Project Insight

2025

Colorado Space Grant Consortium

The University of Colorado, Boulder

Authors: Emma Herzog, Gabriel Ohnstad, Jonathan Leites, Gavin Smith, Ethan Levin, Aur Shalev Merin, Bernadette Weigang

Faculty Advisor: Whit Whittall

Abstract

Project Insight developed a lightweight, modular autonomous rover for the 2025 Mars Rover Challenge (See Figure 1.A), integrating advanced SLAM-based mapping software on the Jetson Nano. Designed to complete a topographical map after navigating an obstacle course, the rover's primary objective was both technological functionality and competition success. The team prioritized a unique approach to software development, emphasizing modularity, adaptability, and efficient performance within a 4 kg design constraint.

Comprising mostly aerospace engineering students alongside members from computer science, physics, and mechanical engineering, the team began with focused research into material stability in sandy environments, movement efficiency, and spatial data collection. This foundation informed key design decisions, such as a gear-stilt clearance system and an adjustable sensor platform for early software testing.

The rover's standout feature is its ability to generate a run overview using camera systems and Simultaneous Localization and Mapping (SLAM) on the Jetson Nano. This system was integrated with a custom program using the Robotic Operating System (ROS) with the 'Move_Base' package for path planning and navigation. The team used an adjustable camera wand to simulate performance ahead of full mechanical integration, allowing iterative refinement.

Insight ensured organized testing on all materials, particularly in solidifying a chassis design and material choices. Testing was formulated by a tagging system and authenticated through isolated testing and system integration verification, particularly with power distribution. Subteams balanced personal goals and needs with greater mission tasks, followed by subsystem combinations overseen by the systems engineer. With these developments, Insight has aimed to showcase the multitude of abilities that can be fit into a compact system.

1.A Final Rover at Competition

Introduction

Project Insight was started in the late fall of 2024 to compete in the 2025 Mars Rover Challenge. The challenge, taking place in mid-April 2025, required each team to develop an autonomous rover, meeting weight requirements in 1kg and 4kg categories. Insight aimed for the 4kg category to minimize technological limitations and maximize headroom for creativity. The team worked diligently in organized subteams to meet competition requirements while setting their own goals: Achieve system modularity, find a creative solution to obstacle clearance, and utilize the Jetson Nano as a core microcontroller to guide system tasks and develop a map post-course completion.

Research was guided by the challenge's requirements while placing subsequent priority on team-specific goals. Holding documentation as a priority, subteams proposed component ideas and goal solutions via an organized presentation system; A holistic review system of component development and method proposals built a cohesive plan going into fabrication and development. With high priorities in weight and budget restrictions, each step of planning was followed by weight estimation checks and budget updates. Project Insight found benefits in recycling existing materials at Space Grant to minimize budget restrictions, allowing dominant allocation of the budget in purchasing an NVIDIA Jetson Nano. Following the major purchases, project management ensured the existing budget could be balanced between unexpected needs and purchases for materials that may hold probable issues.

Finding the ideal method for autonomous rover development given the challenge budget and requirements, Project Insight proceeded according to its mission statement: Design and build an autonomous rover that can efficiently traverse Martian terrain while showcasing thoughtful integration of subsystems. Staying true to this mission, the team found inventive approaches and solutions throughout the project; Namely, an interchangeable gear-stilt system to increase clearance height while maintaining a reliable center of mass, integrating ORBSLAM and ROS frameworks in Jetson software for object recognition and map development, and facilitating a building and design process defined by component modularity for maintenance and agility.

This paper will highlight the processes behind developing Insight, the research and processes that developed the original design, and the transformations that Insight underwent as the project progressed. A literature review will provide insight into the preliminary, big-picture decisions behind the primary methodologies used to build Insight. Readers will find numerous diagrams and illustrations in the materials and methods section to portray complex ideas and showcase developmental illustrations. The Results section will discuss not only the competition outcomes but also individual subteam final results and tech reflections.

Research & Literature Review

Insight was started from scratch, with little to no knowledge of autonomous rover development before the COSGC-provided modules. Research to expand everyone's understanding of the requirements and technology that go into fabricating a rover was integral to begin ideating design possibilities. This section will explore the research and sources that aided the team in initial design ideas.

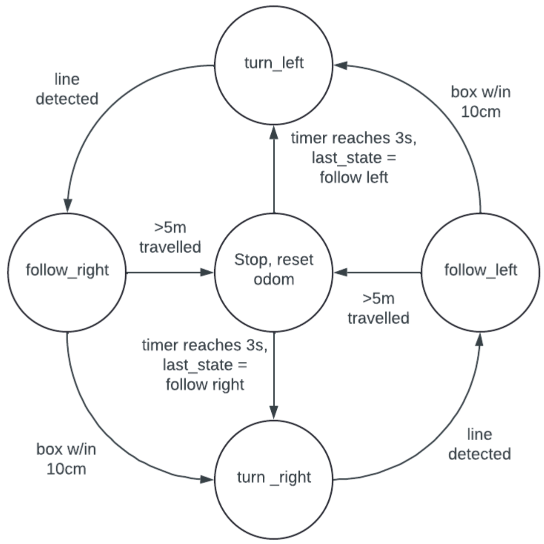
Understanding Autonomy, Basic Motion, and Sensing:

Autonomy is built by a combination of basic motion and sensor systems and, being one of the key requirements, was one of the first technological concepts the team explored. Before organizing team management, individuals on the team completed the *Robotics Workshop* modules, understanding and building lightweight, basic individual rovers. These modules, developed by Space Grant, started with power electronics and went through advanced sensing.

