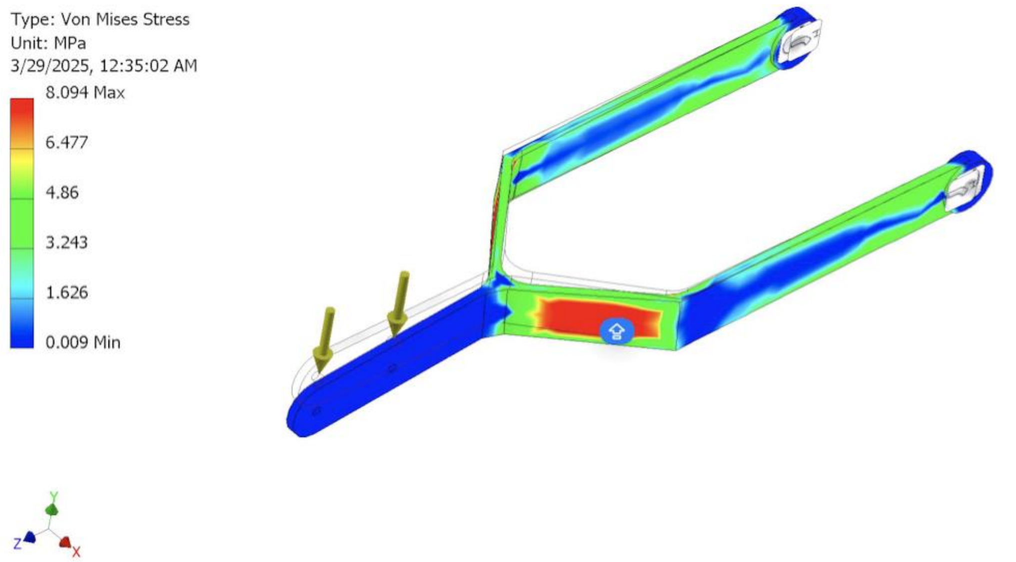


Figure 18: FEA, Von Mises Stress on Lifting Arm. Applied Force of 30 N

Compaction System Analysis

The compaction system for compressing garbage in the hopper uses a lead screw-driven scissor lift mechanism, powering the compactor plate. This design uses a strong mechanical advantage and controlled motion on linear rails resulting in a large compaction force within the hopper. The system incorporates 3D-printed linkages, lead screw system, and a motor, powering the compaction system to move the compaction plate. The lead screw extends or retracts the scissor mechanism, allowing the compaction plate to compress garbage effectively within the hopper.

The force required to operate the compaction system was determined using principles of Newton's second law, which considers the force of the compression plate and modelling the compressed garbage as a resisting spring force from the compression. Through manipulation of the Newton's second law, we create an equation which states the force of the compression, Fp must equal the mass of the plate multiplied by its acceleration in addition to the estimated spring constant of the garbage multiplied by the compression system. This force is then related to the input force at the lead screw, FL, through the geometry of the scissor lift mechanism using FL = 2 * Fp / tan θ.

More so, to ensure effective compression of garbage, a torque analysis on the lead screw was further conducted. The torque needed to power the lead screw was calculated based on the scissor mechanism's geometry, specifically using the lead angle and friction coefficient. For a determined plate acceleration of 3.3 cm/s^2 based on motor's capabilities and a compression distance of 0.2 meters, the required torque was determined to be approximately 1.25 Nm through the equation specified below.

For the rail-plate mount, a 150 N load was applied to one side of the mount, simulating the force applied from the compactor plate as it pushes back garbage. Constraints were placed along the Y-Z plane, simulating its connection with the bearings of the linear rail. The results showed that the maximum von Mises stress occurred at the rail and plate connection, with a maximum stress value of 9.734 MPa. The design of the mount was optimized with a factor of safety of 3, ensuring safe operation while maintaining the lightweight properties of the design. Results of the analysis performed on the compaction plate mount are shown in Figure 19 below.

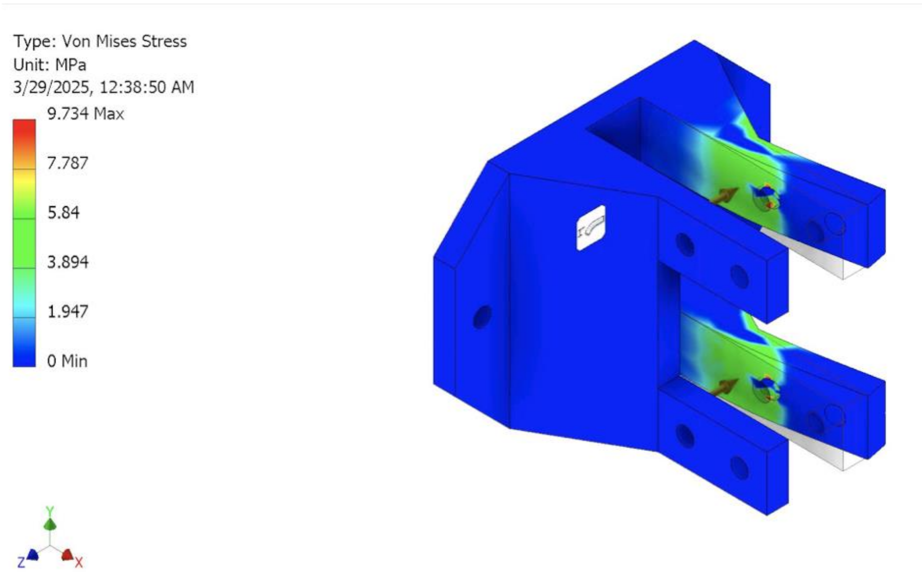


Figure 19: FEA, Von Mises Stress on Compaction Plate Bracket. Applied Force of 30 N

To validate the structural integrity of the linkage in the scissor mechanism, a finite element analysis was performed on the design. A 37.5 N load was applied to each end of the linkage at the pinned connection, simulating the forces exerted during the compaction process. Constraints were applied against the x-z plane to represent the fixed connections. The results showed that the maximum von Mises stress occurred at the pinned connection, with a maximum stress value of 4.293 MPa. This also caused some bending and deflection within the linkage. Through multiple iterations, the linkage was designed with a factor of safety of 4.5, ensuring it could withstand the applied forces while maintaining lightweight properties and reliable performance. Results of the analysis performed on the linkage are shown in Figure 20 below.

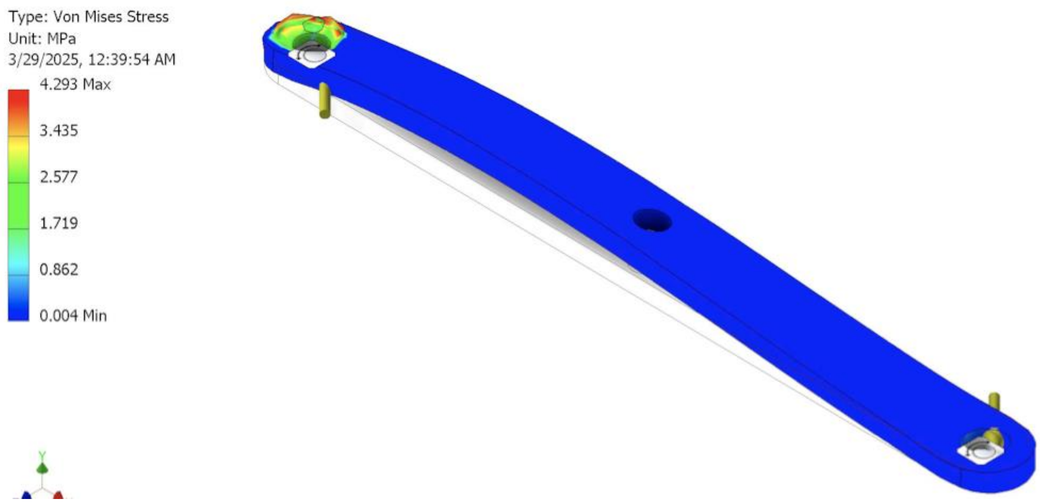


Figure 20: FEA, Von Mises Stress on Linkage. Applied Force of 30 N

Detailed calculations are provided in Appendix D.

Fabrication

To meet the objectives of the project, the fabrication of this project involved creating a collection method to pick up trash (gripper at the end of an arm), a compaction mechanism (scissor lift attached to a compaction plate), and a hopper to store the collected trash in. Fabrication also required creating a place to store the electrical systems in, to avoid contamination from the collected trash. Appendix B contains detailed sketches of the final design with a labelled bill of materials. The fabrication includes all components from the final design, including both mechanical and electrical components. Figure 21 below shows an image of the final fabricated prototype. Additional photos of the fabrication process are included in Appendix D.

Figure 21: Fabricated Prototype

To save on cost, the main manufacturing methods chosen for this project involved 3D printing and laser cutting plywood where possible. Table 3 below details the components that were manufactured by the group and the methods used for each.

Component	Manufacturing Method
Motor mounts (x5)	3D printing
Driven wheels (x2)	3D printing
Hopper	Laser cutting (plywood)
Compaction Plate	Laser cutting (plywood)
Gripper	3D printing
Rack and pinion for gripper	Laser cutting (acrylic)
Bearing mounts (x4)	3D printing
Scissor lift linkages	3D printing
Scissor lift connections (compaction plate and lead screw)	3D printing
Compactor plate brackets	3D printing

Table 3: Manufacturing Methods

In addition to the manufacturing detailed in the table above, several modifications were made to pre-purchased components such as metal rails and shafts. The metal shaft connected to the lifting arm, the metal rails connected to the compaction plate, as well as the metal lead screw, were cut to size in the machine shop using an angle grinder, then deburred and smoothed using a grinder. Additionally, the L brackets that hold the hopper in place to the baseplate were cut from 1/8" angle brackets using a band saw, then drilled and tapped.

As seen in the table, the gripper mechanism was fabricated using a combination of 3D printing and laser cutting. A closeup view of the fabricated gripper can be seen in Figure 22 below. The rack and pinion, connected to a small servo motor with a bolt, was fabricated by laser cutting 3 mm acrylic due to its strength and accuracy. High accuracy was required so that the rack and pinion would accurately mesh, which was beyond the abilities of a 3D printer. The gripper jaws, however, were 3D printed, and a rubber strip was added along the inside of the grippers for added friction. This was added during the testing phase due to some objects slipping out of the gripper when tested. A plate to hold the rack and pinion in place was also laser cut, while a motor mount to hold the servo and connect the gripper to the arm was 3D printed.

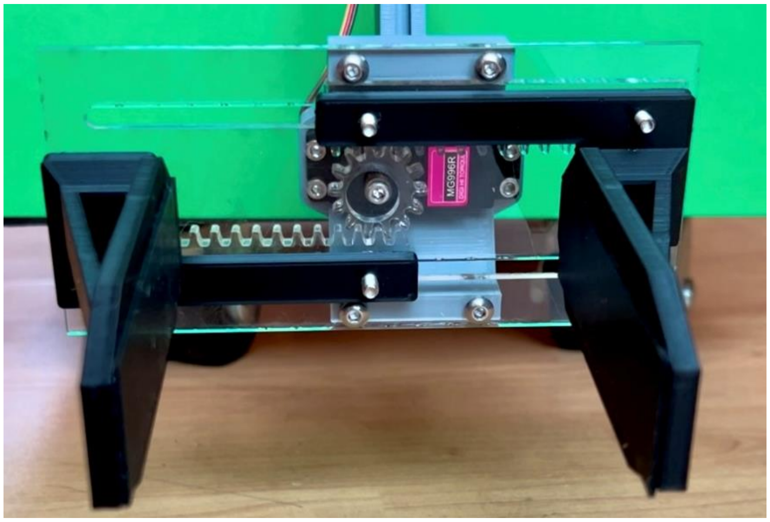


Figure 22: Fabricated Gripper

The majority of the components used for the lifting arm assembly were 3D printed, including the arm itself, the motor mounts, and bearing mounts, as seen in Figure 23. To prevent the arm from torsional stress if only one side of the arm was driven, as was originally designed, a shaft was used through both sides of the arm, coupled to the motor.

Figure 23: Fabricated Lifting Arm

3D printed hubs were used to connect the rotation of the arm to the shaft, as seen in Figure 24, which rely on friction between the screw and the shaft. The same hub design was used to connect the driven rear wheels to the motor shaft.

Figure 24: Fabricated hub for Lifting Arm and Wheels

The compaction system was fabricated using laser cut plywood for the plate, and 3D printed brackets to connect the plate to the purchased linear rails. The scissor lift portion of the compaction mechanism was also 3D printed, as were the motor mount, connection from scissor lift to lead screw, and connection from scissor lift to compaction plate. To allow rotation between the linkages, they were manufactured so that half of the holes were through-holes and half were tapped. Then, M5 bolts were put through both linkages to allow rotation of the linkage with the through-hole. A DC motor was coupled to the lead screw and used to power the compaction plate. Photos of the fabricated compaction mechanism are shown in Figure 25 below.

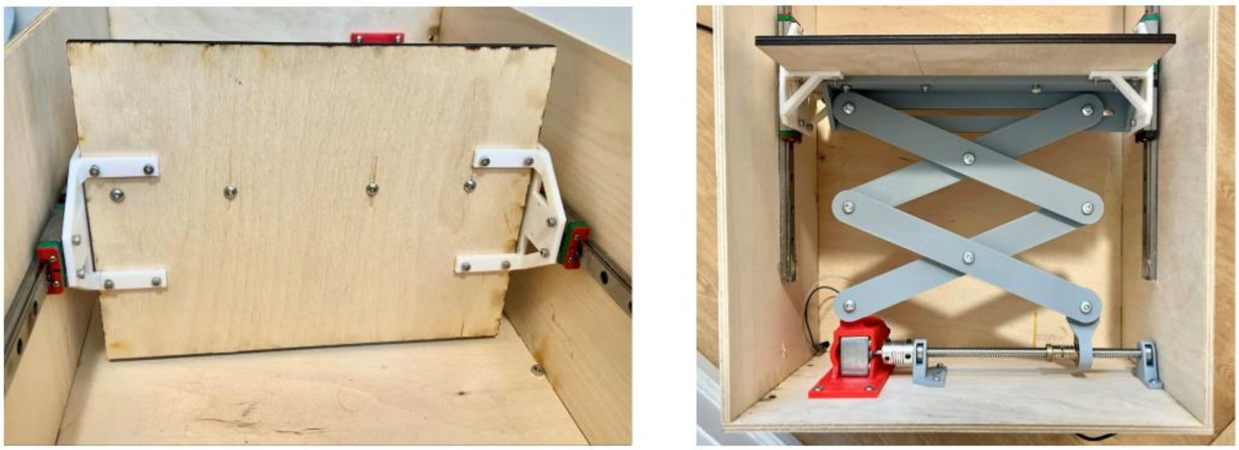


Figure 25: Fabrication of Compaction Mechanism

The base of the robot, similar to the compaction plate, was fabricated by laser cutting plywood. It was fabricated in puzzle-like pieces, designed to fit together with tight tolerances, so that the base could be glued together but have extra durability due to the interlocking pieces. This was done because the plywood was too thin to put screws or nails through, so glue was the best option for assembly. This was also done for the false-bottom which creates the cavity in which the electrical components are stored. To hold the base of the robot to the hopper, several L brackets were cut from a piece of angle bracket, then drilled and tapped. This was done instead of buying pre-made brackets both due to budget reasons and because this gave the brackets a sharp corner, which is unlike most pre-made brackets. Holes were drilled on the hopper at the exterior of each threaded L bracket so that the hopper could be securely attached to the base. However, even without bolts, the placement of the brackets was able to hold the hopper in place. An image of the base with the fabricated L brackets can be seen in Figure 27. Holes were also cut in the hopper in places where wires or motor shafts protruded out from the base, such as where the wheels were mounted.

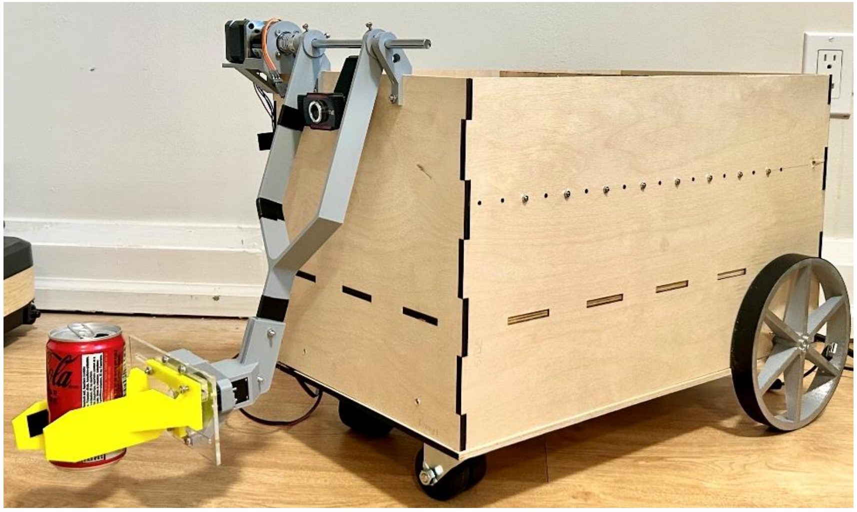


Figure 26: Fabricated Hopper

Figure 27: Fabricated base

Other fabricating included the rear wheels, which were directly connected to the shaft of a DC motor using 3D printed hubs, as mentioned above. The wheels were 3D printed and had a rubber coating around the exterior for added traction. The wheels were fabricated by the team instead of purchased due to the size of the castor wheels, which were borrowed from the machine shop. These wheels caused the rear-driven wheels to require a very specific diameter for the base to lay flat, which was not possible to purchase.

The exterior image, seen in Figure 21, was also designed and fabricated by the group by laser cutting posterboard and attaching it to the sides of the robot. This was done to conceal some of the screws and wires that were exposed to give the prototype a more polished look which is more representative of what the real product would look like.

The electrical systems were integrated as designed, as shown in Figure 28. All components were mounted to the base using M3 bolts and secured in place with nuts on the bottom of the base.

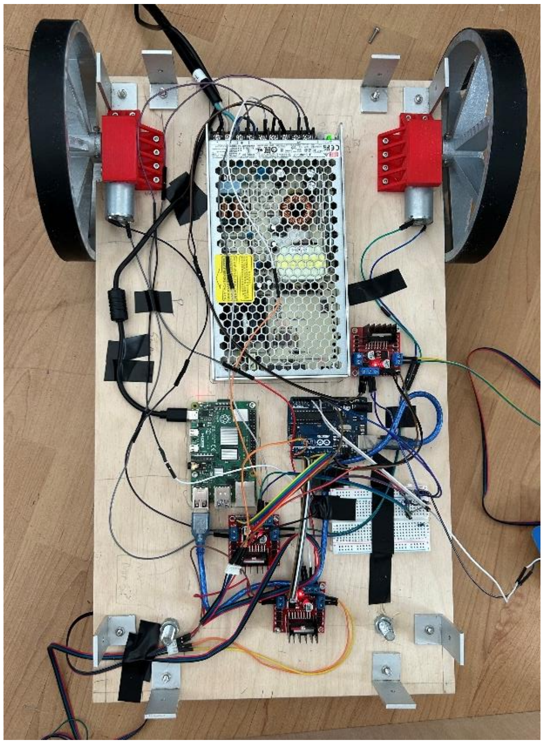


Figure 28: Fabricated Electrical System

Many components were also either purchased or borrowed from the machine shop, and shown in the table below. Borrowing components was done when possible to save on cost, such as for the motors and castor wheels. These borrowed components were often the limiting factors when fabricating the project due to their size and shape. For example, the castor wheels were borrowed from the machine shop, which dictated the required size of the driven rear wheels, causing them to require 3D printing to achieve their custom size, instead of buying pre-fabricated wheels. A breakdown of the components purchased or borrowed by the group is included in the table below.

