Ideation Document under Rover Category from Team of Indian Institute of Technology, Gandhinagar (IITGN)

Team Composition

Mentor

—--------

Team Members

1) Arjun Paan, 2nd-year B.Tech, Mechanical Major (23110041@iitgn.ac.in)
2) Malhar Karade, 3rd-year B.Tech, Mechanical Major (malhar.karade@iitgn.ac.in)
3) Srajan Dehariya, 2nd-year B.Tech, Artificial Intelligence Major (23110320@iitgn.ac.in)

Table of Contents

1. Abstract

The IIT Gandhinagar team presents a project for Robofest 4.0, aiming to create a highly capable autonomous rover by leveraging Jetson Nano and Intel Realsense to achieve seamless navigation and efficient task completion in challenging environments. Features include learning from the environment, autonomous driving guided by a camera, versatile mobility, and data collection. Future enhancements aim to integrate a robotic arm for sample gathering. The goal is to develop an autonomous rover with advanced AI capabilities for exploration and robotics, paving the way for enhanced scientific missions in challenging environments.

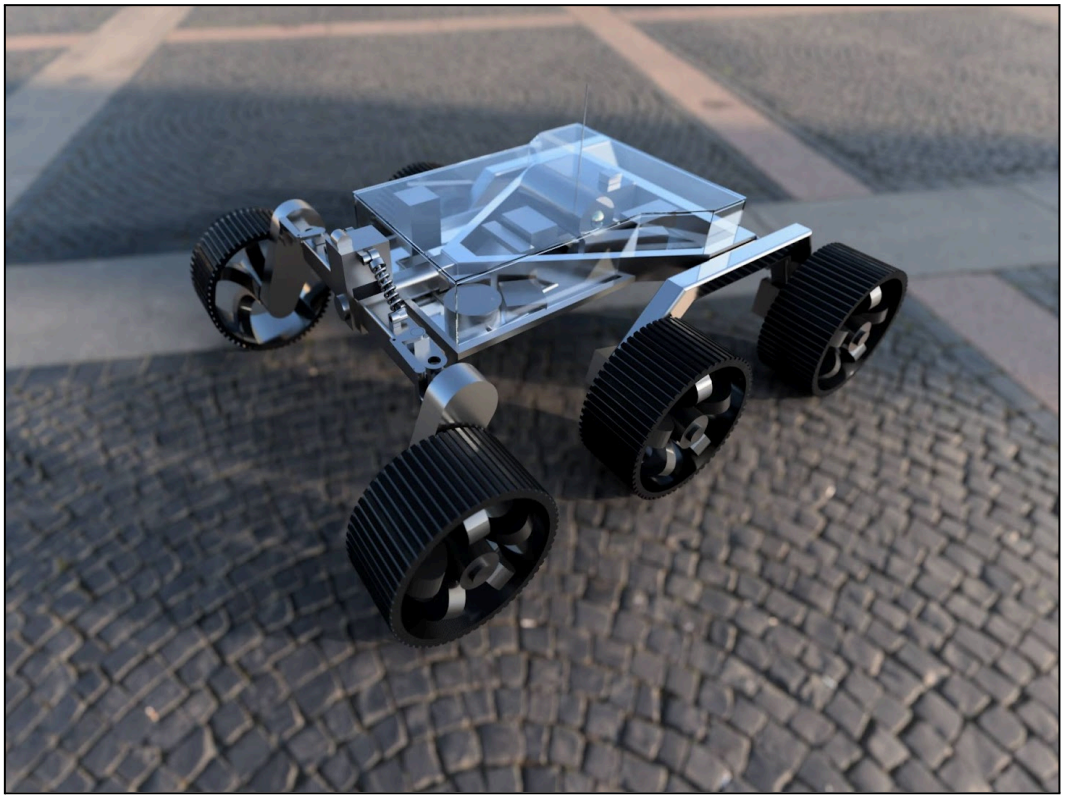


Fig (1) Proposed design of the rover

2. Features

2.1 Mechanical Features

2.1.1 Modified Rocker Bogie mechanism

- The modified Rocker-Bogie suspension system enhances the rover's ability to navigate uneven terrain by flexing and conforming to the ground's contours.
- This flexibility helps maintain stability on tilted surfaces and prevents the rover from getting stuck or toppling.
- It also ensures even weight distribution & avoids any sort of rocking.
- Our structure’s unique feature is having 2 distinct rotation axes for rear & front rockings.
- Unlike the classic mechanism, the rear one rotates about an axis perpendicular to the one about which the front one rotates.
- As per our hit and trial experiments, this mechanism has proven efficient in climbing obstacles of uneven shape and traversing a rough terrain of rocks and debris (an open ground with uneven terrain).

Fig (2) Doosan DA40-5 articulated dump truck (the inspiration for our rover design)