As students with involved engineering interests, team members came in with a basic understanding of motor drive. However, more detailed knowledge of varying supply voltage and motor code was lacking, which is where Lesson 3, “Basic Motion,” came in. This longer overview covered detailed motor and rotary connections while contrasting brushed and brushless DC motors and describing proper voltage supply for varying speeds and directions. This lesson also introduced the concept of “Layers of Abstraction,” building a visual to divide larger problems into smaller, more manageable ones. Being able to break down a problem or setback into the simple core of an issue made ideating the rover’s design and systems far more approachable.

Key takeaways in autonomous development strategies started to be understood in Lesson 4, “Basic Sensing,” where time-of-flight sensors, ultrasonic sensors, infrared light sensors, and rotary encoders were introduced alongside steps to integrate them into a rover. It is a proper combination and integration of these sensors and others that begins to make a rover autonomous via a Finite State Machine (FSM) (See figure 2.A). FSM serves as the foundation of possible states or tasks the rover can encounter, and when transposed with software-developed event-driven programming, the rover will have a ‘brain’ prepared to acknowledge a condition and execute a specific task in response. These short modules and lessons were integral to getting the team started but lacked advanced techniques and methods, as would be expected.

2.A Basic Sensing Diagram Provided in Lesson 4



After learning this basic framework to autonomy, the team wanted to explore more complex methods to autonomous navigation beyond a common ultrasonic sensor, which is when preliminary explorations of the Jetson Nano began.

Understanding the Jetson Nano:

A major consideration while designing the rover was providing space to or coming up with a unique component to stand out amongst other teams in the competition. It quickly became apparent that ideating a more complex aspect at that point would be challenging and risky, which is when the consensus arose to integrate the unique component into the software. Removing the expectation or pressure to have a unique, complex method integrated into the rover design in the first few weeks allowed the team to focus on producing a working rover, highlighting basic structural integrity with electronic integration and communication. However, it was not long before the idea of complex object visualization to develop a map post-course completion sparked.

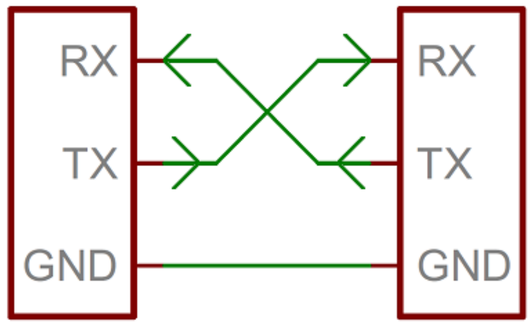
2.B NVIDIA Jetson Nano Version 1

Researching how to achieve this is where the Jetson Nano (See figure 2.B) was first introduced. It was learned that building a map given course data required Simultaneous Localization and Mapping (SLAM) for the rover to understand where it was in the course to properly integrate course data into a map. Running SLAM requires a powerful brain, and the Jetson Nano met these requirements while remaining within budget. Because it consumed the majority of the budget, detailed research to understand if such a system could be used, how it would be, and proper communication with other components was conducted.

Learning How to Make Electronics Communicate:

A crucial aspect of Insight's architecture was the splitting of computing between the Jetson Nano and Arduino. Arduino was ideal for motor control and IO, for which its platform is optimized. Still, due to its very light processor and lack of compatibility with desktop software, machine vision was impossible. The Jetson, however, had compatibility with Ubuntu and a powerful CPU and GPU expressly built for machine learning and vision in edge applications. In this sense, the Jetson was a full computer that you could hold in your palm, and the perfect candidate for Insight. Thus, the structure was that of the Jetson taking in camera inputs, generating our topography, and sending instructions to the Arduino, which would be used as a lightweight motor controller. This architecture of bifurcated computing thus required a communication protocol to carry information from the Jetson to the Arduino for encoding.

Data protocols are abundant and purpose-built, but for our purposes, all we needed was a simple serial communication standard, with the Jetson and Arduino each receiving and sending data on to each other through their channels. We decided to use the already existing TX and RX ports through UART, a back-and-forth communication method, with one computer's transmit pin (TX) connecting to the other's receive pin (RX). The next question was that of the encapsulation of the data itself,



2.C

TX/RX (See figure 2.C), which ROS conveniently had built-in libraries for. Using the RosSerial library on Arduino IDE and libraries on ROS' end, a pipe was built through which directional instructions were sent to the Arduino, which converted this data into motor instructions. Aur Shalev Merin initially used key mapping to send outputs to the motors before fully letting OrbSlam dictate the driving.

Materials and Methods

This section will be divided into 3 subsections, representing Insight's 3 subteams: Mechanical, Electrical, and Software. Organizing materials and methods this way clarifies the organized division of labor, ensuring every member has a specific task in rover fabrication. The Systems engineer oversaw and guided these developments while picking up slack where needed, while the Project Manager organized broader task distribution, budget and weight projections, scheduling and communication, and documentation.

Mechanical Subsection

Deciding Rover Structure and Material:

Nobody on the team had built a rover like one for the competition before, so preliminary ideas regarding structure were based solely on structural stability, lacking consideration from a broader systems perspective on how components will work in a specified structure. This was one of the first matters considered by the team, likely for its significance in being able to sketch what the final rover would look like. However, what mattered more in this design period was *how* the rover would drive. What is the best way for the motors to be placed/attached to the wheels for efficient drive? How does internal/electrical component placement in the body of the rover affect the center of mass? Is clearance height an issue, and how does that affect the wheel design? The first sketch/design iteration utilized a trapezoidal, tank-like body with a rocker-bogie system for the wheels. In figuring out how to achieve this picture, it became clear that considering the previously stated questions would have guided the team to a more simplistic structural design considering priorities in reliable electronics, balance and center of mass, and remaining within weight constraints. A rocker-bogie system, while helpful in traversing hills and bumpy terrain, would have required heavier material structures, axles, and the additional weight of two wheels. While architecturally strong and slightly more aerodynamic, a trapezoidal tank for the chassis of the rover would have presented more space issues for the components inside and is far more challenging to build, making it an unreliable structure.