Table 4: Fabrication Cost Summary

Cost Category	Quantity	Unit Cost	Total Cost
Material (PLA + Plywood)	1	$40	$40
Fasteners + Bearings	1	$14.29+$12.88+$10+$11.99	$49.16
Gripper Motor (MG996R)	1	$18	$18
Motor Drivers + Arduino	1	$31.99+$12.99	$44.98
Miscellaneous	1	$46.60	$46.60
Power Supply	1	$19.97	$19.97
Limit Switch	1	$10.99	$10.99
Adhesive Rubber	1	$15.99	$15.99
Miscellaneous	1	$48.31	$46.60
DC Motors	3	$23.87	$71.61
Current Sensors	1	$13.99	$13.99

Pressure Sensor	1	$16.77	$16.77
Geared NEMA 17 Motor[1]	2	$0	$0
Lead Screw[1]	1	$0	$0
Castor wheels[1]	2	$0	$0
Camera[2]	1	$0	$0
Raspberry Pi[2]	1	$0	$0
Shaft[2]	1	$0	$0
Linear rails[2]	2	$0	$0
		Total cost	**$467.1 (including HST)**

[1]Borrowed from the Mechanical Engineering Undergraduate Laboratory

[2]Pre-owned by team members

The total cost of the project amounted to $467.10 including HST, staying within budget while allowing for a complete and functional prototype. Fasteners included a range of M3, M4, and M5 bolts and nuts to accommodate different component sizes and assembly requirements. Bearings were used to support rotating shafts and reduce friction in key moving parts. The miscellaneous category accounted for various unlisted but essential items such as adhesives, tools, shipping fees, and extra hardware acquired during assembly and testing. By prioritizing cost-effective materials like PLA and plywood, and utilizing accessible electronics such as the MG996R servo, DC motors, and L298N drivers, the team was able to balance affordability with functional performance across all subsystems.

It's also important to note that several key components—such as the lifting arm stepper motor, metal shafts, and the Raspberry Pi 4—were sourced from previous projects or lab stock and were not included in the final cost breakdown. If these items had to be purchased new, the total project cost would have been significantly higher. For example, a Raspberry Pi 4 alone can cost upwards of $80–$100 CAD, and quality stepper motors and precision shafts add further expense. Reusing these components not only helped stay within budget but also emphasized the project's sustainability and efficient use of available resources.

The fabrication of this project meets the objectives by enabling the robot to successfully collect and compact trash, as well as giving the robot the ability to drive around. Given that the mechanical systems of this project (i.e. the collection and compaction mechanisms), were the primary goal, successfully fabricating these systems so that they work as intended means the robot met the objectives.

Testing

Throughout the project, extensive testing was conducted to ensure GarBot met its performance and safety objectives. These tests covered mechanical components, AI detection accuracy, system calibration, and full integration under realistic use cases.

The team began with motor testing to verify that each selected motor could deliver the required torque. Individual tests were conducted using an Arduino to monitor performance under simulated loads. While the gripper servo and wheel DC motors performed reliably, the Nema 17 stepper motor originally selected for the compactor lacked the torque necessary to compress trash effectively. This led the team to replace it with a high-torque DC motor, which proved much more reliable during repeated compaction cycle tests.

After completing individual motor tests, the team proceeded to verify the system's ability to operate multiple motors concurrently. The Raspberry Pi handled high-level control while the Arduino managed motor signals through L298N motor drivers, all powered by a single 12V supply. During testing, the motors performed reliably when operated together, with no signs of instability or power-related issues. This confirmed that the electrical and control systems were capable of supporting simultaneous multi-motor operation, which was essential for the robot's real-time functionality during tasks such as driving, lifting, and compacting.

The gripper mechanism was tested separately to confirm its ability to grip and release objects without slipping. Using PWM signals from the Arduino, it was verified that the servo motor responded correctly and provided enough force to hold lightweight debris. No design changes were needed for this subsystem.

On the AI side, camera calibration tests were performed to fine-tune the focal length used in object distance calculations. A known object was placed at various distances from the camera and applied the standard focal length formula to estimate real-world distance. This improved the system's detection accuracy to within ±1 cm — a crucial factor in ensuring that the robot only attempts to pick up trash within its effective reach. This detection accuracy was computed by placing an object in front of the camera, manually measuring the true distance from the object to the camera, and then comparing to the output of the system.

Figure 29: Calibration of camera using a bottle

A wheel traction test was also performed to evaluate the robot's mobility. The 3D-printed wheels, coated with rubber strips, were tested on smooth indoor surfaces and a low-friction mat. While traction was generally good, minor slippage was observed on smooth floors at low speeds. This was noted as a future improvement area, possibly requiring upgraded tread or material.

A critical test involved the compactor plate brackets, which failed during an early compression test. The original 3D-printed brackets broke under load, exposing a design weakness. As a result, the brackets were redesigned with improved geometry and increased infill to better handle the forces applied during trash compaction. The redesigned brackets performed well in subsequent tests and were used in the final prototype.

Finally, full system integration tests were conducted, during which the robot executed the complete operational sequence from object detection to movement, trash pickup, and compaction. These tests were essential in confirming that all system components, including the Raspberry Pi, Arduino, motor drivers, sensors, and AI detection software, could operate together as a cohesive autonomous system.

However, several challenges surfaced during testing. The camera’s narrow field of view made it difficult for the robot to detect nearby objects unless they were centered within the frame, which reduced overall detection efficiency. Additionally, the Raspberry Pi 4 struggled to handle real-time object detection while simultaneously managing control logic. This resulted in low frame rates and delayed responses, occasionally slowing down the robot’s decision-making process.

While the team was able to optimize the system to work within these limitations, by tuning detection thresholds and simplifying control sequences, these issues highlighted opportunities for improvement. In future iterations, upgrading to a Raspberry Pi 5 could significantly improve performance. Its enhanced processing power and better thermal management would help reduce frame lag and allow for smoother, faster object detection, leading to a more responsive and reliable robot. Despite these constraints, the robot was still able to complete its intended tasks and demonstrate successful autonomous operation.

Final Discussion and Conclusions

In conclusion, the project as a whole was a resounding success with almost all of the major goals being met. The developed robot was able to collect and compact debris, compact debris to less than 50% volume, had higher FOS than the minimum required for the collection and compaction mechanisms, was able to detect people, and accurately identified 92% of debris. Every goal was completed other than the ability to collect 85% of trash within a $100m^2$ area in 1 hour, but this was primarily due to the large lag in the camera due to budget constraints, limiting the project from purchasing a better one.

There were also a few stretch goals that if implemented in the future could greatly elevate the ability of the robot to complete its intended goal. First, the robot would need to be made waterproof so it can run in any weather conditions and can reliably operate outdoors. Additionally, either infrastructure or an augment to the robot needs to be developed to allow for automatic garbage removal so that collected garbage can be disposed of autonomously. Through testing, the robot ran solely off a power supply connected to the wall; however, the implementation of a rechargeable battery pack that runs the entirety of the robot would increase both range and efficiency. Furthermore, charging station infrastructure as well as a charging port should be developed in conjunction with GPS positioning to allow the robot to automatically return to its charging station when the battery is running low. GPS positioning would also be useful with an edge detection camera system that can allow the robot to define its own collection area. Also, the robot being able to optimize a path within its boundaries that covers the full area of the park in the most efficient way and deviates from the path to pick up trash it sees before returning to the path and continuing in search of more trash. Finally, the trash that the robot collects should save the GPS location related to each piece and produce a heat map showing the highest densities of trash in the area. This would help to determine the best locations for new trash cans to be placed as it would show the most likely locations for littering.

References

[1] Ultralytics, "ultralytics/ultralytics GitHub Repository," GitHub, [Online]. Available: https://github.com/ultralytics/ultralytics

[2] Raspberry Pi Foundation, "Installing the Operating System," [Online]. Available: https://www.raspberrypi.com/documentation/computers/getting-started.html#installing-the-operating-system

[3] Raspberry Pi Foundation, "Raspberry Pi OS," [Online]. Available: https://www.raspberrypi.com/software/

[4] Ultralytics, "YOLOv11 Performance Metrics," [Online]. Available: https://docs.ultralytics.com/models/yolo11/#performance-metrics

[5] Ultralytics, "Raspberry Pi Deployment Guide," [Online]. Available: https://docs.ultralytics.com/guides/raspberry-pi/#comparison-chart

[6] Tencent, "ncnn: High-Performance Neural Network Inference Framework," GitHub, [Online]. Available: https://github.com/Tencent/ncnn

Appendix A – Extended Background Information

Currently, manual labour is the main way to collect debris and the large vacuum store cleaners are the closest thing to our design, however those don't compact the trash they collect and are extremely bulky for their purposes leaving little option for fast efficient automatic trash cleanup options which is exactly where this project fits.

Beyond the main objectives of a scaled down prototype, ensuring the safety and efficiency of the operation and the general efficacy of debris collection when compared to traditional means, we also aim to have the robot have autonomous movement, effectively compact debris, be able to locate debris and collect it.

We aim to have the robot detect trash using a camera and an ai model that can classify if items on the ground are trash or not. Then, the robot should center the trash in the camera by pivoting its wheels and detecting the distance to the trash either from an ultrasonic sensor or from calculating it using the focal length, angle and resolution of the camera. Once in the suitable range, the gripper arm should lower from out of its rest position and position itself near the trash where the gripper would close to pick up the trash. Then, the arm would rotate back over the trash bag in the bin on the main frame of the robot and drop the trash and return to its rest position. Finally, the compacting plate will crush the bag towards one of the walls of the bin where the plate runs on tracks and has the trash bag fixed to it and the wall that the plate compresses it into, and the compacting plate will return to its rest position as well.

The AI model [9] will be coded in python and will be provided with an object detection dataset where specific classes will be considered trash which the robot will be looking for while avoiding objects that are not trash. The dataset will be preprocessed, cleaned, and tested to confirm its validity [5] . The most challenging portion of this is finding a dataset that will fit our needs and be flexible enough to classify a large range of objects.

The camera will provide a video feed that the model will be using to detect real world objects and avoid or attempt to collect them. It will likely be on a swivel that has a connected encoder/potentiometer to determine its rotational position over a 90-degree range as well as providing a better field of view for the robot to find trash. The camera along with the AI will need appropriate voltage and amps [1] to control them as well as the ability to house the model on board the microcontroller [15]. The motor for the swivel will need to have an appropriate motor controller as well as torque calculations to determine what motor would be best for the application. The most challenging portion of this is the connection of all the electrical components as well as making sure all their voltage and amperage needs are met without sacrificing or overloading the others.

Once trash is found, the camera will compare the bounding box coordinates of the object to those of the camera frame and move the camera to its 0-degree position as well as centering the trash in the frame by rotating the robot by its wheels. The motors for the wheels [11] will need to have an appropriate motor controller as well as torque calculations and potential gear calculations to determine what motors would be best for the application. [2]. The most challenging portion of this is making sure that the torque of the motor can handle the weight of the arm and gripper and the wheels being strong enough to turn under the weight of the robot making sure that the motors don't overload.

Either the ultrasonic sensor [13] through distance values or the camera though angle, resolution and focal length calculations to find distance [7] will be done to determine if the trash is in the suitable range to be picked up [14]. The most challenging portion of this will be iterating though detection mechanisms to find the best one for the project.

The mechanical arm [12] with a range of 210 to 225-degrees where 0 is the rest position will swing down to pick up position between 210 and 225-degrees where the mechanical gripper [8] will grasp the trash [4], and the arm will the bring the trash to about 60 degrees where it will be dropped in the bin. The motors for the arm and gripper will need to have appropriate motor controllers as well as torque calculations and potential gear calculations to determine what motors would be best for the application. [3] The most challenging portion of this will be finding the suitable motor controller to couple the with the motors already existing and new ones for the arm and gripper and have its power requirements mesh with the other electrical systems requirements.

The mechanical compactor with a range of 10cm – 30cm will compact trash to 50% its original volume [10] after a set amount of time from either a repeating timer or an on-board proximity sensor with a large variety of potential mechanisms such as lead screws, rack and pinions, and lever arms that will require gear calculations as well as torque calculations for the motor as well as choosing an appropriate motor controller for the application. The required stress for compacting the trash as well as various loading conditions will also need to be calculated. [6] The most challenging portion of this will be deciding on and finding the most suitable actuating mechanism to compact the trash.

[1] “How to calculate amps, volts, and Watts,” WebstaurantStore, https://www.webstaurantstore.com/guide/600/how-to-calculate-amps-volts-and-watts.html#:~:text=Volts%20%3D%20Watts%20%2F%20Amps,Watts%20%2F%2010%20Amps%20%3D%20240%20Volts (accessed Oct. 23, 2024).

[2] I. Patsiaouras, “How to calculate torque: Your complete guide,” Bota Systems, https://www.botasys.com/post/how-to-calculate-torque (accessed Oct. 23, 2024).

[3] M. Brain, “How gear ratios work,” HowStuffWorks Science, https://science.howstuffworks.com/transport/engines-equipment/gear-ratio.htm (accessed Oct. 23, 2024).

[4] Ruslan MushkaevRuslan Mushkaev 19211 silver badge1111 bronze badges, cmscms 4, and Chet MillerChet Miller 34.4k33 gold badges2121 silver badges4848 bronze badges, “Understanding the internal forces in a rotating bar,” Physics Stack Exchange, https://physics.stackexchange.com/questions/380374/understanding-the-internal-forces-in-a-rotating-bar (accessed Oct. 23, 2024).

[5] FIRSTIntern and Instructables, “Control System Basics,” Instructables, https://www.instructables.com/Control-System-Basics/ (accessed Oct. 23, 2024).

[6] How to select a brushless DC Motor Controller, https://www.portescap.com/en/newsroom/blog/2023/08/how-to-select-a-brushless-dc-motor-controller (accessed Oct. 23, 2024).

[7] Annael.le-Poullennec, “Resolution and DPI,” PSL Explore, https://explore.psl.eu/en/tools-and-training/tutorials/resolution-and-dpi#:~:text=Understanding%20the%20DPI%20ratio,paper%20should%20count%202480x3507%20pixels (accessed Oct. 23, 2024).

[8] M. Liarokapis and A. M. Dollar, “Learning the post-contact reconfiguration of the hand object system for adaptive grasping mechanisms,” *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, vol. 1, pp. 293–299, Sep. 2017. doi:10.1109/iros.2017.8202171

[9] E. Naf’an, R. Sulaiman, and N. M. Ali, “Optimization of trash identification on the house compound using a convolutional neural network (CNN) and Sensor System,” *Sensors*, vol. 23, no. 3, p. 1499, Jan. 2023. doi:10.3390/s23031499

[10] M. Meena, M. Sharma, M. Singh, and R. Saha, “Modeling of web-based solar powered and battery operated Automatic Trash Compactor,” *2022 IEEE Delhi Section Conference (DELCON)*, vol. 5, pp. 1–9, Feb. 2022. doi:10.1109/delcon54057.2022.9753575

[11] Y.-L. Cho and B.-K. Cho, “Hybrid control of wheeled quadruped robot for enhanced mobility and efficiency in diverse terrains,” *Journal of Institute of Control, Robotics and Systems*, vol. 30, no. 8, pp. 863–870, Aug. 2024. doi:10.5302/j.icros.2024.24.0055

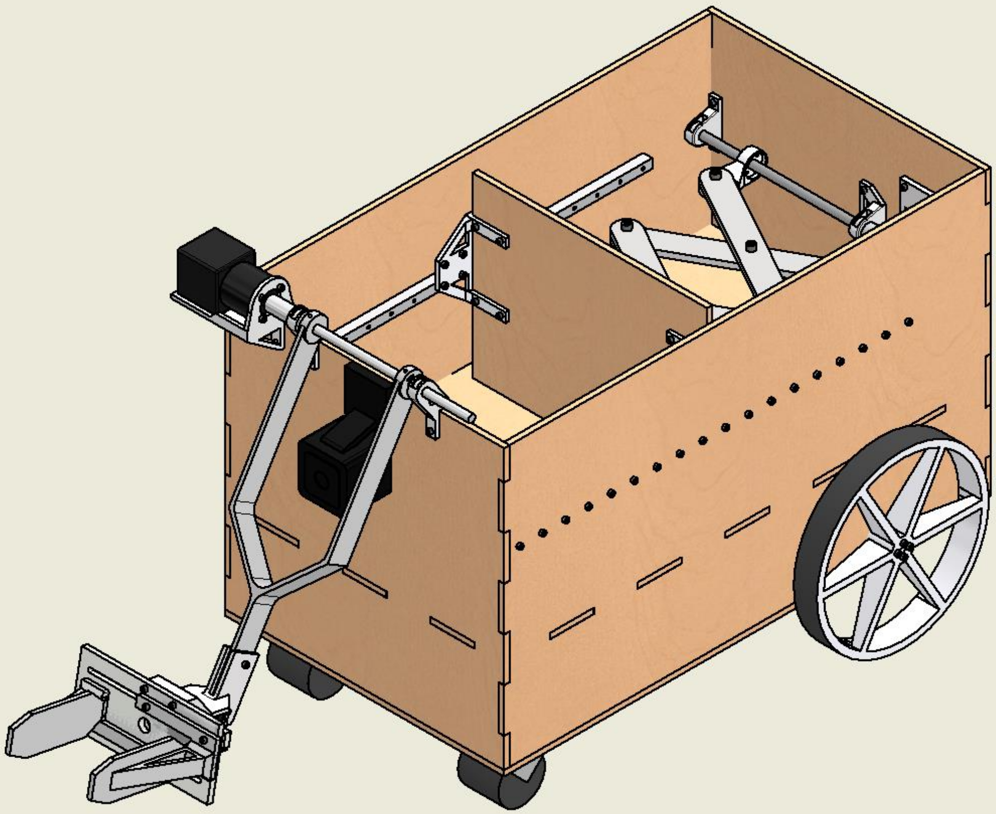
[12] B. Wang *et al.*, “A modular cable-driven humanoid arm with anti-parallelogram mechanisms and Bowden cables,” *Frontiers of Mechanical Engineering*, vol. 18, no. 1, Feb. 2023. doi:10.1007/s11465-022-0722-2

[13] J. Gu, M. Bellone, T. Pivoňka, and R. Sell, “CLFT: Camera-Lidar Fusion Transformer for semantic segmentation in autonomous driving,” *IEEE Transactions on Intelligent Vehicles*, pp. 1–12, 2024. doi:10.1109/tiv.2024.3454971