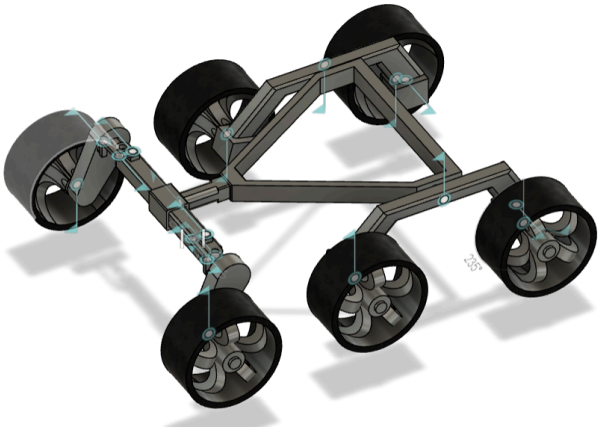


Fig (3) The chassis model

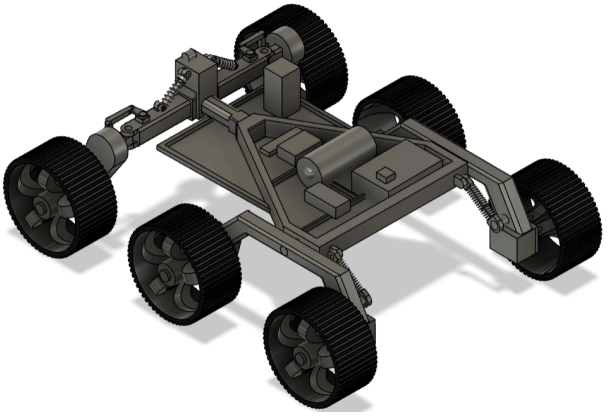


Fig (4) The final assembly

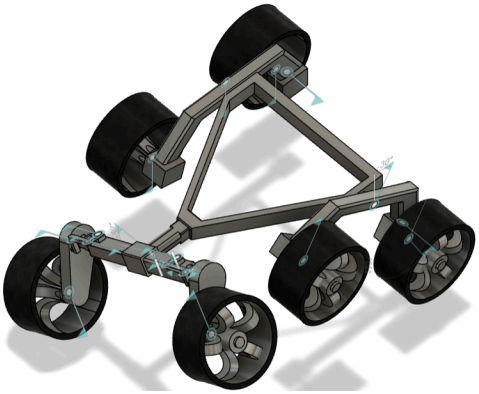


Fig (5) Left rocking (bird view)

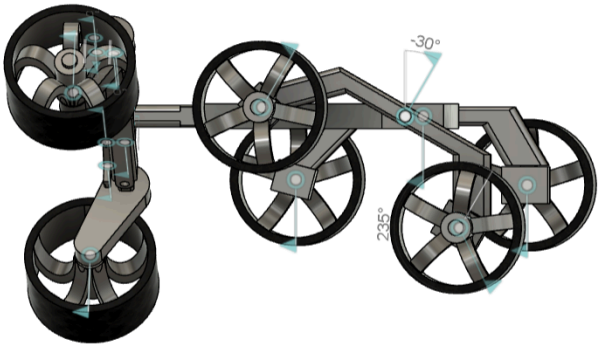


Fig (6) Left rocking (side view)

2.1.2 Navigation mechanism

- The navigation mechanism for the rover is a 6-wheel drive with balloon-type stripped rubber casing (proposed diameter of 10-12 inches (25-30cm)).
- These 6 wheels are supposed to facilitate forward motion, backward motion & on-spot rotations with the power supplied to them by the lipo battery of 12V.
- We've also inculcated a steering mechanism for all 6 wheels, which works on the Ackermann's steering geometry. To actuate that, we've used an MG996R servo motor & a gear mechanism.
- This steering mechanism will allow the rover to move in any direction without turning.

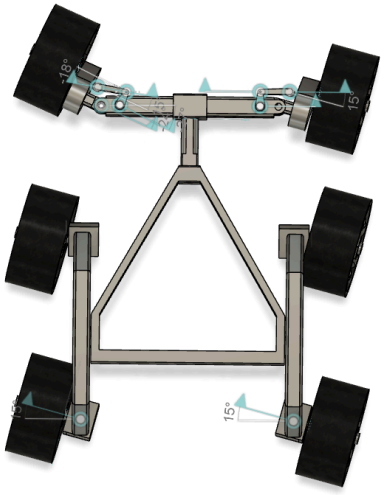


Fig (7) 15° Diagonal orientation

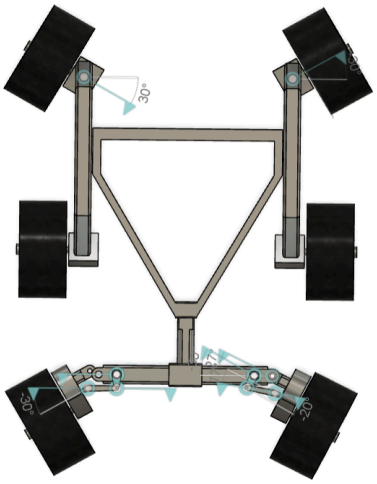


Fig (8) Orientation for In-place rotation

2.1.3 Weight reduction mechanism

- The main chassis is a casing rather than a rigid box-like structure.
- The aluminium extruders are the bones of the rover.
- We've used customised shape acrylic sheets of 3mm as "slabs" to place the components over it. Slabs are supposed to be placed in the grooves of the extrusions.
- All the components are to be screwed over the "slab" using appropriate sort of nuts.

2.1.4 Shock Absorption Technology

- Shock-ups are installed on the axles to absorb the sudden jerks the rover may encounter due to objects in its path.
- These reduce vibrations and ensure a smoother ride over rough terrain.

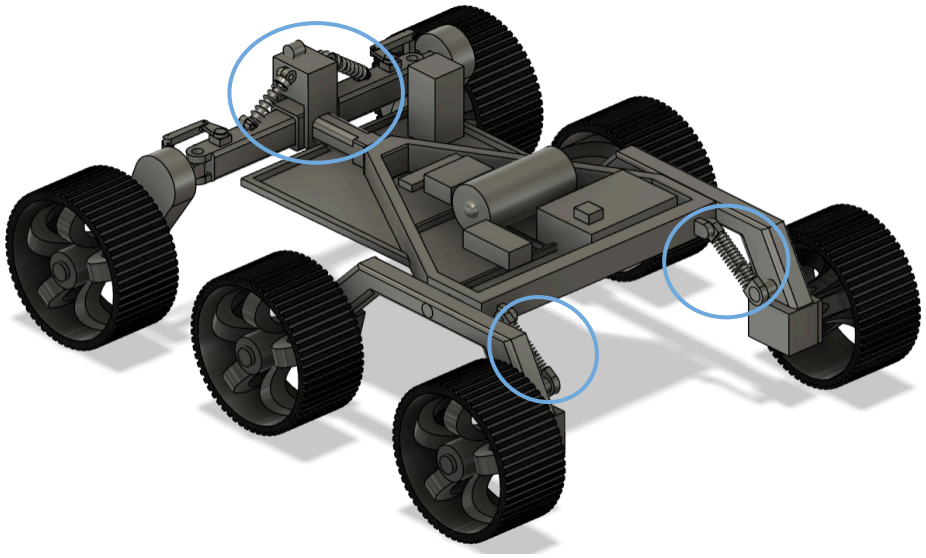


Fig (9) Shock up mechanism

2.2 Technological Features

2.2.1 Electronics

- Lithium batteries of 12V, 30Ah
- Motor driver (L293D)
- PCA 9685
- Intel Realsense Camera
  - To retrieve RGB and Depth data from environment
- Jetson Nano Board
  - For data processing and decision making
- Neo-6M GPS Module
  - To access GPS
- Custom PCB design
  - To add “Plug and Play” functionality
  - To make the circuit compact