It proved highly beneficial to discuss the 'do's and don'ts' of the process with last year's rover team alongside major takeaways, as it helped the team solidify the decision to make a more simplistic rectangular body made from acrylic. Having a rectangular body provided the opportunity to make easily removable hatches to access parts or change batteries during competition in specific locations of the body to isolate other components from sand intrusion.

The decision to use acrylic as the primary material stemmed from previous success, accessibility, relative durability, cost, and the ability to easily fabricate and iterate with laser cutting. Having access to purchase 12"x18" sheets right on campus at a low cost allowed for numerous structural iterations to perfect the integrity and the attachment system.

Starting with a clear goal of making every component of the rover easily removable and replaceable in a short enough time to be practical during competition, the design changed quite a bit as time progressed. Starting with a simple rectangular prism for a chassis, the mechanical team fabricated and tested multiple methods of securing the acrylic at the corners without using permanent fasteners. Ultimately, the most successful design path for the chassis came from working backward. The team created a scale on which each component was listed from least likely to most likely to break or need to be replaced. After taking a look at that scale, the least likely components were used as a base for the most likely components–the ones that might need to be replaced a couple of times. This meant that the bottom of the chassis was designed with as few cuts as possible to maintain structural integrity; then, each of the wall pieces and interior dividers were slotted into joint holes cut into the base. The walls with the motors attached were connected next, as they would need the most stability. Lastly, the walls on the short end of the rover were slotted into “hooks” cut into the ends of the long wall before the base of the wall was pushed into the base, where square shims locked it–and, therefore, the rest of the walls–securely into place.

Iterating the Rover Chassis:

While iterating the chassis took time for the fabrication team, the biggest obstacle they faced was implementing modular stilts. The purpose of the stilts was to give the rover body enough clearance, but with the added challenge of being able to remove the stilts at a moment’s notice. The team started with a belt drive system, encased in an acrylic case, which would click onto the motor shaft mounted in the main body of the chassis. It would extend about 5 inches outward at a 45° angle. Multiple issues were quickly discovered with this design. The acrylic was too fragile and pliable to make it a viable structural material, and the belt in the belt drive system would have been too unreliable, especially the second any sand or environmental factors were introduced into the system. To combat these issues, the belt drive system was quickly transitioned into a purely gear-driven one, and all the components were 3D printed instead of cut out of acrylic. Transitioning to printing not only allowed for faster and more accurate testing of prototypes but also much less pliability, allowing for far more torque conserved from the motor, down the gears, and into the wheel.

Once the design had been finalized, the entire stilt consisted of a two-part casing, three gears, a ball bearing, and one axle. When being utilized, the system would click onto the axle of the motor on each of the four corners of the rover, with square extrusions clicking into the acrylic, and two screws securing the part rigidly in place. The wheel, now removed from the motor, would slide smoothly onto a custom-designed axle that was attached to the bottom gear in the stilt system. When an obstacle called for more stability and less clearance, it was as simple as removing the wheel and the two screws, popping the stilt off of the frame, and sliding the wheel back onto the motor.

Ultimately, the finalized design of the rover and its components was chosen over a long period of theoretical design, fabrication, iteration, and repetition. The final design chosen for

Project Insight was the combination that the team believed would provide the greatest amount of adaptability, longevity, and ease of accessibility.

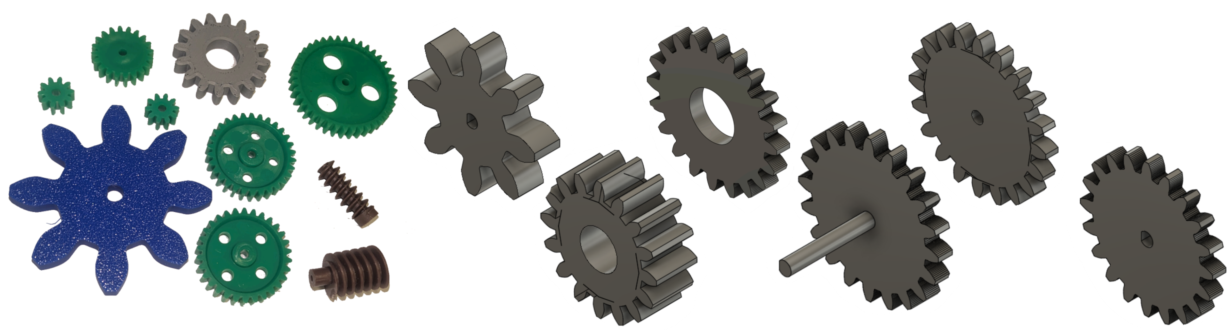
Addressing Clearance Height:

The task of addressing clearance height was postponed after choosing not to include a rocker-bogie system. However, the mechanical team, Jonathan Leites and Gavin Smith, worked diligently to ensure this issue could be properly considered and addressed before the competition. Before adding additional height, the rover's clearance height would have been 1.75 inches, which would not have posed a significant issue, but may have led to time constraints or inconveniences in specific courses. The completed stilts system made the clearance height 4.5 inches, allowing the rover to easily traverse over small rocks and other obstacles that may have led to flipping.