[14] P. Tornero, S. Puente, and P. Gil, “Detection and location of domestic waste for planning its collection using an autonomous robot,” *2022 8th International Conference on Control, Automation and Robotics (ICCAR)*, pp. 138–144, Apr. 2022. doi:10.1109/iccar55106.2022.9782609

[15] W. Liang, P. R. Baldivieso, R. Drummond, and D. Shin, “Tuning the feedback controller gains is a simple way to improve autonomous driving performance,” *2024 UKACC 14th International Conference on Control (CONTROL)*, vol. 10, pp. 72–77, Apr. 2024. doi:10.1109/control60310.2024.10531819

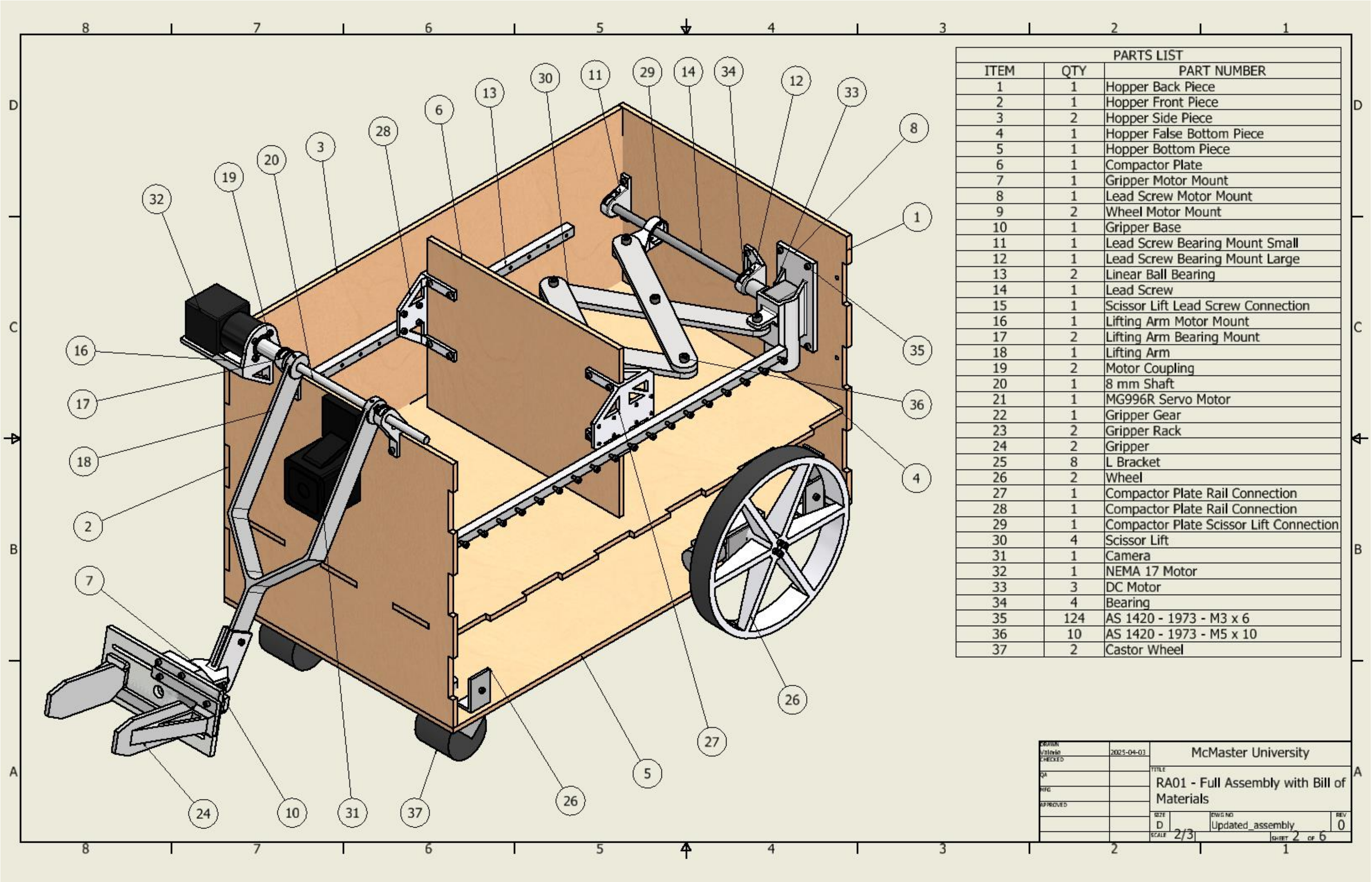
Appendix B – Final Design



PARTS LIST		
ITEM	QTY	PART NUMBER
1	1	Hopper Back Piece
2	1	Hopper Front Piece
3	2	Hopper Side Piece
4	1	Hopper False Bottom Piece
5	1	Hopper Bottom Piece
6	1	Compactor Plate
7	1	Gripper Motor Mount
8	1	Lead Screw Motor Mount
9	2	Wheel Motor Mount
10	1	Gripper Base
11	1	Lead Screw Bearing Mount Small
12	1	Lead Screw Bearing Mount Large
13	2	Linear Ball Bearing
14	1	Lead Screw
15	1	Scissor Lift Lead Screw Connection
16	1	Lifting Arm Motor Mount
17	2	Lifting Arm Bearing Mount
18	1	Lifting Arm
19	2	Motor Coupling
20	1	8 mm Shaft
21	1	MG996R Servo Motor
22	1	Gripper Gear
23	2	Gripper Rack
24	2	Gripper
25	8	L Bracket
26	2	Wheel
27	1	Compactor Plate Rail Connection
28	1	Compactor Plate Rail Connection
29	1	Compactor Plate Scissor Lift Connection
30	4	Scissor Lift
31	1	Camera
32	1	NEMA 17 Motor
33	3	DC Motor
34	4	Bearing
35	124	AS 1420 - 1973 - M3 x 6
36	10	AS 1420 - 1973 - M5 x 10
37	2	Castor Wheel

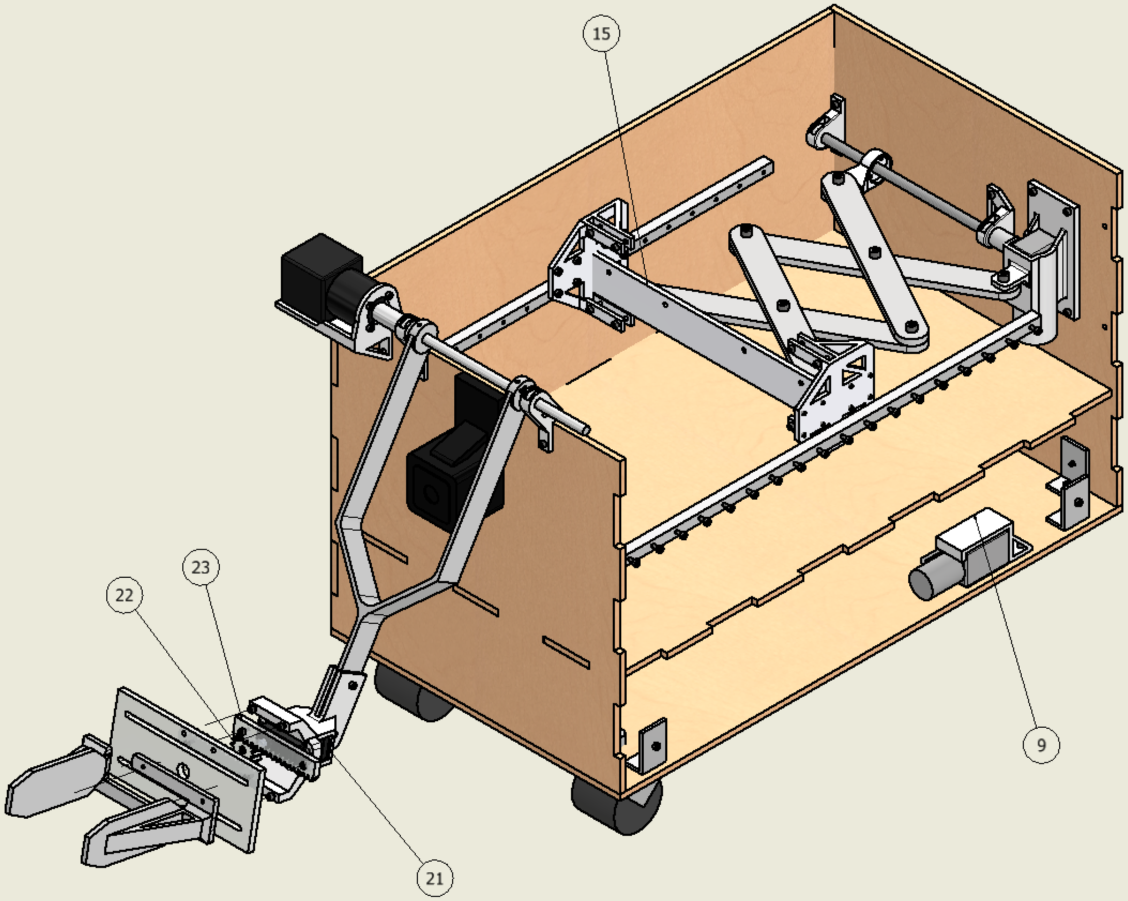
DRAWN Valerie 2025-04-03
McMaster University
TITLE RA01 - Full Assembly
SIZE D DWG NO Updated_assembly REV 0
SCALE 2/3 SHEET 1 OF 6

Figure 30: Final Design Assembly



PARTS LIST		
ITEM	QTY	PART NUMBER
1	1	Hopper Back Piece
2	1	Hopper Front Piece
3	2	Hopper Side Piece
4	1	Hopper False Bottom Piece
5	1	Hopper Bottom Piece
6	1	Compactor Plate
7	1	Gripper Motor Mount
8	1	Lead Screw Motor Mount
9	2	Wheel Motor Mount
10	1	Gripper Base
11	1	Lead Screw Bearing Mount Small
12	1	Lead Screw Bearing Mount Large
13	2	Linear Ball Bearing
14	1	Lead Screw
15	1	Scissor Lift Lead Screw Connection
16	1	Lifting Arm Motor Mount
17	2	Lifting Arm Bearing Mount
18	1	Lifting Arm
19	2	Motor Coupling
20	1	8 mm Shaft
21	1	MG996R Servo Motor
22	1	Gripper Gear
23	2	Gripper Rack
24	2	Gripper
25	8	L Bracket
26	2	Wheel
27	1	Compactor Plate Rail Connection
28	1	Compactor Plate Rail Connection
29	1	Compactor Plate Scissor Lift Connection
30	4	Scissor Lift
31	1	Camera
32	1	NEMA 17 Motor
33	3	DC Motor
34	4	Bearing
35	124	AS 1420 - 1973 - M3 x 6
36	10	AS 1420 - 1973 - M5 x 10
37	2	Castor Wheel

Figure 31: Full Assembly with Labelled BOM



PARTS LIST		
ITEM	QTY	PART NUMBER
1	1	Hopper Back Piece
2	1	Hopper Front Piece
3	2	Hopper Side Piece
4	1	Hopper False Bottom Piece
5	1	Hopper Bottom Piece
6	1	Compactor Plate
7	1	Gripper Motor Mount
8	1	Lead Screw Motor Mount
9	2	Wheel Motor Mount
10	1	Gripper Base
11	1	Lead Screw Bearing Mount Small
12	1	Lead Screw Bearing Mount Large
13	2	Linear Ball Bearing
14	1	Lead Screw
15	1	Scissor Lift Lead Screw Connection
16	1	Lifting Arm Motor Mount
17	2	Lifting Arm Bearing Mount
18	1	Lifting Arm
19	2	Motor Coupling
20	1	8 mm Shaft
21	1	MG996R Servo Motor
22	1	Gripper Gear
23	2	Gripper Rack
24	2	Gripper
25	8	L Bracket
26	2	Wheel
27	1	Compactor Plate Rail Connection
28	1	Compactor Plate Rail Connection
29	1	Compactor Plate Scissor Lift Connection
30	4	Scissor Lift
31	1	Camera
32	1	NEMA 17 Motor
33	3	DC Motor
34	4	Bearing
35	124	AS 1420 - 1973 - M3 x 6
36	10	AS 1420 - 1973 - M5 x 10
37	2	Castor Wheel

