TORONTO METROPOLITAN UNIVERSITY
FACULTY OF ENGINEERING, ARCHITECTURE AND SCIENCE

DEPARTMENT OF MECHANICAL AND INDUSTRIAL ENGINEERING

AUTONOMOUS TRUCK NAVIGATION WITH TRAILER INTEGRATION VIA NATURAL LANGUAGE PROCESSING

Junseo(Jason) Kim, 500872090

Rishabh Raja, 500871672

Ateeb Jawaid, 500888305

Raymond Ha, 500971317

MEC825 - Design Project Report

Submitted in partial fulfillment
Of the requirements for the degree of
Bachelor of Engineering (BEng)

Faculty Advisor: Dr. Sajad Saeedi

Date: April 5, 2024

ACKNOWLEDGEMENTS

We would like to thank our supervisor, Dr. Sajad Saeedi, for his guidance, encouragement, and support. It has been a privilege to work with such a knowledgeable supervisor. We would also like to thank Mr. Matthew Lisondra for the feedback.

ABSTRACT

This report presents an innovative approach to enhancing the autonomous navigation of truck and trailer systems through the application of Natural Language Processing (NLP) and advanced sensor fusion technologies. This study focuses on leveraging NLP for the intuitive interpretation of user commands for destination settings, significantly simplifying the interaction with autonomous vehicles.

At the heart of the system is a sophisticated integration of sensor technologies, including LIDAR and Inertial Measurement Units (IMU), coupled with an Ackerman steering mechanism. This combination not only facilitates precise maneuverability but also ensures the system's adaptability to diverse environmental conditions. The structural integrity and reliability of the design are validated through comprehensive Finite Element Analysis (FEA), showcasing the project's commitment to durability and efficient operation.

Addressing both the mechanical and computational aspects of autonomous system development, this project adopts a holistic approach. It recognizes the imperative of synergizing hardware robustness with software intelligence to achieve a seamless autonomous driving experience. The study outlines a multidisciplinary methodology, incorporating mechanical design, sensor technology, and software development expertise, with a particular focus on NLP's role in enhancing system interaction and navigational decision-making.

Key deliverables of this report include an in-depth analysis of the design rationale, development methodologies, and the strategic integration of NLP and sensor technologies to facilitate autonomous navigation. While the immediate application of NLP marks a significant step towards user-friendly autonomous transportation, the exploration of LLMs for future enhancements is identified as a vital area for continued research, promising further advancements in the system's environmental perception and operational adaptability.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

NOMENCLATURE AND ABBREVIATION

FEA	Finite Element Analysis
FMEA	Failure Mode and Effects Analysis
GPT	Generative Pre-trained Transformer
JSON	JavaScript Object Notation
IMU	Inertial Measurement Unit
LLM	Large Language Model
NLP	Natural Language Processing
ROS	Robot Operating System
SLAM	Simultaneous Localization and Mapping
VLM	Vision Language Models
LIDAR	Light Detection and Ranging

CHAPTER 1 - INTRODUCTION

1.1 INTRODUCTION

In the rapidly evolving field of autonomous transportation, innovative approaches to enhance navigational intelligence and environmental perception are continuously sought. This project introduces a novel paradigm in autonomous truck and trailer systems by employing Natural Language Processing (NLP) for command interpretation and destination setting. The fusion of NLP with sensor technology on an Ackerman steering-based robotic platform aims to redefine standards of autonomous maneuverability and decision-making capabilities.

The concept of autonomous driving, traditionally targeting a spectrum from basic assistance systems to full autonomy, has relied heavily on predefined algorithms and sensor data, which may not capture the dynamic complexities of real-world environments. The introduction of NLP as the cognitive engine for autonomous systems shows a significant advancement, enabling the direct translation of user commands into actionable navigational decisions, thus ensuring adaptable and intelligent operation across various scenarios.

A robustly designed trailer system, directed by an Ackerman steering mechanism and supported by state-of-the-art sensor fusion, including LIDAR and Inertial Measurement Units (IMU), provides the foundational structure. This setup is not only made for precise control and integrity but also undergoes thorough Finite Element Analysis (FEA) to confirm its mechanical robustness and reliability under different operational stresses, ensuring durability and optimizing design for efficient autonomous functionality.

This approach extends beyond academic exploration, addressing the critical need for efficient, reliable, and intelligent transportation solutions, particularly in logistics and urban development. The integration of mechanical precision, advanced sensor technology, and NLP places this autonomous truck and trailer system at the forefront of pioneering autonomous transportation’s future.

This project discusses a wide range of challenges, from the mechanical design intricacies of the trailer and its steering mechanism to the computational rigours of sensor fusion and NLP implementation. Adopting a holistic approach, it recognizes the essential synergy between mechanical components and software in creating an autonomous intelligent driving system.

A multidisciplinary methodology under the project combining expertise in mechanical design, sensor technology, software development, and NLP ensures comprehensive optimization of each system aspect. The deliverables of this final report include a detailed exposition of the design decisions, underlying principles, and methodologies employed to realize the vision of an advanced autonomous truck and trailer system. While the current focus centers on NLP for immediate enhancements in autonomous navigation, exploring LLMs remains a promising direction for future work, potentially offering significant advances in the system's environmental understanding and adaptability.

1.2 OBJECTIVES

This project sets several vital objectives to advance the field of autonomous vehicle technology, specifically targeting an integrated truck and trailer system:

1. To Develop an Intuitive Command Interface: Implementing NLP to allow users to communicate navigation commands in natural language, simplifying the interaction and making the technology more accessible.
2. To Achieve High Precision in Environmental Perception: Utilizing a combination of LIDAR, IMU, and rotary encoders for comprehensive sensor fusion, aiming for high accuracy in mapping and real-time positioning.
3. To Ensure Mechanical Reliability: Conducting Finite Element Analysis on the truck and trailer system to verify structural integrity and durability under a variety of stress conditions and operational scenarios.
4. To Optimize Navigation and Obstacle Avoidance: Integrating sensor data with NLP inputs to facilitate dynamic path planning and obstacle detection, ensuring safe and efficient operation in diverse environments.

1.3 LITERATURE REVIEW AND RELATED WORKS

1.3.1 INTRODUCTION TO AUTONOMOUS DRIVING SYSTEMS

The concept of autonomous driving systems has captivated engineers, researchers, and the general public for a future where vehicles navigate the roads with little to no human intervention. The Society of Automotive Engineers (SAE) defines autonomous driving through

levels 0 to 5, where level 0 represents no automation and level 5 represents full automation, capable of performing all driving functions under all conditions without human input [1]. This framework is essential for understanding autonomous vehicle technology's current capabilities and future goals.

The pursuit of autonomous driving technology began in the 1980s, with early experiments such as the EUREKA Prometheus Project in Europe setting the foundation for many aspects of modern autonomous vehicle technology [2][3]. Since then, advancements in computing power, sensor technology, and machine learning have accelerated the development of autonomous vehicles. Notable milestones include DARPA's Grand Challenges in the early 2000s, which spurred innovation in the United States by challenging teams to navigate autonomous vehicles through complex terrains and urban environments [4].

The development of autonomous driving technology relies heavily on the integration of various systems within the vehicle. This includes but is not limited to, advanced sensor arrays for environmental perception, such as LIDAR, radar, cameras, and ultrasonic sensors, and sophisticated algorithms for sensor fusion, path planning, and decision-making [5]. These technologies enable the vehicle to understand its surroundings, predict the actions of other road users, and navigate safely and efficiently.

One of the primary challenges in autonomous vehicle technology is achieving reliable and safe navigation in complex and dynamic environments. This requires the ability to detect and classify objects accurately, anticipate future states of the environment, and adapt to unforeseen circumstances [6]. The complexity of this task has led to a phased approach to development and deployment, with current commercial efforts focusing on levels 2 through 4, which offer various degrees of automation and driver support [7].

Despite the significant progress, autonomous driving technology faces numerous technical, ethical, and regulatory challenges. These include ensuring the safety and reliability of autonomous systems, addressing legal and liability issues, and gaining public trust [8]. Resolving these challenges is crucial for adopting autonomous vehicles and realizing their potential benefits, including reduced traffic accidents, increased mobility for those unable to drive, and improved traffic flow and efficiency.

1.3.2 DIFFERENTIAL STEERING MECHANISM

Differential steering is a steering method where it consists of two wheels that are independently driven, usually by servomotors in robots. It is the most common type of steering seen on wheeled robots due to its simplicity. This type of steering can also be found in existing vehicles such as wheelchairs, forklifts, and tanks, to name a few [9]. The steering system allows the wheels to go at the same velocity when maneuvering in a straight path and then adjusts the velocities of the two wheels if they want to rotate or turn [10]. Also, this steering system doesn't require the wheels to physically turn like in modern cars, furthering its simplicity to incorporate.

In robotic vehicle development, many studies have incorporated the use of differential steering into their robots, with some studies finding it to be effective through their research. In one such study where they were looking into the optimal steering strategy for a tracked robot, they simulated the robot's ability to steer while avoiding obstacles with center steering and differential steering. Compared to differential steering, center steering requires the robot to halt and rotate itself by having one wheel turn forward while the other backward to make a turn making an inefficient method to maneuver around obstacles [11]. Differential steering was found to be quite efficient in maneuvering around obstacles with great skill by only minimally changing the speed of the two tracks. This is important as having such high maneuverability would allow robots to be deployed in tight spaces. On top of that, they also found that using differential steering led to a decrease in both track skid and slip rates [11].

1.3.3 ACKERMANN STEERING MECHANISM

The Ackermann steering mechanism is an arrangement of linkages that allow a vehicle to turn both inside and outside wheels to avoid any sideways slippage while turning. The mechanism functions by making the axles of each wheel share a common center point. The common center point is achieved by ensuring each wheel is laid on a circle of increasing radii such that when the wheels are turned, they all share the same center of turning radius [12]. A line is extended from the rear wheels, from the inside front wheel, and from the outside front wheel (relative to the turn) to find a center point, as shown in Figure 1.

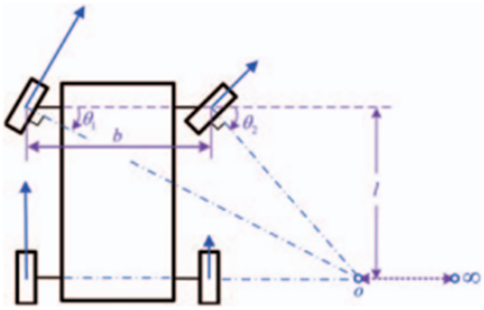


Figure 1: Ideal Ackermann turning geometry [13]

An Ackermann steering system is typically used in cars and works with either a two-wheel drive or a four-wheel drive, with the latter requiring another differential to control the speed of the front wheels as well. Each wheel travels at a different angular velocity based on the distance travelled by each tire. Angular velocity and distance travelled can be listed in the following descending order: 1) the outside front tire, 2) the outside back tire, 3) the inside front tire, and 4) the inside back tire.

The turning of the front wheel is achieved using a four-bar linkage. The linkage is controlled using the Ackermann criteria governing the maximum turning angle for each wheel to satisfy the condition of each wheel having the same center of rotation [13]. In order to achieve an ideal result, the linkage used in an Ackermann steering mechanism must satisfy the aforementioned principle, as significant deviation would negatively affect the tires and increase the chance of the tires slipping.

1.3.4 SENSOR FUSION IN AUTONOMOUS SYSTEMS

As sensor technology has become more advanced, it is beginning to play a role in autonomous systems and robotics. One such example of this is self-driving or automated driving vehicles. Automated driving is becoming a pivotal technology that can revolutionize the future of transportation and mobility [14]. Safety, reliability, and efficiency are paramount in autonomous vehicles. Achieving these goals necessitates a sophisticated understanding and integration of various sensory inputs to perceive and interpret the surrounding environment accurately. This integration process, known as sensor fusion, lies at the heart of autonomous vehicle technology, serving as the cornerstone for decision-making and control.

One study on sensor fusion involved a vehicle localization approach using the Global Navigation Satellite System or GNSS, the Inertial Measurement Unit or IMU, the Distance Measuring Instruments or DMI, and Light Detection and Ranging or LIDAR. The GNSS, IMU, and DMI sensors were fused using the Unscented Kalman Filter to improve the autonomous vehicle's localization accuracy. The research concluded that this method was effective in enhancing the autonomous vehicle's localization accuracy. On top of this, it was also able to estimate curbs in real time using 3D LIDAR, which could prevent self-driving vehicles from hitting sidewalks [15].

Another study involved the use of GNSS, IMU, Odometer or ODO, and LIDAR-SLAM (simultaneous localization and mapping) as a navigational system. The system was used to control a land vehicle in two situations: open-sky areas and tunnel cases. From the tests, it was determined that the navigation system was capable of improving the accuracy and robustness of navigation. The GNSS/INS/ODO integration was able to reduce the root mean square of position drift error by 52.1% - 72.3%. It was concluded that the proposed navigation system was able to fuse information from multiple sources to maintain the SLAM process while mitigating navigation errors [16].

LIDAR and IMU sensor fusion can be used for other vehicles, such as unmanned aerial vehicles or UAVs. Although localizing and navigating UAVs is difficult due to altitude and motion dynamics, it was possible in a study published in 2017. The UAV would use LIDAR and IMU sensor fusion to navigate a pipeline. The navigation system proposed in the study used the Kalman filter to derive the 3D position of the UAV by fusing primary and secondary LIDAR data. The navigation system proved successful in its tests and showed great potential for use in small UAVs for indoor navigation and localization [17].

1.3.5 NATURAL LANGUAGE PROCESSING(NLP) IN NON-TEXTUAL APPLICATIONS

Natural Language Processing (NLP) has long been a crucial technology in understanding, generating, and interpreting human language, with its applications from voice-activated assistants to complex data analysis tools. While traditionally, NLP has been focused on textual data, offering advancements in text completion, translation, and sentiment analysis, its potential extends far beyond different applications, such as autonomous driving systems.

Recent innovations have seen NLP techniques being adapted for the interpretation of environmental data, allowing autonomous vehicles to comprehend and act upon complex commands and descriptions of their surroundings. This adaptation involves the processing of sensor inputs such as those from LIDAR, radar, and cameras, converting them into a format that can be understood and acted upon by the vehicle's decision-making algorithms. For example, voice commands with details of specific navigation instructions or descriptions of nearby obstacles can be parsed and translated into actionable data, letting the vehicle guide through different landscapes or adjust its path in response to sudden environmental changes [18].

The use of NLP in autonomous systems is further enhanced through its integration with sensor fusion technologies. This allows for the creation of a more comprehensive environmental perception framework, where the fusion of data from various sensors is complemented with the interaction of NLP. Such an approach enables the vehicle to not only detect and navigate around obstacles but also to understand and respond to more abstract commands by the user, improving the interaction between humans and machines [19].

Moreover, NLP's role in autonomous driving extends to the simulation of potential future scenarios. Through the processing of environmental descriptions and predictive modelling, NLP can assist autonomous vehicles in evaluating various navigation strategies and choosing the optimal path that maximizes safety and efficiency. This aspect of NLP shows a more dynamic and adaptable decision-making process [20].

Despite these advancements, NLP in such non-textual applications has its own challenges. The accuracy of NLP interpretations in this context heavily relies on the quality of input data for advanced data processing and error-handling mechanisms. Additionally, the computational demands of real-time NLP processing and the need for continuous learning and adaptation to new environmental conditions and languages prove the need for significant computational resources and sophisticated machine-learning models [21][22].

1.3.6 KINEMATICS

The goal of this project is to also incorporate a trailer system with the robot, which adds another level of complexity. It is important to examine the kinematics of the robot in relation to the trailer since the trailer's movement is dependent on the robot itself. A few important

instabilities that need to be considered are the trailer swaying (with and without a load) during motion and the optimal turning angle.

In one study, they began their research by using four-bar linkages, two of which are articulated, as they found from a previous study that articulated linkages provided promising stabilizing properties. They created three connection arrangements to test: backward converging, inter-crossing, and forward converging sidebars $L_{bl}L_{bt}$ and $R_{bl}Rb_{bt}$ [23].

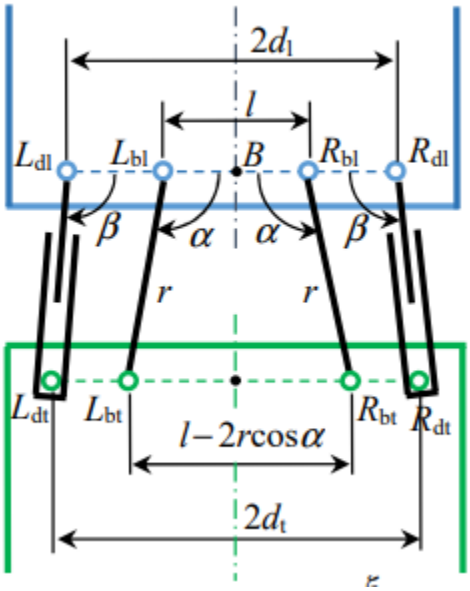


Figure 2: Four-bar linkage trailer connection [23]

It's important to test different arrangements as they directly impact the maneuverability in the space that the vehicle travels in. This was the case when they looked into the relative rotation between the vehicle and trailer. For backward converging, they found it to be not very large, under 90° actually [23]. Conversely, forward converging may be capable of reaching close to 90°. Lastly, inter-crossing resembled more of the pintle-hitch that's on modern vehicles, capable of going past 90° [23].

Another area they explored was the stability of the four-bar linkage by comparing the velocity of the vehicle with the trailer's centre of mass in relation to where the axle is located. The forward converging bar was demonstrated to be more stable than the backward linkage, not requiring dampers, but of course, dampers would further improve the stability [23]. Although it proved to be better, in the end, the stability is more dependent on the velocity and the centre of

mass in relation to the axle of the trailer. Another way to further improve the stability was through the increase of the cornering stiffness of the axle of the trailer [23].

In terms of optimal turning angle, that is done through calculations. In this study, however, the researcher instead of deriving a formula for the angle, instead derived formulas for the ideal and actual radius of curvature of the path [23].

In Principles of Robot Motion Theory(H. Choset), if the robot were designed to drive like a modern car (i.e, using Ackerman steering), then a minimum turning radius can be calculated using the distance between the centres of the front and rear wheels along with the steering angle which can be between 0 to 90 [24]. Also, to ensure a smooth turn, it would have to satisfy a turning radius constraint where its angular velocity has to be less than the ratio of linear velocity to the minimum radius. On the other hand, differential steering does not have a steering angle when it turns, and the formula and constraint are inapplicable to it

The linear velocity of a single attached trailer or multiple trailers can be found by using the velocity that the trailer is pulled at along with the cosine of the relative angle between the trailer and vehicle [24]. For additional trailers afterward, the linear velocity would become the pulling velocity. The angular velocity can also be found by using the ratio of pulling velocity to the distance between the axle and hitch multiplied by the sine of the relative angle[24].

Dublin's path can be used to help determine the optimal path of travel, as that is the purpose of Dublin's path. Given two points in a plane, it will produce the shortest path consisting of curves and straight lines [25]. The paths are computed and compared to one another, which can take a while. One study explored using the logical classification scheme to figure out the shortest path without having to go through all the computations. They found that the elements of the Dubins set can be organized into equivalency groups, which are based on the angle quadrants of the corresponding pairs of the initial and final orientation angles [25]. The group has different classes of paths, but all turn out to be equivalent. This reduces the analysis time. Their findings allowed them to create a decision table to find the shortest path. However, note that their research was done only for the long-path case, and not the short-path case. The two cases are similar, however, the short path would take more work overall [25].