Iterating the Gear Stilts:

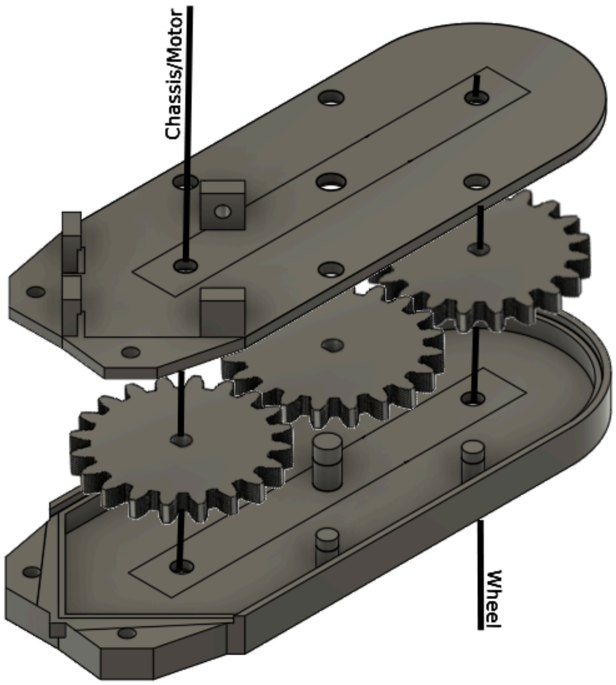
The following are examples of the tested gear designs after the drive belt had been discarded, followed by the finalized designs for the casing of the stilt (Figure 3.A)

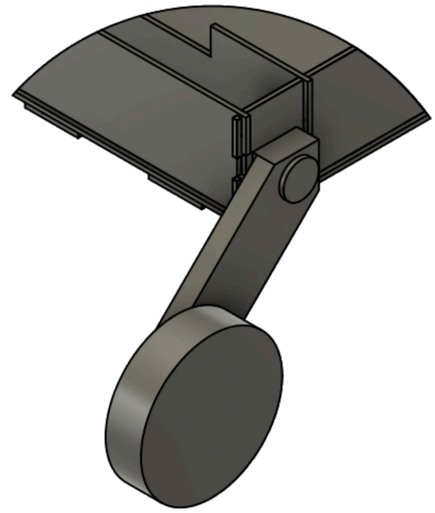
3.A Gears



After many iterations with many different styles of fastening, the team decided to settle on a 3D printed gear made with relatively small teeth, for maximum strength and lowest chance of breaking.

3.B Final Gear Stilt Deisgn





The designs above (Figure 3.B) show the finalized design for the legs of the rover. The top part of the left figure would click onto the side of the chassis, as shown on right, and the wheel would slide onto an axle that is attached through the bottom casing plate directly to the gear.

<u>Electrical Subsection</u>

The electrical subteam was made up of a one-man team, Ethan Levin, who was able to successfully address the electrical requirements (i.e., What components are necessary? How should power distribution be considered?), fabricate said components, and integrate all connections in an organized manner within the rover chassis. Electrical lead, Ethan Levin, followed a fairly systematic approach to the rover's electronics. By prioritizing components from a broader level (explained below) to perfecting specific pieces later on, he faced few setbacks.

Selecting a Microcontroller:

Since the goal of the rover was to be able to map its surroundings, an ordinary microcontroller like the Arduino would not suffice alone. This is due to the mapping program being extremely resource intensive. After researching various microcontrollers, the team decided to utilize a Jetson Nano to run the program. Additionally, the team sought it wise to utilize an Arduino in tandem with the Jetson to run low intensive tasks, such as motor control. The Jetson would function as the core microcontroller and give directions over UART to the Arduino.

Motors and Power Budgeting :

The team selected DC motors due to their availability and ease of integration, which allowed for rapid prototyping. These motors were salvaged from the previous year's rover, contributing to significant budget savings. Consequently, the electrical system had to be designed around the specific requirements of these motors. Since the motors demanded high currents, a suitable motor driver was necessary. Fortunately, the previous team's Arduomoto board–compatible with the motors–was repurposed and used.

The Ardumoto was mounted on an Arduino Uno, powered by a 2200 mAh, 11.1 V Lithium battery. To confirm safe operation, the team referenced Arduino specifications and determined that the board could accept input voltages between 7 V and 12 V, allowing direct connection of the battery via the varrel jack.

However, the Jetson Nano presented a stricter requirement, operating only with a 5V input. To meet this constraint, the team integrated a buck converter to step down the 11.1 V battery output to the required 5 V for the Jetson. Once all major components were selected, the team developed a power budget, estimating current draw under maximum power usage. Although documentation on the Jetson suggested a very small window of tolerence for transient spikes and current variance, the buck converter and battery, when tested with the system, worked nominally, even allowing for two batteries to be switched in parallel without the Jetson crashing.

This Theoretical analysis suggested a minimum operational battery life of approximately 45 minutes. Given that the robotics challenge spanned a full two-hour window, this insight prompted the strategic decision to carry multiple fully charged batteries to ensure uninterupted operation throughout the competition.

Software Subsection

Finding Broad Software Structure Based on Requirements :

The software engineering team, led by Aur Shalev Merin, was in charge of implementing all of the code the robot needed to function and avoid obstacles, as well as the additional, self-imposed requirement for the rover to create a map while it traversed the terrain. SLAM algorithms, Simultaneous Localization and mapping, were deemed the perfect choice for this mapping mission. The algorithms create a map based on the sensory inputs the rover receives and then proceed to place the rover within that map, allowing for intricate knowledge of the rover's surroundings, path planning, and mission reusability, among other advantages. Arduino boards, while excellent at motor control and taking in sensor inputs, are not nearly powerful enough for this computationally intense algorithm; therefore, it was deemed necessary to separate the processing section of the software system and the motor control section. The solution the software team came up with was to have one board, the Jetson Nano, run the slam algorithm and send commands to another board, the Arduino Uno, which in turn converts that to direct commands to the motors. The tech stack had numerous iterations, but it turned out to be as such: Ubuntu 18.04 for the Operating System, ROS (Robot Operating System) Melodic for the middleware, ORBSLAM 2 for the SLAM algorithm, Move_base for path planning within the given map, geometry_msgs/Twist for the message structure, and rosserial to transfer those messages between the jetson and the ardunio using a USB Cable. It is also important to note that, due to further testing, it was deemed that the Arduino should play a much more significant role, which will be described in a later paragraph.