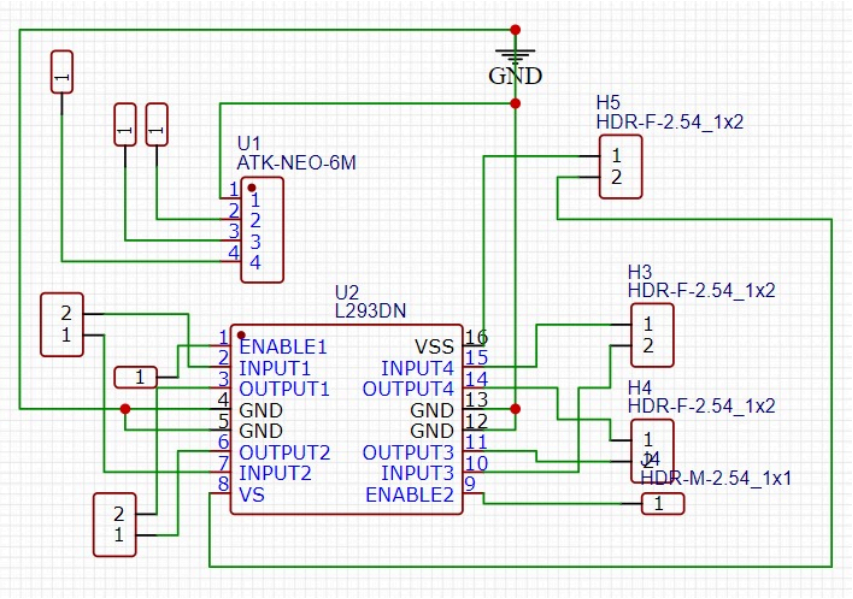


Fig (11) Tentative schematic of the custom PCB that includes GPS module & motor driver

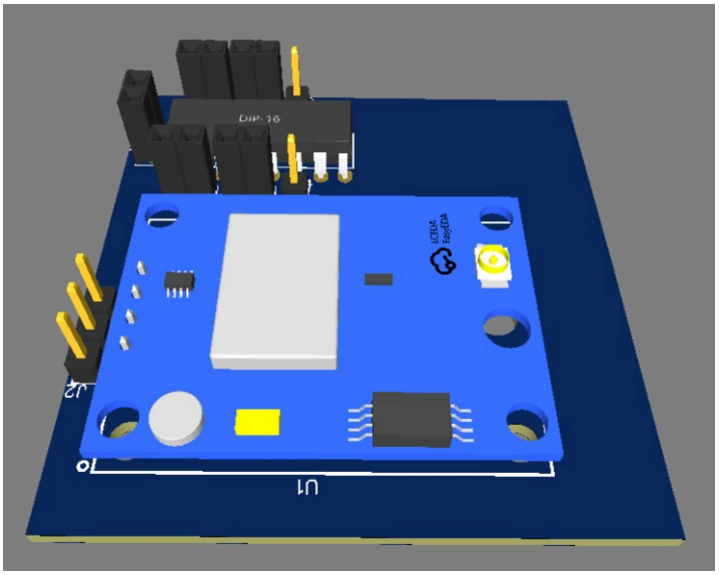


Fig (12) Tentative location of GPS module & motor driver

2.2.2 Algorithm

- YOLOv7 inspires the object detection algorithm.
- It's supposed to utilise a neural network of roughly 10-12 layers.
- The libraries that will be utilised for computer vision and machine learning aspects are PyTorch and OpenCV.
- It's an algorithm written for edged GPUs, so it can run on boards of the Jetson family. The power required for processing is the reason for dropping Raspberry Pi and using Jetson Nano instead.