1.3.7 MECHANICAL DESIGN AND STRESS ANALYSIS

The mechanical design required in this project involves, the design of the Ackermann steering system, the linkages used for the steering system and the trailer connection or hitch. The stress analysis focuses on areas the critical areas of the design, such as the floors, and the trailer hitch. Additionally, load analysis is also required for the trailer if applicable.

The Ackermann system requires a linkage allowing for the front wheels to turn, a typical Ackermann steering mechanism makes use of a four-bar trapezoidal linkage. The linkage needs to satisfy the Ackermann principle and ensure that each wheel shares a common center point of turning [13].

The trailer connect / hitch is a critical point in our design as the trailers will be connected using the hitch and turn about the hitch as the driving car turns. The type and detail of the analysis depend on the type of hitch designed. There are various types of hitches used, ranging from simple pin lock hitches to hitches with links used for heavy machinery such as a tractor [26].

The wheels in the driving car are an important component, if the wheel is improperly chosen, it directly affects the quality and safety of the vehicle. Therefore, it is important to conduct an analysis of the wheels used, this includes the load distribution, the reliability of the wheel, and an FEA study to determine the stress state distribution of the wheel. [27]

The trailer used in this project will be designed to carry the load as such, load analysis is an important process to ensure the trailer can carry the load. The placement of the load on the trailer also needs to be considered as the weight distribution of the trailer can cause sway [28]

1.3.8 BEST PRACTICES AND GAPS IN CURRENT RESEARCH

In developing autonomous driving systems, a combination of advanced sensor technologies, robust mechanical designs, and sophisticated algorithms has emerged as a best practice. Integrating multiple sensors, such as LIDAR, radar, and cameras, facilitates comprehensive environmental perception, which is critical for safe navigation [29]. Best practices also emphasize the importance of redundancy in systems design, ensuring that the failure of a single component does not compromise overall system safety [30]. Moreover,

adopting a modular approach to a system architecture enables more accessible updates and maintenance, contributing to the longevity of autonomous driving technologies [31].

Applying LLMs to non-textual domains, particularly in processing environmental data for autonomous systems, has adopted several best practices. One such practice is the use of transfer learning, where models pre-trained on large textual datasets are fine-tuned on domain-specific non-textual data, significantly reducing the time and resources required for model training [32]. Additionally, the integration of domain-specific knowledge into LLMs has been shown to improve the model's performance and interpretability in tasks related to environmental perception.

Despite these advancements, several gaps remain in the research. One of the most significant gaps is the lack of standardized datasets for training and evaluating LLMs in non-textual applications. Unlike in NLP, where large-scale, annotated datasets are abundant, equivalent resources in non-textual domains, such as environmental perception for autonomous driving, are scarce. This limitation hampers the development and benchmarking of models across different research groups.

Another critical gap is the challenge of ensuring safety and reliability in the decision-making processes of autonomous systems. While effective in many different scenarios, current LLMs still struggle with specific scenarios that require deep learning. Developing models incorporating uncertainty in their decision-making remains a significant research challenge.

CHAPTER 2 - PROBLEM DEFINITION

2.1 PROBLEM STATEMENT

The integration of autonomous technologies into transportation has shown a significant stride toward enhancing safety, efficiency, and environmental sustainability. However, the complexity of autonomous navigation escalates substantially with the addition of a trailer to a truck. This configuration introduces complex challenges in kinematics and path planning, exacerbated by the trailer's distinct movement patterns and the increased difficulty in maneuvering and stability control. Conventional autonomous driving systems, primarily designed for single-unit vehicles, struggle to accurately predict and adapt to the dynamic behaviours of truck-trailer assemblies in real-time operational scenarios. The static environment, while eliminating variables associated with dynamic obstacles, requires an acute understanding of spatial constraints and the ability to plan paths that account for the extended physical properties of the vehicle assembly. Additionally, the current interfaces for command input in autonomous systems lack the intuitiveness and flexibility needed for continuous human-machine interaction, particularly in specifying navigational commands for vehicles towing trailers. Addressing the difficulties associated with autonomous truck and trailer systems, from advanced kinematic models to sophisticated path planning and intuitive command interfaces, remains a pivotal yet unmet need for truly autonomous transportation solutions. This project aims to bridge these gaps, focusing on the unique challenges posed by the integration of trailers in autonomous driving to develop a more adaptable, reliable, and user-friendly system.

2.2 SCOPE OF THE PROJECT

This project is dedicated to the design, development, and implementation of an advanced autonomous driving system specifically tailored for a truck with an attached trailer navigating within static environments. It targets the development of a system capable of understanding and executing complex navigational commands provided through Natural Language Processing, thereby enhancing the user experience by simplifying the interaction between the vehicle and the driver. A foundation of this initiative is the integration of sensor fusion technology, combining data from LIDAR, IMU, and rotary encoders. This integration is crucial for creating a detailed

and accurate environmental map that the vehicle relies on for navigation, obstacle detection, and path planning.

Furthermore, the project encompasses a mechanical analysis of the truck and trailer system, utilizing FEA to ensure structural integrity and operational safety under various conditions. Due to the trailer's kinematic challenges and path planning complexity, maneuverability, and stability optimization were mainly considered within predefined static environments. The project's scope includes extensive simulation and real-world testing phases to validate the proposed method's effectiveness and reliability. Through a structured and systematic approach, the project aims to deliver a robust autonomous truck and trailer system that sets a new benchmark for efficiency and safety in the domain of autonomous logistics and transportation while also laying the groundwork for future exploration into dynamic environments and more complex scenarios.

CHAPTER 3 - DESIGN AND ANALYSIS

3.1 DETAILED DESIGN

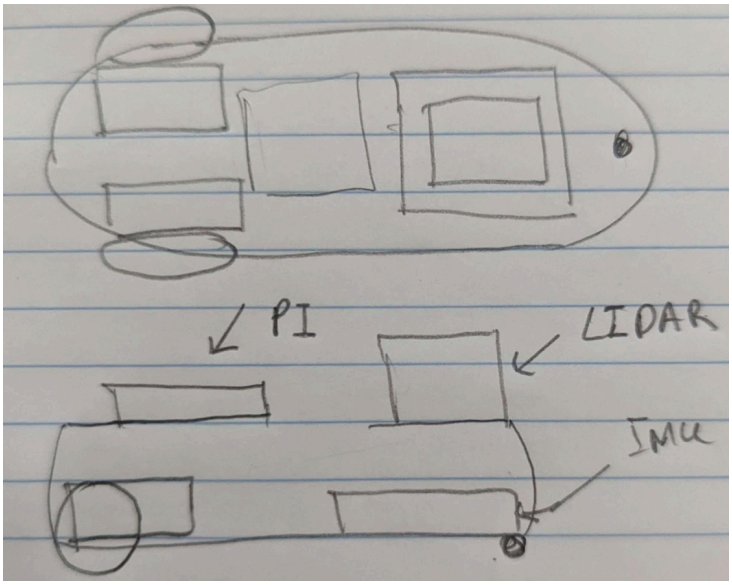


Figure 3: Concept 1

The design philosophy behind Concept 1 was aimed at creating a body that could accommodate all the necessary sensors and components while minimizing the footprint of the vehicle. This was achieved by basing the design on that of a kayak, with sufficient space to house the battery, motor, and IMU internally. The Raspberry Pi and LIDAR were mounted externally due to space constraints.

A differential drive and a free-rotating ball were utilized, enabling the vehicle to turn. Additionally, the LIDAR was aligned on the z-axis with the board containing the IMU, simplifying the calibration of relative position.

Plans for further enhancing this design included adapting it to incorporate an Ackerman steering system and optimizing the load on the body for even distribution. Transitioning to an Ackerman steering system would require additional space for the steering mechanism, and the positioning of the components might need to be adjusted to maintain the balance of the vehicle.

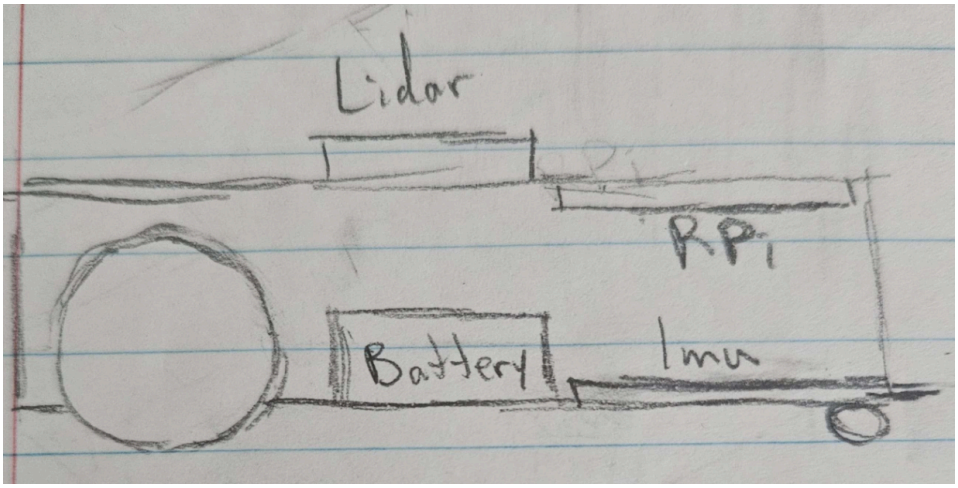


Figure 4: Concept 2

In the design of concept 2, the goal was to make it more compact (i.e., reduce the height) with a more boxy shape so that it would resemble a truck and reduce the amount of materials needed. In order to achieve this, instead of having another floor to house the Raspberry Pi, the Raspberry Pi is instead mounted upside down on the underside of the upper level. The LIDAR has to be kept on top of an open surface because it can't have any obstructions to be able to observe the surroundings.

The battery is placed in the middle because the front is occupied by the wheels' motors for a differential drive as well as having the battery in the middle evens out the weight distribution so that the front wheels aren't bearing most of the load. The back has a free-rotating ball to achieve turning. Additionally, there are plans to incorporate Ackerman steering where the differential drive is and be able to swap between the two. With the remaining space, the IMU would be placed behind the battery.

Trailer

The initial idea for the trailer was to design one based on an already existing trailer on the market. The trailer has a flatbed, with a ramp that would allow objects to enter from it, a ball joint connection for the hitch(mentioned later on), and a still axle with free spinning wheels. However, this design was not ideal. The ramp merely served as an aesthetic, as nothing was actually going to be loaded from the ramp, as well as it couldn't be used even if we wanted to because it isn't proportional with the trailer as seen in Figure 5. The trailer would have been

made longer to work with a shallow angle because the wheels are quite large, making the height of the trailer bed too high for the designed length. Additionally, there were no walls to prevent the load from sliding off or areas to secure the load to the trailer. The trailer was redesigned with those issues in mind as seen in Figure 6. The walls were designed higher than necessary to be conservative if we ever test with a tall load. There are now four securing points on the trailer, one on each side of the wall. The securing points are designed with two open gaps so a string/hook can wrap around the solid part. Due to the tires covering the middle walls, the two on the side had to be placed elsewhere which could be anywhere but was chosen to be at the back of the trailer as that would create the most amount of space from the truck if any adjustments were needed to the securement method in use. In addition to those changes, the wheels were also changed. Due to a lack of wheels available to use, it was changed to use the same wheel but 3D printed instead.

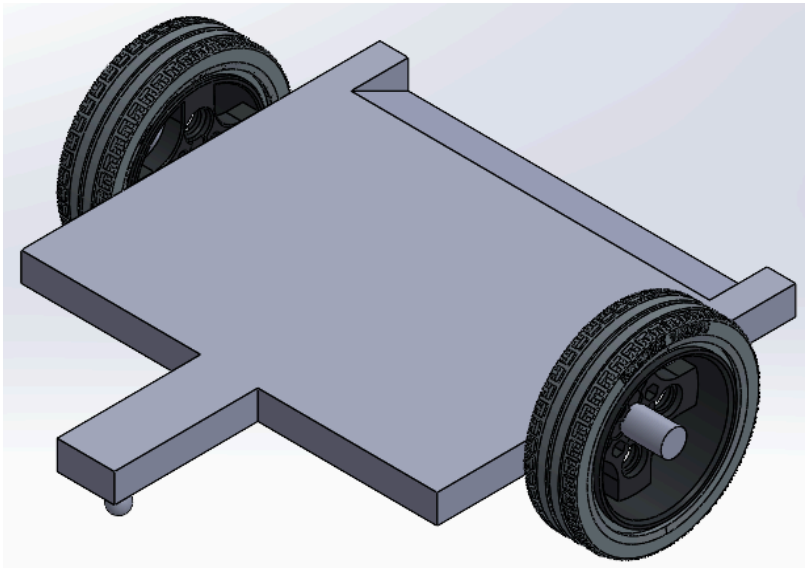


Figure 5: Original trailer design

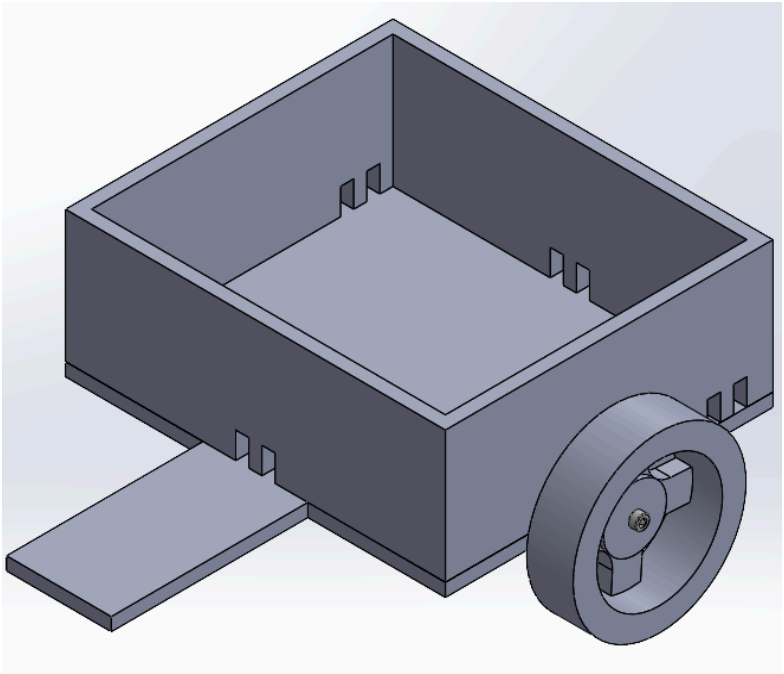


Figure 6: New trailer design

Original hitch

Furthermore, on the trailer design, the initial hitch design was to use a ball and socket joint coupled with a nut to tighten the connection. The socket joint was designed with a cutout that divided its walls into four quarters. It was designed like this because the idea is to use a nut that is slightly smaller in diameter than the socket so it would slightly compress the socket walls inwards, tightening around the ball joint, resulting in a secured connection. However, this ended up not working as we hoped when prototyping it. The socket joint is directly connected to the first floor, so it ended up being printed laterally with the floor. This is bad because this resulted in the print lines being parallel with the direction of the force being applied, making it easy to fail due to shear. It ended up breaking just as we expected during testing. However, the dimensions of the design also contributed to the failure. The walls of the socket were relatively thin, having a thickness of 2mm, further contributing to its breaking. There was an attempt to change the cutout from quarters to thirds as well as increase the thickness of the walls, together creating a sturdier socket. However, this was not enough, as shown in Figure 8. Additionally, the ball joint experienced the same fate due to the print lines being parallel with the force. In order to have a chance of the socket working, it would require the walls to be a lot thicker than

what was currently designed. However, even if it's made thicker, it's still at risk of breaking in the future, again, due to the print lines being parallel to the force.

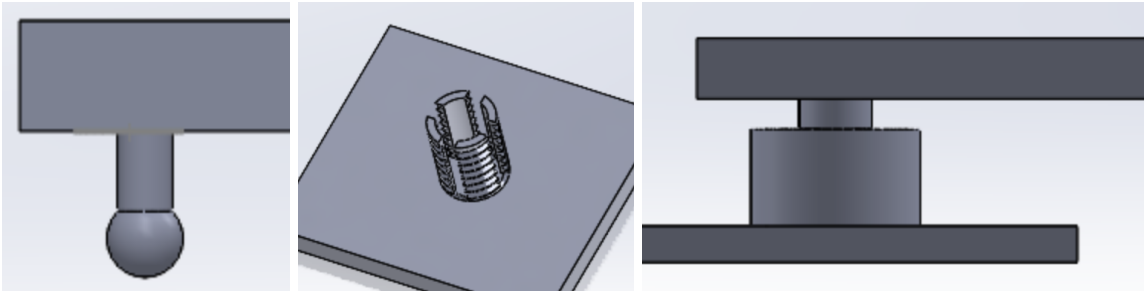


Figure 7: Trailer ball hitch connection

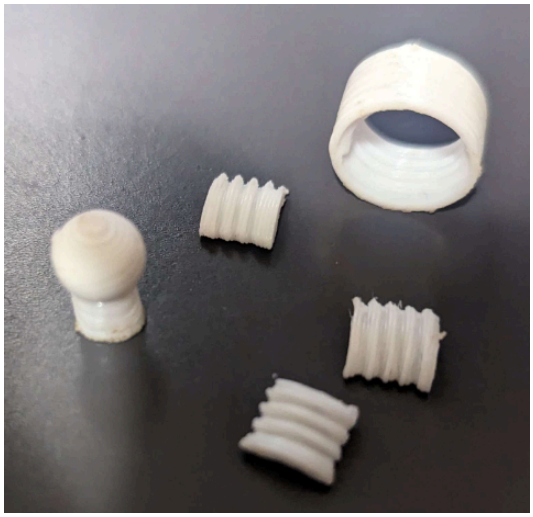


Figure 8: Broken ball and socket joint

Revised hitch

Increasing the thickness of the socket walls would allow for a stronger hitch however, due to the manufacturing process used, the significant weakness would remain, leaving the hitch prone to breaking. The 3D-printed layer lines would always be parallel to the stress on both the ball joint and socket end, as they were designed to be printed as part of the floor and trailer base, respectively. Due to the issues presented by the hitch and in an effort to save 3D printing time, a new hitch that could be mounted onto the floor would be better suited. From brainstorming and researching, we stumbled upon one where the trailer clips onto two gaps on the truck as seen in Figure 9. This gave us an idea to create something similar. We designed new hitch mounts onto the trailer and truck by sliding onto them; the same goes for the turning connection

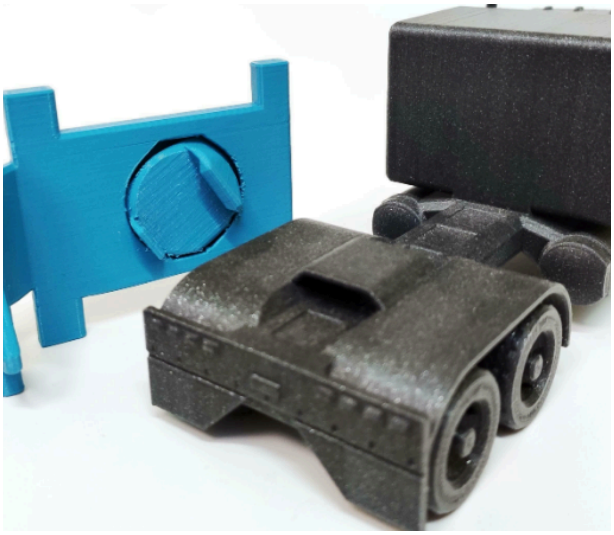


Figure 9: Hitch reference [37]

The hitch was split into two components. The first component featured a distinct groove attached to the truck using a sliding fit. The second component, which was also connected to the trailer using a sliding fit, was designed to fit into the grooves of the first component. This allowed the hitch to be connected while a rotating component within the hitch ensured the trailer retained its ability to turn. An added advantage of this design was the ability to regulate the maximum angle at which the trailer could turn in relation to the truck. The shape of the grooves on both components was designed to maximize the area of contact between the two components to allow for a strong sliding fit.

