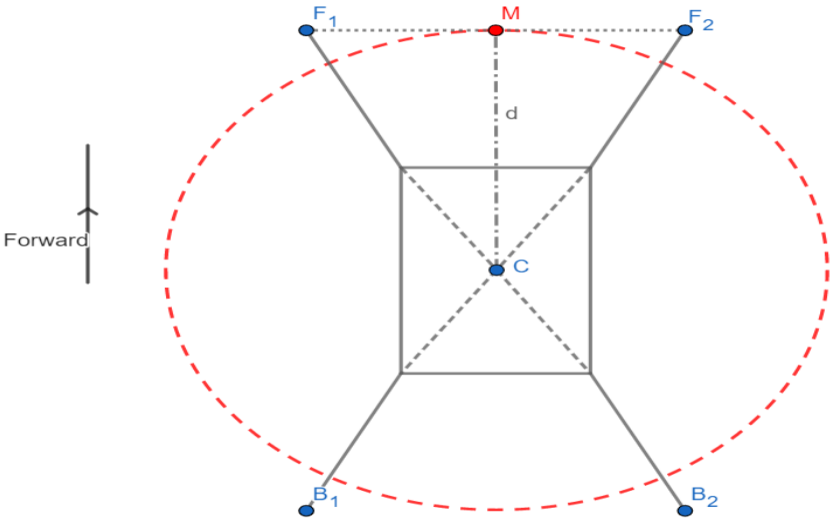
Introduction :-

Airy - 0 : Is a crater inside the larger Airy Crater on Mars, whose location defines the position of the prime meridian of that planet. It is about 0.5 km (0.3 mile) across and lies within the region Sinus Meridiani. This definition maintains the position of the center of Airy-0 at 0° longitude, within the tolerance of current cartographic uncertainties.

The Airy-0 location will be used as an alignment point for all of the autonomous missions.

Navigation and Movement:-

Face Theory :-



Dia: 1

Diagram Symbols:-

F_1 = Front rotor 1

F_2 = Front rotor 2

B_1 = Back rotor 1

B_2 = Back rotor 2

C = Centre point of the UAV

M = Dynamic midpoint of the line F_1F_2

d = distance between the centre C and the midpoint M

Direction Arrow = Forward pointing in the direction of the front of the UAV

Due to absence of GPS technology and no sole magnetosphere, it is fairly difficult to estimate the direction and the position of the vehicle dynamically. Hence we introduce the concept of the face of the UAV. The midpoint M of the line F_1F_2 is taken as the face of the UAV. In all cases this will be the point that has to be rotated in order for the UAV to change its orientation and face the new coordinate to which it needs to travel. The circle with centre C and radius d is the track on which the UAV will rotate at its position. Hence it is imperative to calculate the value of 'd'. This value is a mechanical value and not a geometric value hence this point can be assumed and calculated on any type of vehicle design. Important note is that this value should always be calculated keeping in mind the front direction of the UAV, i.e. the forward direction in which the UAV moves (forward arrow mentioned in the diagram).

Conversion of Latitude and Longitude to cartesian coordinates:

Formula:-

x = R * cos (latitude) * cos (longitude) y = R * cos (latitude) * sin (longitude) **1.1**

where,

R= radius of planet in consideration(for mars=3389.5 km), Latitude and Longitude in degrees.

Distance Calculation:

To calculate distance between two coordinate points, we use the **Haversine formula**

Definition of Haversine formula:-

The **Haversine** formula calculates the shortest distance between two points on a sphere using their latitudes and longitudes measured along the surface.

The below algorithm explains how the Haversine formula works:-

```
var lat1, lat2, lon1, lon2
dLat = (lat2 - lat1) * math.pi / 180.0
dLon = (lon2 - lon1) * math.pi / 180.0
lat1 = (lat1) * math.pi / 180.0
lat2 = (lat2) * math.pi / 180.0
a = (pow(math.sin(dLat / 2), 2) + pow(math.sin(dLon / 2), 2)* math.cos(lat1) * math.cos(lat2))
rad = 3389.5
c = 2 * math.asin(math.sqrt(a))
finaldist = rad * c
```

1.2

Here, dLat is the distance between two latitudes and dLon between two longitudes

lat1 and lat2 are the latitude and longitude converted into radians

a is the haversine angle converted into sin form

c is the inverse haversine angle formula, final dist is the distance between the 2 coordinates.

Why the **Haversine Formula** over the **Vincenty Formula** ?

The Haversine formula is easy for computation but it's less accurate like the vicinity formula is. The haversine formula although gives output much easier than vincenty because the vincenty is more computationally intensive and will consume more battery and performs slower than haversine formula. Due to this, we chose the haversine formula over the vincenty formula.

