Summer Project Review

Rover

Abstract-

The main objective is to make a remote-controlled Rover. Its goal is to move over various terrains and to overcome different obstacles in its path like footpaths or a pit. It can go in all directions ie front, back, right, left and clockwise and anticlockwise. Many more features will be added throughout the making of this. It is used in planetary space exploration and in places where humans can't go.

Introduction-

Basic design:

Rover to be made with Rocker Bogie Mechanism with 6 wheels to overcome obstacles on the path like a pit or a stone and the setup was done in a way that each motor could individually lift/pull a significant portion of the total weight of the robot. A differential bar has also been added in the project so as to provide more traction for wheel and no tilt occurs. This will most likely need powerful and high torque DC motors. A Li-Po battery to power the rover for journey. All electronic items in a box inside rover under two border of PVC pipes in between spacers of same material. Plates, screws, nuts, bolts and washers etc and many more items of various dimensions.

Dimensions and Specs of the rover-

height-370 mm

length-480 mm

Width-526 mm

Mass- 800g

Avg Speed- 0.30 Km/hr

Run Time- 2 hour (continuous)

Process:

Electronics:

* Arduino UNO will be used as microcontroller.
* And will be connected by L298 Motor Driver Module for connecting all motors to arduino
* Breadboard for all stuff to connect
* Jumper wires for connecting and
* 10m long wire for connecting all stuff.
* And **1000mAh 3S 40C/80C Lithium polymer battery Pack** to power the whole thing. its voltage is 11V.
* Servo Motor to move ultrasonic sensor
* Ultrasonic sensor for RADAR to detect object on our path
* Bluetooth sensor to controling and moving it.

Hardware:

* U-PVC Has been used for all body particularly rocker boogie in different sizes and 45’ and 90’ connecters to connect them throughout.
* Screws, nuts-bolts, washers etc.
* As strong motors have to be used so 6 X 30 rpm 12V DC Johnson Geared Motor of high torque will be used to move wheels over different terrains.

Rated torque- 1000rpm

Stall torque- 254 rpm

* Wheels- Rubber body wheels for greater traction and grip which can overcome different terrains. Its dimensions are:

125 mm diameter and

60 mm width for

Software used:

* Fusion-360
* Arduino UNO
* Bluetooth App

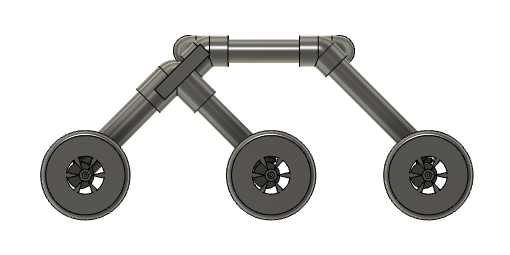
Process Used:

* Drilling
* Sawing
* Cutting
* Soldering
* Coding
* Assembling

Rocker Boogie Suspension:

A rover is considered to have a high degree of mobility in natural terrain if it can surmount obstacles that are large in comparison to the size of its wheels.

S it must have enough traction from its rear wheels to push the front wheels against an obstacle with enough force so that they can climb up it. Typically, a four wheeled rover cannot climb obstacles larger than a wheel radius because the rear wheels do not have enough traction. Without traction the wheels will slip and there will not be enough forward thrust to keep the front wheels in contact with the obstacle. The rocker bogie suspension can surmount obstacles head on that are larger than a wheel diameter because it uses an extra set of wheels to provide more forward thrust. The extra wheels also reduce the normal force on each wheel by about 1/6 the weight of the rover. Less forward thrust is required because the front wheels only have to lift