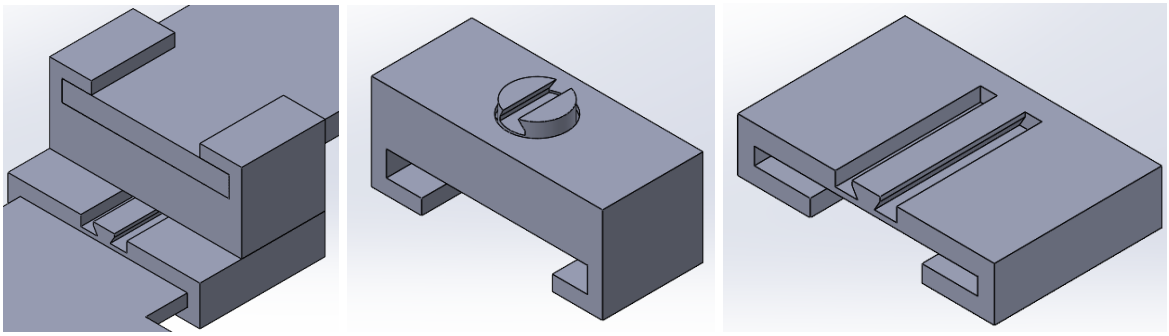


Figure 10: Redesigned hitch

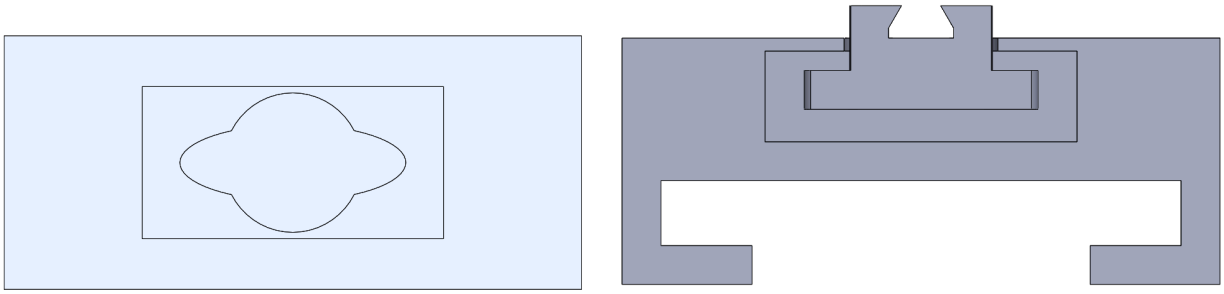


Figure 11: Internal view of the hitch

The second component of the hitch incorporates an elliptical shape with an extruded cylinder internally. This component slides into a groove in the first component. The maximum rotation of the hitch is determined by the dimensions of the ellipse and the available internal space. The design allows for single-piece printing using supports, which are subsequently removed to enable rotation of the hitch about that cylinder.

Ackerman steering design

The Ackerman steering system employs a simple 4-bar linkage with one fixed link. The system begins with a fixed link that also acts as a mounting point, attaching the steering system to the overall assembly. The subsequent links connect the linkage to the wheels and axles, allowing the wheel to turn based on the movement of the second and third links. Lastly, the middle link connects links 2 and 3, enabling relative movement between them. A servo and servo mount are attached to manipulate the middle link. The servo's movement affects the middle link, which, in turn, moves links 2 and 3. The maximum turning angle was determined to be 25 degrees, mimicking the range commonly used by most vehicles (20-30 degrees).

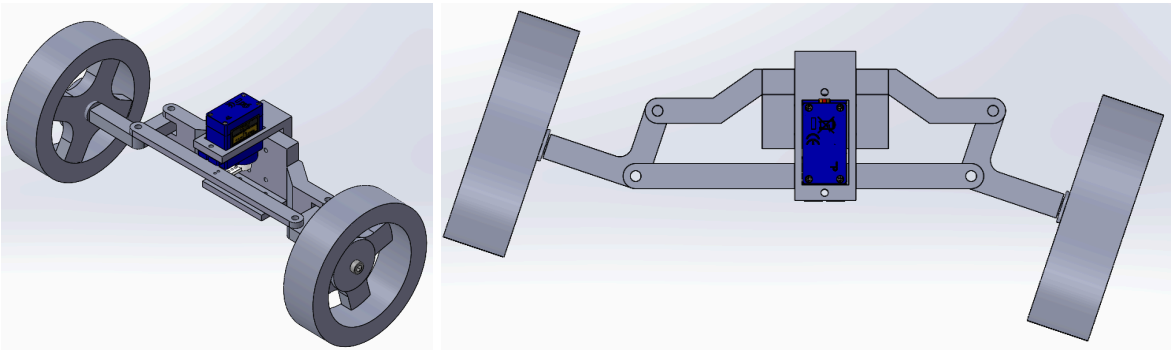


Figure 12: Ackerman steering system

Final design

The final design of the truck and trailer incorporates all of the mentioned components from earlier (i.e. trailer, hitch, Ackerman steering). The Ackerman steering system turned out to be capable of acting as differential steering as well, resulting in not requiring to have two separate steering systems. The final design also incorporates changes from the concepts that were discussed with the team. For one, the idea of making it compact was not ideal as space is required to do the wiring. Due to this, the team agreed to use three floors to house all the components. To further make it easy to wire, all the boards were housed on the second floor and oriented in a specific orientation to help with that, and the LIDAR was on the top floor, just like in the concept. Additionally, the LIDAR and IMU are now aligned because in order to do sensor fusion, the relative positions must be known. By aligning the two as much as possible, it simplifies the relative position. In this case, only need to know the relative position along the z-axis. Supports were used to hold up the second and third floors. They were placed near corners to minimize the deflection that the components would cause and a few additional ones were under where the majority of the weight from components would be.

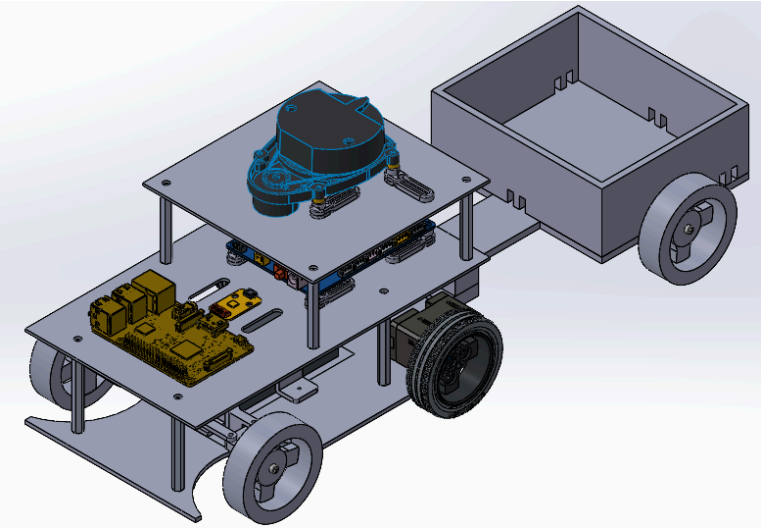


Figure 13: Final Assembly

3.3 KINEMATICS

3.3.1 MOTION

Not only was the vehicle supposed to look like a truck, but also to mimic one to demonstrate how it can be applicable to real life. We determined the following parameters for motion as seen in the table below:

Table 1: Truck motion parameters

Parameter	Value
Minimum Translational Velocity	0.11m/s
Maximum Translational Velocity	0.22m/s
Minimum Angular Velocity	1.37rad/s
Maximum Angular Velocity	2.75rad/s
Maximum Acceleration	$2.5m/s^2$
Maximum Angular Acceleration	$3.2m/s^2$

These values were determined by examining the specifications of current trucks on the market to get an initial sense of the numbers. Then when it came to scaling it to fit our map, we also used the TurtleBot3 as another reference for appropriate numbers for an indoor setting as our design is somewhat similar to it.

3.3.2 TURNING ANGLES

The turning angle for the Ackerman steering system was determined by studying the angles commonly used in existing vehicles on the market. Most vehicles have a turning angle ranging from 20 to 30 degrees relative to the axis of revolution of the rear driving wheels. For our steering system, we chose an angle of 25 degrees for two primary reasons:

1. Compatibility: By selecting 25 degrees, we ensured that our system falls within the range commonly found in existing vehicles.

2. Component Fit: Additionally, this angle was chosen to harmonize with other components of the truck. A turning angle of 30 degrees required more space for the steering system to be compatible with the frame of the truck, taking out this material from the frame would potentially affect the strength of the bottom floor. On the other hand, a 20-degree turning radius was also considered, but 25 degrees struck a balance within the typical range.

Furthermore, the chosen turning angle of 25 degrees results in a turning radius of 332 mm based on equation (1), which has been deemed sufficient for the intended application.

$$Turning\ radius\ (tr)$$

$$Turning\ angle\ (ta)\ =\ 25\ degrees$$

$$Wheel\ base\ (wb)_{\ distance\ between\ front\ and\ back\ wheels}\ =\ 155mm$$

$$(1)\ tr\ =\ wb\ \div\ tan(ta)\ =\ 155mm\ \div\ tan(25)\ =\ 332.40mm$$

Ideally, a trailer should be capable of making a maximum 90-degree turn relative to the truck. This 90-degree trailer turn is primarily used for back parking or maneuvering the trailer into tight spaces.

However, for the specific scope of our application, a 90-degree turn for the trailer is unnecessary because the truck will not be performing such extreme movements. After analyzing trailers attached to other vehicles, we determined that an angle of 25-35 degrees would be sufficient. This range allows the trailer to navigate turns without requiring an excessively wide angle between the trailer and the car.

The hitch and trailer design ensure that the trailer has a maximum turning angle of 35 degrees relative to the truck—a suitable range for the truck's anticipated movements.

3.5 FEA (FINITE ELEMENT ANALYSIS)

3.5.1 STRESS ANALYSIS

The stress analysis was carried out in SolidWorks and was used to generate Von Mises stress plots as seen in the figures below. In Figure 14, it can be seen that the holes near the corners of the supports are the highest points of stress. This makes sense as those areas are supporting the weight of the LIDAR. The thin slots also have some stress as they bear a bit of the weight of the LIDAR. Note that the deformation seen in this and the following figures are exaggerated by SolidWorks.

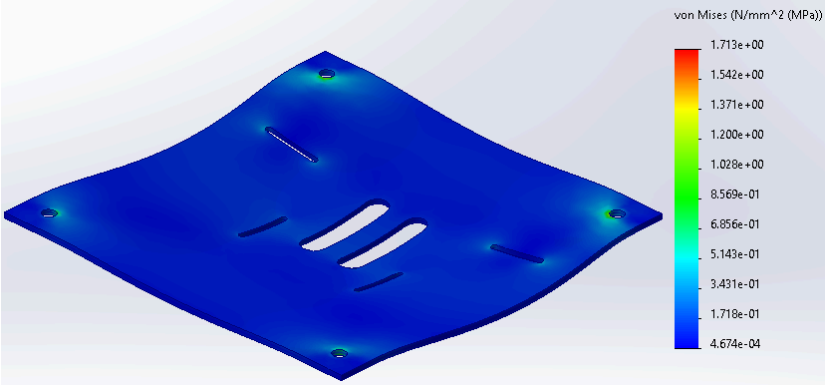


Figure 14: 3rd Floor Von Mises Stress

Similarly in Figure 15, the holes of the supports have the highest stresses followed by the thin slots due to supporting the components from the third floor as well as the Raspberry Pi, and IMU.

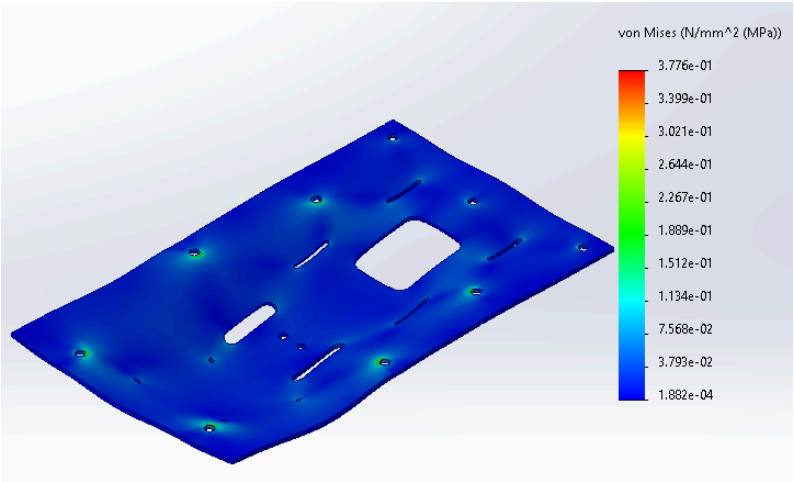


Figure 15: 2nd Floor Von Mises Stress

In Figure 16 the floor was simulated with the weight of the steering system, the battery and the motors, the wheels, and the weight of the top two floors distributed by the supports. All of the wheels were fixed in place for this simulation. The majority of the deformation would be caused by the weight of the battery. All of the holes where the supports are attached have the highest strain as expected.

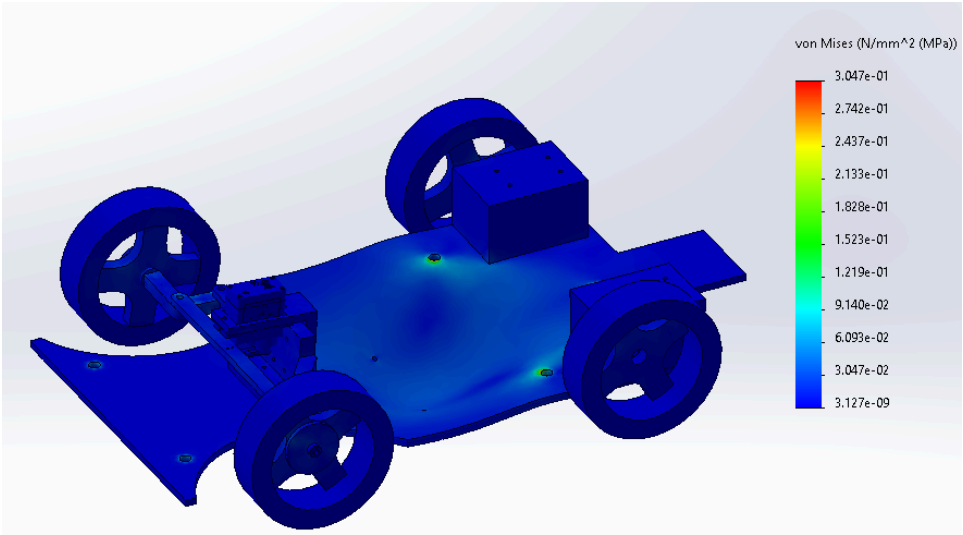


Figure 16: 1st Floor Von Mises Stress

In Figures 17 and 18 the hitch was simulated with the pulling force on each part of the hitch from the motors for the bottom half and the friction force from the trailer for the top half. As seen in Figure 18 most of the stress for the bottom half is where the hitch is attached to the first floor of the truck. Most of the stress on the top half of the hitch is on the groove that slides into the bottom half. The results are consistent with our testing as the groove on the top half of the hitch is a potential point of failure as further discussed in the FMEA section. While the bottom half stress is explained by the pulling force and the tight sliding fit.

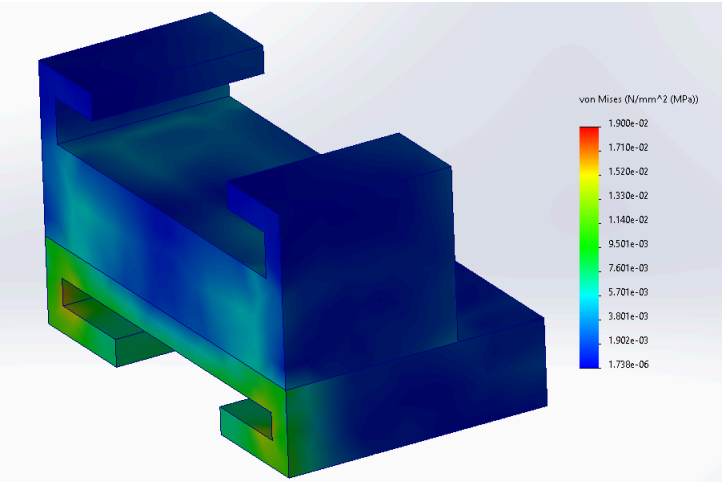


Figure 17: Hitch assembly Von Mises Stress

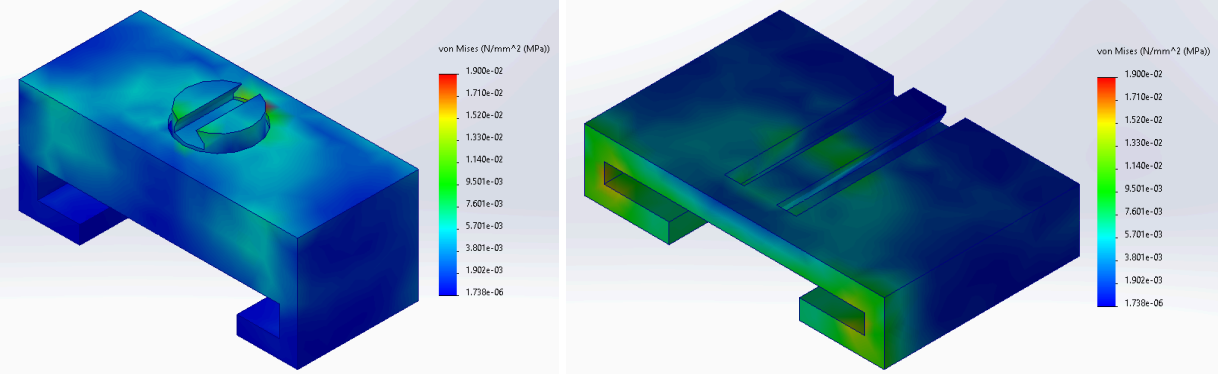


Figure 18: Hitched pieces separated Von Mises Stress

In Figure 19 the full assembly was simulated with the wheels fixed. As seen in the figure, the full assembly does not have any major deformations and has relatively low Von Mises stress. Most of the stress is found in the supports which is expected as they bear most of the load.