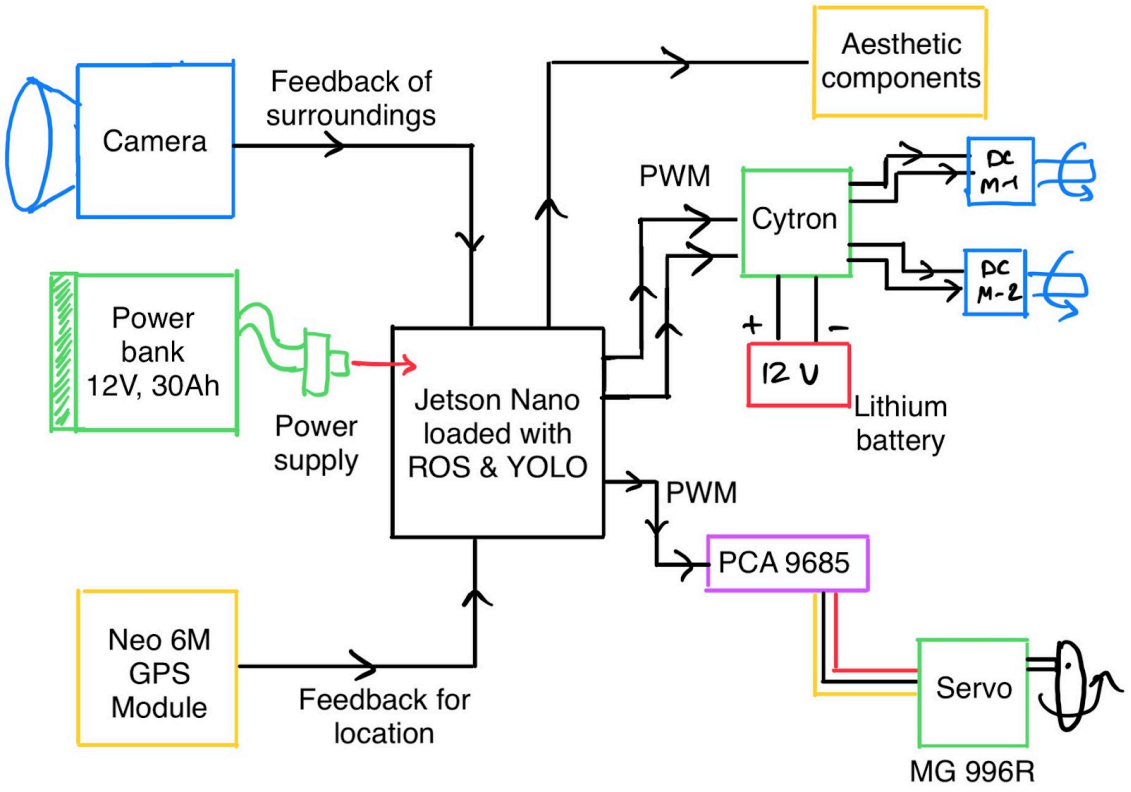


Fig (10) Control signal diagram of the circuit

3. Design

This section develops an intuitive overview of the structural aspects of the proposed Rover.

3.1 Methodology of assembling

3.1.1 Mechanical structure assembly

- The rover's body comprises aluminium extrusions (2cm x 2cm cross-section) of needed dimensions.
- Some aluminium extrusions are welded together, while others are joined using L-shaped brackets.
- For the screwing purposes, we've used the M4 types of nuts & bolts.
- For the felicitation of smooth rotations around the rocker-bogie mechanisms, we've used the pillow block bearings at all the rotation points.
- The axles are made up of steel carbon rods.
- The steering mechanism is a gear-coupled MG996R servo motor. This will help the rover to move in any direction without turning (refer to Fig (7)).
- The shock-ups are attached to the extrusions made using CNC models to transfer a shock directly to something as sturdy as metal.

- The camera is mounted on a pedestal to ensure sufficient ground clearance, allowing for accurate image capture and depth data acquisition from the Intel Realsense camera.
- The entire circuitry is enclosed in a casing made of acrylic sheets to ensure the protection and organisation of components.
- This model is under iterative adjustments based on the experiment's results to refine the rover's capabilities and ensure smooth operation.

3.1.2 Electronics & circuits

- The Jetson Nano is supposed to be powered by a battery of 12V, 30Ah.
- We may need an extra power supply for driving the motor as 6 motors draw a considerable amount of current. As we want the battery to function for as long as possible, we may consider having two individual power sources.
- Another component that will utilize some power will be the GPS module, Neo-6M. As it does not need more than 5V, we can run it on JNano's power supply.
- The rover is supposed to move from one specified point to another while avoiding obstacles. So, the rover will receive control signals from the GPS module and camera.
- Thus, the code is written such that the rover stays on a specified path (with the coordinates from Neo 6M) and moves such that the obstacles are avoided locally with the help of the camera's input.
- The servo motors require a PWM signal to function. So we've used an I²C-bus controlled 16-channel PWM driver that helps us to manage many PWMs (here 6, as we've 6 motors to steer) together, a PCA 9685.
- All the necessary components are compiled over one PCB to make the rover a simple plug-and-play model.

3.2 Description of Materials

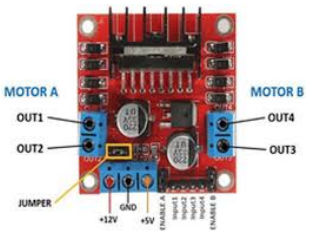
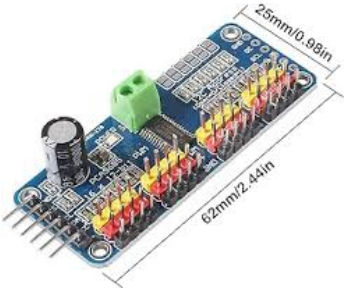
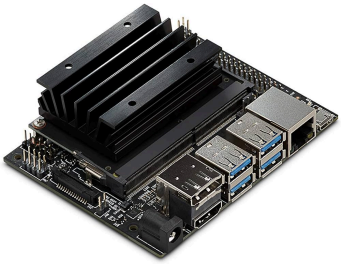
3.2.1 Components for the Mechanical body

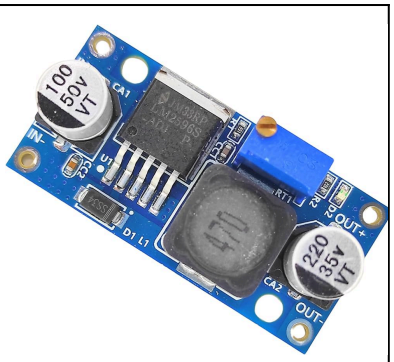
Components and Materials	**Application**	**Specification**
Aluminium extrusions	used for constructing lightweight, durable chassis frames	2cm×2cm cross-section
Slider square-fitted nuts	Provide adjustable and secure attachment points to the extrusions	M4 type sliders
Servo Horns	Transfer torque from servo motors to mechanical linkages to the tiers.	Metallic (aluminium)

L-shaped extrusion connector	Support modular assembly, offering stability and easy connection between extrusions.	Aluminium casted L shaped connector with 4 holes of M4 type
Flange Bearings	For smooth motion and structural support to the servo motor	10mm diameter
Pillow Bearing	For smooth motion and structural support to the extrusion for free rolling	10mm diameter
Steel carbon rod	Attach between the bearing and Hold the load of the chassis	10mm diameter
Bolts	Bolts secure components and structures in rover assembly	M4 type
3D print	Supports for servos and motors and for connectors in the chassis	ABS & PLA
Aluminium sheets	For the motor mounting and used for the 150-degree angle connection of the extrusion	3mm thickness
Acrylic sheet	For Electronics enclosure and final touch	5mm thickness