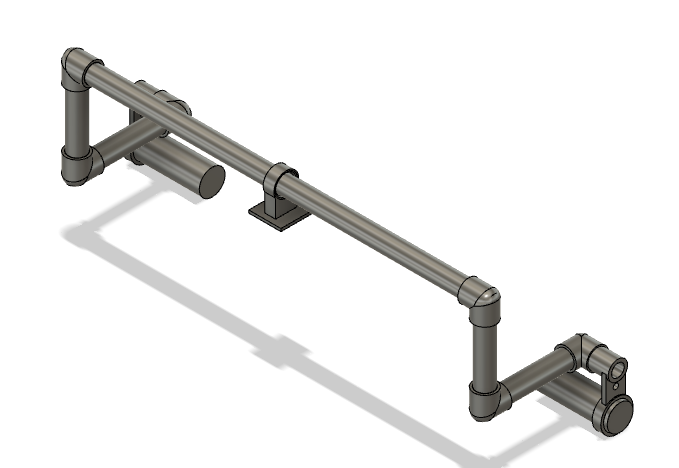
Mass distribution per wheel with the help of each rover is M/6

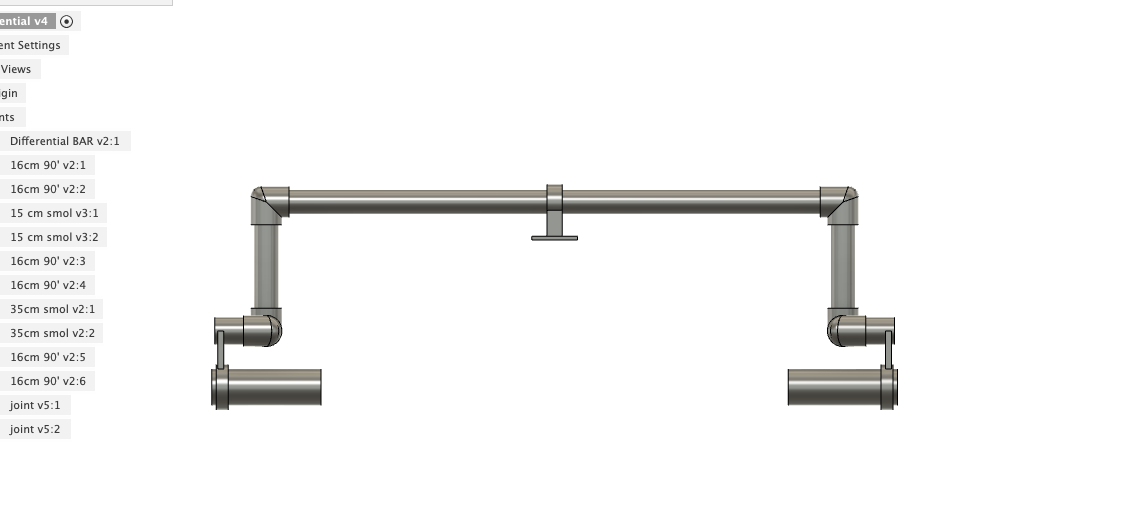
So rover Mass is 800gm then

800/6=134g per wheel which is quite manageable

Differential Bar-

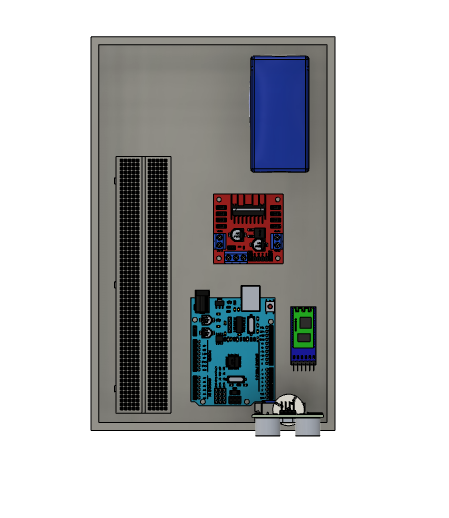
The middle of the bar is connected to the body with a pivot and the two ends are connected to the two rockers through some short links. If you hold the model rover body steady in midair and tilt one rocker up, one end of the bar will go back, the other end will go forward, and the other rocker will tilt down.It’s purpose is to stabilize the rover when traversing rough terrain. If either of the shafts rotated, then it caused the other shaft to rotate in the opposite direction as shafts were connected to each side of the rocker-bogie suspension system. If the front wheel on the left side of the rover went upward over a large obstacle, then the differential would push the three wheels on the right side downward, and vice versa.