<u>Why the Jetson Nano?</u>

The team decided on the NVIDIA Jetson Nano because of the board's balance of computational capability, energy efficiency, relative cheapness, and compact form factor, making it a strong choice for autonomous rover applications operating in remote environments such as deserts. Specifically, for SLAM-based navigation using ORB-SLAM2 with monocular webcam input, the Jetson Nano's integrated 128-core Maxwell GPU enables real-time visual feature extraction and pose estimation, while its quad-core ARM Cortex-A57 CPU supports the necessary system orchestration and ROS integration. Its low power consumption—typically around 5 to 10 watts—ensures compatibility with battery-operated field systems, which is crucial for extended deployment in environments where power resources are limited. Additionally, the platform's broad compatibility with Ubuntu-based ROS environments and camera drivers simplifies software development and integration, enabling reliable performance even in thermally challenging and connectivity-constrained desert conditions.

Integrating ROS and SLAM:

Implementing Simultaneous Localization and Mapping (SLAM) with the Robot Operating System (ROS) framework presented significant challenges, particularly on resource-constrained platforms like the Jetson Nano. The limited storage capacity caused the team to utilize the careful optimization of ROS packages and SLAM parameters to maintain system stability. The team “bricked” the jetson multiple times, resulting in losing all work and requiring a time-costly reflash of the operating system. Camera integration posed substantial difficulties, including synchronization issues between stereo webcams, where even minor timing discrepancies between frames significantly impacted feature matching and depth perception. Camera calibration proved especially problematic, with incorrect or missing camera intrinsics causing critical failures in the SLAM pipeline, causing the team to eventually decide on simply using monocular SLAM. Additionally, dependency conflicts between OpenCV versions and ROS packages created unexpected compatibility issues. The computational demands of visual SLAM algorithms further strained the Jetson Nano's resources, requiring thoughtful parameter tuning to balance mapping accuracy with real-time performance constraints. Numerous SLAM algorithms were tested, including RTAB-Map, ORBSLAM 3, and Vslam, but ORBSLAM 2 was the best mix of efficiency and performance.

The Arduino’s Changing Role:

As the competition neared, the team identified a few potential issues with the existing SLAM system. Although the Jetson is quite powerful, a computationally intensive process such as visual SLAM would take a while to initialize and would often lag, despite the various optimizations the team made; on top of that, the resulting map from the slam system was two dimensional, great for testing in the lab, but not great for a terrain as complex as the desert. Therefore, it was decided to add three ultrasonic sensors as a backup to the more sophisticated SLAM system. The sensors were positioned to detect obstacles in a forward-facing configuration: one aligned centrally and two angled at approximately 30 degrees to the left and right, respectively. The robot utilizes these sensors to detect and avoid obstacles dynamically in real time. When an obstacle is detected within a predefined threshold, the robot executes a sequence of maneuvers that includes reversing, turning 90 degrees toward the safer side (based on comparative sensor data), and performing lateral displacements until a clear path is found. It then reorients to its original heading and resumes forward motion. The Arduino script that was programmed with three different modes, the first being taking messages from the Jetson Nano as initially planned but only while the ultrasonic sensors did not detect an incoming collision, the second was relying totally on the ultrasonic sensors, and lastly a mode which simply tells the rover to go straigh ahead without listening to any sensor input. These three modes allowed the team to test what works and what doesn't on the fly while in the field. For example, it was discovered that the rover could climb some dunes that the ultrasonic sensor was picking up and avoiding, so the straight-ahead mode worked best. However, most tracks did require obstacle avoidance.

Results and Reflections

The conclusive test of Project Insight's research and development was COSGC's Robotics Challenge at the National Sandunes. The team's priorities leading up to the challenge were to complete as many of the challenge courses as possible while demonstrating advanced technology in software and modularity.

The challenge was comprised of 5 main courses, each increasing in difficulty and complexity. Insight was able to complete 4 of the 5 courses flawlessly, while nearing completion on the 5th course but not reaching the end. During the competition, numerous adjustments or modifications were made to the rover, namely, the removal of the stilts after the 1st course and a later alteration of the stilts for an untested approach to the 5th course.

Research in power and voltage requirements proved important as an extra battery was purchased on top of the existing two before the competition, and all 3 fully charged batteries were drained by the end of the 3-hour round. Insight's electronics performed efficiently and consistently, demonstrating reliability in the technology's communication. Additionally, the organized electronics remained protected from the sand inside the electronics, as designed by the puzzle-like chassis walls and sealed surroundings. Any software uploads made during the competition were completed efficiently due to Systems Engineer Gabriel Ohnstad's attention to detail in cable management and Ardumoto modifications made for increased pins in voltage and ground.

Removal of the stilts after the first course was decided under the conditions of increasing wear of the internal gears connecting the motors and the wheels. Because the gears were 3D-printed, their PLA was easily worn by the high torque in the motors. Even after additional reinforcements to the gears the night before the competition, they still showed unreliability as the wheel rotations would skip and the engines would stall due to inconsistent movements. Increased stilts testing would have resolved this issue, bringing attention to a needed material change for the gears; machine-manufactured aluminum or metal gears would likely have been more reliable, allowing the stilts to remain on for a greater majority of the competition.

A later alteration of the stilts was proposed by software engineer Bernadette Weigang after seeing that increased clearance height would be beneficial for the vertical gradient in course 5. The modification involved using only 2 of the stilts instead of all 4 to add clearance height (See figure 4.A) without reducing the torque from the worn gears too much. The stilts had been previously modified by a permanent addition of cones at the base to increase width, but these cones had to be removed for the rover to still be able to turn with only 2 of the stilts being added. Although the cones required forceful prying, they were successfully removed, and the lifted rover was able to overcome more of the gradient and larger rocks in course 5. The ultimate reason for its failure of full completion was down to the wheels being too small to continue traversing such a gradient; having larger wheels would have prevented the rover from digging into the sand and restricting its path to the top.