Note:- The list of mechanical elements is subject to change. We may or may not use the elements listed above or add any if necessary; however, the most probable ones are listed above. Any sort of change will be documented in the repository (link on pg 20)

3.2.2 Electrical Component Description

Components and Materials	Application	Images
Motor driver (L293D)	To precisely control the Motors	
PCA 9685	Controlling pulse width modulation (PWM)	
Intel Realsense Camera	Capture RGB and depth data.	
Jetson Nano Board	Computer for processing data and decision-making	
Neo-6M GPS Module	To aid in navigation using GPS	
Batteries	Power supply (12V, 3000mAh)	

Buck Converter	Step-down transformer (5V DC output)	
Servo motor	Wheel steering (MG996R Servo)	
DC Motors • 60 RPM speed • 38 kg-cm torque	Locomotion of the rover, i.e. for providing speed and torque.	

3.3 Rough dimensions proposed for the proof of concept using CAD

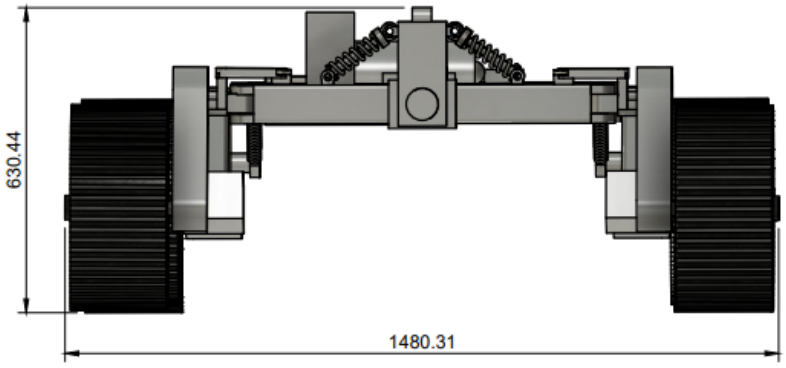


Fig (11) Front view

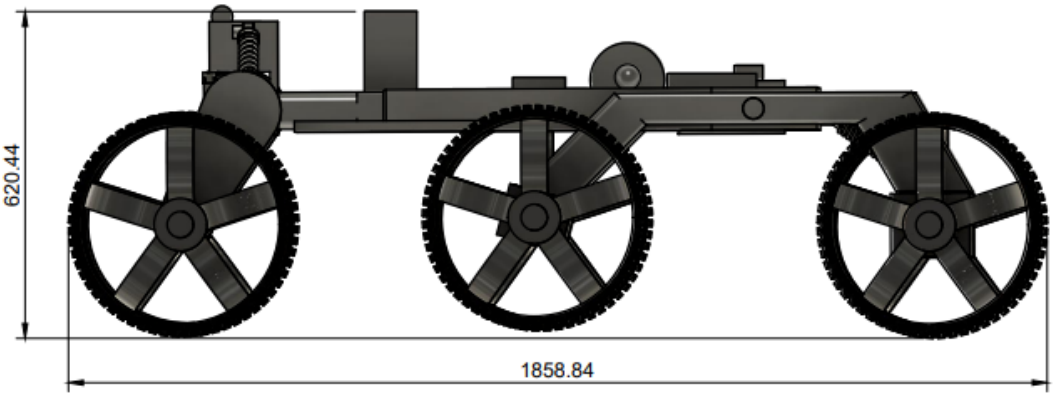


Fig (12) Side view

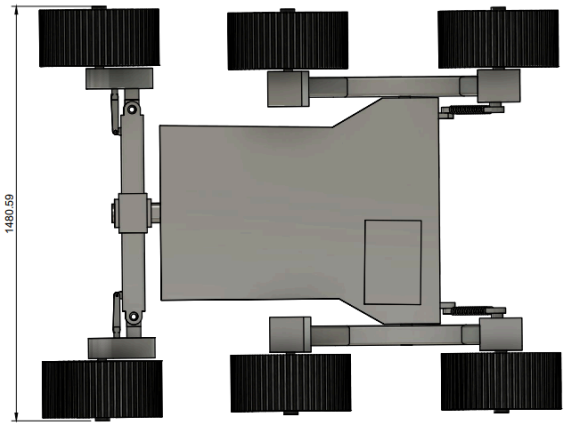


Fig (13) Bottom view

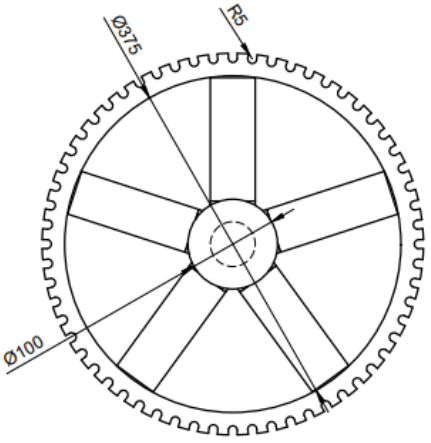


Fig (14) Wheel

3.4 Dimensions of the Model Proposed as Prototype (Actual Version)

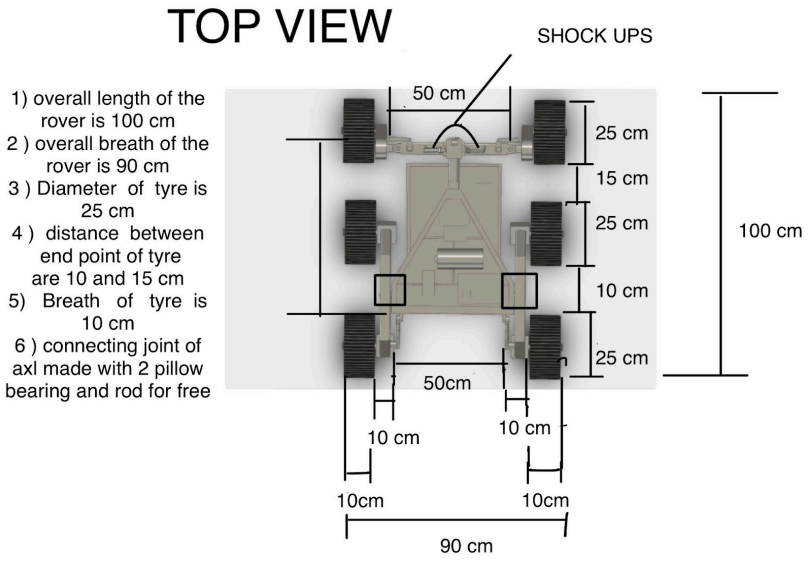


Fig (15) Top view of Rover

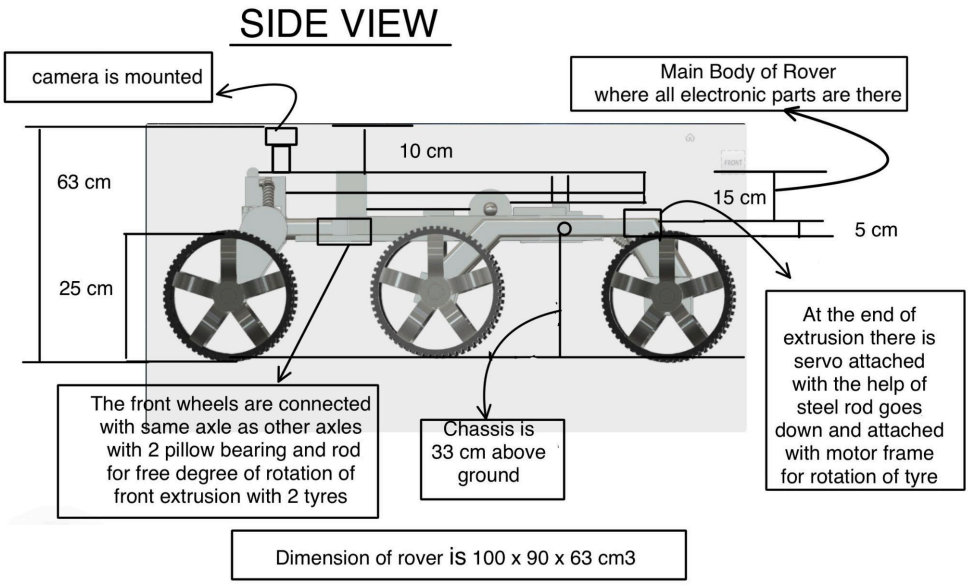


Fig (16) Side view of Rover

4. What's running on Jetson Nano?

NVIDIA® Jetson Nano™ Developer Kit is a small yet powerful AI computer that we propose to use for running our object detection algorithms and other decision-making processes.

It has the following features that favour our use:

1) low power demand of 5W to 10W
2) 472GFLOPS for AI algorithms
3) 128-core NVIDIA Maxwell™ architecture GPU
4) CPU Max Frequency of 1.43GHz
5) small size of 70×45mm
6) 40 GPIO pins for additional peripherals and devices

It also provides a Python library for directly accessing the GPIO pins [1].
Below is a sample program for detecting an event from GPIO pin 4 and giving output to pin 18

```
import Jetson.GPIO as GPIO

GPIO.setmode(GPIO.BOARD)  # setup board for accessing pins them
out_channel = 18  # output pin
in_channel = 4    # input pin
# setting out_channel pin to HIGH
GPIO.setup(out_channel, GPIO.OUT, initial=GPIO.HIGH)

# adding event detection
GPIO.add_event_detect(in_channel, GPIO.BOTH)