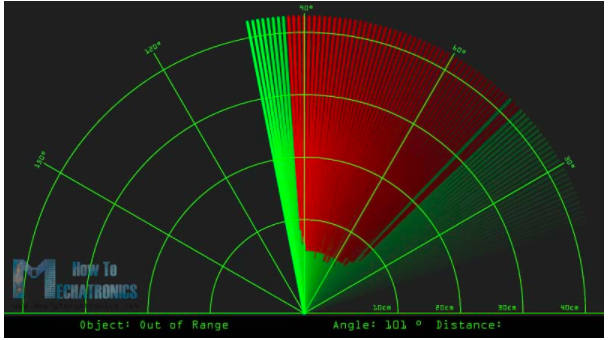
Electricals:

A box for electrical items is used to keep it safe from above world inside 2 PVC pipes bordering. Arduino Uno microcontrollers has been used to control which will control wheels with dual control motor driver and with the help of Bluetooth sensor it will be able to control it through a mobile app and will act as a remote-controlled vehicle. A servo motor has also been added below of a ultrasonic sensor will be used for a RADAR system for which Arduino code is given below.



RADAR system:

A simple Ultrasonic sensor will rotate with the help of Servo motor which will map out obstacles in the path and will give direct feed on PC. The code for it is given below:



Calculations:

To find torque

F= Force

M=mass of rover in grams

g=earth’s gravity in m/s^2

F=(Mxg)/6

= (800 g x 9.82 m\*s^-2)/6

= 1.30933 N

F= force on wheels in N

r= radius of wheel in cm

T= Torque

T= F x r

= 1.30933 x 6.25 N-cm

= 8.183 N-cm

So torque per wheel is \*.183 N-cm

Slope:

height-370 mm

length-480 mm

Width-526 mm

Angle of tilt-: tan^-1 (y/x)

tan^-1(370/480)= 37.6’

The maximum slope angle that rover can theoretically climb is approximately 40 degree. This assumes that the centre of gravity is near the top of the body. Once tilts past this angle it will flip over and will not be able to over it. It can drove up 38o − 39o slopes but eventually slid back down due to lack of traction on the wheels.

Software:

Arduino code-

For moving the rover via

For RADAR-

import processing.serial.\*; // imports library for serial communication

import java.awt.event.KeyEvent; // imports library for reading the data from the serial port

import java.io.IOException;

Serial myPort; // defines Object Serial

// defubes variables

String angle="";

String distance="";

String data="";

String noObject;

float pixsDistance;

int iAngle, iDistance;

int index1=0;

int index2=0;

PFont orcFont;

void setup() {

size (1920, 1080);

smooth();

myPort = new Serial(this,"COM4", 9600); // starts the serial communication

myPort.bufferUntil('.'); // reads the data from the serial port up to the character '.'. So actually it reads this: angle,distance.

orcFont = loadFont("OCRAExtended-30.vlw");

}

void draw() {

fill(98,245,31);

textFont(orcFont);

// simulating motion blur and slow fade of the moving line

noStroke();

fill(0,4);

rect(0, 0, width, 1010);

fill(98,245,31); // green color

// calls the functions for drawing the radar

drawRadar();

drawLine();

drawObject();

drawText();

}

void serialEvent (Serial myPort) { // starts reading data from the Serial Port

// reads the data from the Serial Port up to the character '.' and puts it into the String variable "data".

data = myPort.readStringUntil('.');

data = data.substring(0,data.length()-1);

index1 = data.indexOf(","); // find the character ',' and puts it into the variable "index1"

angle= data.substring(0, index1); // read the data from position "0" to position of the variable index1 or thats the value of the angle the Arduino Board sent into the Serial Port

distance= data.substring(index1+1, data.length()); // read the data from position "index1" to the end of the data pr thats the value of the distance