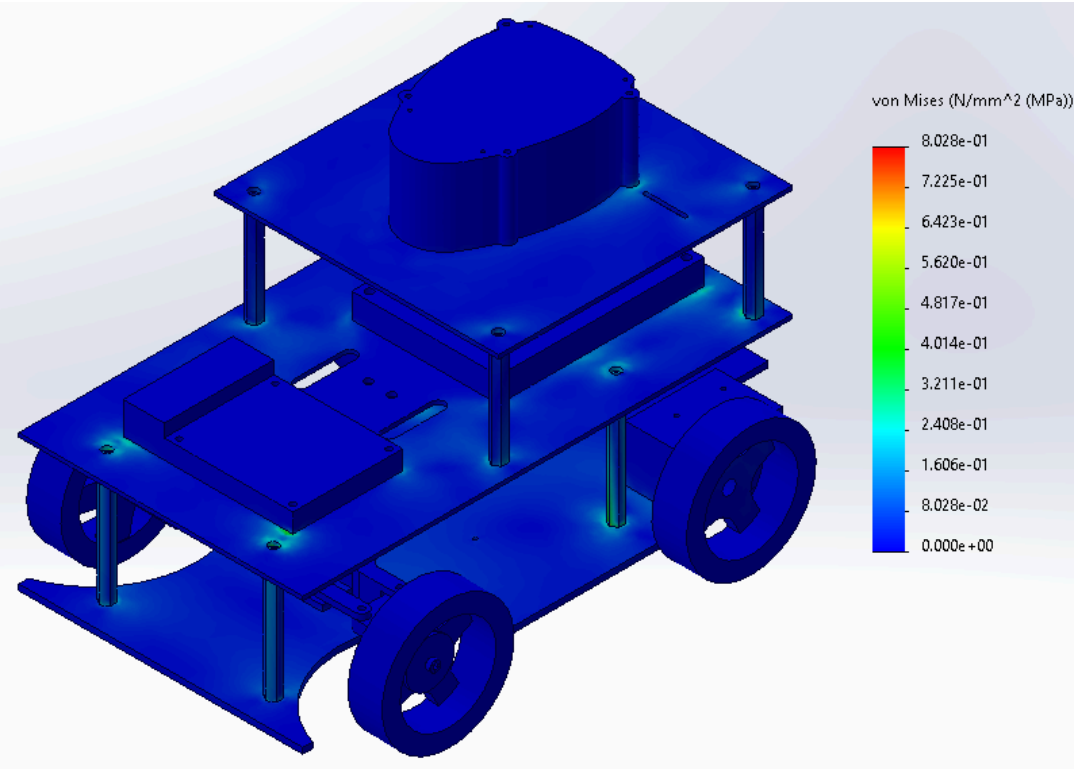


Figure 19: Final Assembly Von Mises Stress

3.5.2 LOAD ANALYSIS

Analysis of structural supports for the truck: Figure 19 reveals that the maximum stress on the truck's structural supports occurs at the back three supports, which are close to the motors and wheels. This outcome aligns with expectations, given that the majority of the truck's weight is concentrated toward the rear. Additionally, the Von Mises stress plot indicates that the current load on the truck will not lead to any fractures or failures.

Considerations for the trailer load: The trailer attached to the truck serves as a platform for a dummy load during demonstrations. When placing the load on the trailer, attention must be paid to its positioning. Shifting the load's placement affects the center of gravity of the trailer and may induce swaying (i.e. swaying may occur if there isn't enough tongue weight). To mitigate this, it is advisable to position the load near the center of the trailer. By doing so, we can minimize or eliminate the resulting sway while the truck is in motion.

3.6 FMEA (FAILURE MODE AND EFFECTS ANALYSIS)

Failure Mode and Effects Analysis (FMEA) is a tool used in design. Its purpose is to identify potential problems of a product and assess its impact and potential solutions for it. For this project, two points of failure were identified. The first failure point identified is at the hitch, specifically the component that allows the trailer to rotate about. This component has a slot which allows the trailer to connect to the truck. As well as it also has two “wings” on each side that control the maximum rotation allowed, as seen in Figure 11. Through prototyping, we noticed that the portion near the slot(as seen in Figure 20) can break first if it is forced to turn beyond its limit. This can be seen in Figure 21 of the stress analysis. It can’t be directly seen in the simulation but there is stress there. Furthermore, how the component was printed was the main cause of its breaking. The component was oriented upright when printed making the print lines to be parallel with any lateral force that gets induced. However, in reality, the component won’t break during actual performance because the truck has been set up with specific parameters that are appropriate to the map that we’ve made (see Figure 28). Although it may not break, we can take a corrective approach by changing how it's manufactured. That can be changing how the part is printed by reorienting it so that the print lines are not parallel to the force or using another method such as injection moulding.

The second failure point is at the 3D-printed axles in both the front of the truck and at the trailer. The purpose of the axle is to connect the wheels to the vehicle in order to allow the vehicle to move, as well as maintain the relative position of the wheels to each other. The axles can fail from experiencing an upward or downward force, but realistically, upwards. This would be from maneuvering over bumpy terrain. Just like the rotational component from the hitch, the axles' cause of failure is due to them being printed in the non-ideal orientation, resulting in the print lines being parallel to the force making it susceptible to sheer. Additionally, having no suspension contributed to its failure too. Having some form of suspension would at least allow the axle to withstand more force before breaking. The axles have yet to break from our testing; however, to fix this, just like the rotational component, the axles can be manufactured differently, whether that be reorienting so the print lines are perpendicular to the force or using a method like injection moulding. As well, the steering system could be redesigned to incorporate suspension to absorb the force.

Refer to Appendix A.1 on the completed FMEA table

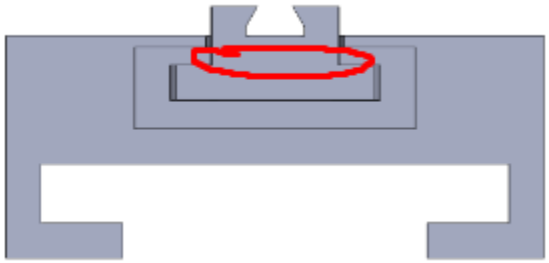


Figure 20: Location of breakage from testing

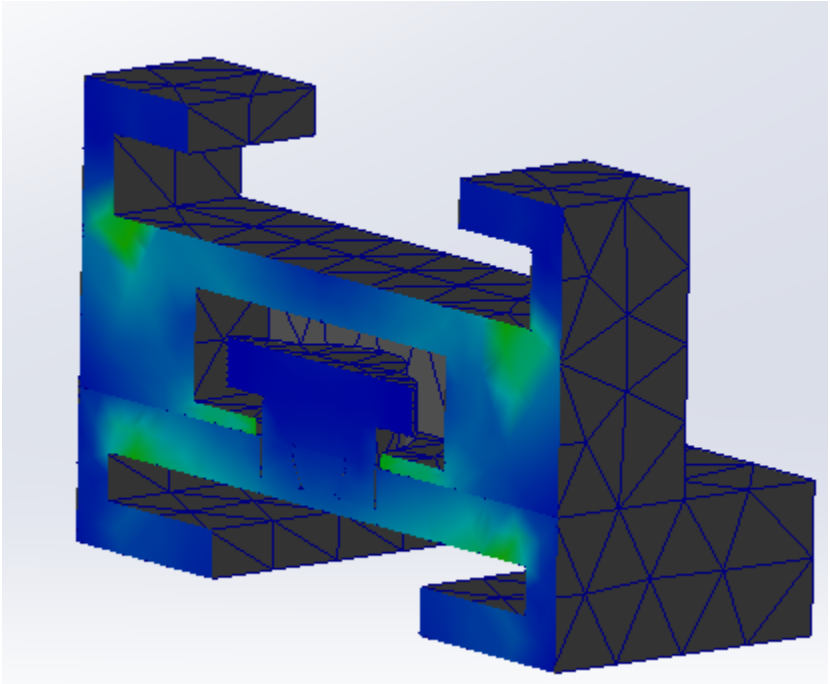


Figure 21: Section of Hitch Von Mises

CHAPTER 4 - SENSOR INTEGRATION

4.1 LIDAR

LIDAR, which stands for Light Detection and Ranging, is a remote sensing method that uses light as a pulsed laser to measure variable distances to the Earth. It operates on the principle of emitting laser pulses and measuring the time it takes for the light to return after reflecting off surfaces. This information can then be used to create detailed three-dimensional maps or models of objects or environments [33]. To further this, in comparison to other sensors, lasers are capable of higher levels of precision, which gives them applications in high-speed vehicles such as self-driving cars [34]. As such, the LIDAR proved to be paramount as a sensor for use in obstacle avoidance and navigation for the autonomous truck and trailer system.

Figure 22: LIDAR Mounted on the Truck and Trailer System

The autonomous truck and trailer system relies on a LIDAR to provide a complete map of its surroundings. The LIDAR is capable of sensing 360 degrees around the vehicle, which is extremely helpful in Simultaneous Localization and Mapping. With this technology, the system can detect and avoid obstacles more effectively, resulting in a safer driving experience. The LIDAR is mounted at the top of the truck to ensure that it has clear visibility of its surroundings,

as shown in Figure 22. This placement gives the system a better vantage point and allows it to detect objects from a higher perspective.

4.2 IMU

IMU stands for Inertial Measurement Unit. It's a sensor device that measures and reports specific force, angular rate, and sometimes magnetic field surrounding the device. IMUs often contain multiple accelerometers and gyroscopes to provide measurements along different axes. By combining the data from these sensors, an IMU can determine the device's orientation, velocity, and changes in position over time[35].

The use of an Inertial Measurement Unit (IMU) was deemed crucial for the success of the autonomous truck and trailer system. The IMU allows the system to determine its orientation at any given time and navigate through different environments. To achieve this, the IMU used in this project contained a 3-axis gyroscope, which measures the rotational motion of the system, and a 3-axis accelerometer, which measures the linear acceleration of the system. In combination, these sensors provide the system with 9-axis measurements, enabling it to accurately detect its movement and position.

To ensure optimal performance, the IMU was implemented directly into the motherboard of the system. It was strategically placed as close to the center of the truck as possible to align its axis with the axis of the truck as closely as possible. This placement ensures that the IMU can accurately detect the movements and orientation of the truck and trailer, allowing for safe and efficient autonomous operation.

4.3 ROTARY ENCODER

The implementation of Rotary encoders for the autonomous truck and trailer system was essential in determining the movement of the trailer system. This is because the rotary encoder converts a shaft's angular position or rotation to digital signals in the form of pulses. The output can be processed by a microcontroller or other digital/analog circuits for further use. The particular type of rotary encoder used was an incremental encoder. These produce a series of pulses as the shaft rotates. Each pulse represents a small increment of rotation. They don't give absolute position information but instead track the change in position relative to a reference point [36].

An experiment was performed with the rotary encoder attached to the trailer linkage in order to determine whether it was able to read the changes in angular position. When performing this experiment, a program was made to determine how the encoder would determine angular motion. The program would determine the movement of the rotary encoder's shaft using steps as a form of digital measurement. This would then be converted to angles, and the information would be used to produce a graph. The results of the experiment can be seen in the graph below.

Figure 23: Graph of Rotary Encoder Angular Position

As depicted in Figure 23, the rotary encoder displayed a movement increment of approximately 15 degrees. If the linkage is connected to the system, it can provide a reading up to a maximum of 30 degrees and a minimum of -30 degrees. This aligns perfectly with the angular movement of the linkage that it is attached to as it means that the system can detect a range of angular motion within that specified limit. The code for the program used can be found in APPENDIX A.2—Rotary Encoder Experiment Program.

4.4 SENSOR DATA INTEGRATION

Simultaneous Localization and Mapping, or SLAM, is a computational problem in robotics and computer vision that involves building a map of an unknown environment while simultaneously keeping track of the robot's position within that environment. Essentially, SLAM

allows a robot to navigate and understand its surroundings without prior knowledge of the environment's layout. SLAM consists of two main processes: Localization and Mapping. Localization involves determining the robot's position within its environment, while mapping involves making a representation of the robot's environment [34].

To achieve this, the method of SLAM used was LIDAR SLAM. The output values from the LIDAR would be in 2D or 3D point cloud data. Movement is estimated sequentially by registering point clouds. The calculated movement is used for localizing the vehicle. On the other hand, the IMU also performs localization by measuring and tracking the movement of an object in space based on its acceleration and angular velocity. Furthermore, the data acquired by the IMU can be used to refine the data from the LIDAR by filtering out errors caused by motion, such as motion blur. After this, the relevant features are extracted from the sensor data to be used for mapping and localization. These features could be critical points in images, such as geometric features from LIDAR scans. Then, the extracted features are matched with existing map features or determine if they represent new features in the environment. This step is essential for correctly updating the map and estimating the robot's position and orientation or pose. Estimating the robot's pose based on sensor data and the current map. This involves integrating sensor measurements over time to update the robot's position estimate. Finally, updating the map with new information obtained from sensor data. This includes adding new landmarks, updating the positions of existing landmarks, and refining the map's structure. Figure 24 provides a flowchart to explain this process.

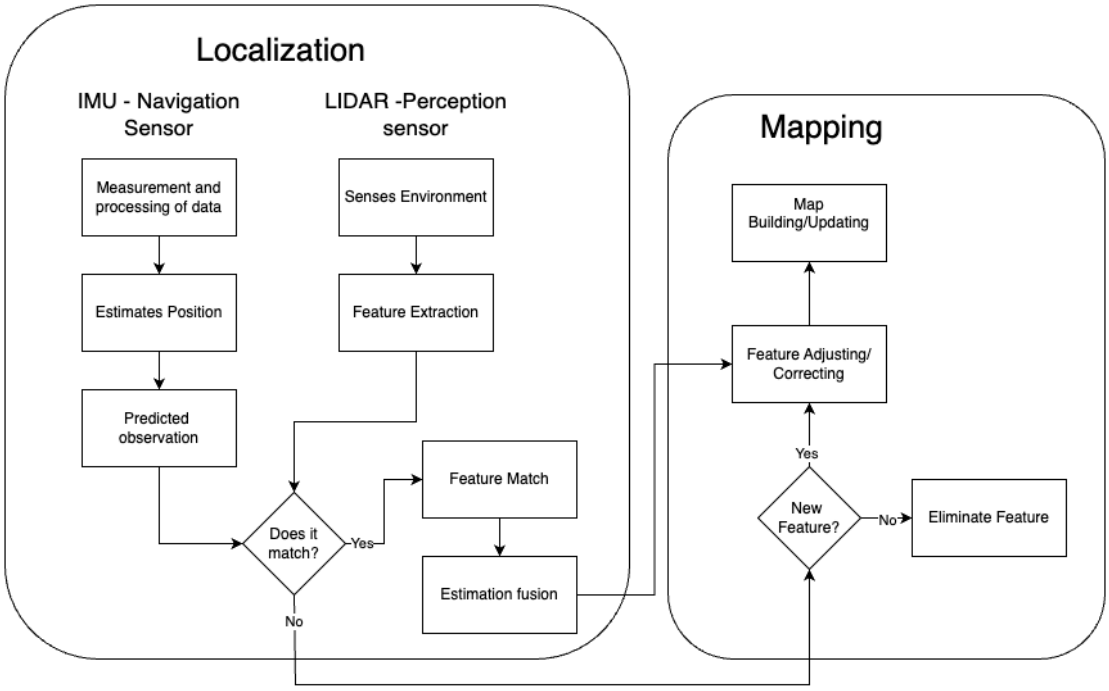


Figure 24: Flowchart of SLAM Process

The process of localization and mapping can be greatly improved by using a LIDAR and IMU sensor fusion. This technique helps to enhance the accuracy and robustness of the system. In order to simplify the process of finding the pose of the robot, the x and y axes of the LIDAR and IMU were aligned. This means that when performing the sensor fusion, the z-axis will contain the most information, making it easier to determine the position and orientation of the robot in 3D space. This is particularly useful in applications such as autonomous vehicles and drones, where precise localization and mapping are essential for safe and effective operation. As such it was crucial for programming a successful autonomous truck and trailer system.

CHAPTER 5 - NATURAL LANGUAGE PROCESSING

5.1 IMPLEMENTATION OF NLP FOR COMMAND INTERPRETATION

The implementation of Natural Language Processing (NLP) for command interpretation within this autonomous truck and trailer system represents a step toward realizing an intuitive interaction model between humans and machines. This approach enables the system to understand and execute complex navigational commands in natural language, thereby eliminating the traditional barriers posed by rigid command interfaces. At the core of this framework is an algorithm designed to parse and interpret user inputs, transforming them into actionable data that the autonomous navigation system can understand and act on.

To achieve this, the project combines regular expressions and advanced parsing techniques to analyze the user's input commands. This process begins with the extraction of numerical values and directional instructions from the natural language commands using regular expressions. The algorithm identifies and isolates key pieces of information, such as coordinates (x,y) and orientation angles, embedded within the sentences. This capability is crucial for allowing users to specify destinations and navigation paths in a manner that is both natural and intuitive.

Following the extraction phase, the parsed command data undergoes a validation process to ensure its integrity and applicability for navigation purposes. This step is vital to mitigate errors and enhance the system's reliability in real-time scenarios. Once validated, the command is converted into a structured format that can be compatible with the autonomous system's navigation module. This command data is then communicated directly to the system's movement and path-planning algorithms, enabling the vehicle to execute the navigational task with precision and fidelity.

The NLP system design emphasizes scalability and adaptability for future enhancements, including the integration of more complex linguistic models and the accommodation of diverse languages and dialects. Moreover, the implementation is engineered to operate efficiently within the ROS (Robot Operating System), with custom ROS nodes and topics for continuous data exchange between the NLP interface and the autonomous navigation system. Specifically, the project employs a Flask-based server as an intermediate, which serves as the platform for receiving user commands and publishing them to a dedicated ROS topic. This design ensures

that the NLP system is not only robust and responsive but also easily integrated with other components of the autonomous driving framework.

5.2 DESIGN OF THE USER INTERFACE FOR COMMAND INPUT

The user interface (UI) for command input in the autonomous truck and trailer navigation system is designed to ensure interaction between the user and the system, leveraging Natural Language Processing capabilities. The core of this interaction model is a web-based application developed using Flask [38], a lightweight WSGI web application framework, as shown in Figure 25.

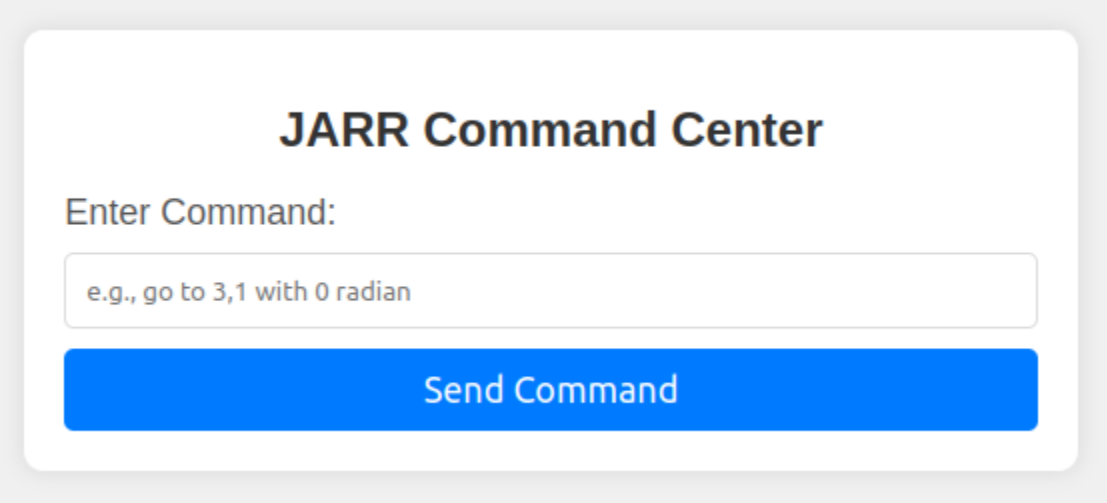


Figure 25: Flask Server User Interface

Flask's simplicity and flexibility make it an ideal choice for creating a user-friendly interface that can process and interpret natural language commands.

The UI workflow begins at 'index.html,' the primary interface where users can input their commands in natural language form. This page is designed for clarity and ease of use, featuring a simple text input field where users can type commands such as "Navigate to coordinates X:10, Y:20, with an orientation of 30 degrees." Upon submitting a command, the Flask server processes the request and extracts the relevant navigation parameters using the NLP techniques outlined in the project.

After successful extraction and parsing of command parameters, the Flask application routes the user to 'success.html,' a webpage designed to acknowledge the successful processing of the command and provide feedback on the action initiated by the system. This feedback

mechanism is crucial for user assurance and system transparency, allowing users to verify whether their instructions have been understood correctly.

In cases where the command input does not meet the expected format or crucial information is missing, the Flask application redirects users to a failure page. This page informs users of the issues and allows users to try again. The failure handling ensures that users are supported in correcting their inputs and are educated on the system's requirements for successful command interpretation.