DRAWN Valerie 2025-04-01
CHECKED
QA
MFG
APPROVED

McMaster University

TITLE RA01 - Full Assembly, Hidden Parts Revealed

SIZE D | DWG NO Updated_assembly | REV 0

SCALE 2/3 | SHEET 3 OF 6

Figure 32: Full Assembly with Labelled BOM, Side Wall and Wheel Not Visible

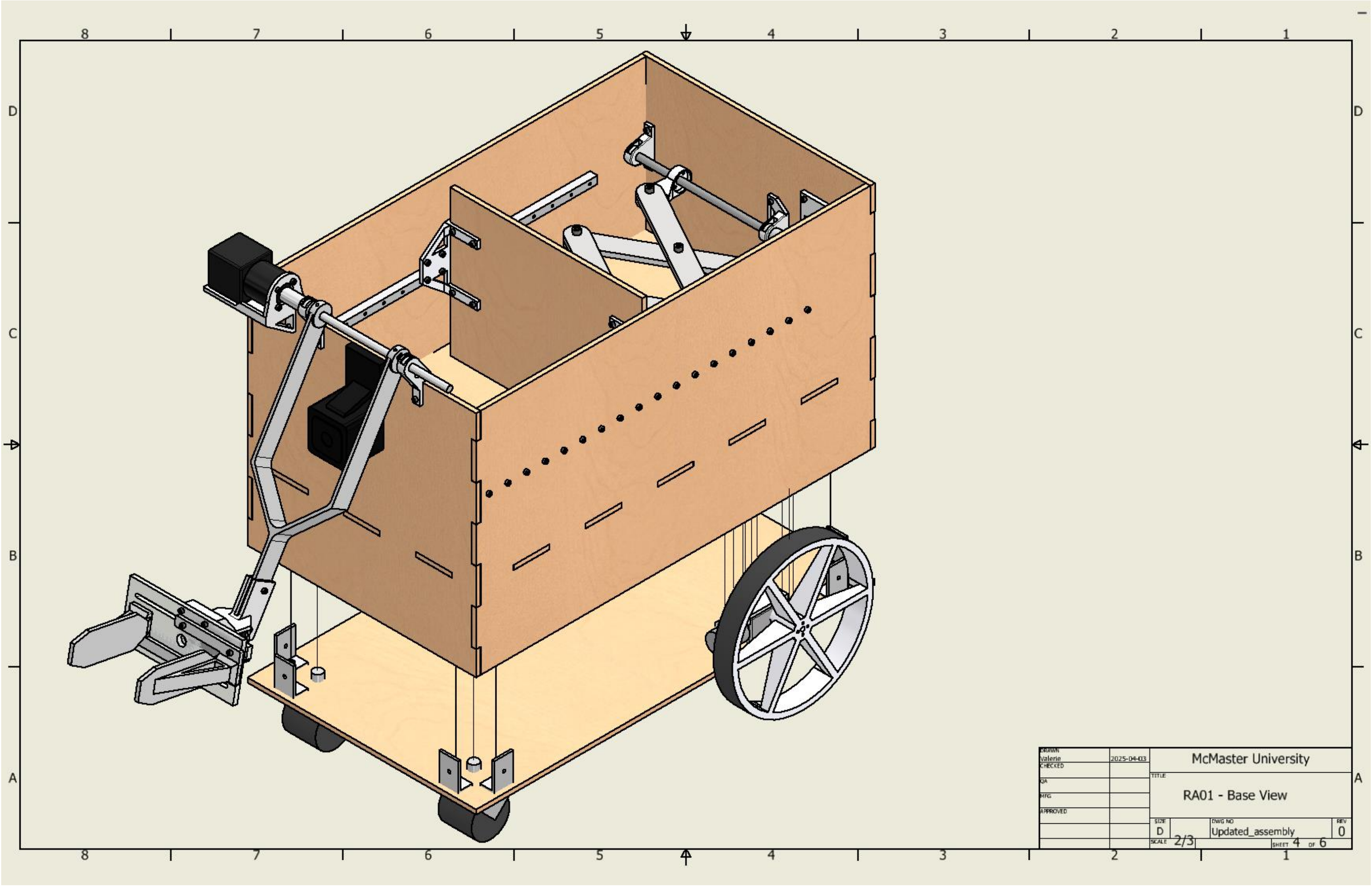


Figure 33: View of Base Under False Bottom

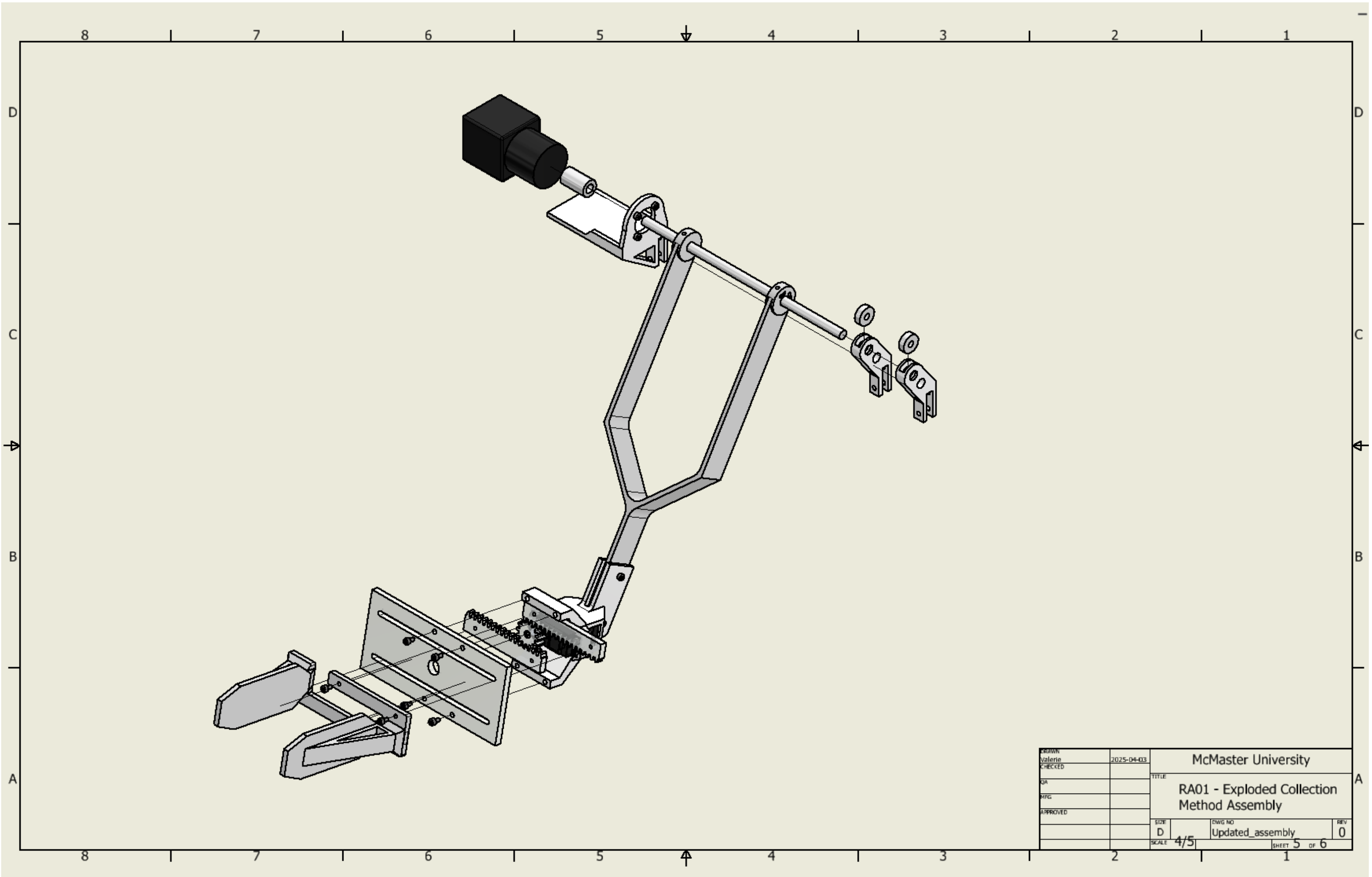


Figure 34: Exploded View of Collection Mechanism

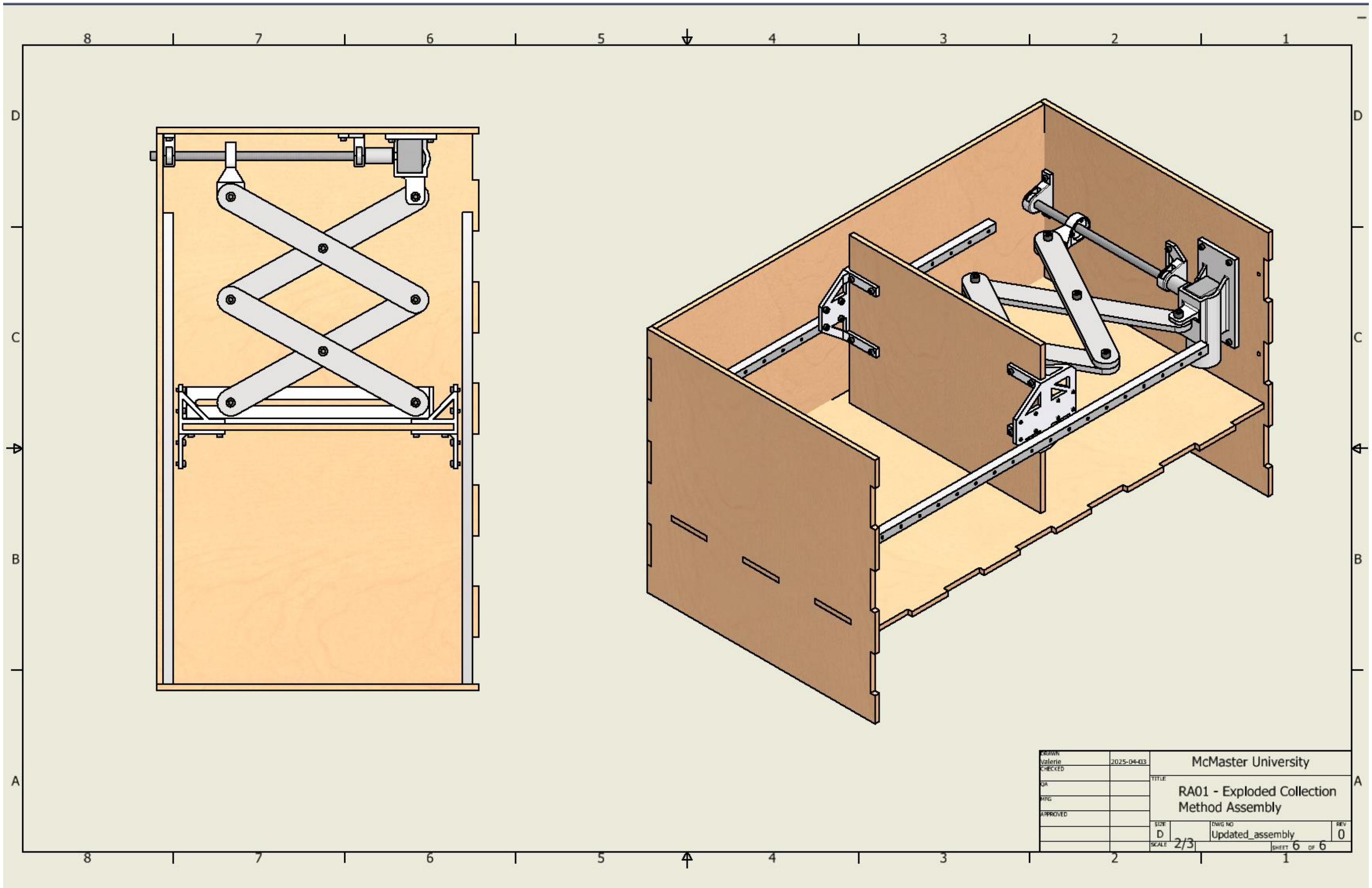


Figure 35: View of Compaction Mechanism

Appendix C – Part Drawings

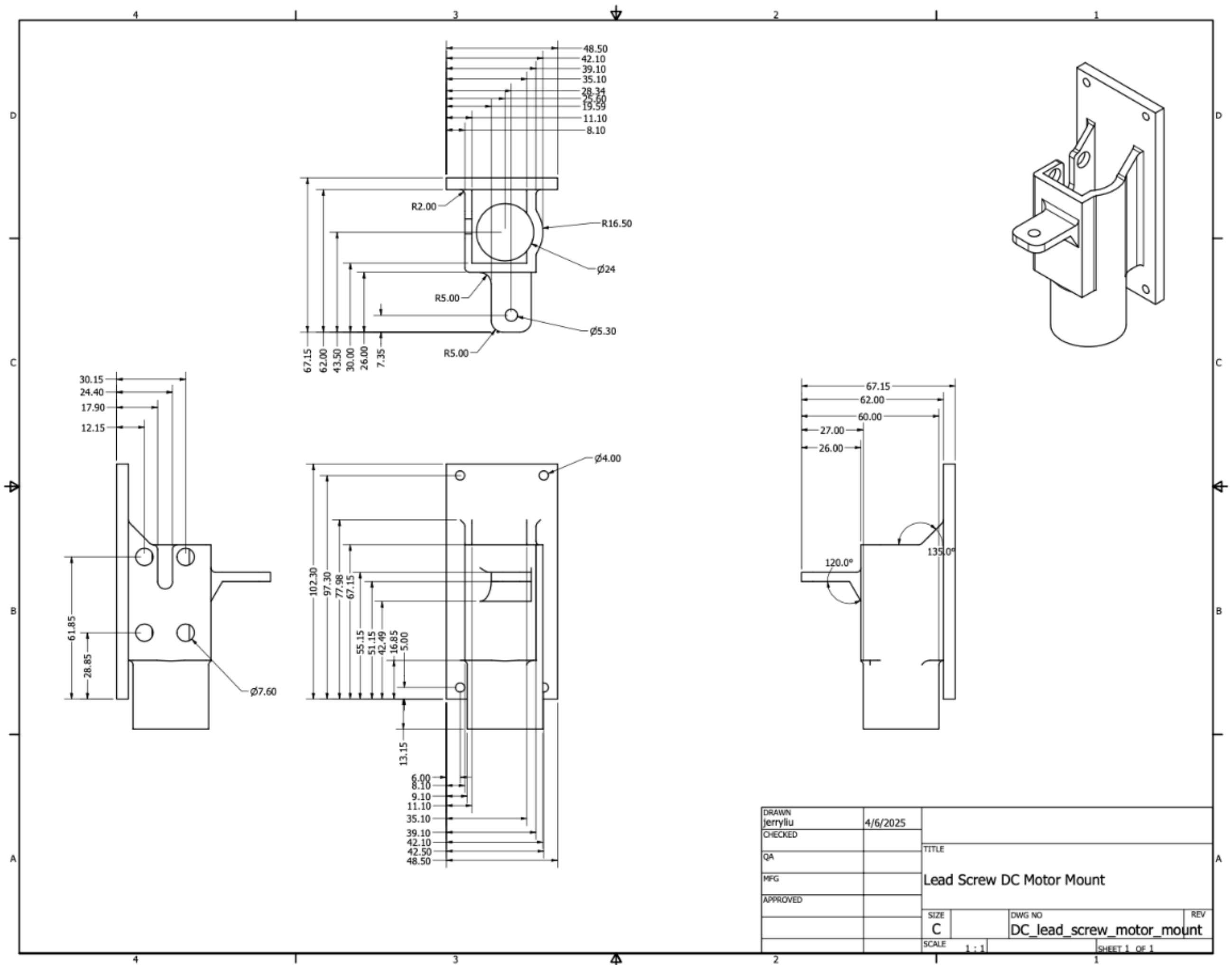