// converts the String variables into Integer

iAngle = int(angle);

iDistance = int(distance);

}

void drawRadar() {

pushMatrix();

translate(960,1000); // moves the starting coordinats to new location

noFill();

strokeWeight(2);

stroke(98,245,31);

// draws the arc lines

arc(0,0,1800,1800,PI,TWO\_PI);

arc(0,0,1400,1400,PI,TWO\_PI);

arc(0,0,1000,1000,PI,TWO\_PI);

arc(0,0,600,600,PI,TWO\_PI);

// draws the angle lines

line(-960,0,960,0);

line(0,0,-960\*cos(radians(30)),-960\*sin(radians(30)));

line(0,0,-960\*cos(radians(60)),-960\*sin(radians(60)));

line(0,0,-960\*cos(radians(90)),-960\*sin(radians(90)));

line(0,0,-960\*cos(radians(120)),-960\*sin(radians(120)));

line(0,0,-960\*cos(radians(150)),-960\*sin(radians(150)));

line(-960\*cos(radians(30)),0,960,0);

popMatrix();

}

void drawObject() {

pushMatrix();

translate(960,1000); // moves the starting coordinats to new location

strokeWeight(9);

stroke(255,10,10); // red color

pixsDistance = iDistance\*22.5; // covers the distance from the sensor from cm to pixels

// limiting the range to 40 cms

if(iDistance<40){

// draws the object according to the angle and the distance

line(pixsDistance\*cos(radians(iAngle)),-pixsDistance\*sin(radians(iAngle)),950\*cos(radians(iAngle)),-950\*sin(radians(iAngle)));

}

popMatrix();

}

void drawLine() {

pushMatrix();

strokeWeight(9);

stroke(30,250,60);

translate(960,1000); // moves the starting coordinats to new location

line(0,0,950\*cos(radians(iAngle)),-950\*sin(radians(iAngle))); // draws the line according to the angle

popMatrix();

}

void drawText() { // draws the texts on the screen

pushMatrix();

if(iDistance>40) {

noObject = "Out of Range";

}

else {

noObject = "In Range";

}

fill(0,0,0);

noStroke();

rect(0, 1010, width, 1080);

fill(98,245,31);

textSize(25);

text("10cm",1180,990);

text("20cm",1380,990);

text("30cm",1580,990);

text("40cm",1780,990);

textSize(40);

text("Object: " + noObject, 240, 1050);

text("Angle: " + iAngle +" °", 1050, 1050);

text("Distance: ", 1380, 1050);

if(iDistance<40) {

text(" " + iDistance +" cm", 1400, 1050);

}

textSize(25);

fill(98,245,60);

translate(961+960\*cos(radians(30)),982-960\*sin(radians(30)));

rotate(-radians(-60));

text("30°",0,0);

resetMatrix();

translate(954+960\*cos(radians(60)),984-960\*sin(radians(60)));

rotate(-radians(-30));

text("60°",0,0);

resetMatrix();

translate(945+960\*cos(radians(90)),990-960\*sin(radians(90)));

rotate(radians(0));

text("90°",0,0);

resetMatrix();

translate(935+960\*cos(radians(120)),1003-960\*sin(radians(120)));

rotate(radians(-30));

text("120°",0,0);

resetMatrix();

translate(940+960\*cos(radians(150)),1018-960\*sin(radians(150)));

rotate(radians(-60));

text("150°",0,0);

popMatrix();