Direction Calculation:-

Determining the Location of the face of the UAV :

To determine the coordinate of X_1 , we use section formula to find the coordinate of any point on the line between point A and B.

By using section formula:-

$$x = \frac{mx_2 + nx_1}{m+n} \qquad y = \frac{my_2 + ny_1}{m+n} \qquad \textbf{1.3}$$

In our case, we considered the length of line AX_1 as **m=d,** the d is the very same distance from the diagram 1, hence the length of line X_1A i.e **'n'** will be :-

n = total length of AB - d

The Algorithm shows the stepwise calculation of point and its coordinates:-

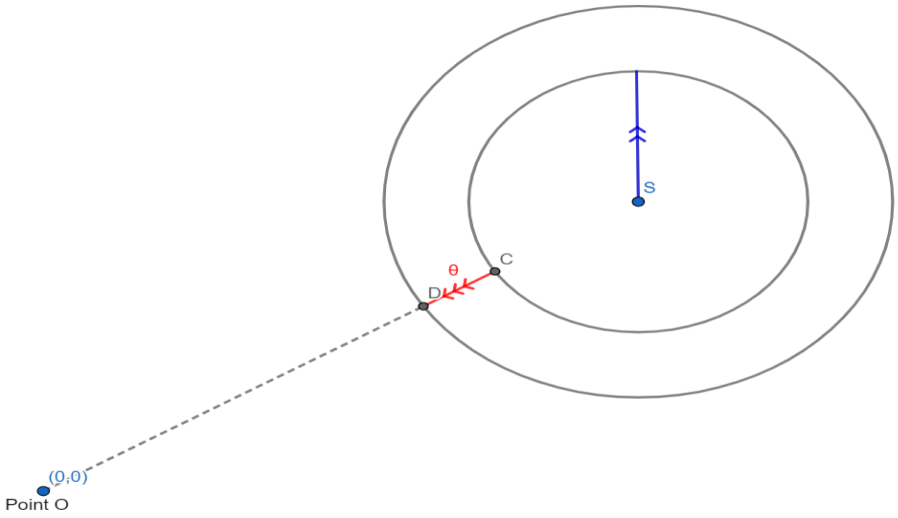
```
td = haversine(lat1, lat2, lon1, lon2)
d = input("enter the value of d (in kms): ")
n = float(td) - float(d)
m = float(d)
x_start = get_cartesian_x(lat1, lon1)
x_end = get_cartesian_x(lat2, lon2)
y_start = get_cartesian_y(lat1, lon1)
y_end = get_cartesian_y(lat2, lon2)
x = (((m * x_end) + (n * x_start))/(float(td)))
y = (((m * y_end) + (n * y_start))/(float(td)))
```

1.4

Equation 1.3 is the formula for coordinates of x and y respectively of point $\mathbf{X_1}$.

Determining the orientation (Direction):

Initial Orientation -



Dia: 2.1

Diagram symbols :-

S = Starting point or Launch Point

Point O = Origin Point of Mars (Airy-0)(0,0)

θ = The direction in which the origin point exists

C = Illumination point to be calculated for determination of first alignment

As there is no physical north on mars, for our navigation, we will consider Airy-0 as the origin point (Latitude and Longitude both are zero). The UAV has to align itself with the origin in such a way that its front (face) directly points to the origin. To achieve this on mars without any guided system is physically impossible. Hence we are introducing the technique of Launch Pad Orientation Fix. This technique allows the use of a custom built launch pad as shown in the above diagram.

Determination of **C** -

1 - After converting the values of Launch Pad and Airy-0 to cartesian coordinates, we use section formula from 1.3 and substitute value of A as the coordinates of the Launch pad, point B will be Airy-0.

2 - Therefore point C will coincide with X_1 and by the algorithm, the cartesian value of point C will be received. This is the point which will light up the particular direction arrow pointing towards Airy-0.

The inner circle is a rotable mechanism that grips the UAV for the first flight. It has to be manually aligned so that the first orientation calculation is complete. The area between the inner circle and outer circle will be designed to have strips of LED lights(). The LED lights will combine to form an arrow which will point to Airy-0.The direction will be calculated by sending the coordinates of the Launch Pad to equation 1.3, in return a coordinate value will be received that will light up the particular direction LED which will in turn form the arrow pointing to the origin. The inner and outer circular design together form the Launch Pad. Once the two arrows are perfectly aligned. The UAV is ready for take off.