state = 0
if GPIO.event_detected(in_channel):
      if state:
            GPIO.output(out_channel, GPIO.HIGH)
            state = 0
      else:
            GPIO.output(out_channel, GPIO.LOW)
```

4.1 Libraries

1. Numpy: For efficient numerical computation
2. SciPy: For scientific computation
3. Matplotlib: For creating interactive visualisations
4. Pyrealsense2: Python wrapper for Intel Realsense Camera
5. PyTorch: For building and training deep learning models
6. OpenCV: Computer vision library

4.2 Pseudo Codes

All the codes mentioned in this section are sample codes to demonstrate basic functionality.

1. DC Motor drive
    - This code is a sample code for switching on & off a pin on the board.
    - The code demonstrates digital signals (HIGH or LOW, nothing else).
    - To inculcate the in-between values (analogue output), we can change the frequency & duty cycle of the input wave with appropriate delays (using time. sleep(XYZ)).

```
import Jetson.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)

out = 16   # PWM output pin
GPIO.setup(button1, GPIO.HIGH)

try:
      while True:
            GPIO.output(out, GPIO.HIGH)
            time.sleep(2)
            GPIO.output(out, GPIO.LOW)
            time.sleep(2)
except KeyboardInterrupt:
      GPIO.cleanup()
```

Fig(24) On-off of 16th GPIO Pin with a time period of 4 sec

2. Servo motor using PCA 9685
    - PCA9685 driver
    - Can direct **16 vivid PWM** signals into different slots.
    - Thus, we can choose to operate **16 different servo motors** on a maximum of one PCA9685.
    - Although we may not need as many as 16, discussing how to handle 16 or more is essential.
    - The code below shows how to operate the servo connected to the 1st position on PCA9685. Using the exact mechanism, we can drive the remaining motors (change index 0 to need one, a number between 0 and 15).
    - If one needs to handle more than 16 motors, the person may choose to solder 2 PCA9685 together & add channels 32 instead of 16 in the ServoKit() function of the code.

```
from adafruit_servokit import ServoKit
import time

pca = ServoKit(channels=8)

pca.servo[0].angle = 0

for i in range(100):
      pca.servo[0].angle = i
      time.sleep(0.1)

for i in range(180, 0, -1):
      pca.servo[0].angle = i
      time.sleep(0.1)
```