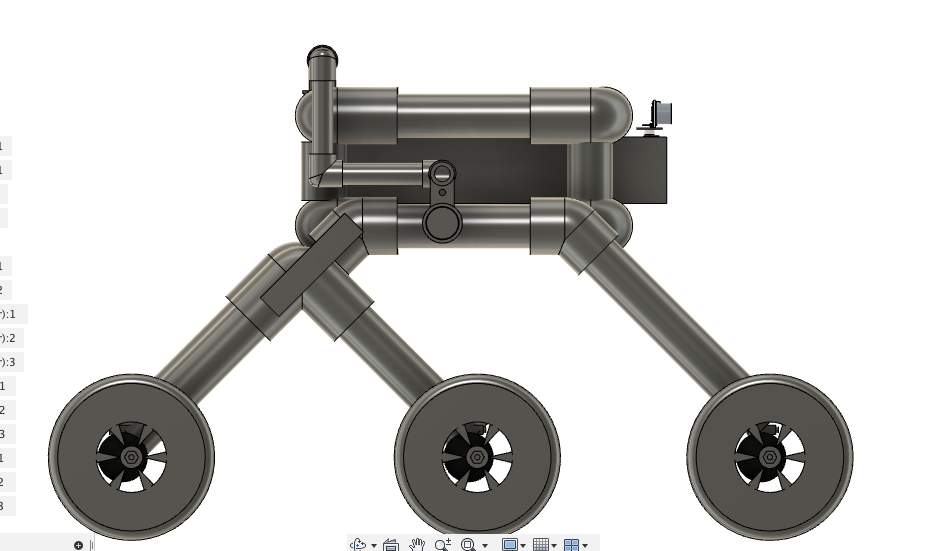
}

Final Product:

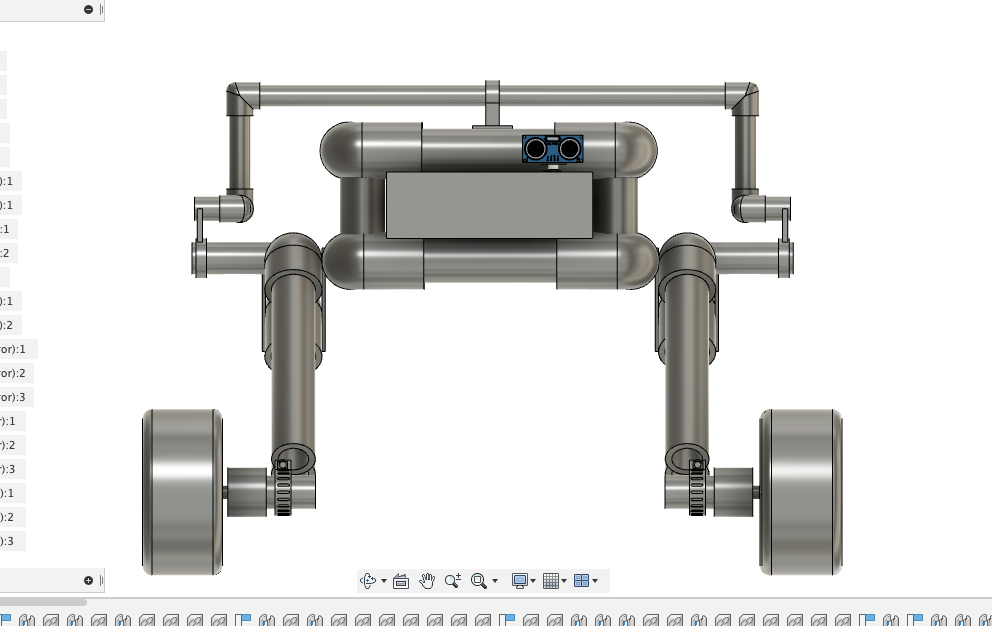
Orthogonal View:



Side View:



Front View



Cost of the Project – Still in process

Conclusion-

With this as a first Major project it helped me a lot with learning lots of things in all the major fields of Mechanical, Electrical and Computer Science. I hope it will help me in all the subjects and help me grasp the subject of Engineering a little better. All

Future Additions- I plan to add many other things in this like a solar panel to charge battery, gripper arm, LCD screen and Live feed Camera along with it if possible.