Figure 36: Lead Screw Motor Mount

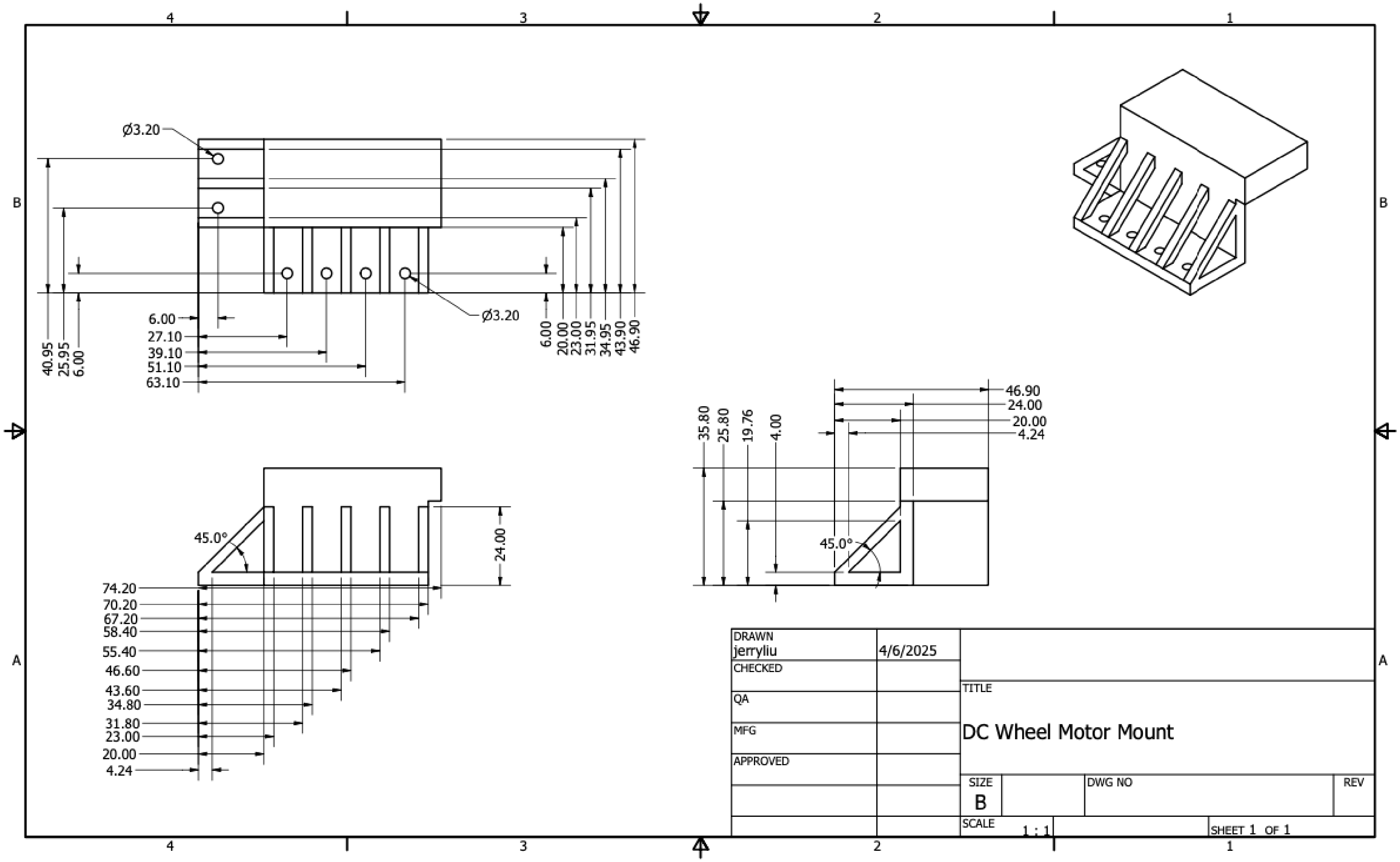


Figure 37: Wheel Motor Mount

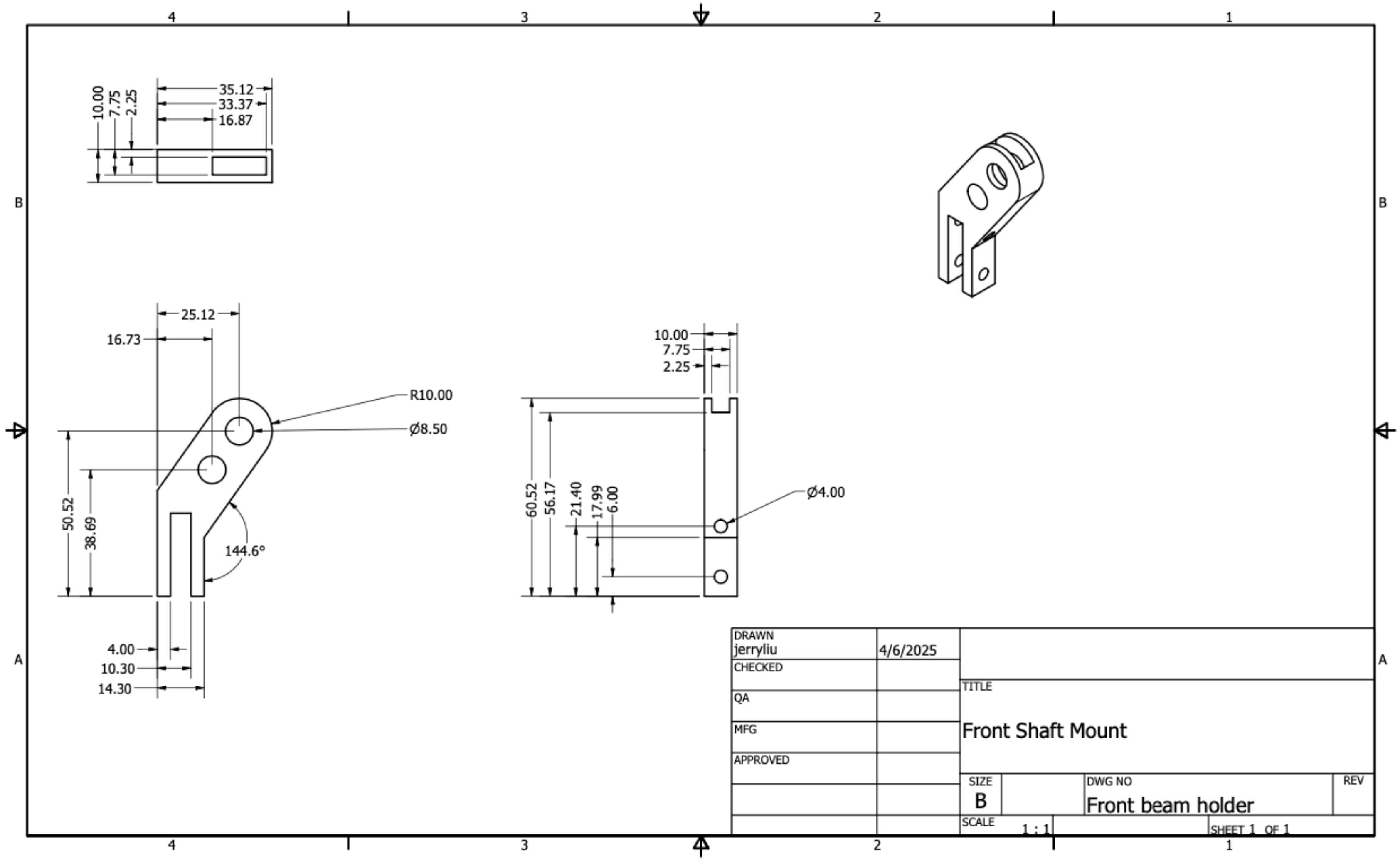


Figure 38: Front Shaft Mount

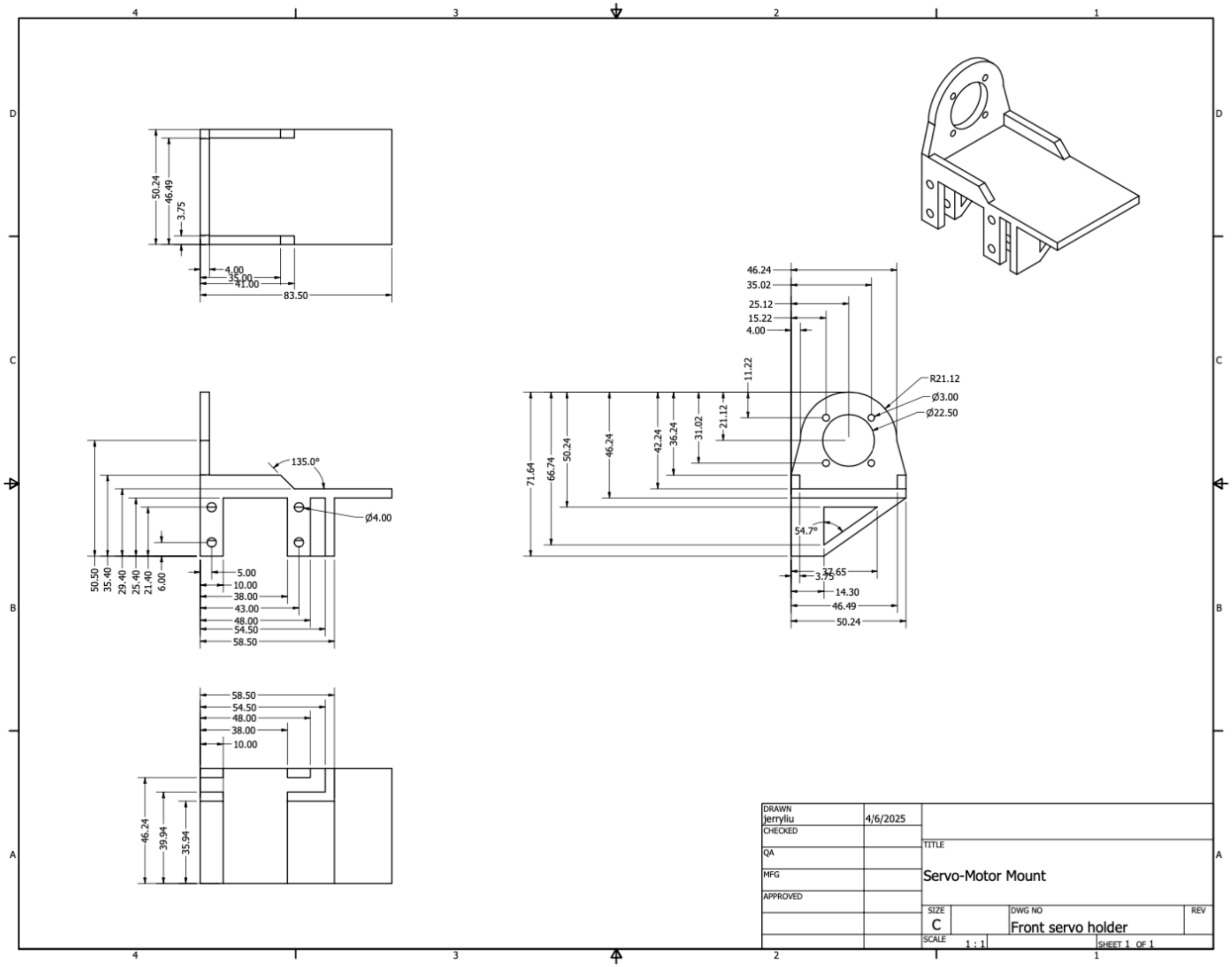


Figure 39: Servo Motor Mount

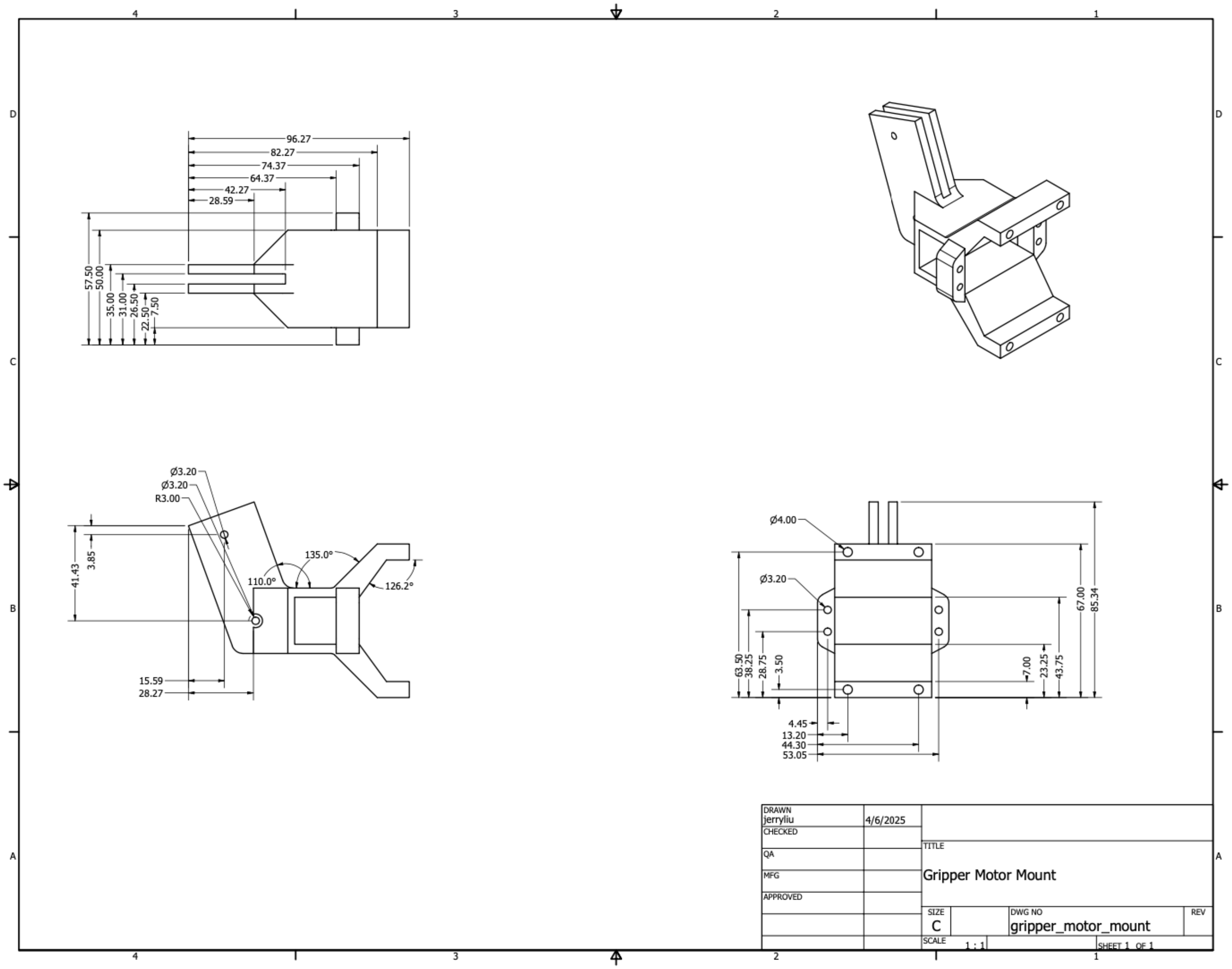


Figure 40: Gripper Motor Mount

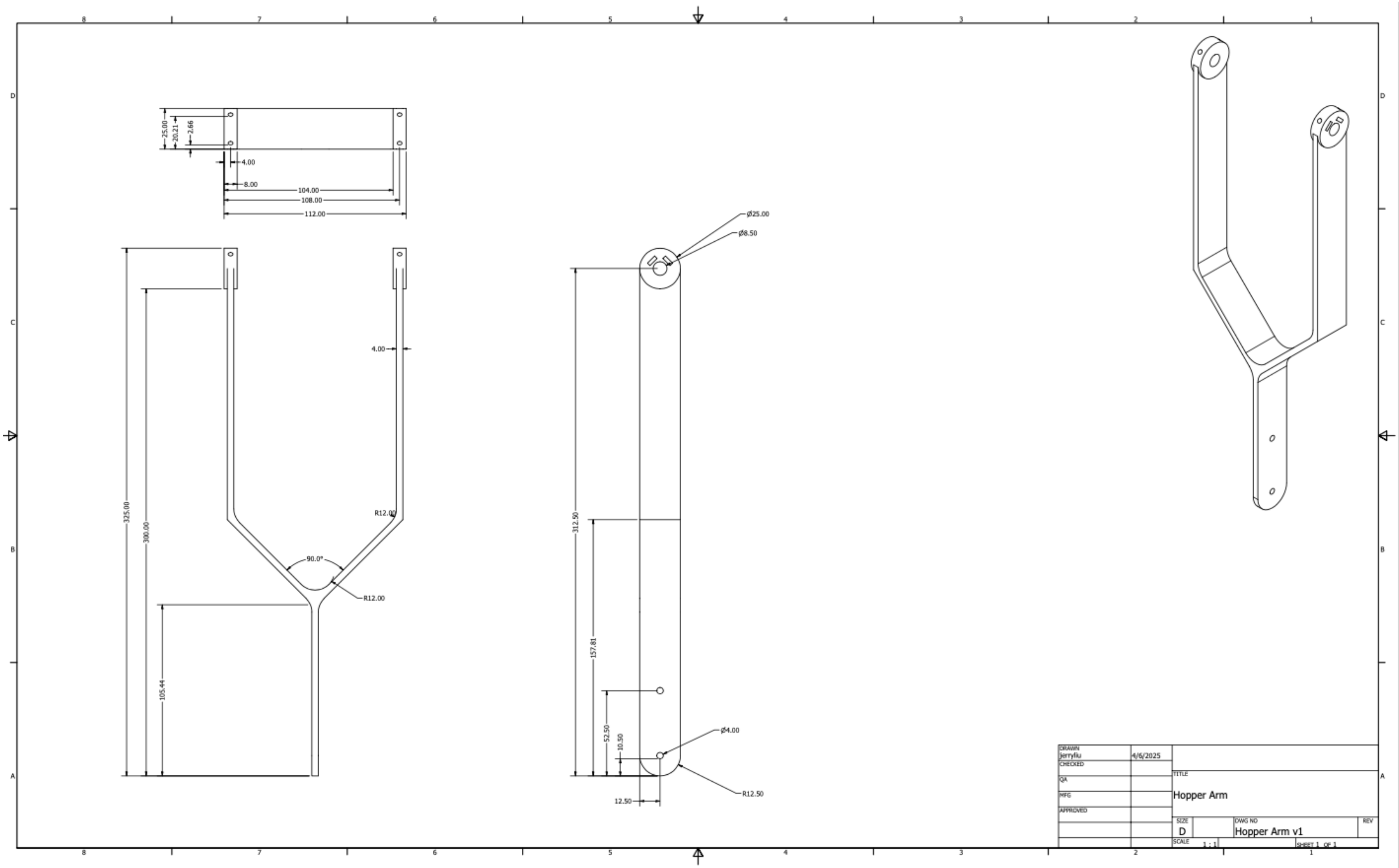


Figure 41: Hopper Arm Mount

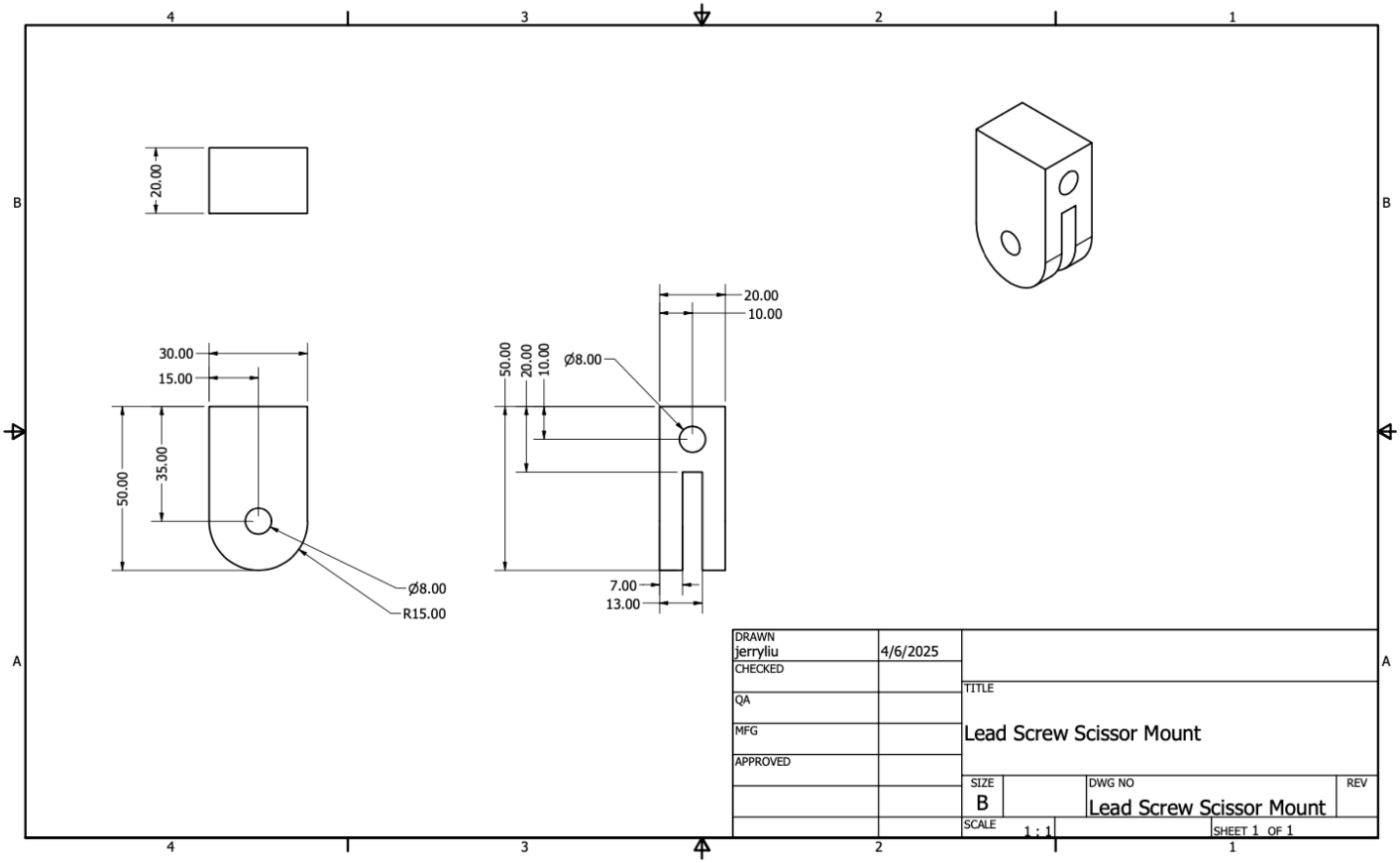


Figure 42: Lead Screw Scissor Mount

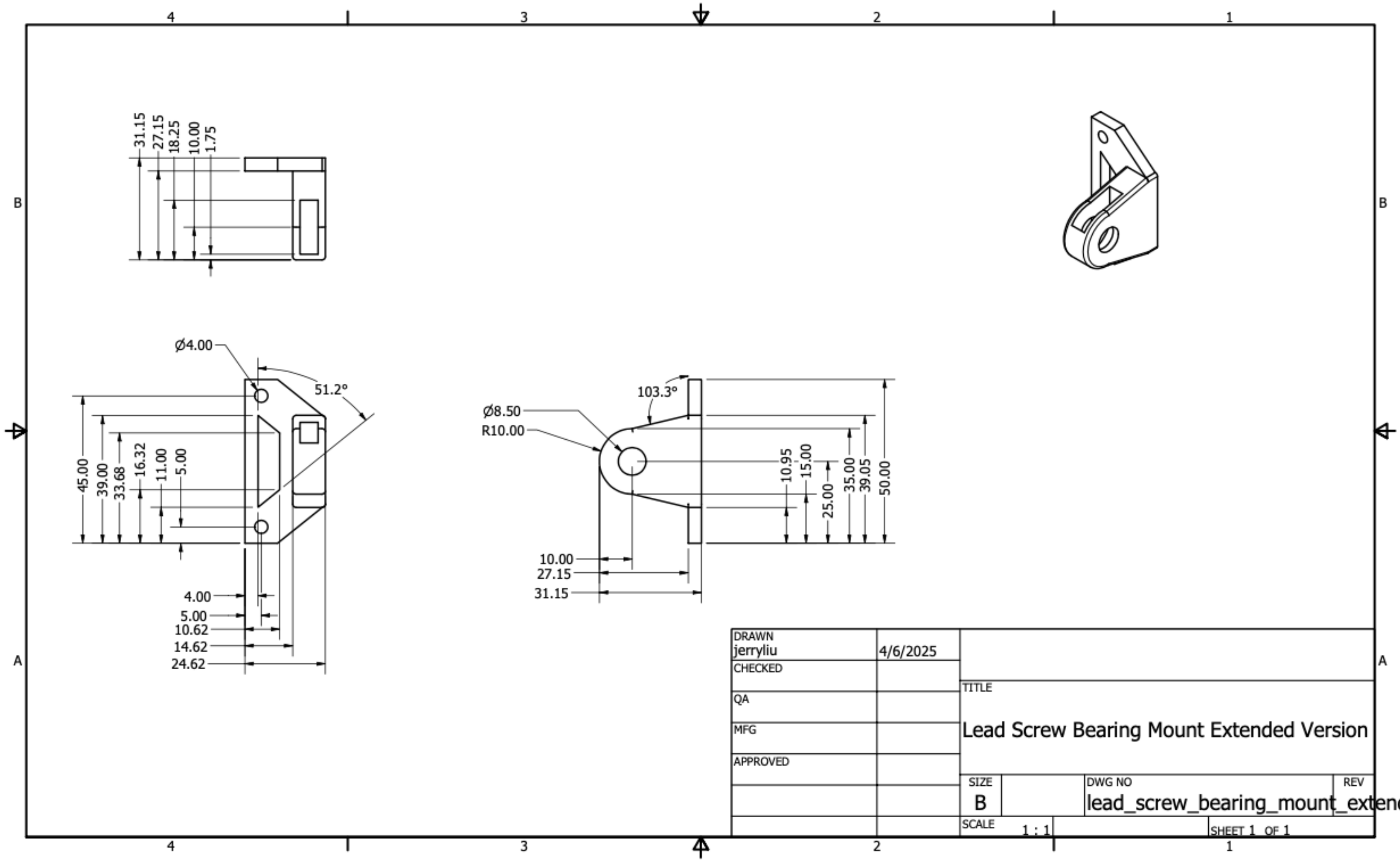


Figure 43: Lead Screw Bearing Mount (Extended Model)