Fig (25) Sweeping a servo attached to the 1st channel of PCA9685 using the PWM from Jetson

3. GPS ModuleGPS Module
    - The GPS module can provide much information regarding the object's position to which it's attached.
    - Out of that whole chunk of information, **we need to retrieve only latitude & longitude coordinates.**
    - The following code is for the same.

```
import serial
import time
import pynmea2

while True:
      port = "501"
```

```
      s = serial.Serial(port, 9600, 0.8)
      data = s.readline()
      if data[0:6]==”$GPGLL”:
            location = pynmea2.parse(data)
            latitude = location.latitude()
            longitude = location.longitude()
            print(“Latitude:”,latitude,”\tLongitude”,longitude)
```

Fig(26) Retrieving coordinates from the GPS module

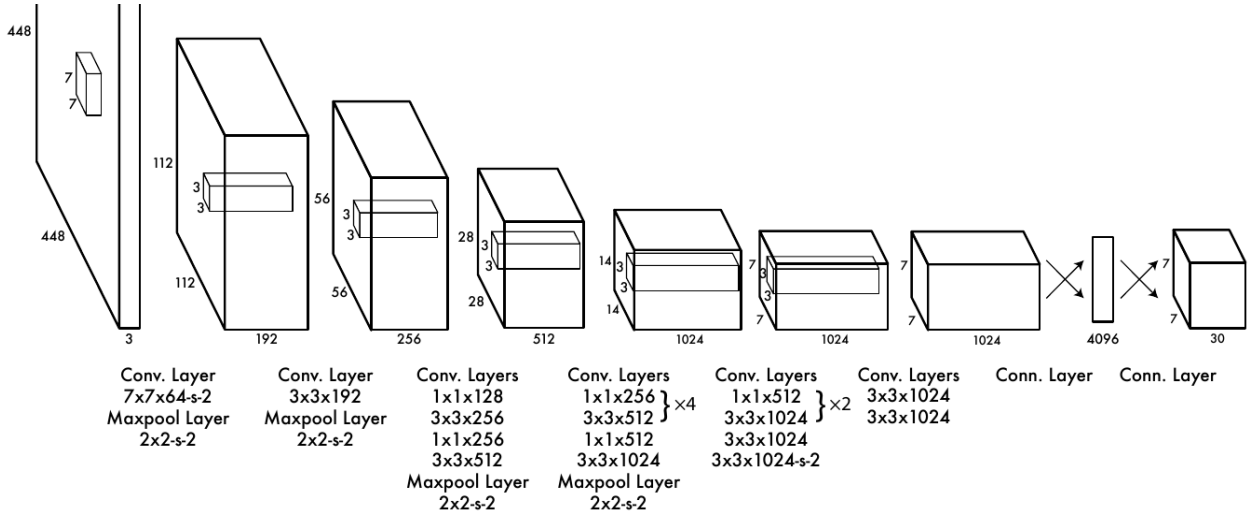
4. Data acquisition from Intel’s Realsense camera
    - Intel’s Realsense is a camera that is capable of doing depth analysis in addition to color analysis (RGB data).
    - As we are recording a video, we need to get each frame and then extract RGB & depth data from each of it & pass for further analysis.
    - The code below extracts that data from the videos & displaying them on a screen as remote video windows.

```
import pyrealsense2 as rs
import numpy as np
import cv2

pipe = rs.pipeline()
cfg = rs.config()
cfg.enable_stream(rs.stream.color, 640, 480, rs.format.bgr8, 30)
cfg.enable_stream(rs.stream.depth, 640, 480, rs.format.z16, 30)
pipe.start(cfg)

while True:
      frame = pipe.wait_for_frames() # object corresponding to
      each frame
      depth_frame = frame.get_depth_frame() # object corresponding
      to depth data of each frame
      color_frame = frame.get_color_frame() # object corresponding
      to color/RGB data of each frame

      # Conversion to numpy arrays
      depth_img = np.asanyarray(depth_frame.get_data())
      color_img = np.asanyarray (color_frame.get_data())
```