The entire pipeline, from command input through 'index.html' to feedback via 'success.html' or 'error.html,' shows a user-centric design philosophy. This approach facilitates an intuitive interaction with autonomous navigation and handles the complexity of the NLP and navigation processes, making advanced functionality accessible to users without technical expertise in robotics or programming.

Appendix A lists Related Codes, including "index.html," "success.html," and "error.html."

5.3 INTEGRATION WITH THE SYSTEM NAVIGATION MODULE

Integrating the user interface designed on the Flask server with the autonomous system's navigation module shows a refined process that connects the natural language commands to precise robotic navigation. This integration transforms user input sentences into specific navigational actions by the autonomous truck and trailer system.

When a user inputs a navigation command through the Flask web interface, employing natural language to denote their intended destination. This command is then processed by the Flask server, where NLP techniques are used to extract critical parameters such as desired coordinates and orientation angles. This phase involves dissecting the natural language input to identify and isolate numerical values and direction, converting an unstructured command into a structured dataset interpretable by robotic systems.

After this extraction, the command's structure data is formatted into a JavaScript Object Notation (JSON) object and then published onto a ROS topic specifically for navigation instructions. This publication step is done by a ROS node with the Flask application, acting between the web interface and the ROS system. Upon this data being broadcasted over the ROS topic, a robust ROS navigation stack activates for autonomous movement.

The primary system of the navigation module uses global planner ‘navfn,’ which uses Dijkstra’s algorithm for computing the optimal path to the target based on the map generated through the SLAM process with the ‘gmapping’ algorithm. This map is dynamically updated with real-time data from LIDAR and IMU sensors for accurate localization and obstacle mapping. At the same time, the DWA algorithm serves as the local planner to calculate the vehicle’s immediate movements to ensure it follows the global path while dynamically adjusting for obstacles and kinematic constraints specific to the truck-trailer configuration.

As the vehicle reaches the set goal, its navigation stack continuously processes sensor inputs to refine the vehicle’s trajectory and maneuver around obstacles to make sure that the planned route is both feasible and safe. Any updates or necessary route adjustments are communicated back through the ROS framework to give real-time feedback. This bi-directional communication keeps the user informed of the system’s status and shows the system’s ability to navigate complex environments autonomously.

CHAPTER 6 - SIMULATION

6.1 SIMULATION ENVIRONMENT SETUP

The simulation environment setup for the autonomous truck and trailer system employs the Turtlebot3 platform [39], renowned for its wide acceptance within the robotics research community. TurtleBot3 serves as an ideal model for simulating the complex dynamics of autonomous vehicles, compatibility with various sensors, and support for advanced robotic functionalities. This simulation leverages the TurtleBot3 model as the foundation agent for the study of autonomous navigation algorithms in a controlled environment.

The simulation uses the Robot Operating System (ROS) navigation stack, which is for path planning and localization. The navigation stack works with the TurtleBot3 platform to operate simulations of sophisticated navigational tasks. There are several key components, including the ‘move_base’ node, which interacts between global and local planners to navigate the robot toward a designated goal while avoiding obstacles.

Gazebo [40], an open-source 3D robotics simulator, is used to simulate the physical environment. It provides a realistic setting for testing the robot’s navigational capabilities with detailed physics and high-fidelity sensor emulation. Within Gazebo, the robot is placed in various environments that are very similar to the real-world conditions ranging from urban landscapes to indoor environment settings. These environments are made with static and dynamic obstacles to challenge the navigation ability to guide the robot efficiently.

The integration of the TurtleBot3 with the Ros navigation stack using the Gazebo simulation environment is important for several reasons. First, this setting allows for the validation of planning and navigation algorithms under various conditions without the risks and costs. Second, this facilitates the iterative development of tuning the algorithms, where parameters can be adjusted, and their impacts can be immediately observed and analyzed. Lastly, performance in executing commands interpreted through Natural Language Processing (NLP), specifically focusing on how the robot interprets these commands within the spatial context of its simulated environment, can be studied.

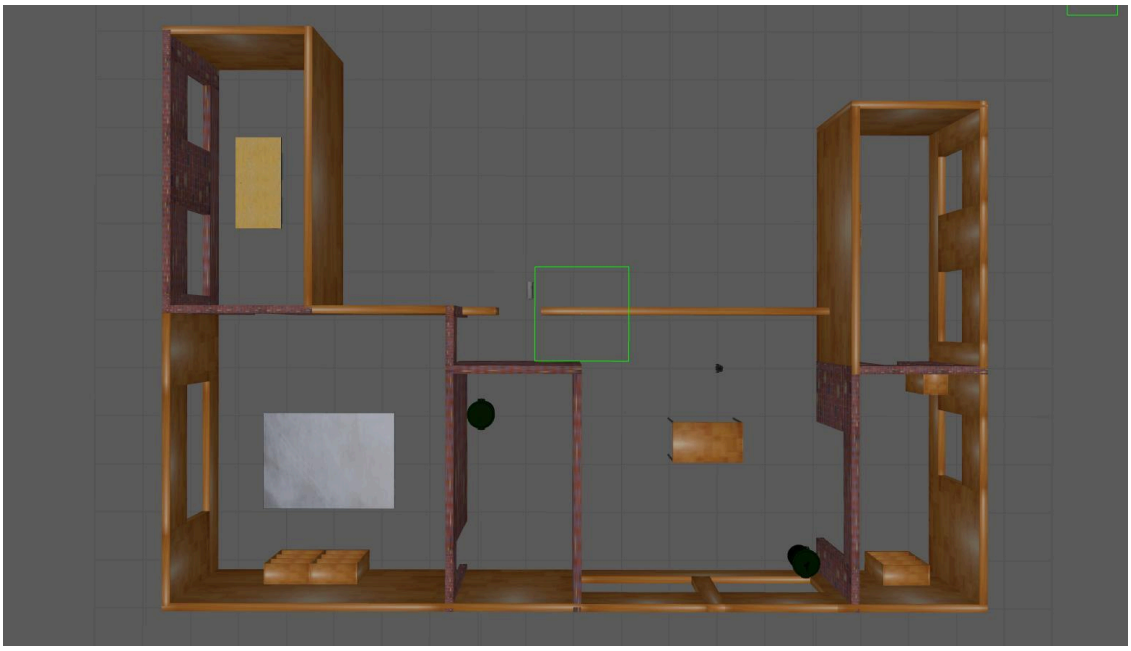


Figure 26: Gazebo Simulation Environment

6.2 PLANNING AND NAVIGATION ALGORITHMS

The planning and navigation algorithms are important components that determine the system's efficacy in dynamic environments. These algorithms are categorized into two main types: global planners and local planners. The global planner employs Dijkstra's algorithm to compute the most efficient path from the robot's current position to its designated goal. This planner works by systemically exploring all possible paths to find the one that minimizes the cumulative cost from the start to the goal. This cost can factor in various elements such as distance, time, or energy required to go through the path. Dijkstra's algorithm guarantees the discovery of the optimal path if one exists. However, its computational intensity can be a drawback, especially for large maps or in situations where we need immediate response due to dynamic changes in the environment.

On the other hand, the local planner DWA operates on a more immediate scale. It takes the global path as input and dynamically adjusts the robot's trajectory in real time, responding to unforeseen obstacles or other changes in the environment. DWA takes the robot's current velocity and all possible velocities within a short timeframe known as the "dynamic window" with constraints from the robot's kinematics and the environment. For each possible velocity, DWA predicts the robot's trajectory and evaluates it based on criteria such as proximity to

obstacles. This allows DWA to dynamically select the optimal velocity that maximizes path following and obstacle avoidance.

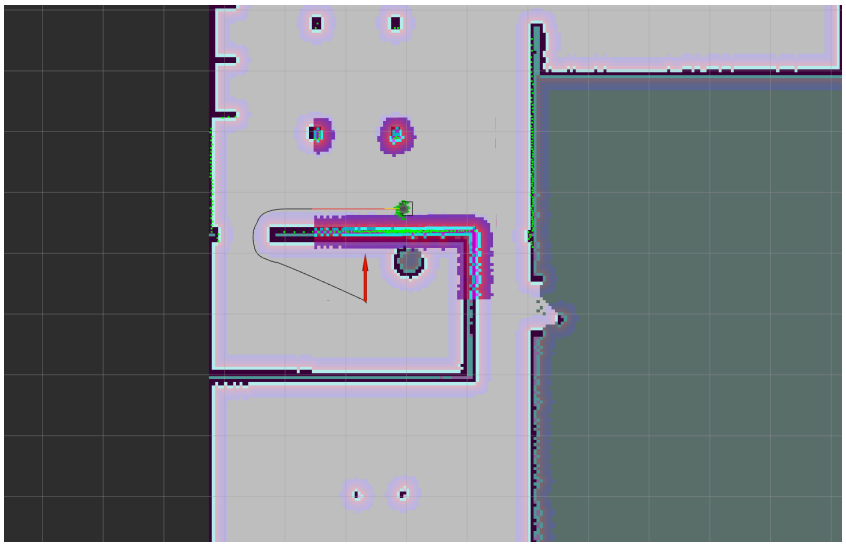


Figure 27: Global and Local Planner

Integrating Dijkstra's algorithm and DWA for the autonomous navigation system gives a comprehensive solution to the challenges of autonomous mobility. Dijkstra's algorithm provides a reliable method for long-term path planning, allowing the robot to find its way in complex environments. DWA ensures that the robot can safely and efficiently navigate its immediate surroundings, making second adjustments as needed to avoid collisions and maintain smooth motion. This makes the truck and trailer system navigate with precision and adaptability.

CHAPTER 7 - EXPERIMENT

7.1 EXPERIMENTAL SETUP

The transition from simulation to real-world experiment requires an experimental setup that mirrors the controlled conditions of the Gazebo simulator while navigating the physical environments. Similar to the simulation, the project uses ROS navigation stack to conduct hardware experiments. The autonomous truck and trailer system uses custom-made hardware components with the LIDAR and IMU.

The main goal of this experiment setup is to validate the autonomous navigation algorithms developed and refined within the simulation environment. To achieve this, a controlled environment is constructed using boxes and trash bins to mimic the static obstacles in indoor navigation scenarios. The layout is designed to test the truck's capabilities in path planning, obstacle avoidance, and the use of navigation commands interpreted through NLP.

Figure 28: Hardware Experiment Map

The creation of a real-world map converted to a digital map is achieved through a manual drive of the truck around the experimental environment utilizing the SLAM techniques called "gmapping," which will be deeply discussed in the next section. This process ensures that the

navigation stack has an accurate representation of the environment, including the location and dimensions of obstacles, which is crucial for effective path planning.

Once the environment map is generated, the truck-trailer system is tasked with executing navigation commands. Commands are input through the Flask-based NLP interface with the robot's responses and movements observed and recorded. The focus is on validating how well the global planner (Dijkstra's algorithm) and the local planner (DWA), navigate to specified destinations while avoiding obstacles.

The experimental setup also includes a data collection and analysis framework. Sensors on the truck and trailer continuously gather data on the robot's position, movement, and interactions with obstacles. This data is used to compare the robot's performance in the real world against the simulated environments.

7.2 DATA COLLECTION

Accurate environmental mapping is critical to effective path planning. Therefore, this project employs the gmapping algorithm, a technique in SLAM for generating a static map of the experimental environment. This approach is particularly practical in creating detailed maps even with minimal sensor data, which is ideal for the initial phase of real-world experiments.

Gmapping integrates data from the truck's LIDAR and IMU sensors to construct a comprehensive map of the surroundings incrementally. LIDAR sensors provide precise distance measurements to surrounding objects, giving the image of the environment's layout, including walls, obstacles, and open spaces. At the same time, IMU gives specific orientation and acceleration data to accurately track the robot's movement and position within the map. Utilizing these sensors and the gmapping algorithm allows for the dynamic response with the high-fidelity representation of the static environment.

The gmapping algorithm operates on the principle of particle filters where the hypotheses about the robot's position are continuously updated based on incoming sensor data. Each particle represents a potential pose of the robot within the map, with sensor readings used to evaluate the likelihood of each pose. Over time, as more data is collected, particles that consistently align with sensor observations become more probable and output the true layout of the environment. This probabilistic approach gives gmapping to deal with the uncertainties efficiently and to provide a robust and accurate map.

Once the map is generated, this static map is fed into the system for an autonomous truck to have an immediate understanding of its surroundings. The map informs the global planner in determining the optimal path to a destination while the local makes real-time adjustments for obstacle avoidance and trajectory optimization. As the truck navigates the environment, sensor data regarding its interaction with the map, such as deviations from planned paths and efficiency of path execution, are collected and analyzed. This data collection process ultimately enhances the system's capability to navigate complex real-world environments with precision and reliability.

CHAPTER 8 - DISCUSSION AND CONCLUSIONS

8.1 IMPACT ON AUTONOMOUS TRANSPORTATION

The integration of Natural Language Processing and advanced sensor fusion within an autonomous truck and trailer system gives significant advancements in autonomous transportation. This project not only demonstrates the feasibility of interpreting and executing complex navigational commands provided in natural language but also shows enhanced environmental awareness through sophisticated sensor technologies. The successful implementation of these technologies offers a step toward the future of autonomous logistics and urban mobility.

The use of NLP as a command interpretation makes autonomous systems accessible and user-friendly. By allowing operators to communicate with vehicles using natural language, this shift to more intuitive human-machine interfaces could accelerate the development of autonomous vehicles across various sectors. The implications for efficiency and safety are profound. With operators able to give complex instructions effortlessly, vehicles can navigate more effectively, reducing misunderstanding issues.

Moreover, the project's focus on the rotary encoder for the truck-trailer system is essential to perceive and understand its environment fully. By knowing the relative pose between the truck and the trailer, the truck system can have optimal path planning considering the overall length and the turning angle of the vehicle. With the sensor fusion between LIDAR and IMU, the system can accurately identify obstacles and make optimal routes for safety and efficiency.

The experimental validation through simulations and real-world testing shows their potential to transform autonomous transportation. The findings show that with further development and refinement, vehicles equipped with NLP and advanced sensor fusion can achieve a level of operational flexibility and reliability. These vehicles will enhance logistical operations and contribute to safer roads by reducing human error.

8.2 CONCLUSIONS

This project fully demonstrates the integration of Natural Language Processing (NLP) and advanced sensor fusion into a custom-designed autonomous truck and trailer system. The development and deployment of custom hardware for both the truck and trailer, supported by

comprehensive Finite Element Analysis (FEA), have set a robust foundation for the system's mechanical integrity and reliability. The incorporation of a rotary encoder for the trailer has also been successful in acquiring the relative pose of the trailer which is a critical factor for optimizing path planning to ensure the system's adaptability to real-world challenges.

The implementation of NLP for interpreting and executing navigation commands shows human-machine interaction within the autonomous vehicle domain. This method enhanced the system's accessibility and efficiency of the navigation stack by providing a direct translation of verbal instructions into navigation tasks.

The project's simulation and real-world experiment provided insights into the system's performance under various scenarios. Simulations conducted in the Gazebo environment facilitated an evaluation of the navigation algorithms, sensor capabilities, and NLP integration in a controlled setting. Subsequently, the hardware experiment in a physical environment validated the system's effectiveness in real-world applications. The project has achieved a combination of custom hardware development, innovative software integration, and rigorous testing to advance autonomous transportation. By addressing the unique challenges associated with autonomous truck and trailer systems, including dynamic obstacle avoidance, precise environment mapping, and user-friendly command interpretation.

CHAPTER 9 - LIMITATIONS AND FUTURE WORKS

9.1 LIMITATIONS

Despite integrating NLP for command interpretation in autonomous transportation systems, several limitations have been identified, primarily rooted in the semantic understanding capabilities of NLP and the general drawbacks of NLP in complex operational contexts. One of the main challenges is the limitation of current NLP technologies to fully understand the semantic environment for higher precision and clarity. Another limitation comes from the general drawbacks associated with NLP in processing highly technical language that may be used in the context of autonomous driving. The variability in linguistic expression among users, including dialects, slang, and terminology specific to navigation and driving, creates a challenge for NLP systems to interpret commands without extensive training accurately.

Furthermore, the project faced specific challenges in executing complex controls, such as back parking of the truck with an attached trailer. The kinematic complexity and precision required for such controls exceed the current capabilities of the NLP-driven command interpretation framework and the system's path-planning algorithms. The difficulty in accurately executing commands for back parking shows a critical area where the system falls short in guiding the vehicles with high levels of spatial awareness and precision.

9.2 FUTURE WORKS

Addressing the limitations in the current project shows several future works, particularly in the fields of large language models (LLMs) and Vision Language Models (VLMs). These AI technologies hold the potential to enhance the semantic understanding capabilities of autonomous transportation systems significantly.

First, future work will explore the integration of LLMs such as GPT (Generative Pre-trained Transformer) into the autonomous navigation system. LLMs are known for their superior natural language understanding and generation capabilities. Training these models on domain-specific datasets will be crucial to bridge the semantic gap identified in the current NLP implementation.

Next, the combination of visual data with NLP through VLMs presents a direction for overcoming the limitation with spatial controls such as back parking. VLMs can provide a richer dataset for the autonomous system to make more informed navigation decisions. This could

enhance the system's understanding of spatial relationships and environmental context for more precise controls.

Lastly, to address the variability in linguistic expression and improve the system's adaptability to different dialects and specialized languages, expanding the training datasets for the NLP models will be necessary. Future work will focus on collecting and incorporating a broader range of linguistic inputs, including regional dialects and industry-specific terminologies to enhance the model's accuracy and reliability.

REFERENCES

---------- Literature Review

[1] SAE International. "*Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*" SAE Standard J3016, 2021.

[2] Dickmanns, Ernst D. "The development of machine vision for road vehicles in the last decade." *Intelligent Vehicle Symposium, 2002. IEEE*. Vol. 1. IEEE, 2002.

[3] Williams, M. "PROMETHEUS-The European research programme for optimising the road transport system in Europe." *IEE Colloquium on Driver Information*. IET, 1988.

[4] Buehler, Martin, Karl Iagnemma, and Sanjiv Singh, eds. *The DARPA urban challenge: autonomous vehicles in city traffic*. Vol. 56. springer, 2009.

[5] Thrun, Sebastian. "Toward robotic cars." *Communications of the ACM* 53.4 (2010): 99-106.

[6] Levinson, Jesse, et al. "Towards fully autonomous driving: Systems and algorithms." *2011 IEEE intelligent vehicles symposium (IV)*. IEEE, 2011.

[7] Anderson, James M., et al. *Autonomous vehicle technology: A guide for policymakers*. Rand Corporation, 2014.

[8] Fagnant, Daniel J., and Kara Kockelman. "Preparing a nation for autonomous vehicles: opportunities, barriers and policy recommendations." *Transportation Research Part A: Policy and Practice* 77 (2015): 167-181.

[9] "Steering (differential steering, rack & pinion steering and front vs. rear wheel steering)." University of Florida. Accessed: Feb. 16, 2024. [Online]. Available: https://mae.ufl.edu/designlab/class%20projects/background%20information/steering.htm

[10] M. Crenganis, C. Biris, and C. Girjob, "Mechatronic Design of a Four-Wheel drive mobile robot and differential steering," in MATEC Web of Conferences, Les Ulis, France: EDP Sciences, 2021. doi: 10.1051/matecconf/202134308003.

[11] C. Wang *et al.*, "Simulation and Validation of a Steering Control Strategy for Tracked Robots," *Applied Sciences*, vol. 13, no. 19, 2023, doi: 10.3390/app131911054.

[12] V. Kolluru, K. Nikhil, K. Theja, K. Sravan, and K. Kumar, "Design and Fabrication of Ackerman Steering Mechanism Combining with ABS," pp. 2321–0613, Jan. 2016.