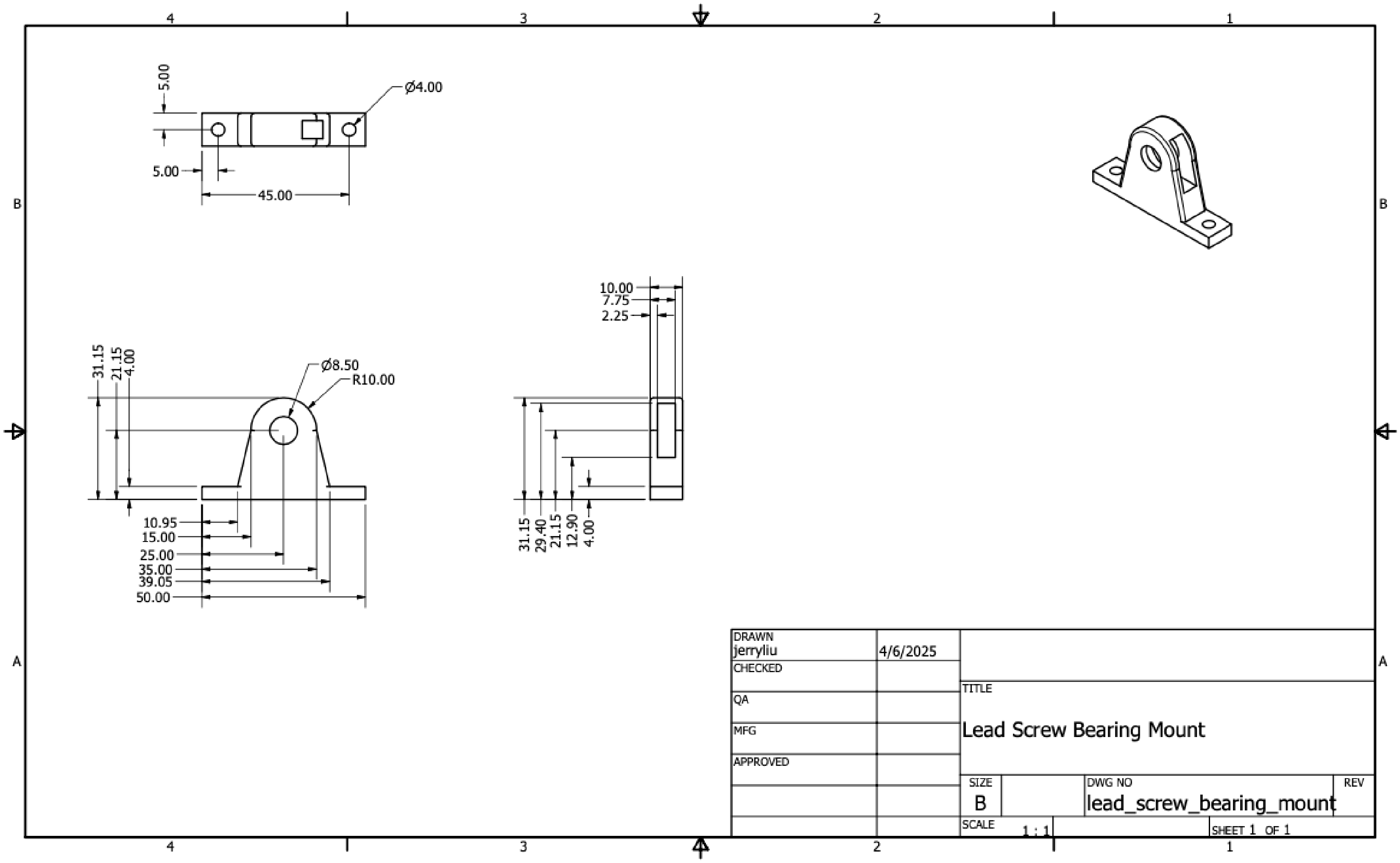


Figure 44: Lead Screw Bearing Mount

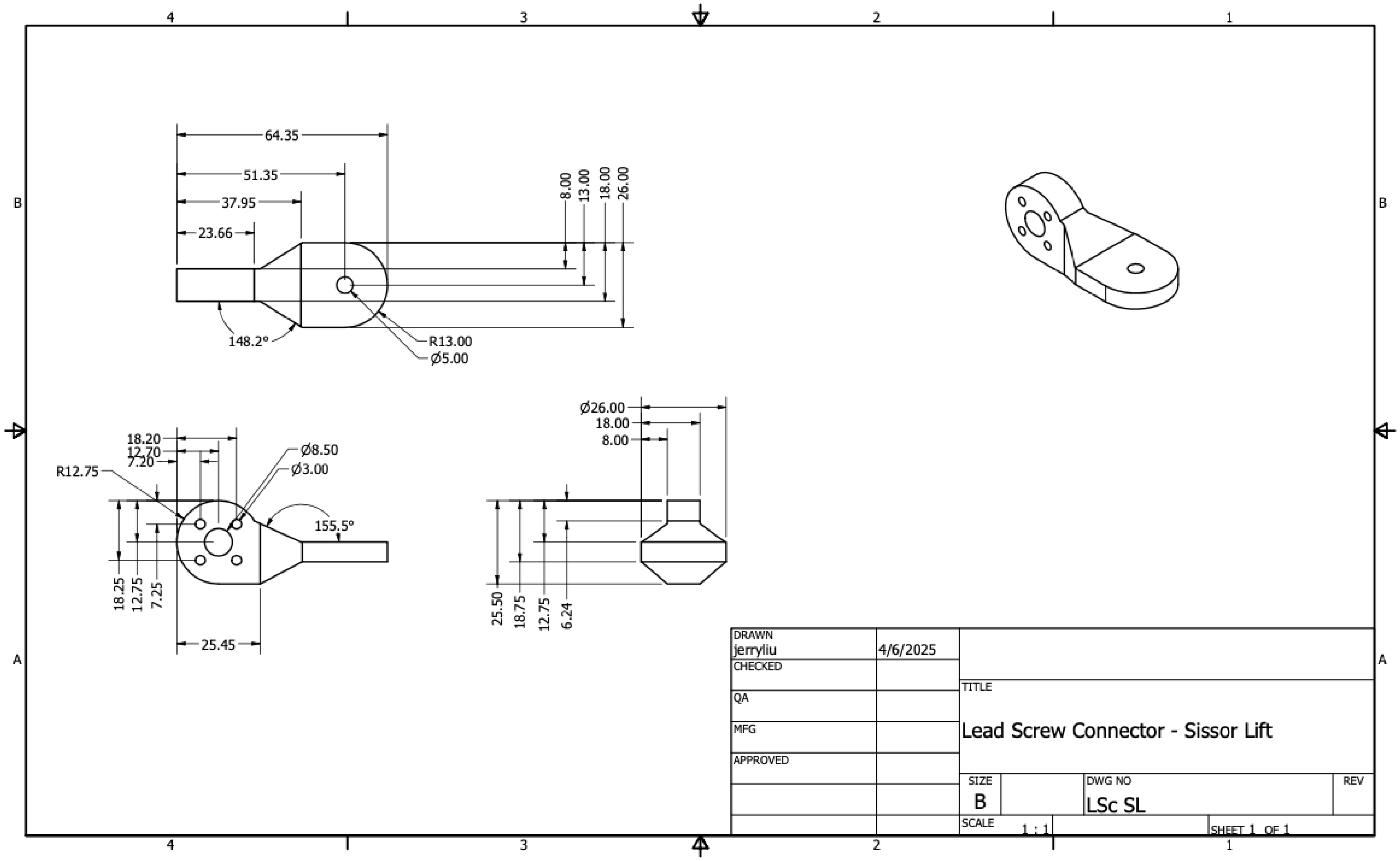


Figure 45: Lead Screw Connector – Scissor Lift

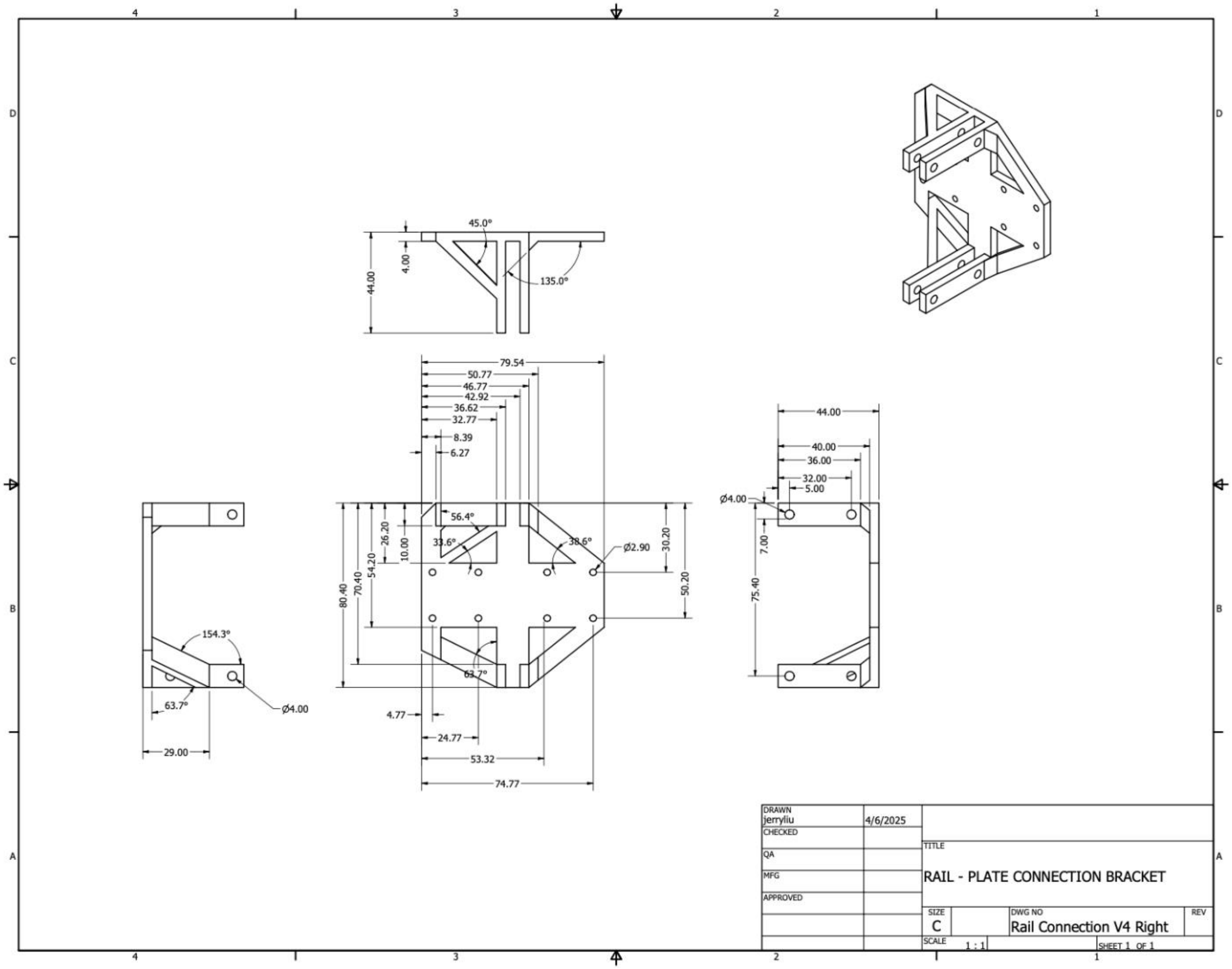


Figure 46: Rail – Plate Connection Bracket

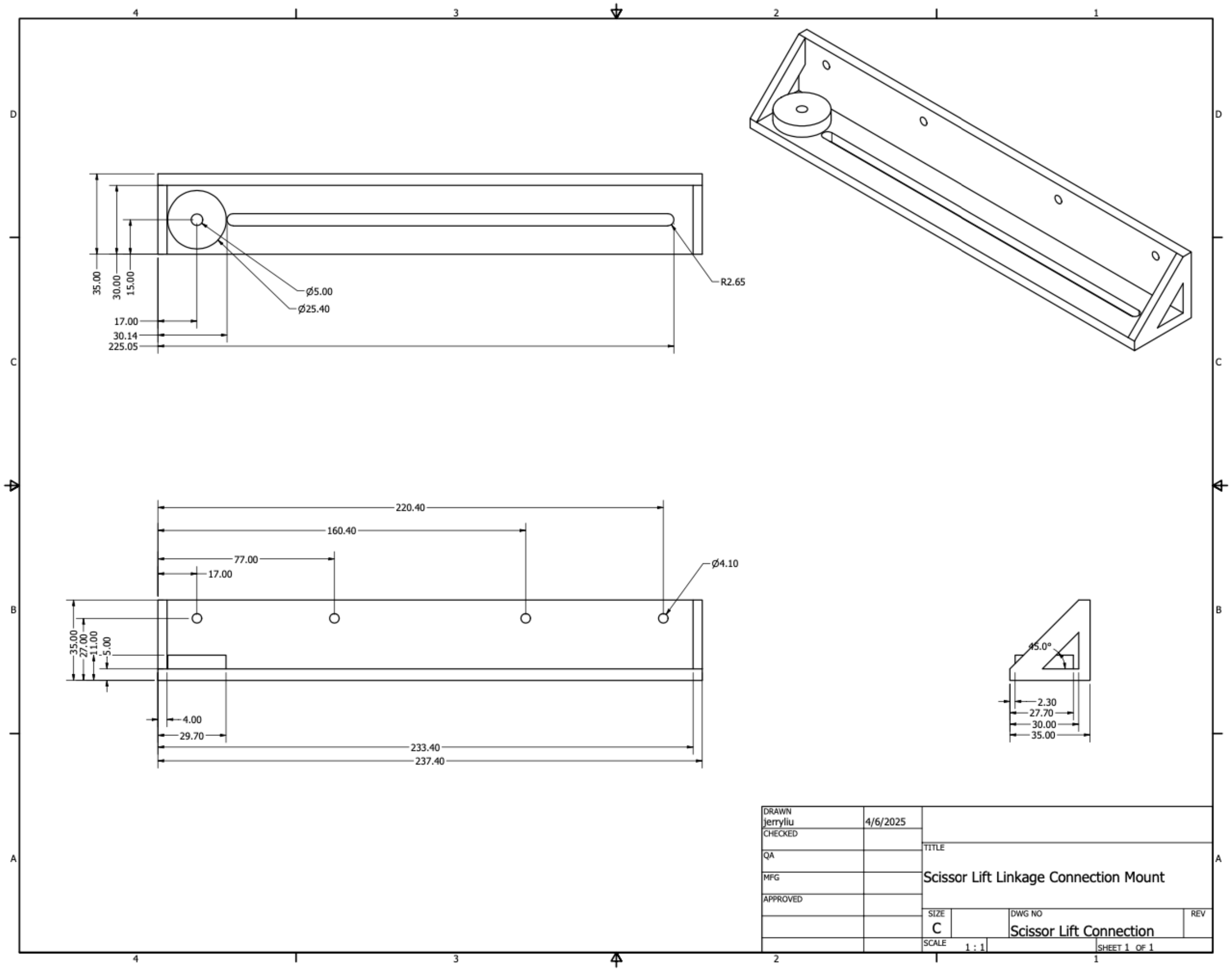


Figure 47: Scissor Lift Linkage Connection Mount

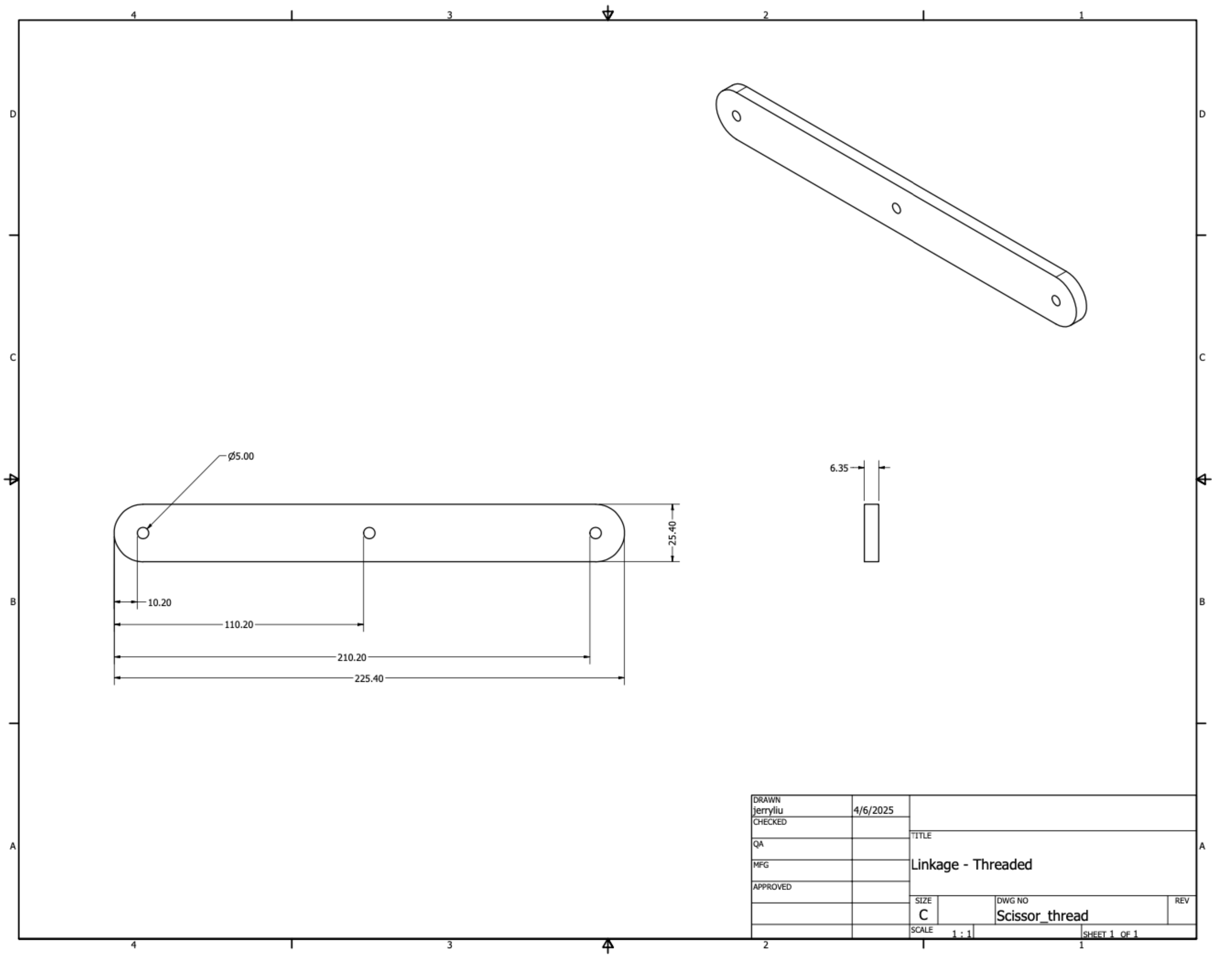


Figure 48: Linkage (Threaded Model)

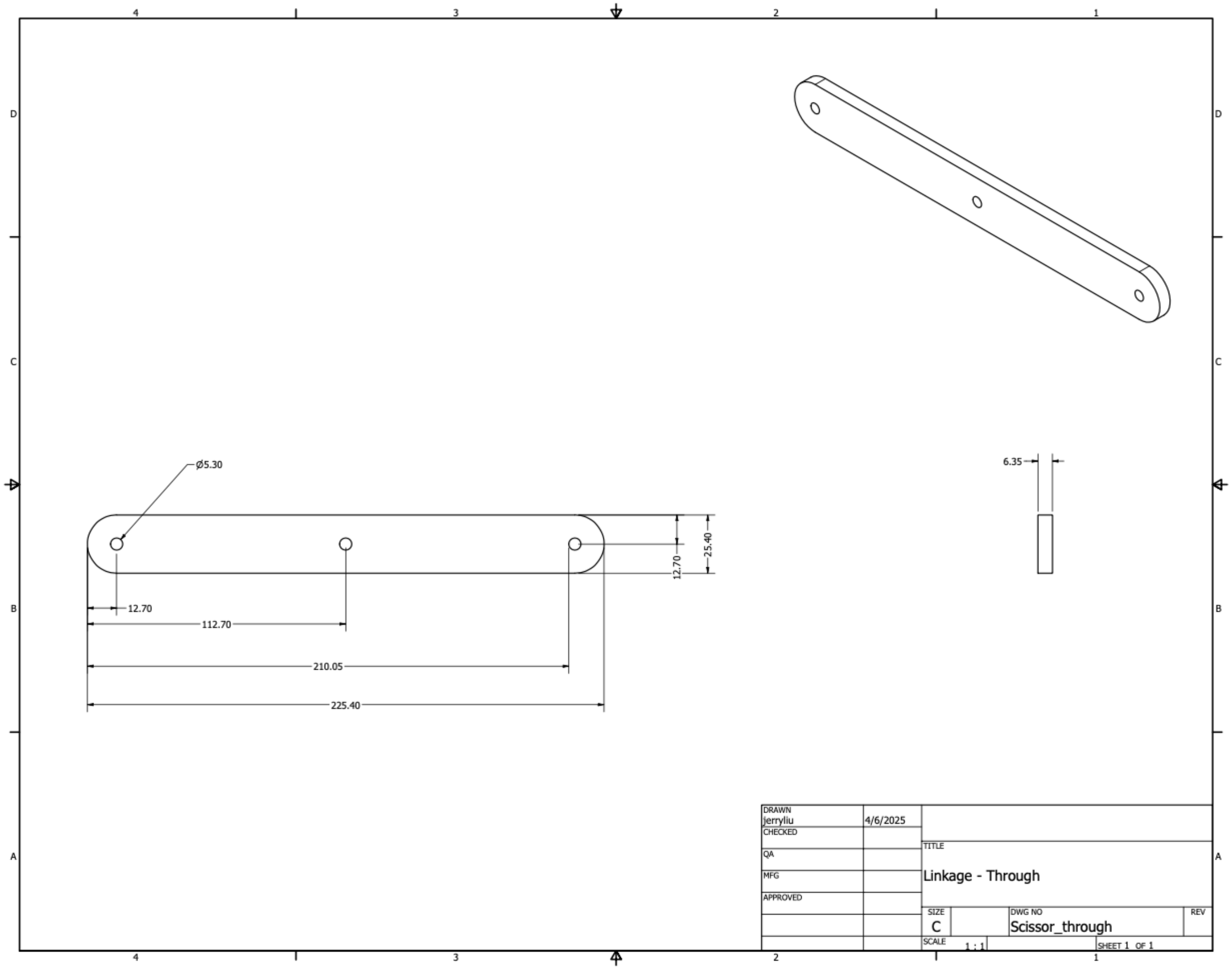


Figure 49: Linkage (Non Threaded Model)