Throughout the flight our Lidar scanner will give us the information about the presence of obstacles around us in 360°. Lidar records the precise time from when the laser left the system to when it returned to calculate the range distance between the sensor and the obstacle. So any obstacle within the radius range of 25m will be detected and through this information Jetson Nano will send required flight commands to the flight controller through MAVlink protocol. Mavlink protocol is a way of communicating with unmanned vehicles. It has different commands for throttling , changing orientation, checking altitude, setting the speed of the UAV which we use for avoiding obstacles.

Movement Between two points-

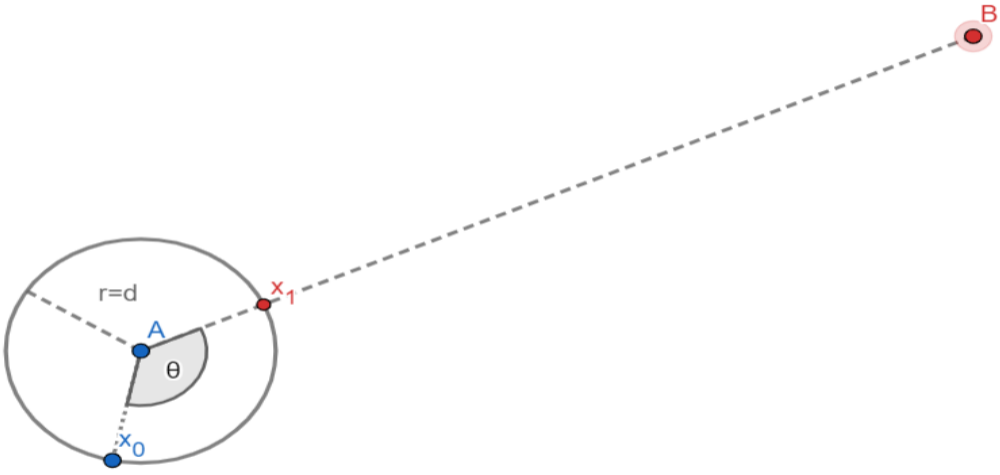


Diagram symbols :-

X_0 = Orientation reached from previous movement

d = Distance between the face and the centre point or the point of hovering

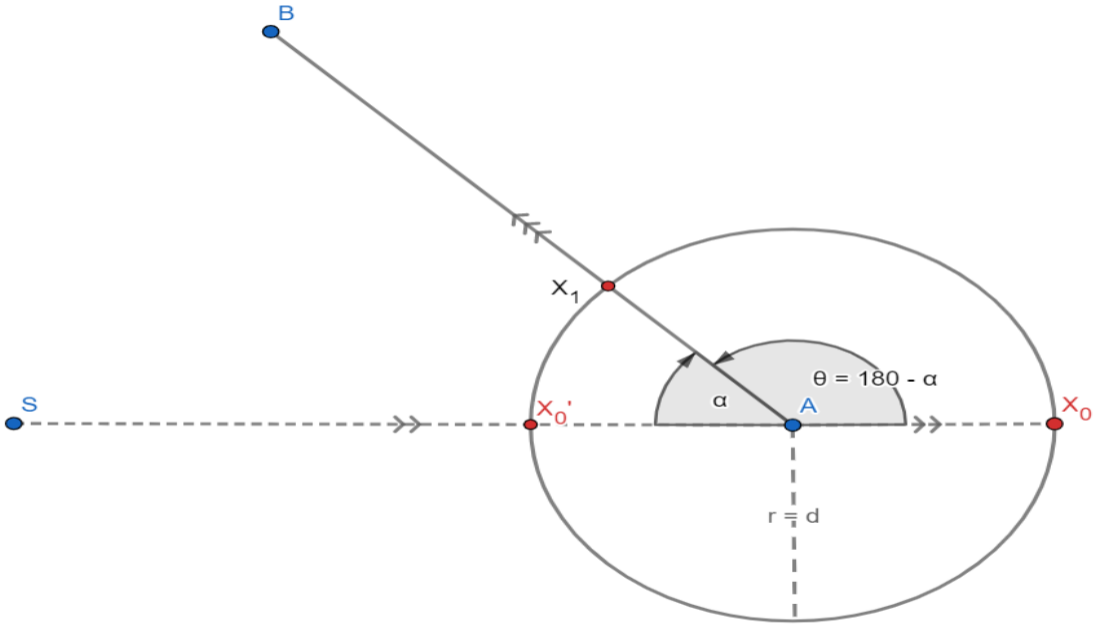
A = Current point

B = Destination point

X_1 = New orientation

θ = angle by which the UAV has to rotate in order for it to reach the new orientation