[13] J.-S. Zhao, Z.-J. Liu, and J. Dai, “Design of an Ackermann Type Steering Mechanism,” *Journal of Mechanical Engineering Science*, vol. 227, Nov. 2013, doi: 10.1177/0954406213475980.

[14] D. J. Yeong, G. Velasco-Hernandez, J. Barry, and J. Walsh, “Sensor and Sensor Fusion Technology in autonomous vehicles: A Review,” *Sensors*, vol. 21, no. 6, p. 2140, Mar. 2021. doi:10.3390/s21062140.

[15] X. Meng, H. Wang, and B. Liu, “A robust vehicle localization approach based on GNSS/IMU/DMI/Lidar Sensor Fusion for Autonomous Vehicles,” *Sensors*, vol. 17, no. 9, p. 2140, Sep. 2017. doi:10.3390/s17092140.

[16] L. Chang, X. Niu, and T. Liu, “GNSS/IMU/Odo/LIDAR-SLAM Integrated Navigation System using IMU/Odo Pre-Integration,” *Sensors*, vol. 20, no. 17, p. 4702, Aug. 2020. doi:10.3390/s20174702

[17] G. Kumar, A. Patil, R. Patil, S. Park, and Y. Chai, “A LIDAR and IMU Integrated Indoor Navigation System for uavs and its application in real-time pipeline classification,” *Sensors*, vol. 17, no. 6, p. 1268, Jun. 2017. doi:10.3390/s17061268

[18] Putri, Tsarina Dwi. "Intelligent transportation systems (ITS): A systematic review using a Natural Language Processing (NLP) approach." *Heliyon* 7.12 (2021).

[19] Duffhauß, Fabian. *Deep Sensor Data Fusion for Environmental Perception of Automated Systems*. Diss. Universität Tübingen, 2024.

[20] Oterino-Bono, Roberto, et al. "Deep Learning Methods Integration for Improving Natural Interaction Between Humans and an Assistant Mobile Robot in the Context of Autonomous Navigation." *EPIA Conference on Artificial Intelligence*. Cham: Springer International Publishing, 2022.

[21] Cui, Can, et al. "A survey on multimodal large language models for autonomous driving." *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2024.

[22] Chen, Hong, et al. "Feedback is all you need: from ChatGPT to autonomous driving." *Science China Information Sciences* 66.6 (2023): 1-3.

[23] F. Sorge, “Optimization of vehicle-trailer connection systems,” J. Phys.: Conf. Ser., vol. 744, p. 012209, Sep. 2016, doi: 10.1088/1742-6596/744/1/012209.

[24] H. Choset, Principles of Robot Motion Theory, Algorithms, and Implementation. Bradford Book.

[25] A. M. Shkel and V. Lumelsky, “Classification of the Dubins set,” Robotics and Autonomous Systems, vol. 34, no. 4, pp. 179–202, 2001, doi: https://doi.org/10.1016/S0921-8890(00)00127-5.

[26] J. Tarighi, H. R. Ghasemzadeh, M. Bahrami, S. Abdollahpour, and A. Mahmoudi, “Optimization of a Lower Hitch Link for a Heavy Duty Tractor Using Finite Element Method,” *Journal of Failure Analysis and Prevention*, vol. 16, no. 1, pp. 123–128, Feb. 2016, doi: 10.1007/s11668-015-0054-1/

[27] L. Cheng, R. Nianzu, Y. Qingsong, C. Chao, T. Cheng, and L. Zhengguo, “Stress State Measurement and Result Analysis of Car Wheels,” *IOP Conference Series: Materials Science and Engineering*, vol. 784, no. 1, p. 012021, Mar. 2020, doi: 10.1088/1757-899X/784/1/012021.

[28] “Basics of Load Analysis”, “Loads for Durability”, “Response of Mechanical Systems” in *Guide to Load Analysis for Durability in Vehicle Engineering*, John Wiley & Sons, Ltd, 2013, pp. 15-29, 31–106, 169-201. doi: https://onlinelibrary.wiley.com/doi/book/10.1002/9781118700518

[29] Kocić, Jelena, Nenad Jovičić, and Vujo Drndarević. "Sensors and sensor fusion in autonomous vehicles." *2018 26th Telecommunications Forum (TELFOR)*. IEEE, 2018

[30] Berk, Mario, et al. "Exploiting redundancy for reliability analysis of sensor perception in automated driving vehicles." *IEEE Transactions on Intelligent Transportation Systems* 21.12 (2019): 5073-5085.

[31] Jahn, Uwe, Carsten Wolff, and Peter Schulz. "Concepts of a modular system architecture for distributed robotic systems." Computers 8.1 (2019): 25.

[32] Zhao, Andrew, et al. "Expel: Llm agents are experiential learners." arXiv preprint arXiv:2308.10144 (2023).

[33] N. O. and A. A. US Department of Commerce, “What is Lidar,” NOAA’s National Ocean Service,

https://oceanservice.noaa.gov/facts/lidar.html#:~:text=Lidar%2C%20which%20stands%20for%20Light,variable%20distances)%20to%20the%20Earth. (accessed Apr. 4, 2024).
[34] “What is SLAM (simultaneous localization and mapping) – matlab & simulink,” What Is SLAM (Simultaneous Localization and Mapping) – MATLAB & Simulink - MATLAB & Simulink, https://www.mathworks.com/discovery/slam.html#section-2b (accessed Apr. 4, 2024). (MATLAB, What is SLAM (simultaneous localization and mapping) – matlab & simulink)

[35] “What is an inertial measurement unit?,” VectorNav, https://www.vectornav.com/resources/inertial-navigation-articles/what-is-an-inertial-measurement-unit-imu?gad_source=1&gclid=Cj0KCQjwn7mwBhCiARIsAGoxjaINOtAsq8tzUuLAgSk6xGQevfn4vRCt0QgoXYyGIVDx-qeDANLFKrcaAugFEALw_wcB (accessed Apr. 4, 2024).
[36]B. Gilbert et al., “How rotary encoder works and how to use it with Arduino,” How To Mechatronics, https://howtomechatronics.com/tutorials/arduino/rotary-encoder-works-use-arduino/ (accessed Apr. 4, 2024).

---Chapter 3 DESIGN AND ANALYSIS--------

[37] vit.budina, “PrintinPlace modular trailer by vit.budina,” MakerWorld, https://makerworld.com/en/models/206012#profileId-286519.

---Chapter 5 NLP--------
[38] Grinberg, Miguel. *Flask web development*. " O'Reilly Media, Inc.", 2018.

---Chapter 6 Simulation

[39] “TURTLEBOT3,” ROBOTIS.US, https://www.robotis.us/turtlebot-3/.

[40] Koenig, Nathan, and Andrew Howard. "Design and use paradigms for gazebo, an open-source multi-robot simulator." *2004 IEEE/RSJ international conference on intelligent robots and systems (IROS)(IEEE Cat. No. 04CH37566)*. Vol. 3. Ieee, 2004.

APPENDIX A.1 - FMEA Table

FAILURE MODES AND EFFECT ANALYSIS WORKSHEET													
ITEM	**FUNCTION**	**POTENTIAL FAILURE MODE**	**Severity**	**POTENTIAL CAUSES OF FAILURE**	**Occurrence**	**CURRENT DETECTION OR PREVENTION METHODS**	**Detection**	**Current RPN**	**SUGGESTED CORRECTIVE ACTIONS TO BE TAKEN**	**Severity (new)**	**Occurrence (new)**	**Detection (new)**	**Updated RPN**
Hitch rotational component	Connects trailer to truck and allows trailer to rotate	Turning beyond maximum allowable amount	10	Print lines being parallel to force	1	Unlikely to occur due to motion parameters being safe	1	10	Manufacture in an alternative way (e.g. print differently or use injection moulding)	10	1	1	10
Axles	Connect wheels to vehicle to allow movement and maintain relative positioning between wheels	Maneuver over bumpy terrain	10	Print lines being parallel to force	1	Maneuver over flat terrain	1	10	Manufacture in an alternative way (e.g. print differently or use injection moulding) Redesign, incorporating a suspension to absorb force	10	1	1	10

APPENDIX A.2 - Rotary Encoder Experiment Program

```
#define CLK 2
#define DT 3
#define SW 4

const int STEPS_PER_REVOLUTION = 24;

int stepCount = 0;
float currentAngle = 0.0;
int currentStateCLK;
int lastStateCLK;
String currentDir ="";
unsigned long lastButtonPress = 0;

void setup() {

  pinMode(CLK, INPUT);
  pinMode(DT, INPUT);
  pinMode(SW, INPUT_PULLUP);
```

```
  Serial.begin(9600);

  lastStateCLK = digitalRead(CLK);

  Serial.println("@PLOT");
  Serial.println("@SIZE=500,400");
  Serial.println("@CH1=Angle vs Time");
}

void loop() {


  currentStateCLK = digitalRead(CLK);


  if (currentStateCLK != lastStateCLK && currentStateCLK == 1){

    if (digitalRead(DT) != currentStateCLK) {
      stepCount--;
      currentDir = "CCW";
    } else {
      stepCount++;
      currentDir = "CW";
    }

    currentAngle = 360.0 * (float(stepCount) / float(STEPS_PER_REVOLUTION));

    Serial.print("Direction: ");
    Serial.print(currentDir);
    Serial.print(" | Step count: ");
    Serial.print(stepCount);
    Serial.print(" | Current Angle: ");
    Serial.println(currentAngle);

    Serial.print("@DATA,1,");
    Serial.println(currentAngle);
  }

  lastStateCLK = currentStateCLK;
```

```
  int btnState = digitalRead(SW);

  if (btnState == LOW) {

    if (millis() - lastButtonPress > 50) {
      Serial.println("Button pressed!");
    }

    lastButtonPress = millis();
  }

  delay(1);
}
```

APPENDIX A.3 - Flask Server Code

```
from flask import Flask, request, jsonify, render_template
import rospy
from std_msgs.msg import String
import json

app = Flask(__name__)

rospy.init_node('flask_command_publisher', anonymous=True)
command_publisher = rospy.Publisher('flask_commands', String, queue_size=10)

@app.route('/')
def home():
    return render_template('index.html')

@app.route('/command', methods=['POST'])
def handle_command():
    command = request.form['command']
    command_publisher.publish(json.dumps(command))
```

```
    return render_template('success.html')


if __name__ == '__main__':
    app.run(debug=True, host='0.0.0.0', port=5000)
```

APPENDIX A.4 - NLP Path Planning Code

```
#!/usr/bin/env python3
import rospy
import actionlib
import json
import re
from move_base_msgs.msg import MoveBaseAction, MoveBaseGoal
from geometry_msgs.msg import PoseStamped, Quaternion
from tf.transformations import quaternion_from_euler
from std_msgs.msg import String

def move_to_position_in_map(x, y, orientation):
    client = actionlib.SimpleActionClient('/move_base', MoveBaseAction)
    client.wait_for_server()

    goal = MoveBaseGoal()
    goal.target_pose.header.frame_id = "map"
    goal.target_pose.header.stamp = rospy.Time.now()
    quaternion = quaternion_from_euler(0, 0, orientation)
    goal.target_pose.pose.position.x = x
    goal.target_pose.pose.position.y = y
    goal.target_pose.pose.orientation = Quaternion(*quaternion)

    client.send_goal(goal)
    client.wait_for_result()

def command_callback(msg):
    parsed_values = parse_command(msg.data)

    if None in parsed_values:
        rospy.logerr("Invalid command")
    else:
        x, y, orientation = parsed_values
```

```
        move_to_position_in_map(x, y, orientation)

def parse_command(command):
    numbers = re.findall(r"[-+]?\d*\.\d+|\d+", command)

    if len(numbers) >= 3:
        x, y, orientation = numbers[:3]
        return float(x), float(y), float(orientation)
    else:
        return (None, None, None)


if __name__ == '__main__':
    rospy.init_node('turtlebot3_command_listener', anonymous=True)
    rospy.Subscriber('flask_commands', String, command_callback)
    rospy.spin()
```

APPENDIX A.5 - Index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Command Center</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            background-color: #f0f0f0;
            display: flex;
            justify-content: center;
            align-items: center;
            height: 100vh;
            margin: 0;
        }
        form {
            background-color: white;
            padding: 20px;
            border-radius: 10px;
            box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
        }
```

```
        h1 {
            color: #333;
            font-size: 24px;
            text-align: center;
        }
        label {
            font-size: 18px;
            color: #555;
        }
        input[type="text"] {
            width: 100%;
            padding: 10px;
            margin: 10px 0;
            border: 1px solid #ddd;
            border-radius: 5px;
            box-sizing: border-box; /* Ensures padding doesn't affect the overall width */
        }
        button {
            width: 100%;
            padding: 10px;
            background-color: #007bff;
            color: white;
            border: none;
            border-radius: 5px;
            cursor: pointer;
            font-size: 18px;
        }
        button:hover {
            background-color: #0056b3;
        }
    </style>
</head>
<body>
    <form action="/command" method="post">
        <h1>Command Center</h1>
        <label for="command">Enter Command:</label>
        <input type="text" id="command" name="command" placeholder="e.g., go to 3,1 with 0
radian">
        <button type="submit">Send Command</button>
    </form>
```

```
</body>
</html>
```

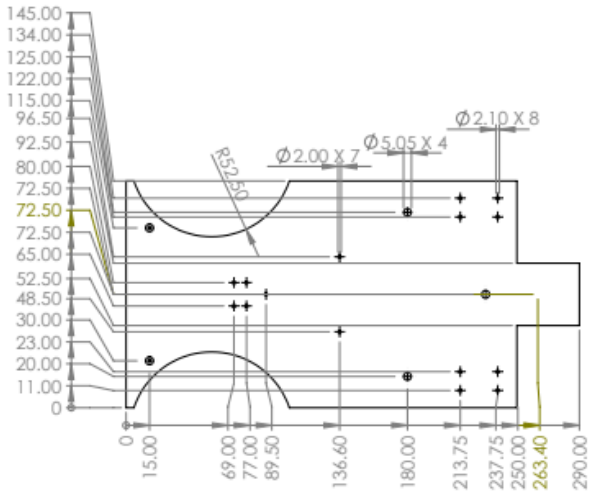
APPENDIX A.6 - Success.html

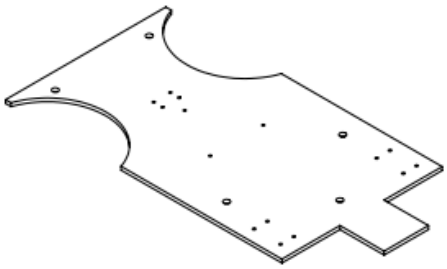
```
<!DOCTYPE html>
<html>
<head>
   <title>Command Success</title>
</head>
<body>
   <h1>Command successfully received!</h1>
   <a href="/">Back to Home</a>
</body>
</html>
```

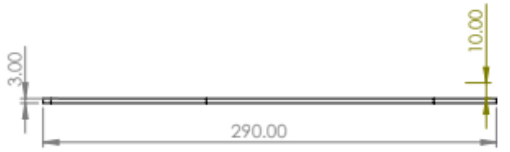
APPENDIX A.7 - Error.html