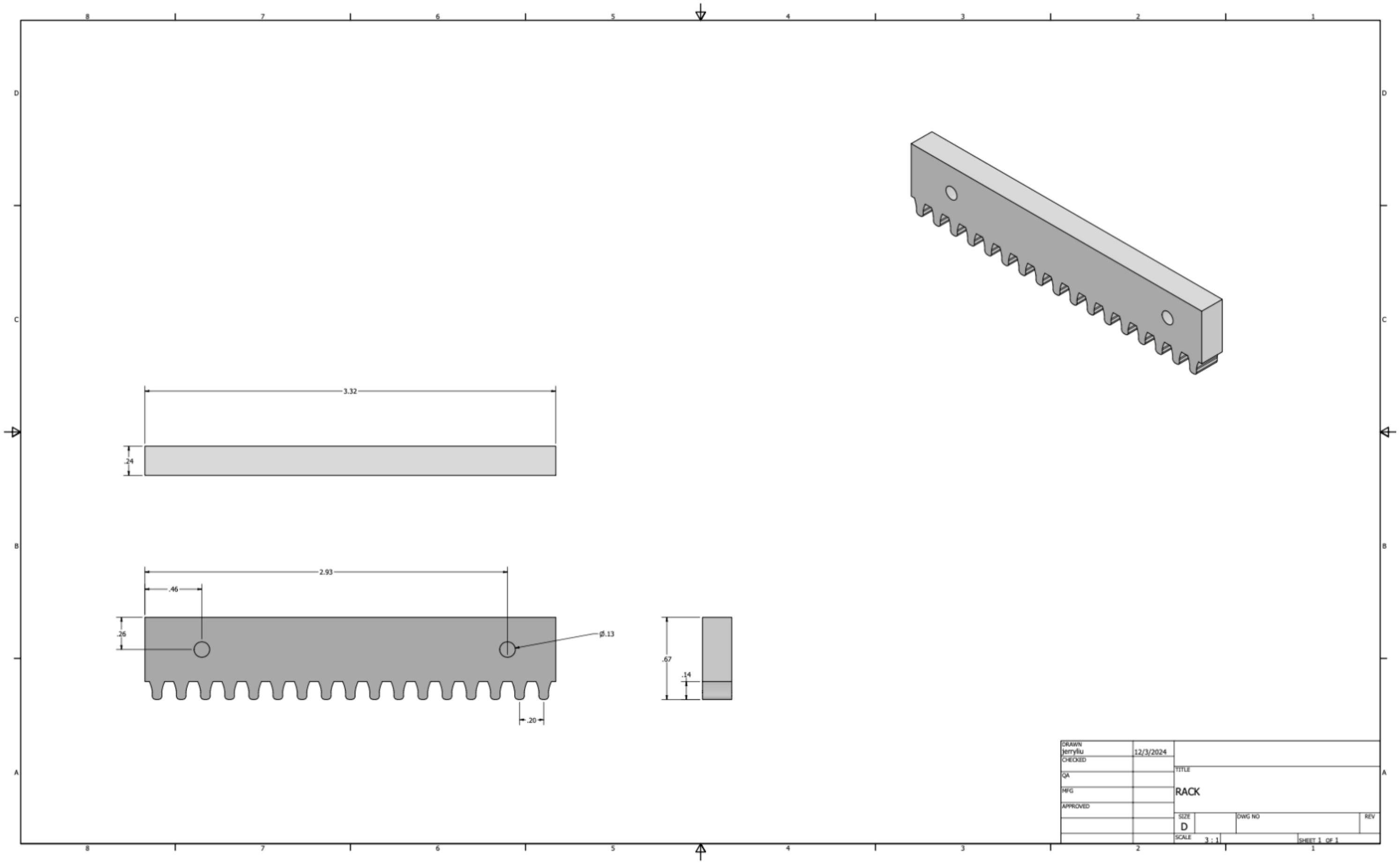


Figure 50: Gripper Rack

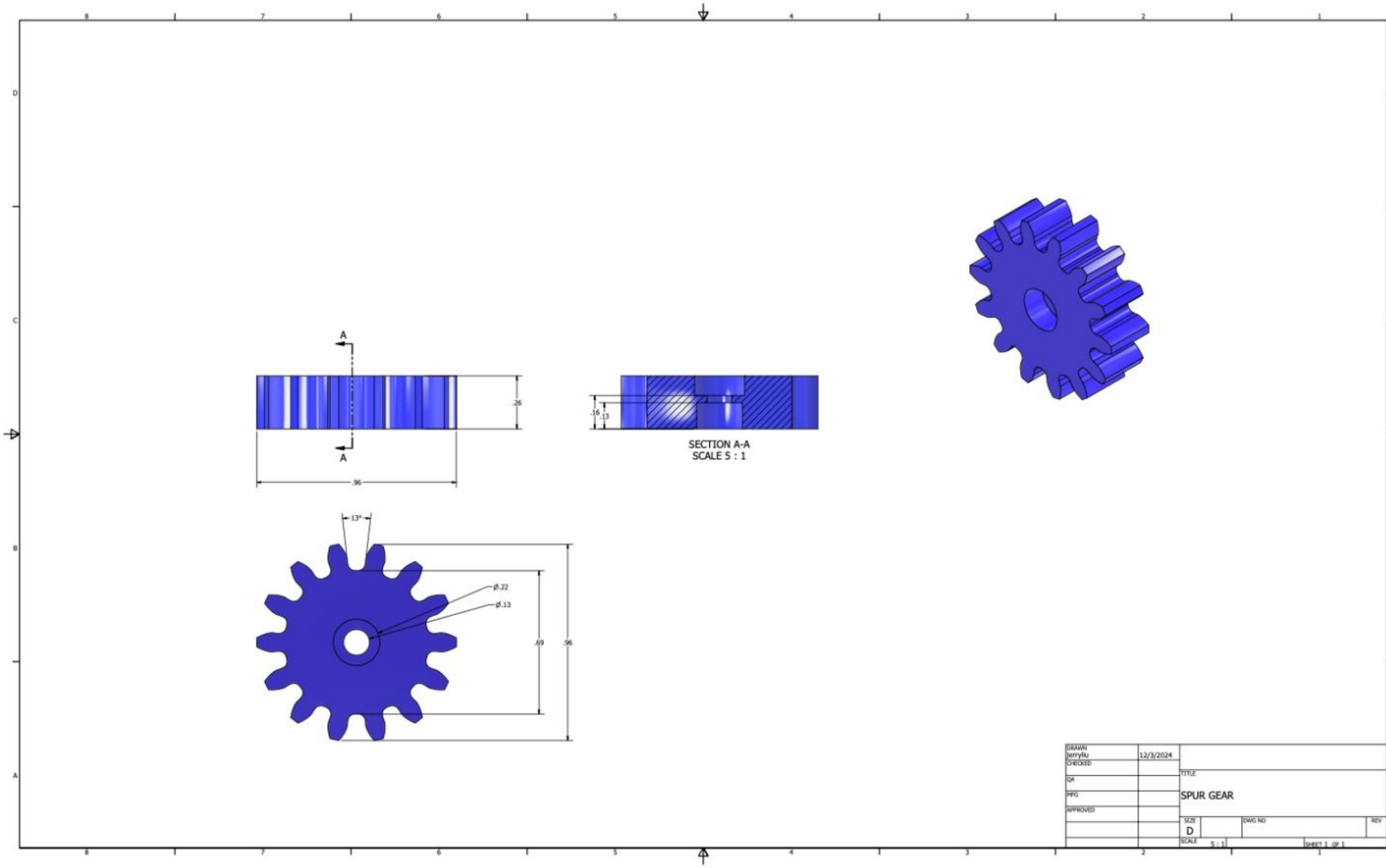


Figure 51: Gripper Spur Gear

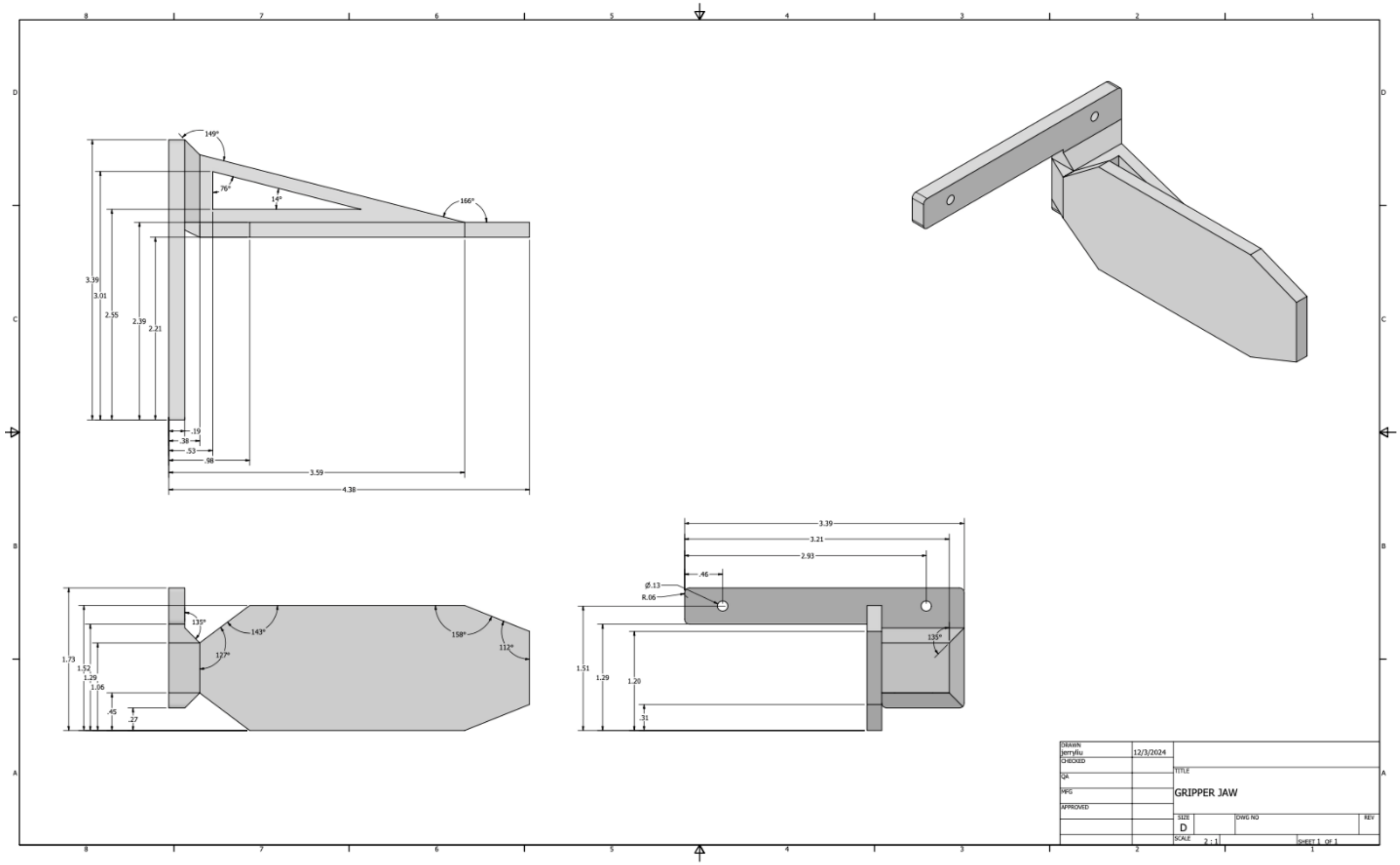


Figure 52: Gripper Jaw

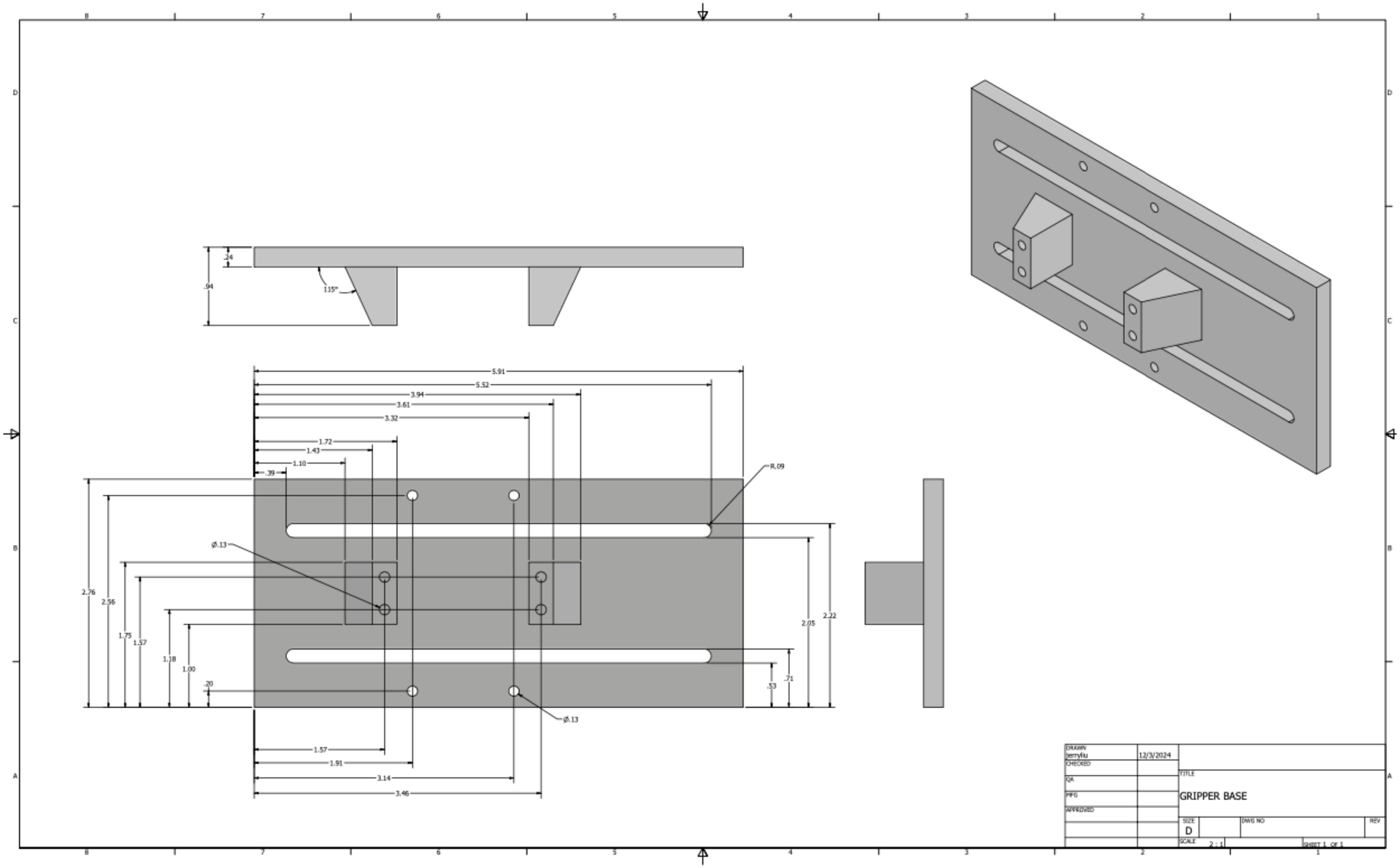


Figure 53: Gripper Base Plate

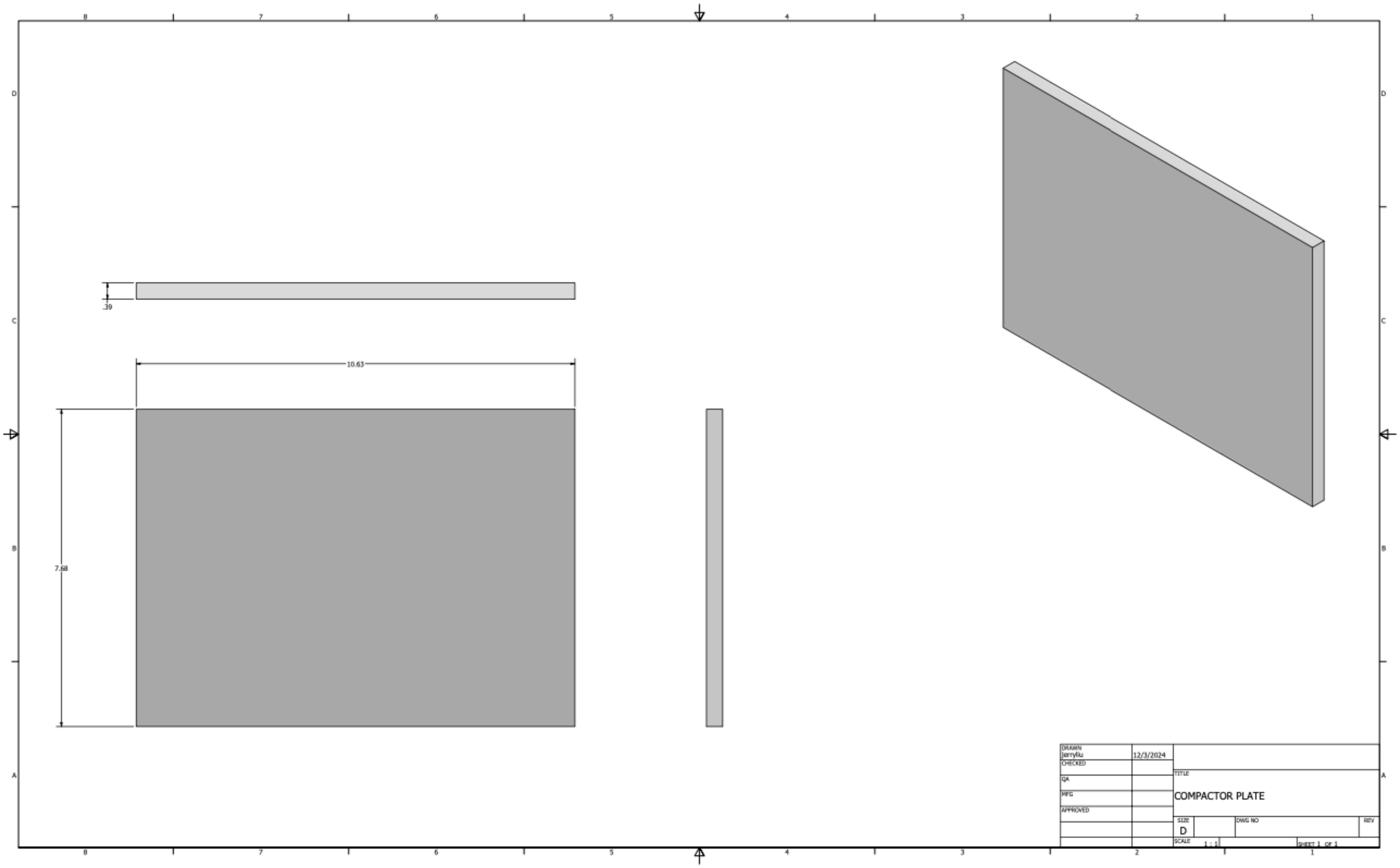


Figure 54: Compactor Plate

Appendix D – Calculations

Properties		
Gravity	9.81	m/s
PLA Density	1310	kg/m^3
MDF Density	792.9167	kg/m^3
Acrylic Density	1190.24	kg/m^3
Plywood Density	686.5079	kg/m^4