4.A Lifted Rover with Addition of 2 Stilts

Software development of the Jetson Nano with ORBSLAM had some initial time setbacks due to limited space in the Jetson and an unexpectedly large learning curve concerning the Linux-specific technologies. However, head software engineer Aur Shalev Merin worked diligently up to the competition to decipher how to successfully achieve the team's goal of map development. His self-education in ROS and SLAM proved useful, resulting in success with the map development. However, the team expected that 2 basic Logitech cameras would be sufficient for the Jetson's object and pathway recognition. Unfortunately, these cameras showed challenges with lagging, and the software team was ultimately unable to properly calibrate the dual camera system, making the Jetson's localization timing inefficient in the competition. A longer working period and more advanced cameras would have facilitated a more effective integration of the Jetson Nano, allowing the rover to use the advanced system to its fullest potential.

Systems design and integration were unquestionably successful, as designed by Gabriel Ohnstad. The different subsystems and components worked flawlessly in the competition, placing a great deal of security surrounding the overall reliability when making modifications before and during the competition. Communication between electronics and mechanics worked precisely, supported by meticulous planning and circuit layouts before fabrication and integration. As documentation remained a priority throughout the entirety of the process, the team was able to rely on comprehensive notes and research in the decision-making processes surrounding the puzzle of component connections, both mechanically within the chassis and in software applications.

Conclusion

A team of 5 freshmen, a sophomore, and a junior–most with little to no prior experience in robotics–successfully designed and fabricated an autonomous rover within a single semester. While the final rover was an impressive feat in itself, the deeper value of Project Insight lies in the process: an iterative journey defined by innovation, interdisciplinary collaboration, and problem-solving under real-world constraints. This project pushed students to navigate unfamiliar technologies, adopt new engineering mindsets, and confront technical challenges head-on, all while staying grounded in systems thinking and documentation.

The Insight rover is more than a product–it is a reflection of a mindset. The project demonstrated how modular design, creative material reuse, and ambitious system integration can yield functional, scalable solutions within budgetary and technical constraints. From the mechanically adjustable gear-stilt system to the power-aware electrical framework to the Jetson Nano's SLAM-based mapping software, each subsystem tells a story of thoughtful engineering decisions informed by research, experimentation, and reflection.

Beyond technical development, the team's experience revealed the importance of flexibility in design, collaboration across disciplines, and the value of iteration over perfection. Early assumptions about structural needs and sensor performance were reshaped by testing and field results. When 3D-printed gears proved unreliable, the team adapted by redesigning the stilt configuration mid-competition. When software setbacks emerged, members stepped into new roles to assist with calibration and last-minute programming. These responses show not only competence but resilience and adaptability–skills fundamental to future engineering success.

Moreover, Insight's development embodied a unique blend of academic learning and hands-on discovery. By synthesizing classroom knowledge with real-world implementation–navigating everything from ROS architecture to chassis stress points–team members gained a robust understanding of how engineering is lived, not just learned. The holistic integration of documentation, testing protocols, and iterative design cycles ensured that each decision was transparent, traceable, and open to collaborative critique.

Project Insight ultimately illustrates the potential of undergraduate innovation when technical curiosity is met with structured teamwork and institutional support. The team's hard work was further supported by their success in competition, winning the award for "Outstanding Domstration of Advanced Autonomy" (See Figure 5.A). This rover, while compact in physical size, represents an expansive educational experience–one that has shaped its contributors as both engineers and learners. It stands not just as success in competition but as a model for how young engineers can think creatively, act decisively, and work collectively to transform a conceptual goal into a tangible, working system.

5.A Awards Ceremony Post Competition, with Barbra Sonhani

Project Insight was Awarded 'Outstanding Demonstration of Advanced Autonomy'

Sources

2b.:NVIDIA. *NVIDIA Jetson Nano 2GB Developer Kit (945-13541-0000-000)*. Amazon, https://www.amazon.com/dp/B08J157LHH. Accessed 14 Apr. 2025.

2c.: "Untitled Schematic Image." *SparkFun Electronics*, https://cdn.sparkfun.com/assets/2/5/c/4/5/50e1ce8bce395fb62b000000.png. Accessed 14 Apr. 2025.