```
<!DOCTYPE html>
<html>
<head>
   <title>Command Error</title>
</head>
<body>
   <h1>Error processing your command</h1>
   <p>{{ error }}</p>
   <a href="/">Try Again</a>
</body>
</html>
```

APPENDIX A.8 - CAD Drawings



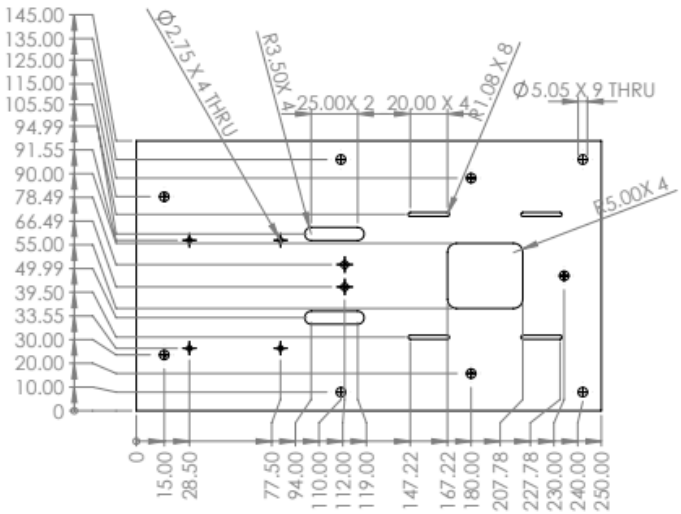


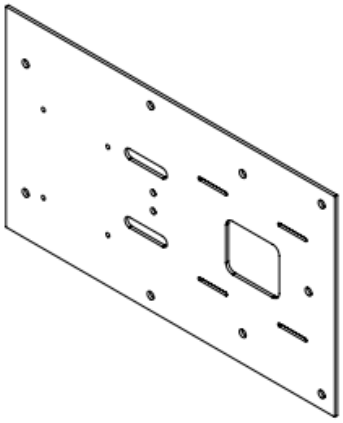


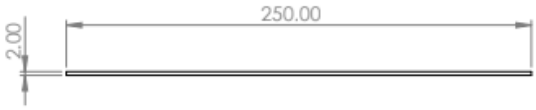
NOTE:
ALL HOLES ARE THROUGH ALL

MEC222 DRAWING TEMPLATE			
NAME	TEAM JARR		
STUDENT ID			
SECTION NO.	DR. SAJAD SAEEDI		
SIZE	SCALE	UNIT	DWG. NAME
A	1:2.5	MM	FIRST FLOOR

5 4 3 2 1



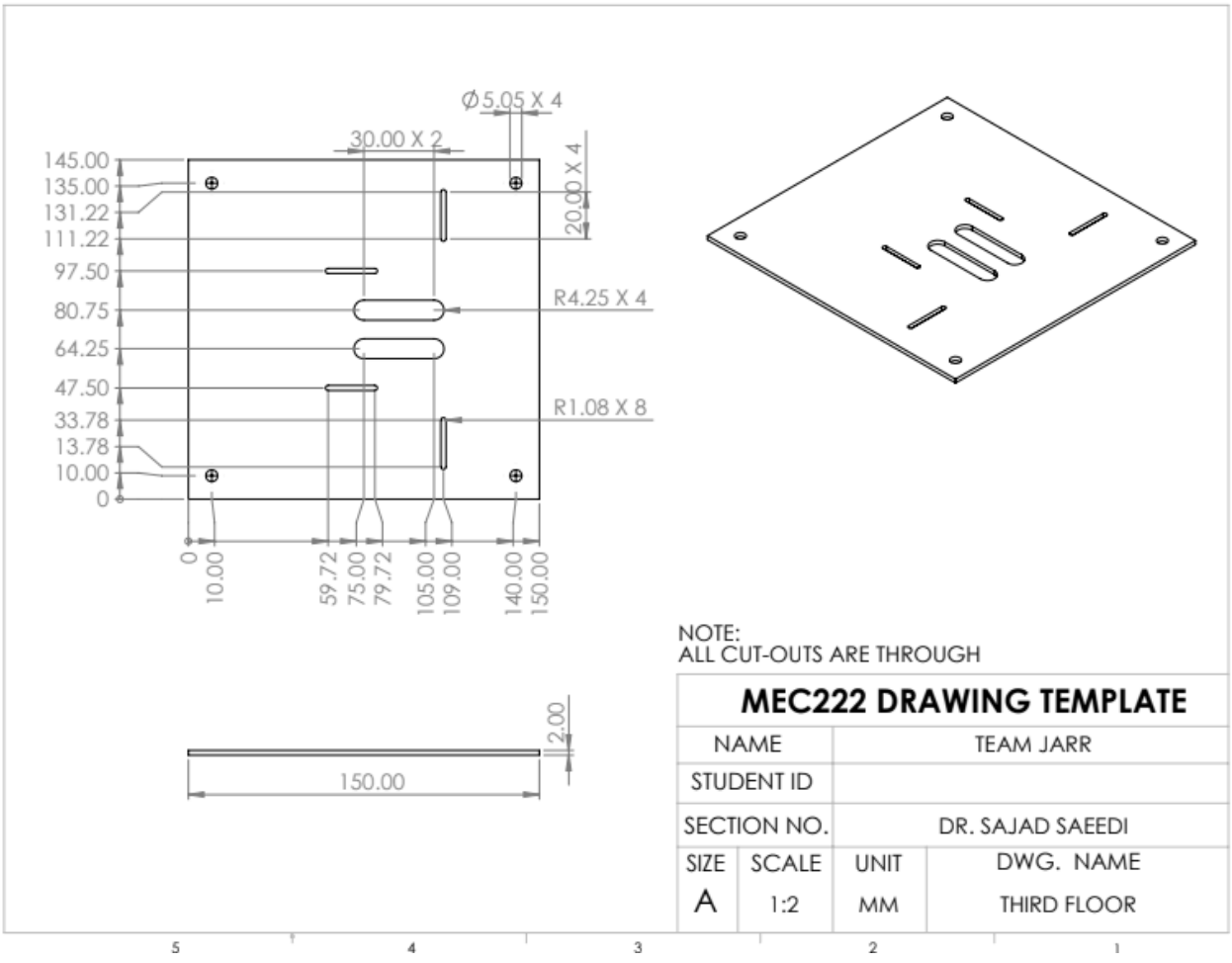


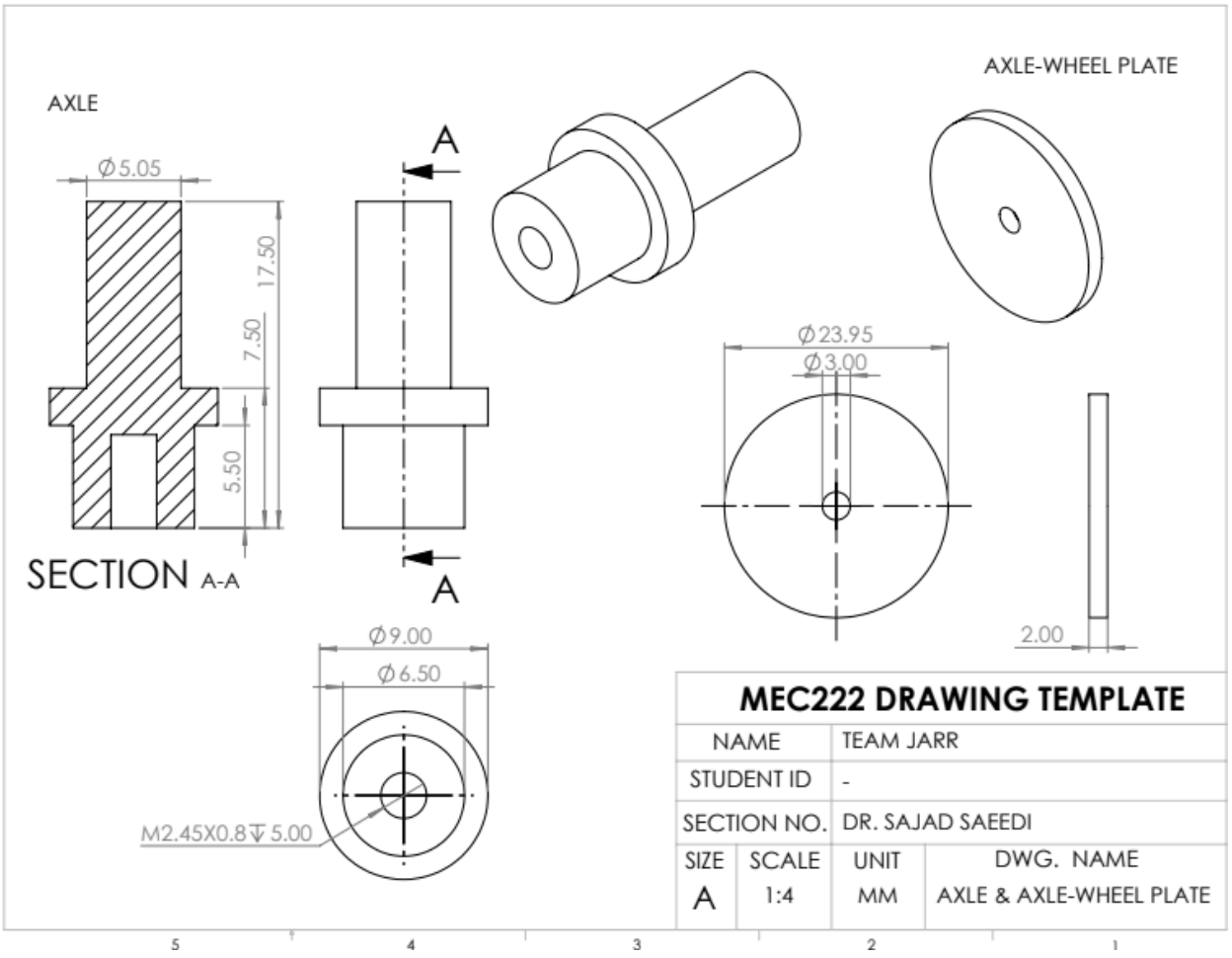


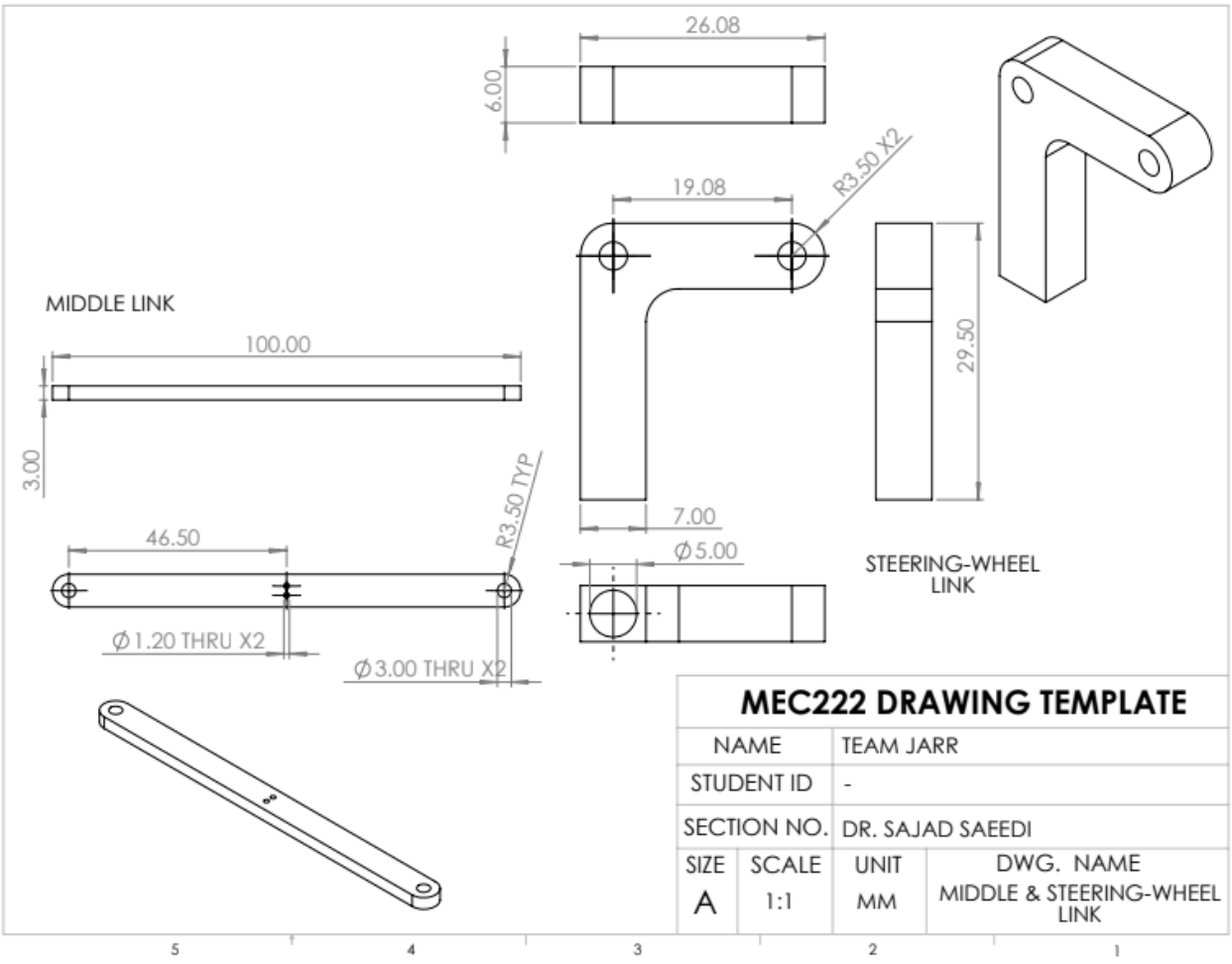
NOTE:
ALL CUT-OUTS ARE THROUGH

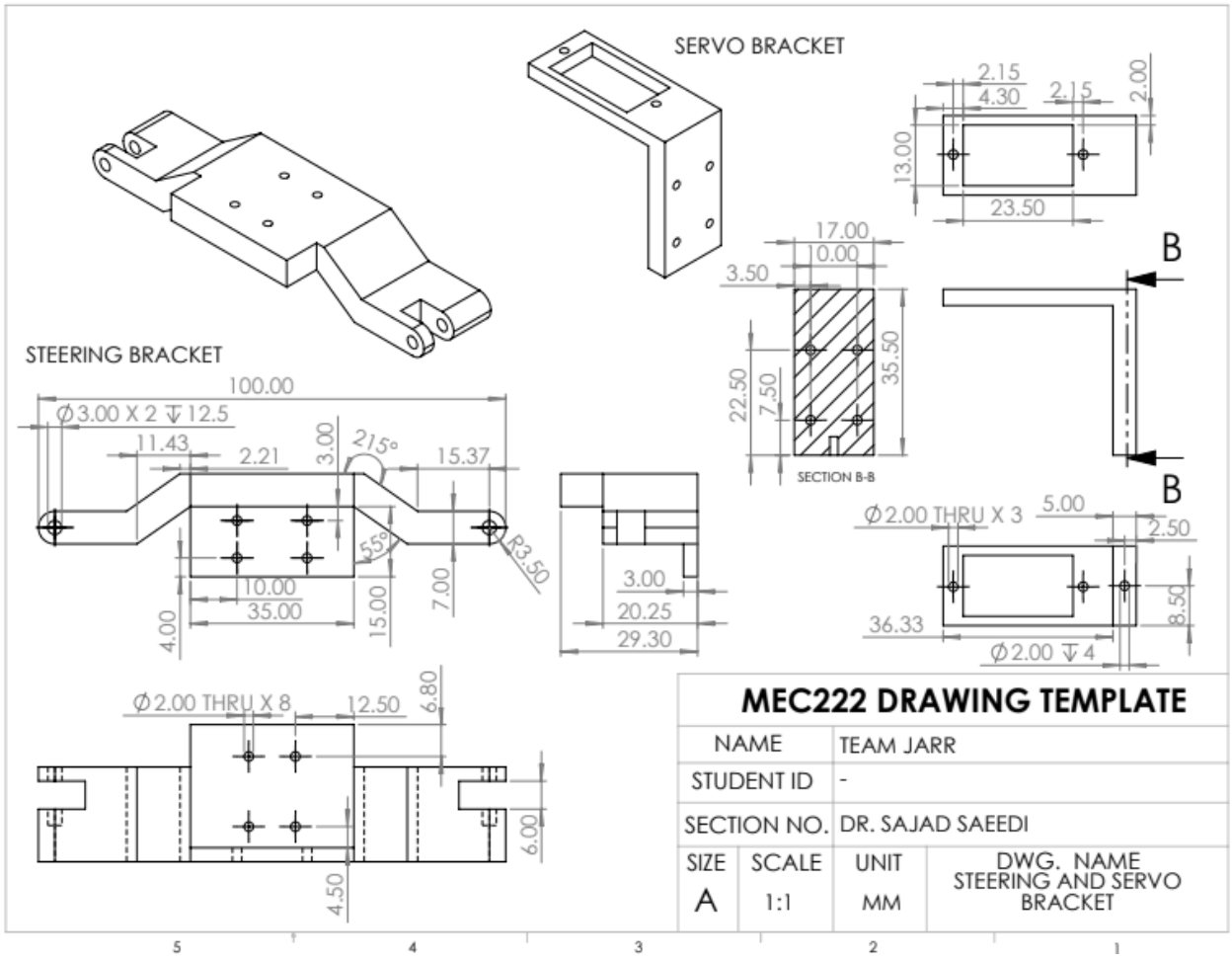
MEC222 DRAWING TEMPLATE			
NAME	TEAM JARR		
STUDENT ID	-		
SECTION NO.	DR. SAJAD SAEEDI		
SIZE	SCALE	UNIT	DWG. NAME
A	2:5	MM	2ND FLOOR

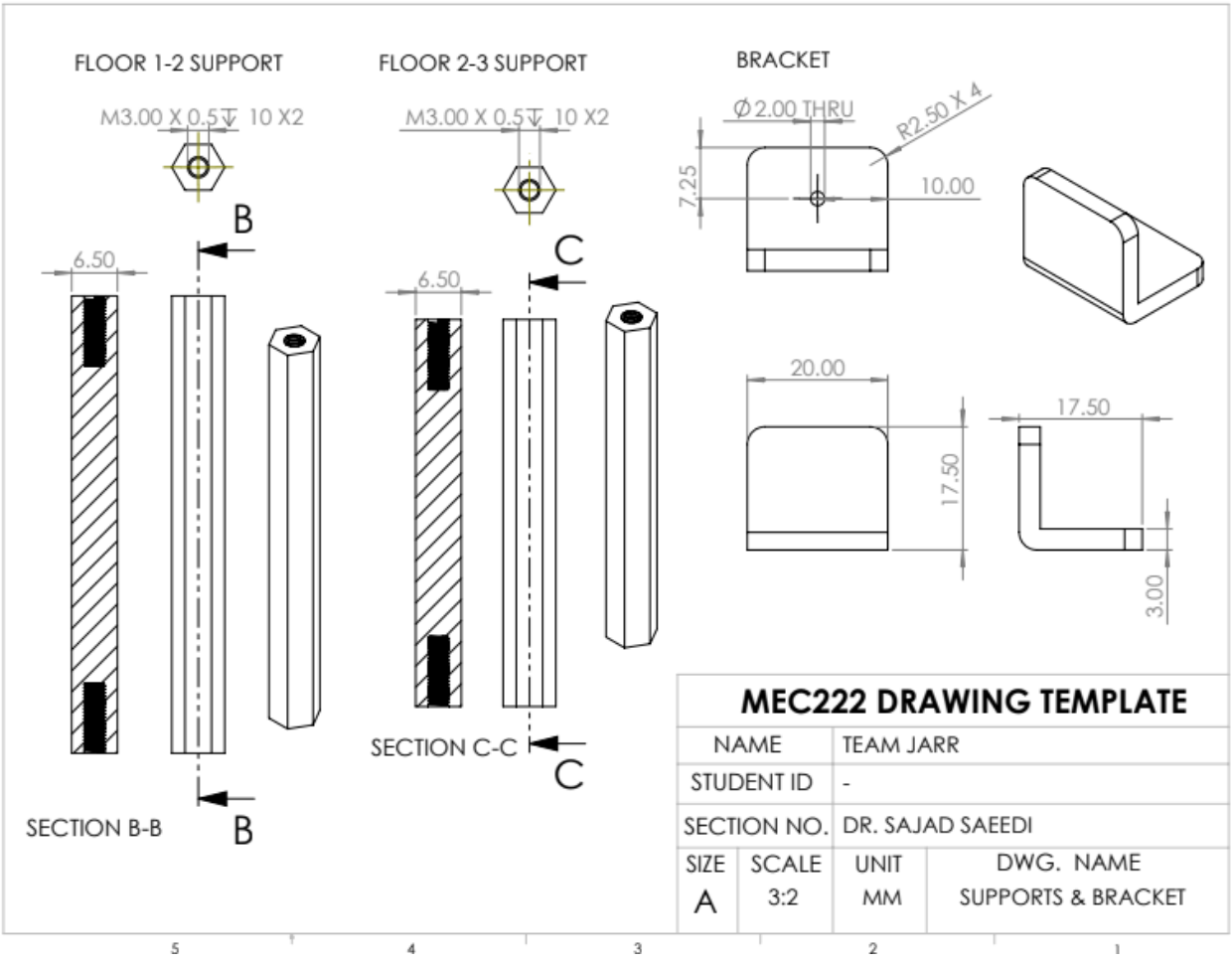
5 4 3 2 1

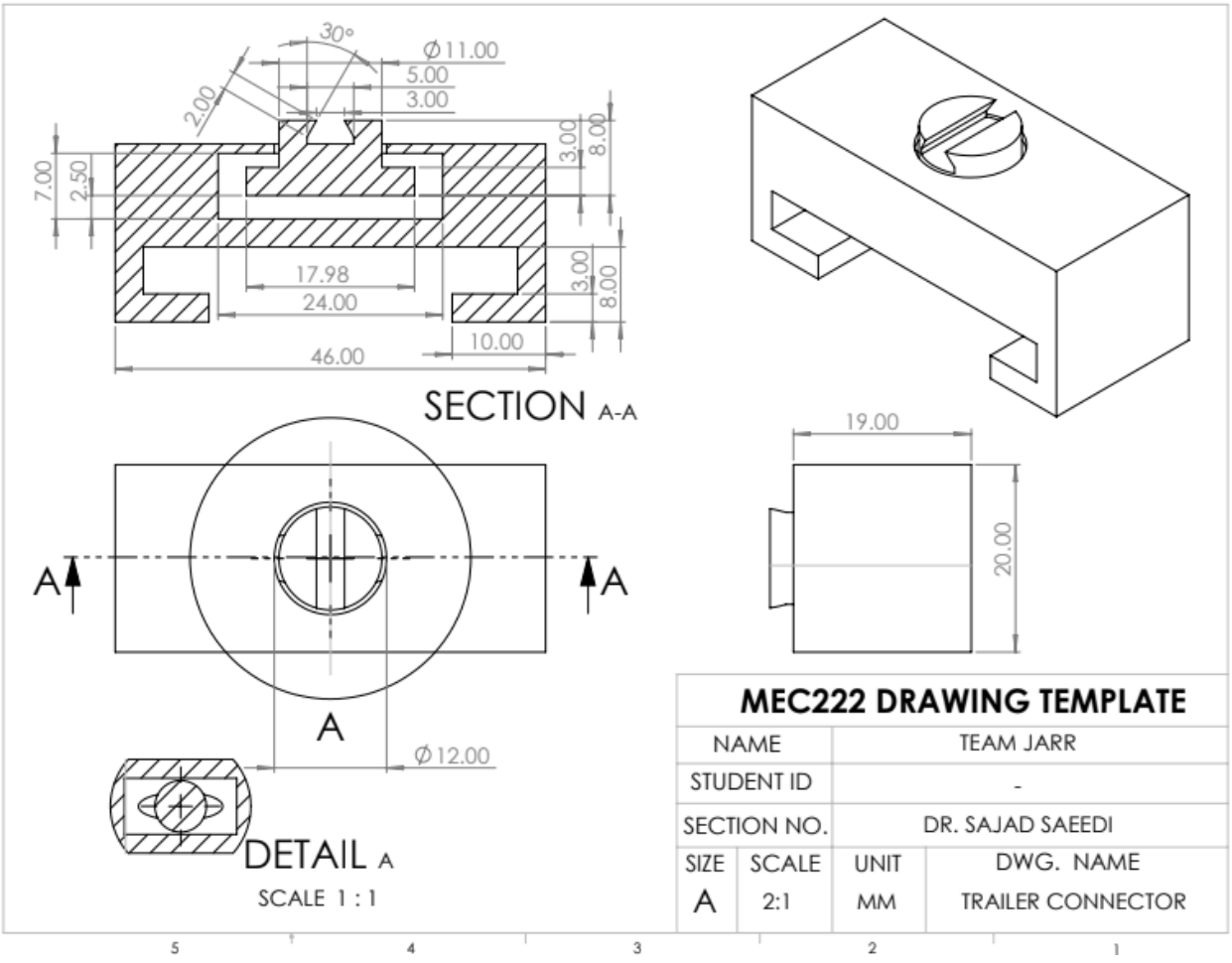

Ø5.05 X 4
30.00 X 2
20.00 X 4
R4.25 X 4
R1.08 X 8
145.00
135.00
131.22
111.22
97.50
80.75
64.25
47.50
33.78
13.78
10.00
0
0
10.00
59.72
75.00
79.72
105.00
109.00
140.00
150.00
2.00
150.00
NOTE:
ALL CUT-OUTS ARE THROUGH
MEC222 DRAWING TEMPLATE
NAME
TEAM JARR
STUDENT ID
SECTION NO.
DR. SAJAD SAEEDI
SIZE
SCALE
UNIT
DWG. NAME
A
1:2
MM
THIRD FLOOR
5
4
3
2
1

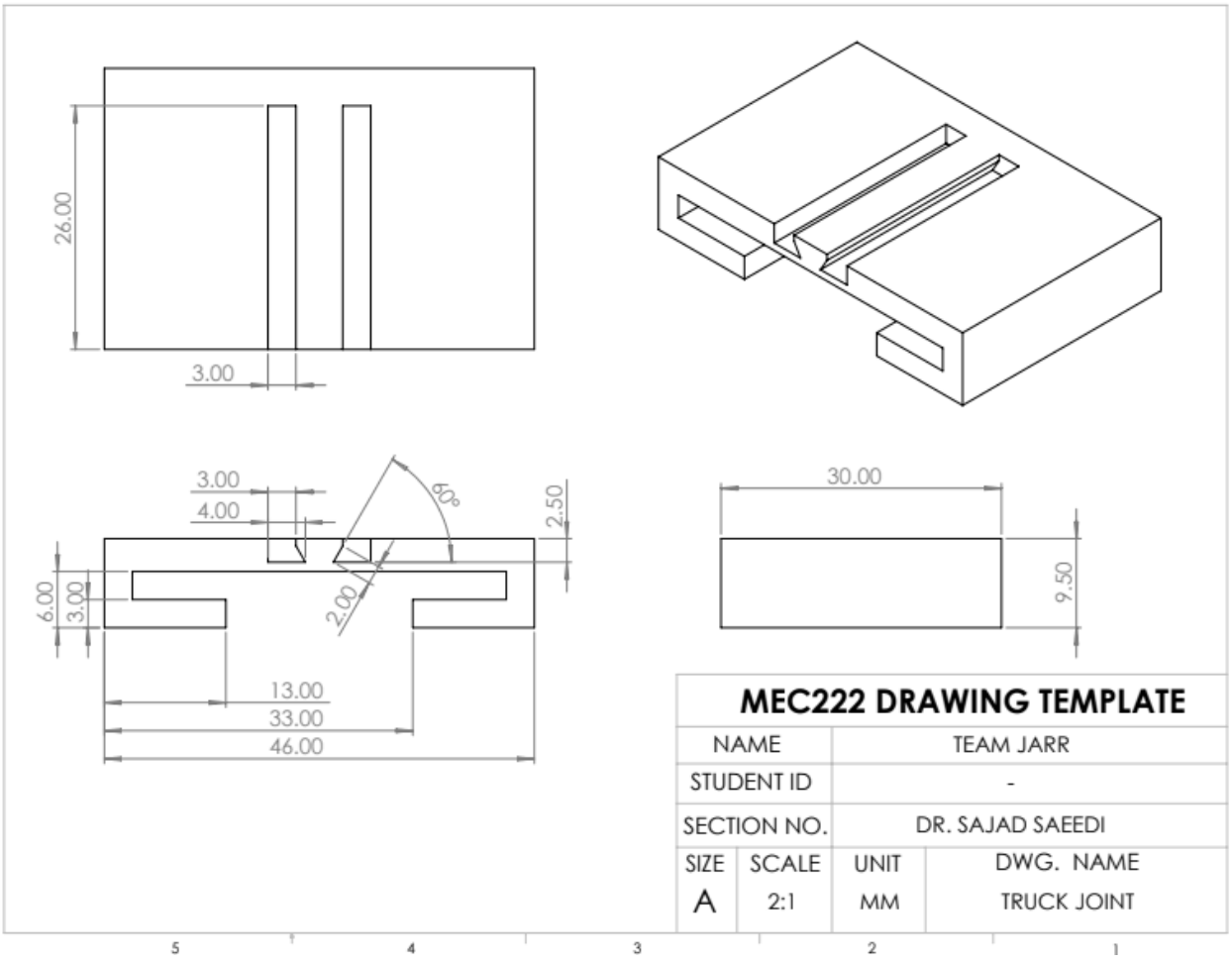

AXLE
AXLE-WHEEL PLATE
Ø5.05
17.50
7.50
5.50
A
A
SECTION A-A
Ø23.95
Ø3.00
2.00
Ø9.00
Ø6.50
M2.45X0.8↧5.00
MEC222 DRAWING TEMPLATE
NAME
TEAM JARR
STUDENT ID
-
SECTION NO.
DR. SAJAD SAEEDI
SIZE
A
SCALE
1:4
UNIT
MM
DWG. NAME
AXLE & AXLE-WHEEL PLATE
5
4
3
2
1


26.08
6.00
R3.50 X2
19.08
MIDDLE LINK
100.00
3.00
29.50
R3.50 TYP
46.50
7.00
Ø5.00
STEERING-WHEEL LINK
Ø1.20 THRU X2
Ø3.00 THRU X2
MEC222 DRAWING TEMPLATE
NAME
TEAM JARR
STUDENT ID
-
SECTION NO.
DR. SAJAD SAEEDI
SIZE
A
SCALE
1:1
UNIT
MM
DWG. NAME
MIDDLE & STEERING-WHEEL LINK
5
4
3
2
1


SERVO BRACKET
STEERING BRACKET
2.15
4.30
2.15
2.00
13.00
23.50
17.00
10.00
3.50
35.50
22.50
7.50
SECTION B-B
B
B
100.00
Ø3.00 X 2 ↧12.5
11.43
2.21
3.00
215°
15.37
55°
R3.50
10.00
35.00
15.00
7.00
4.00
3.00
20.25
29.30
Ø2.00 THRU X 3
5.00
2.50
8.50
36.33
Ø2.00 ↧4
Ø2.00 THRU X 8
12.50
6.80
6.00