Dimensions		
Arm length	30	cm
Arm COM (from hinge)	15	cm
Wall thickness	10	mm
Plate thickness	10	mm
Plate area	0.04	m^2
Gripper gear avg diamete	0.022	m
Wheel radius	17.78	cm
Lead screw lead	0.008	m
Lead screw efficiency	0.308173	
Max compaction distance	0.2	m
Scissor arm length	0.2	m
Compaction plate accel	3.3	cm/s

Assumptions		
Max garbage weight	0.015	kg
Garbage k value	1310	N/m
Friction garbage	0.1	
Friction tires	0.36	
Acceleration of gripper		
Acceleration of full robot	3.3	cm/s

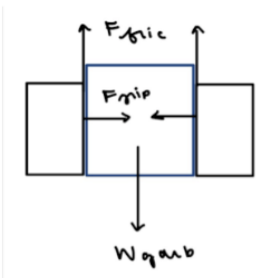
Gripper:

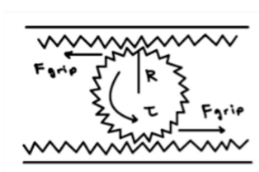
$$2F_{fric} = 2F_{grip}\mu = W_{garb}$$

$$\tau_{min} = 2F_{grip}R = \frac{W_{garb}R}{\mu}$$

For μ of 0.1 (worst-case) [2]:

Required Torque: 0.016 Nm



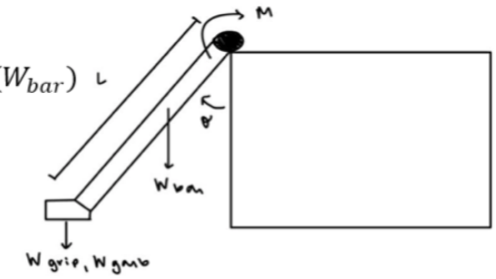


Lifting arm:

$$\Sigma M_{motor} = L\cos\theta\left(W_{grip} + W_{garb}\right) + \frac{L}{2}\cos\theta(W_{bar})$$

$$\tau_{max} = Lg(m_{grip} + m_{garb} + \frac{m_{bar}}{2})$$

Required torque: 1.23 Nm



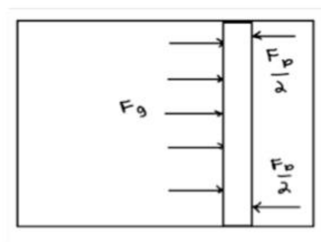
Compaction plate:

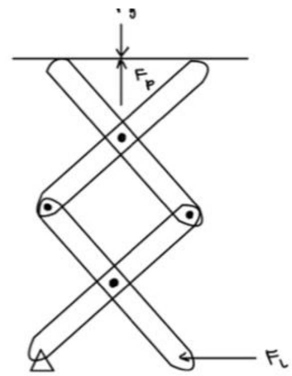
$$\Sigma F = ma = 2F_p - F_g = F_p - kx$$

$$F_p = ma + kx$$

$$F_L = \frac{2F_p}{\tan\theta}$$

$$\tau = F_L \frac{Lead}{2\pi E} = F_L \frac{Lead}{2\pi(\frac{\tan\alpha}{\tan\alpha+\mu})}$$





For a plate acceleration of 3.3 cm/s² and compression distance of 0.2 m

Required Torque: 1.25 Nm

Final Calculations:

Torque Calculations		
Torque for lifting arm	1.93	Nm
Torque to hold object	0.02	Nm
Torque to compact	1.25	Nm
Torque per wheel	2.19	Nm

Appendix E – Manufacturing

Figure 55: Manufacturing Session

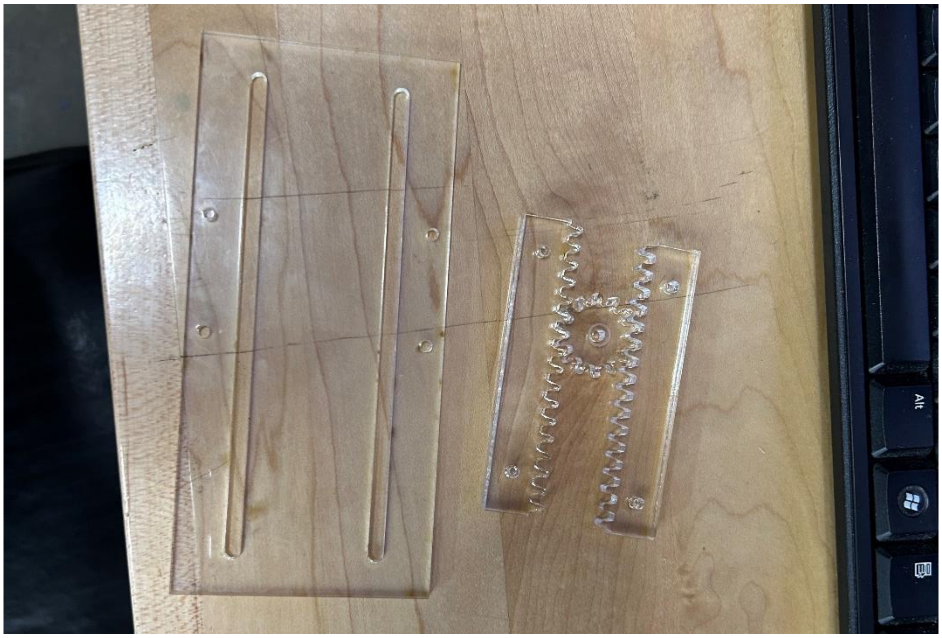


Figure 56: Laser cut components for gripper rack and pinion

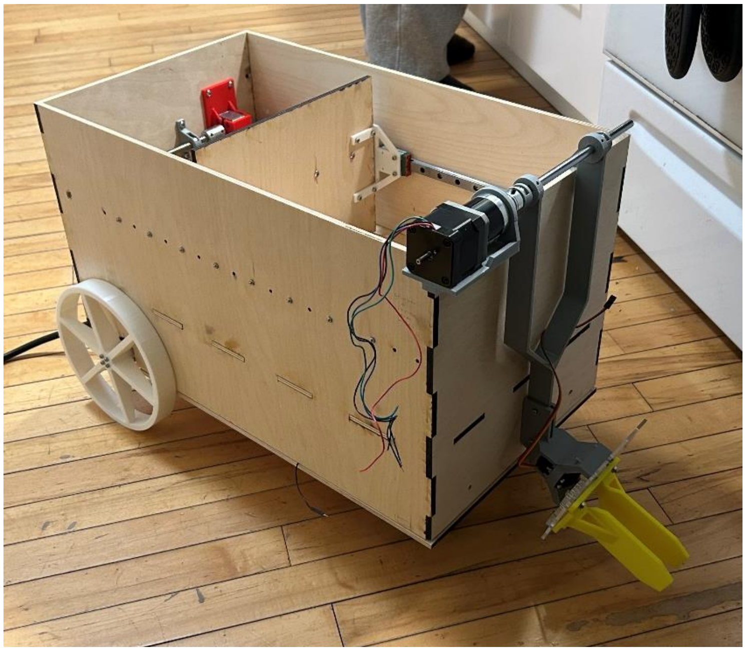


Figure 57: Fabricated Prototype before cosmetic improvements

Appendix F – Code Used for Testing

Arduino Code

```
#include <Servo.h>
#include <Stepper.h>
Servo myservo;
```

```
// RP-S40-SR Force Sensing Resistor
// Printing out resistance values from the voltage divider
int FSR_pin = A0;    // select the input pin for the potentiometer
int avg_size = 10; // number of analog readings to average
float R_0 = 510.0; // known resistor value in [Ohms]
float Vcc = 5.0; // supply voltage

int stepsPerRev = 3800;//2048;
int rpm = 10;
Stepper myStepper(stepsPerRev, 1,2,3,4);
int gpos = 0;
const int wheelL1 = 9; // IN1
const int wheelL2 = 10; // IN2
const int wheelPWML = 6; // ENA (PWM pin)

const int wheelR1 = 8; // IN1
const int wheelR2 = 13; // IN2
const int wheelPWMR = 5; // ENA (PWM pin)

const int scissor1 = 11; // IN1
const int scissor2 = 12; // IN2
const int scissorPWM = 7; // ENA (PWM pin)

void setup() {
  Serial.begin(9600);
  myStepper.setSpeed(rpm);
  // put your setup code here, to run once:
  myservo.attach(0);  // attaches the servo on pin 9 to the servo objectu
  myservo.write(0);   // rotate slowly servo to 0 degrees immediately
  delay(200);
  //myStepper.step(stepsPerRev);

  // Set Lwheel control pins as outputs
  pinMode(wheelL1, OUTPUT);
  pinMode(wheelL2, OUTPUT);
  pinMode(wheelPWML, OUTPUT);

  // Set Rwheel control pins as outputs
  pinMode(wheelR1, OUTPUT);
  pinMode(wheelR2, OUTPUT);
  pinMode(wheelPWMR, OUTPUT);

  // Set scissor control pins as outputs
  pinMode(scissor1, OUTPUT);
  pinMode(scissor2, OUTPUT);
  pinMode(scissorPWM, OUTPUT);
}

void scissor() {
  // Spin motor in one direction
  digitalWrite(scissor1, LOW);
  digitalWrite(scissor2, HIGH);
  analogWrite(scissorPWM, 255); // Set speed (0-255)
  delay(5000); // Run for 2 seconds
```

```
  /*// Spin motor in the opposite direction
  digitalWrite(scissor1, HIGH);
  digitalWrite(scissor2, LOW);
  analogWrite(scissorPWM, 255); // Set speed (0-255)
  delay(20000); // Run for 2 seconds*/
}

void Lwheel() {
  // Spin motor in one direction
  digitalWrite(wheelL1, HIGH);
  digitalWrite(wheelL2, LOW);
  analogWrite(wheelPWML, 200); // Set speed (0-255)
  delay(2000); // Run for 2 seconds

  // Spin motor in the opposite direction
  digitalWrite(wheelL1, LOW);
  digitalWrite(wheelL2, HIGH);
  analogWrite(wheelPWML, 255); // Set speed (0-255)
  delay(2000); // Run for 2 seconds
}

void Rwheel() {
  // Spin motor in one direction
  digitalWrite(wheelR1, HIGH);
  digitalWrite(wheelR2, LOW);
  analogWrite(wheelPWMR, 200); // Set speed (0-255)
  delay(2000); // Run for 2 seconds

  // Spin motor in the opposite direction
  digitalWrite(wheelR1, LOW);
  digitalWrite(wheelR2, HIGH);
  analogWrite(wheelPWMR, 255); // Set speed (0-255)
  delay(2000); // Run for 2 seconds
}

void drive() {
  /* Spin motor in one direction
  digitalWrite(wheelL1, HIGH);
  digitalWrite(wheelL2, LOW);
  analogWrite(wheelPWML, 200); // Set speed (0-255)
  digitalWrite(wheelR1, HIGH);
  digitalWrite(wheelR2, LOW);
  analogWrite(wheelPWMR, 200); // Set speed (0-255)
  delay(2000); // Run for 2 seconds*/



  // Spin motor in one direction
  digitalWrite(wheelL1, HIGH);
  digitalWrite(wheelL2, LOW);
  analogWrite(wheelPWML, 200); // Set speed (0-255)
  digitalWrite(wheelR1, LOW);
  digitalWrite(wheelR2, HIGH);
  analogWrite(wheelPWMR, 200); // Set speed (0-255)
  delay(5000); // Run for 2 seconds

  // Spin motor in one direction
  digitalWrite(wheelL1, LOW);
  digitalWrite(wheelL2, HIGH);
```

```
  analogWrite(wheelPWML, 200); // Set speed (0-255)
  digitalWrite(wheelR1, HIGH);
  digitalWrite(wheelR2, LOW);
  analogWrite(wheelPWMR, 200); // Set speed (0-255)
  delay(5000); // Run for 2 seconds
}

float pressure() {
  float sum_val = 0.0; // variable for storing sum used for averaging
  float R_FSR;
  for (int ii=0;ii<avg_size;ii++){
    sum_val+=(analogRead(FSR_pin)/1023.0)*5.0; // sum the 10-bit ADC ratio
    delay(10);
  }
  sum_val/=avg_size; // take average
  R_FSR = (R_0/1000.0)*((Vcc/sum_val)-1.0); // calculate actual FSR resistance
  return R_FSR;
  Serial.println(R_FSR); // print to serial port
  delay(10);
}

void current() {
  int adc = analogRead(A1);
  //float currsens = adc/1024*100;
  Serial.println(adc);
}

void loop() {
//float R_FSR = pressure();
  //current();
  pressure();
  //scissor();
  //scissor();
/*
  myservo.write(160);   // rotate slowly servo to 0 degrees immediately
  delay(5000);
  myStepper.step(stepsPerRev);
  delay(5000);
  myservo.write(0);   // rotate slowly servo to 0 degrees immediately
  myStepper.step(0);
  delay(5000);
  myStepper.step(-stepsPerRev);
  delay(5000);
  // // Spin motor in the opposite direction
  digitalWrite(scissor1, LOW);
  digitalWrite(scissor2, HIGH);
  analogWrite(scissorPWM, 255); // Set speed (0-255)
  delay(10000); //  Run for 2 seconds// Run for 2 seconds
  */
/*
  //myStepper.step(stepsPerRev);
const int inc = 30;

  //myStepper.step(-stepsPerRev);
  //delay(500);
  while (R_FSR > 500) {
    gpos += inc;
    myservo.write(gpos);
  }*/
  //drive();
  /*
```

```
  // // Spin motor in the opposite direction
  digitalWrite(scissor1, HIGH);
  digitalWrite(scissor2, LOW);
  analogWrite(scissorPWM, 255); // Set speed (0-255)
  delay(8000); //  Run for 2 seconds// Run for 2 seconds
  //myservo.write(0);   // rotate slowly servo to 0 degrees immediately
  */

  //delay(500);
  // put your main code here, to run repeatedly:
  //myservo.write(0);
  //delay(5);
  //myservo.write(90);
  //delay(5);
  //digitalWrite(motor1pin1, HIGH);
  //digitalWrite(motor1pin2, LOW);
  //delay(1000);

  //digitalWrite(mo)
  // for (int angle = 0; angle <= 180; angle += 10) {  // rotate slowly from 0 degrees to 180 degrees, one by one degree
  //   // in steps of 1 degree
  //   myservo.write(angle);  // control servo to go to position in variable 'angle'
  //   delay(10);         // waits 10ms for the servo to reach the position
  // }

  // for (int angle = 180; angle >= 0; angle -= 10) {  // rotate from 180 degrees to 0 degrees, one by one degree
  //   myservo.write(angle);                        // control servo to go to position in variable 'angle'
  //   delay(10);                                   // waits 10ms for the servo to reach the position
  // }
}
```

Object Detection Code

```
import cv2
import numpy as np
from ultralytics import YOLO
import serial
import time



# === SET PARAMETERS ===
MODEL_PATH = "yolo11n_ncnn_model"
KNOWN_WIDTH = 6.0  # cm (actual width of object)
FOCAL_LENGTH = 920  # Replace with calibrated focal length
DISTANCE_THRESHOLD = 20.0  # cm



# Initialize serial communication with Arduino
arduino = serial.Serial('/dev/ttyACM0', 9600, timeout=1)
time.sleep(2)
```

```
# Load YOLO model
model = YOLO(MODEL_PATH)


# Open camera
cap = cv2.VideoCapture(0)
cap.set(3, 1280)
cap.set(4, 720)


# Function to calculate distance
def calculate_distance(known_width, focal_length, width_in_frame):
    if width_in_frame > 0:
        distance = (known_width * focal_length) / width_in_frame
        return distance
    return None


# Function to send motor command to Arduino
def send_command_to_arduino(command):
    arduino.write(command.encode())
    time.sleep(0.05)


# Function to adjust motor speed based on distance
def adjust_motor_speed(distance):
    if distance > 50:
        speed = 255  # Full speed
    elif distance > 30:
        speed = 150  # Medium speed
    else:
        speed = 100  # Slow speed

    return speed


while True:
    ret, frame = cap.read()
    if not ret:
```

```
        break

    # Run YOLO object detection
    results = model(frame)



    object_distance = None
    object_name = None

    for result in results:
        for box in result.boxes:
            x1, y1, x2, y2 = box.xyxy[0]
            label = result.names[int(box.cls[0])]
            object_width = x2 - x1



            # Calculate distance
            object_distance = calculate_distance(KNOWN_WIDTH, FOCAL_LENGTH,
object_width)
            object_name = label



            if object_distance:
                # Draw bounding box and display distance + object name
                cv2.rectangle(frame, (int(x1), int(y1)), (int(x2), int(y2)), (0, 255, 0), 2)
                cv2.putText(frame, f"{label} {object_distance:.2f} cm", (int(x1), int(y1) - 10),
                            cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 0), 1)



    # === DC Motor Control ===
    if object_distance is not None:
        if object_distance > DISTANCE_THRESHOLD:
            speed = adjust_motor_speed(object_distance)
            send_command_to_arduino('F')  # Move forward
            print(f"Moving towards {object_name} at {speed} speed ({object_distance:.2f} cm
away)")
        else:
            send_command_to_arduino('S')  # Stop when close
            print(f"Reached {object_name} — {object_distance:.2f} cm away")
```

```
    # === Servo Motor Control ===
    if object_distance is not None:
        if object_distance > 30:
            send_command_to_arduino('O')  # Open servo
        else:
            send_command_to_arduino('C')  # Close servo



    # Display the frame
    cv2.imshow('Object Detection and Motor Control', frame)

    # Stop on 'q' key
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break



# Cleanup
send_command_to_arduino('S')  # Ensure motor stops
send_command_to_arduino('C')  # Ensure servo resets
arduino.close()
cap.release()
cv2.destroyAllWindows()
```