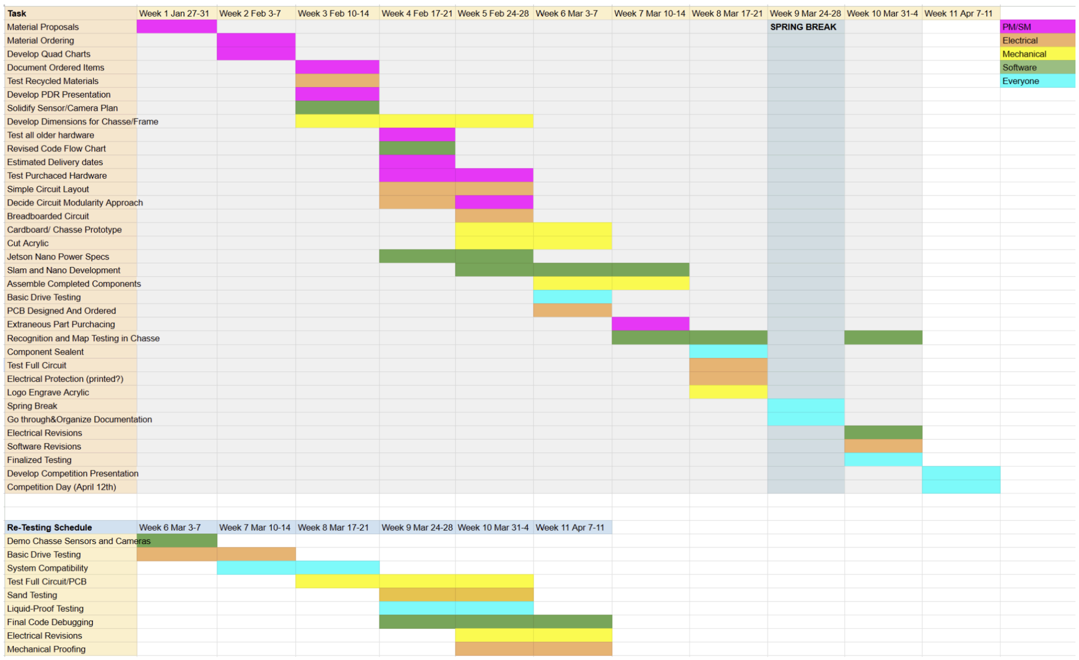
Appendix

Budget and Weight Summary/Tracker

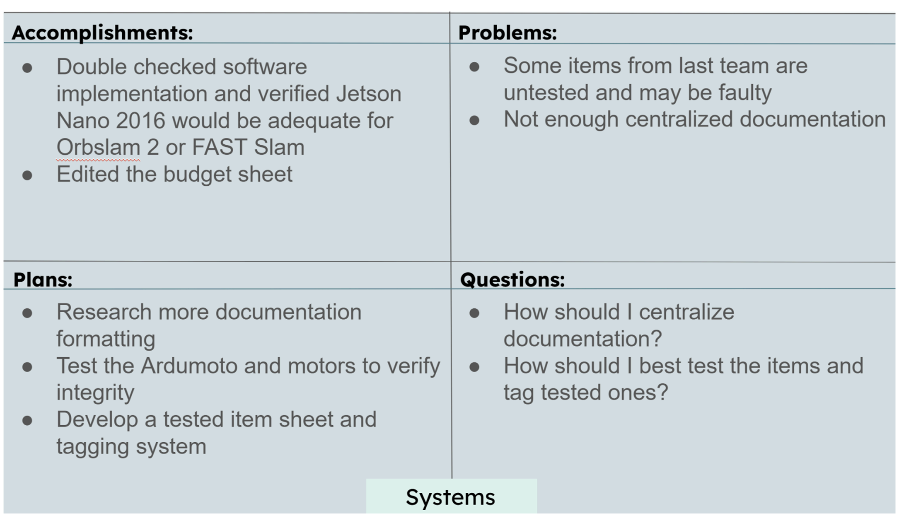
N	QTY	Kg/ Unit	Kg Total	# Lbs/U	Lbs Total	V	A	W/H	$/U	Price	Link	Need for/by	Tested	NOTES
Ovonic 3S 2200mAh 35C (Battery)	2	0.1401614805	0.1401614805	0.309	0.618	11.1	76	24.4	0	17	Batteries			
Motor Driver	4	0.02993740361	0.1197496144	0.066	0.264			0	10	40	Amazon			
Step-down Converter	1	0.03629356346	0.03629356346	0.08001279	0.08001279			0	1	7	~~Arduino Store~~			Actual Link
XT60	6	0	0	unknown/negligible	0			0	1.6	9.6	DigiKey			
Jetson Nano	1	0.12	0.12	0.48987976	0.48987976	5	3	15	199	199	Amazon			
Aruino UNO (Rev3)	1	0.02494783634	0.02494783634	0.055	0.055	5	0.05	0.25	0	0				
Ardumoto	1	0.009979134537	0.00997913453	0.022	0.022			0	0	0	Ardumoto			
3" Rim, 4.25" Tires	4	0.1388006895	0.5552027579	0.306	1.224			0	0	0				
12V Motors 150:1	4	0.1982218997	0.7928875987	0.437	1.748	12	5.5	15.2	0	0	Motors			150:1 Metal Gearmotor 37Dx73L mm 12V with 64 CPR NO Encoder (Helical Pinion)
Acrylic	7	0	0	tbd				0	5	35	ITLL			
Camera	2	0.1304091445	0.260818289	0.2875		5	0.5	5			Cameras			
Ultrasonic Sensor	2	0.008	0.016											
256 GB SD Card	1	0	0							19				
IMU	2	0.002	0.004						0	0	IMU			

Category	KG Total	KG Remaining	LBS Total	Remaining	Detail	Detail Value
Weight	2.080040275	1.919959725	8.8 lbs	4.29910745		
Power (burst)			843.6	783.75	Max battery life	41.29760226
Spending			500	173.4		

Gantt Chart and Re-Testing Schedule



Example Weekly Quad Chart for Weekly Team Meetings



Arduino Code for receiving the Jetson messages:

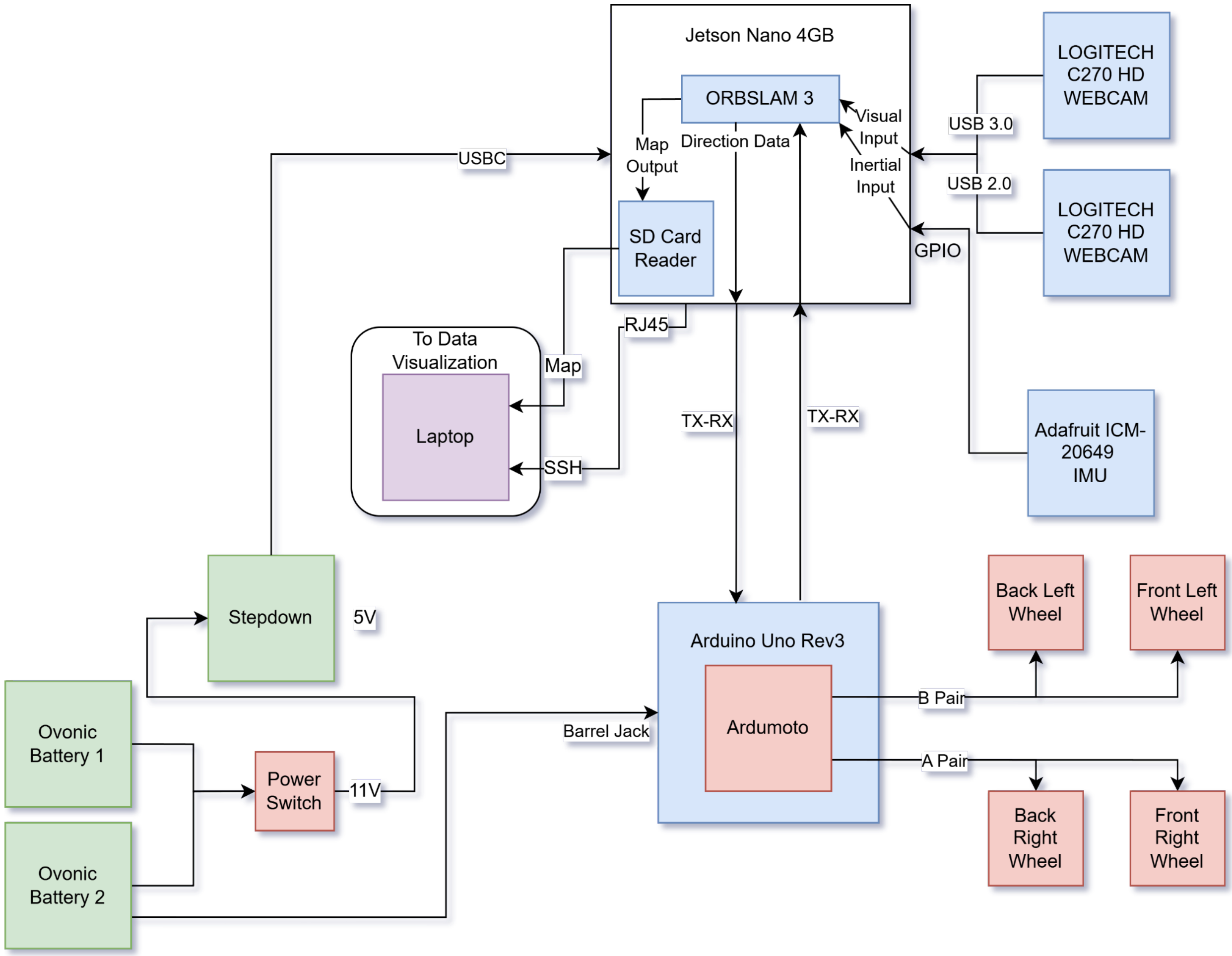
```
// ROS callback for cmd_vel messages.
void cmdVelCallback(const geometry_msgs::Twist &twist) {
  lastCommandTime = millis();
  linear_x = twist.linear.x;
  angular_z = twist.angular.z;
  if (currentState == FOLLOWING_COMMANDS) {
    leftSpeed = (int)((linear_x - angular_z) * MAX_PWM);
    rightSpeed = (int)((linear_x + angular_z) * MAX_PWM);
    leftSpeed = constrain(leftSpeed, -MAX_PWM, MAX_PWM);
    rightSpeed = constrain(rightSpeed, -MAX_PWM, MAX_PWM);
    setMotors(leftSpeed, rightSpeed);
  }
}
```

Arduino logic for ultrasonic-based obstacle avoidance:

```
/*
   Obstacle avoidance routine (avoidObstacle90()):
  1. Determine safe side via side sensor readings.
  2. Back up briefly.
  3. Turn 90° toward the safe side using IMU-based yaw.
  4. Move forward sideways.
  5. Repeat until the path is clear.
  6. Reorient to the original heading.
*/
void avoidObstacle90() {
  float leftDist  = getDistance(TRIG_LEFT, ECHO_LEFT);
  float rightDist = getDistance(TRIG_RIGHT, ECHO_RIGHT);
  bool safeSideLeft = (leftDist > rightDist);

  int netTurn = 0; // Accumulate net turn in degrees.

  while ((getDistance(TRIG_CENTER, ECHO_CENTER) < OBSTACLE_DIST) ||
         (getDistance(TRIG_LEFT, ECHO_LEFT) < OBSTACLE_DIST) ||
         (getDistance(TRIG_RIGHT, ECHO_RIGHT) < OBSTACLE_DIST)) {

    // Step 2: Back up briefly.
    setMotors(-FORWARD_SPEED, -FORWARD_SPEED);
    delay(REVERSE_TIME);
```

```
    stopMotors();
    delay(100);

    // Step 3: Turn 90° toward the safe side.
    if (safeSideLeft) {
      turnByDegrees(90);
      netTurn += 90;
    } else {
      turnByDegrees(-90);
      netTurn -= 90;
    }

    // Step 4: Drive forward sideways.
    setMotors(FORWARD_SPEED, FORWARD_SPEED);
    delay(FORWARD_SIDE_TIME);
    stopMotors();
    delay(100);

    // Update safe side determination.
    leftDist  = getDistance(TRIG_LEFT, ECHO_LEFT);
    rightDist = getDistance(TRIG_RIGHT, ECHO_RIGHT);
    safeSideLeft = (leftDist > rightDist);
  }

  // Step 5: Reorient by counter-turning the net rotation.
  if (netTurn != 0) {
    turnByDegrees(-netTurn);
  }

  // Resume forward motion.
  setMotors(FORWARD_SPEED, FORWARD_SPEED);
  delay(500);
  stopMotors();
}
```

Initial Block Diagram



Final Block Diagram