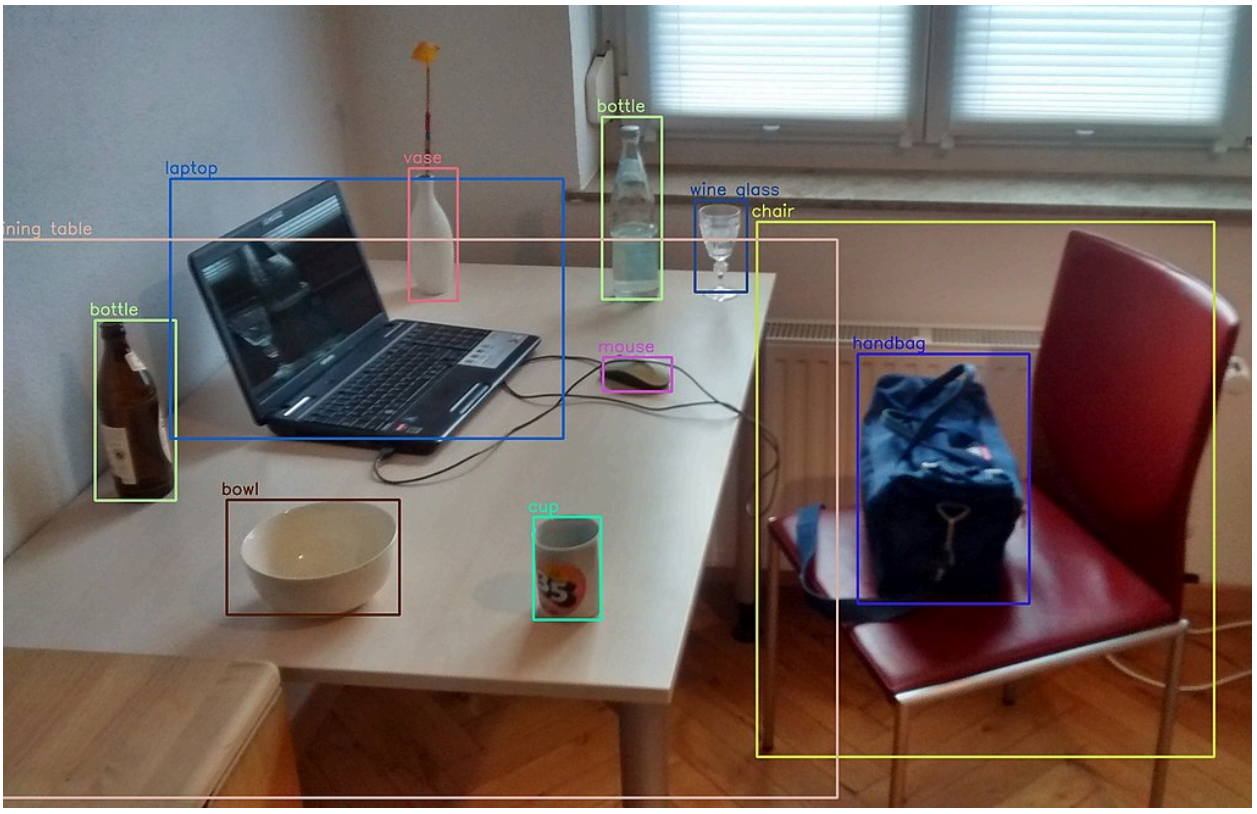
```
        # Applying color map to a binary depth image
        depth_colormap = cv2.applyColorMap(cv2.convertScaleAbs
        (depth_img, alpha = 0.5), cv2.COLORMAP_JET)

        # Showing the image on the screen
        cv2.imshow("RGB image", color_img)
        cv2.imshow("Depth data", depth_colormap)

        # Quitting the program
        if cv2.waitKey(1) == ord('q'):
        break
pipe.stop()
```

4.3 YOLOv7

YOLOv7 is one of the fastest-known algorithms for object detection [2]. It uses a very efficient architecture of convolution neural networks and other mathematical operations for object detection.



Typical Neural Network architecture of YOLO [3]



Example of Object detection using YOLO [4]

The significant advantages of YOLOv7 over other models are the ease of use, fast inference speed, and the ability to detect multiple objects in a single frame with confidence values.

5. Applications

- Self-guided navigation & AI abilities allow the rover to traverse various landscapes, enabling exploration beyond human reach.
- Data collection ability in the form of digital & image-storing capabilities allows it to expand the domains of space exploration.
- It can be used for mountain climbing and transport supplies in the camps
- Rover can be used for surveillance and monitoring purposes in sensitive or high-risk areas. It can patrol perimeters, monitor activity, and detect intrusions, enhancing security measures in various environments.
- In disaster-stricken areas, the rover can assist in search and rescue operations by navigating through debris and hazardous terrain to locate survivors. It can also deliver essential supplies and equipment to inaccessible locations.
- Potential places of use are:-
    - Space explorations
    - Rescue missions in case of calamities & disasters
    - Sample collection from challenging terrains
    - Military uses such as delivery bots & patrolling
    - Construction sites

    - Agricultural uses such as monitoring, plantation & harvesting
    - Mobile transport systems in internal spaces such as offices & townships

Description

The proposed rover offers various applications across various industries and fields. From environmental monitoring and disaster response to agricultural support and surveillance, the rover's versatility and advanced capabilities make it a valuable asset in diverse scenarios. Whether navigating through rugged terrain or conducting scientific research, the rover demonstrates its potential to contribute to exploration, innovation, and problem-solving in real-world contexts.

6. Repository

Github repository link

- The repository will be a storage space for all essential files, including CAD models, 3D printing outputs, Jetson code, object detection algorithms, circuit diagrams, and other required materials.
- This also allows us to collaborate efficiently with the team.
- This repository will be maintained throughout the project.

Timeline of Our Project

March 1 - March 10

Initial team meetings
Defining project scope
Preliminary research

March 16 - March 31

Selection of electronic components
CAD modeling and initial design sketches

April 1 - April 15

Detailed design of mechanical structures
Begin assembly of mechanical parts

April 16 - April 30

Refining and revising ideation document
Incorporating research, trials, and corrections

May 1 - May 15

Integration of electronics
Focus on enhancing autonomous capabilities using Jetson Nano, Intel Real sense, and YOLOv7

May 16 - May 31

Assembly and testing of object detection systems
Preparation for navigation and coordinate plotting with GPS module

June 1 - June 15

Final refinements to ideation document
Submission of ideation document for Level 1 of Robo fest 4.0

June 16 - June 30

Preparing proof of concept video
Finalizing electrical setup for autonomous navigation

July 16 - July 31

Submission of proof of concept video
Level 2 Comprehensive system testing and debugging

July 16 - July 31

Final round preparation
Comprehensive rover task performance testing

7. References

[1] NVIDIA, “GitHub - NVIDIA/jetson-gpio: A Python library that enables the use of Jetson’s GPIOs,” *GitHub*, 2019. https://github.com/NVIDIA/jetson-gpio (accessed Jun. 12, 2024).

[2] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, “YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors,” *arXiv.org*, 2022. https://arxiv.org/abs/2207.02696 (accessed Jun. 12, 2024).

[3] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You Only Look Once: Unified, Real-Time Object Detection,” *arXiv.org*, 2015. https://arxiv.org/abs/1506.02640 (accessed Jun. 12, 2024).

[4] “File:Detected-with-YOLO--Schreibtisch-mit-Objekten.jpg - Wikimedia Commons,” *Wikimedia.org*, 2022. https://commons.wikimedia.org/wiki/File:Detected-with-YOLO--Schreibtisch-mit-Objekten.jpg (accessed Jun. 12, 2024).