4.50
MEC222 DRAWING TEMPLATE
NAME
TEAM JARR
STUDENT ID
-
SECTION NO.
DR. SAJAD SAEEDI
SIZE
A
SCALE
1:1
UNIT
MM
DWG. NAME
STEERING AND SERVO BRACKET
5
4
3
2
1

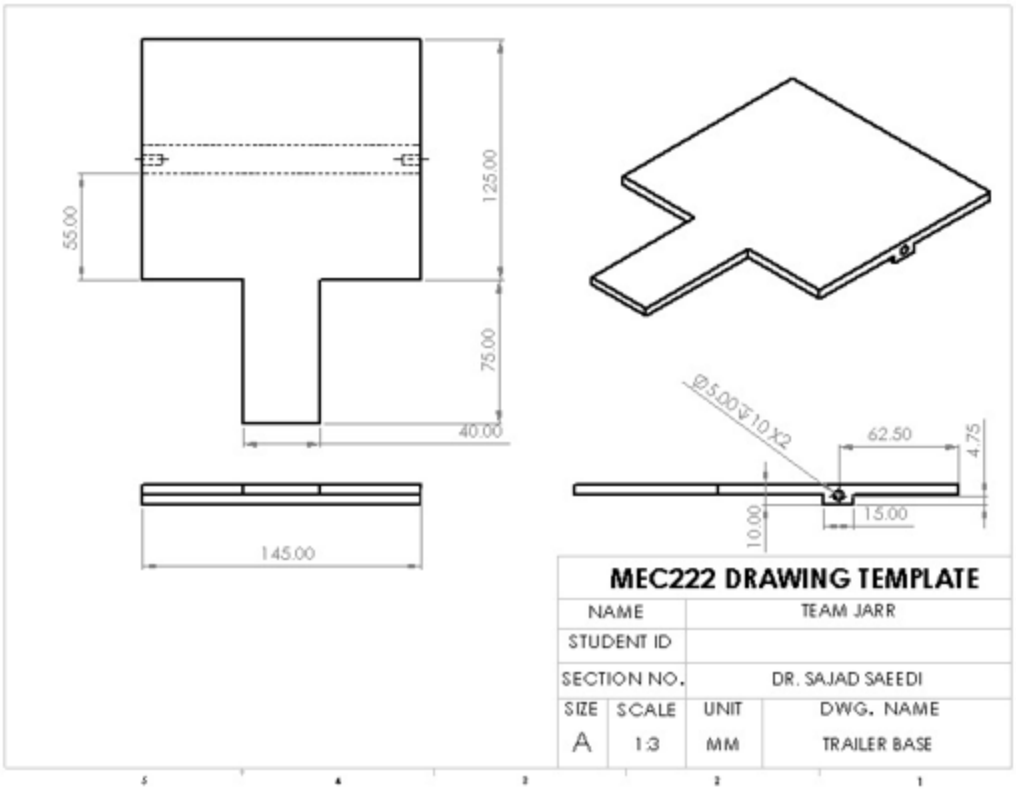

FLOOR 1-2 SUPPORT
M3.00 X 0.5↧ 10 X2
B
6.50
SECTION B-B
B
FLOOR 2-3 SUPPORT
M3.00 X 0.5↧ 10 X2
C
6.50
SECTION C-C
C
BRACKET
Ø2.00 THRU
R2.50 X 4
7.25
10.00
20.00
17.50
17.50
3.00
MEC222 DRAWING TEMPLATE
NAME
TEAM JARR
STUDENT ID
-
SECTION NO.
DR. SAJAD SAEEDI
SIZE
A
SCALE
3:2
UNIT
MM
DWG. NAME
SUPPORTS & BRACKET
5
4
3
2
1

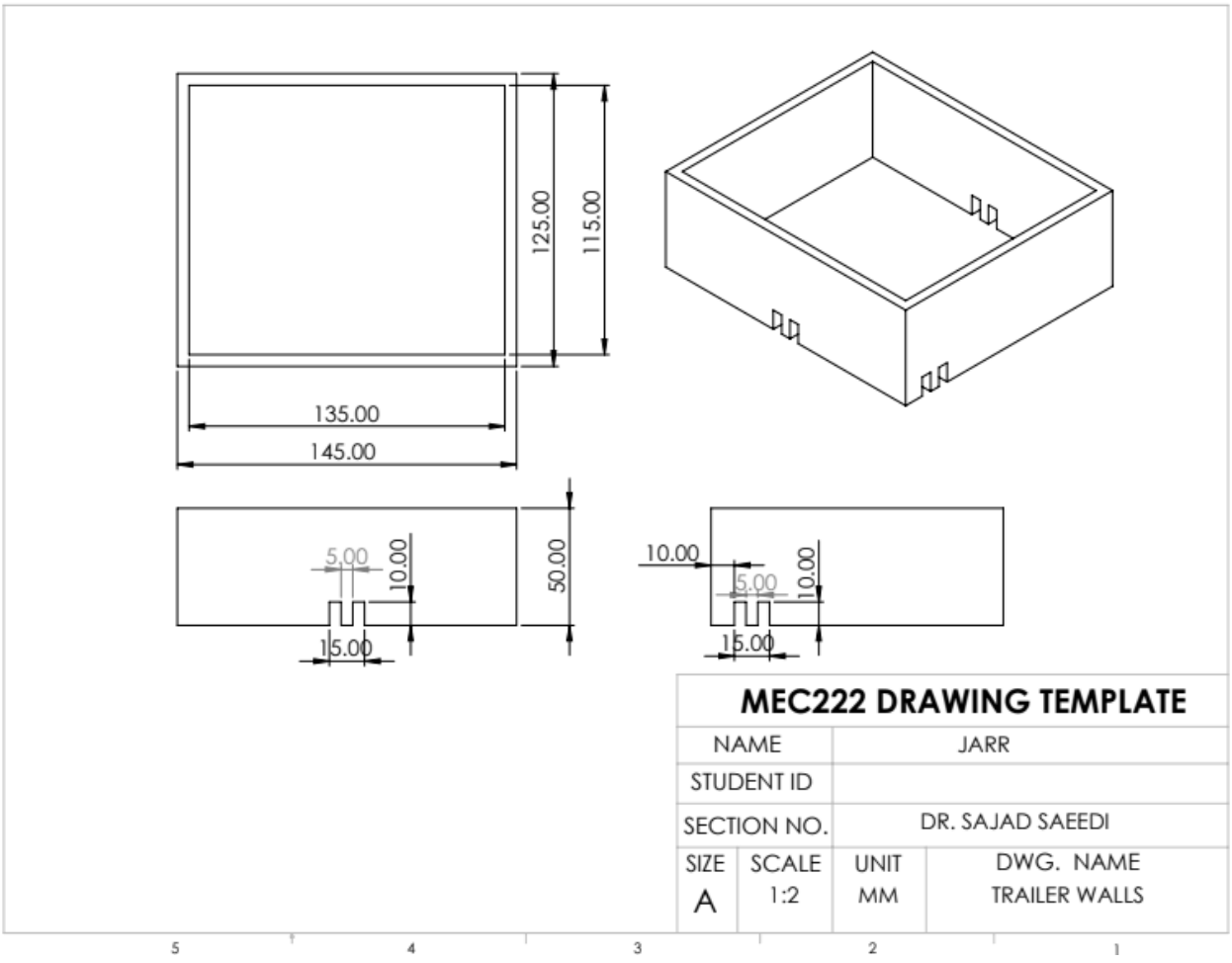

30°
Ø11.00
5.00
3.00
2.00
7.00
2.50
3.00
8.00
3.00
8.00
17.98
24.00
46.00
10.00
SECTION A-A
A
A
A
Ø12.00
DETAIL A
SCALE 1 : 1
19.00
20.00
MEC222 DRAWING TEMPLATE
NAME
TEAM JARR
STUDENT ID
-
SECTION NO.
DR. SAJAD SAEEDI
SIZE
SCALE
UNIT
DWG. NAME
A
2:1
MM
TRAILER CONNECTOR
5
4
3
2
1



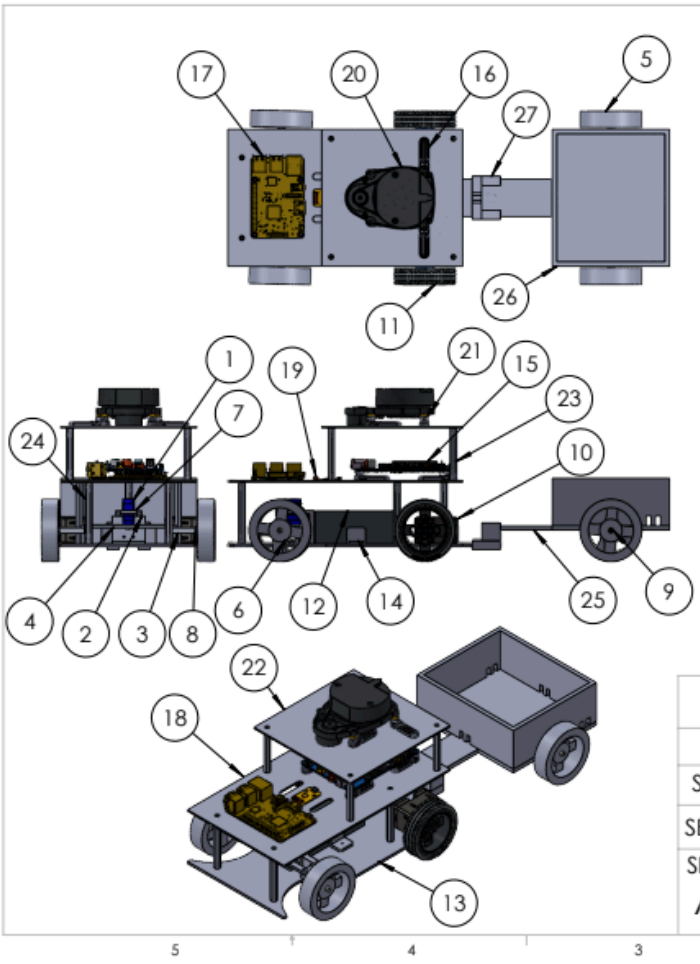
MEC222 DRAWING TEMPLATE			
NAME	TEAM JARR		
STUDENT ID	-		
SECTION NO.	DR. SAJAD SAEEDI		
SIZE	SCALE	UNIT	DWG. NAME
A	2:1	MM	TRUCK JOINT

5 4 3 2 1


55.00
125.00
75.00
40.00
145.00
Ø5.00∓10 X2
62.50
4.75
10.00
15.00
MEC222 DRAWING TEMPLATE
NAME
TEAM JARR
STUDENT ID
SECTION NO.
DR. SAJAD SAEEDI
SIZE
SCALE
UNIT
DWG. NAME
A
1:3
MM
TRAILER BASE



MEC222 DRAWING TEMPLATE			
NAME	JARR		
STUDENT ID			
SECTION NO.	DR. SAJAD SAEEDI		
SIZE	SCALE	UNIT	DWG. NAME
A	1:2	MM	TRAILER WALLS



ITEM NO.	PART NUMBER	DESCRIPTION	QTY.
1	SG90 - MICRO SERVO 9G - TOWER PRO		1
2	STEERING MOUNT		1
3	STEERING-WHEEL LINK		2
4	MIDDLE LINK		1
5	SPARE WHEEL		4
6	WHEEL-AXLE CONNECTION		4
7	SERVO MOUNT		1
8	AXLE		4
9	MC MASTER CARR91292A112	18-8 STAINLESS STEEL SOCKET HEAD SCREW	4
10	XL430_DUMMY.PRT		2
11	WHEEL		2
12	BATTERY		1
13	1ST FLOOR		1
14	BRACKET		2
15	OPENCR		1
16	PR30_IBB_01_3D	NOT SPECIFIED	8
17	DUMMY_RASPBERRYPI3.PRT		1
18	2ND FLOOR		1
19	USB2LDS		1
20	LIDAR		1
21	m_PR07_A01_SPACER_RIVET	PART-M_PR07_A01_SPACER_RIVET-DESC	4
22	3RD FLOOR		1
23	SUPPORTS 2ND-3RD FLOOR		4
24	SUPPORTS 1ST-2ND FLOOR		5
25	TRAILER BASE		1
26	TRAILER WALLS		1
27	TRAILER HITCH ASSEMBLY		1

MEC222 DRAWING TEMPLATE

NAME	JARR		
STUDENT ID	-		
SECTION NO.	DR. SAJAD SAEEDI		
SIZE	SCALE	UNIT	DWG. NAME
A	1:5	MM	ASSEMBLY DRAWING

5 4 3 2 1