The movement from one coordinate to another is done by determining the new required orientation (X_1). The orientation X_0 is taken from the previous movement of the UAV. The angle θ is the angle by which the UAV has to rotate its face in order for it to reach the new orientation. Calculating the coordinates of the previous orientation (X_0) by tracing it on the circle is mathematically more complex and will require more time for computation. To tackle this problem, we use the following the diagram:



dia 2.2

Diagram Symbols :-

S = Starting Point or previous location

Line SA = Path traced by the UAV from Point S to Point A

Line AB = Path to be traced by the UAV from Point A to Point B

Point B = New destination point

Radius of the circle = d (distance between the face and the centre)

X_0 = Current Orientation of the UAV

X_0' = Orientation of the UAV on the opposite side of A

X_1 = New calculated Orientation

α = Angle between X_0' and X_1 with respect to the centre A

θ = Angle between X_0 and X_1 with respect to the centre A

In the diagram, after the UAV has completed its movement from point S to A, the orientation of the vehicle i.e. its front face is assumed to be along the path it has traced which is line SA. On reaching point A it will be facing towards X_0. Now to move from Point A to Point B, the UAV has to first rotate its face towards point B and then move forward. Hence, we take X_0' as it lies on the line (in between) point S and point A. In order to calculate '**α**' , since the points X_0' and X_1 lie on the circle, they form a isosceles triangle with point A. Hence the angle can be calculated with the use of the cosine formula:

$$X^2 = Y^2 + Z^2 - 2YZ\cos(\boldsymbol{\alpha})$$

In our case, Y and Z are the sides next to the angle **α**, which are the radius of the centre A, X is the distance between the points X_0' and X_1. So we get:

$$X^2 = 2r^2 - 2r^2\cos(\boldsymbol{\alpha})$$

$$\boldsymbol{\alpha} = \cos^{-1}\left(\frac{2r^2 - X^2}{2r^2}\right) \qquad 1.4$$

In the diagram, summation of '**α**' and 'θ' is equal to 180° and the radius is the distance d from the face, therefore we get

$$\theta = 180 - \boldsymbol{\alpha} \qquad 1.5$$

θ is the angle by which we have to rotate the UAV at its position so that it faces towards the new point where it has to travel.

Direction of Rotation -

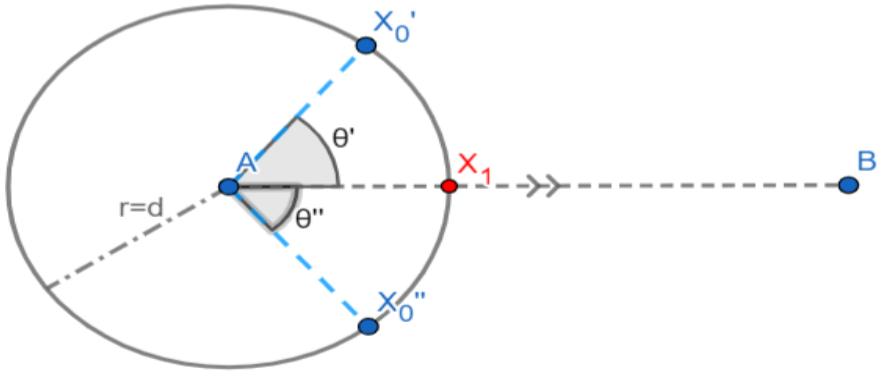


Diagram symbols :-

X_0' =1st case of Orientation reached from previous movement

X_0'' = 2nd case of Orientation reached from previous movement

d = Distance between the face and the centre point / the point of hovering

A = Current point

B = Destination point

X_1 = New orientation

θ' = angle between X_0' and X_1 with respect to the centre A

θ'' = angle between X_0'' and X_1 with respect to the centre A

During the movement, there can be two cases in which the angle of rotation from two orientations will be the same, i.e. θ' = θ''. Hence in order to determine whether the UAV has to rotate in clockwise direction or anticlockwise direction, we compare the values of the latitude of X_0' with X_1, if X_0'>X_1 then the UAV will rotate in clockwise direction else it will rotate in anticlockwise direction. A corner case arises when the latitude value of X_0' and X_1 is exactly the same, so we will compare the longitude value of X_0' with X_1 i.e. X_0' > X_1 then the UAV will rotate in clockwise direction else it will rotate in anticlockwise direction.

Mission Plan:-

The Mission plan involves three major challenges which are Reconnaissance, Logistics and Atmospheric Analysis.

Before conducting any of these missions, there is a preflight checkup and calculations required. The process is : Align the UAV on the launch pad as mentioned in diagram (), from this the initial orientation X_0 can be achieved by supplying coordinate values of Airy-0 and the Starting Point/Launch pad to the algorithm 1.4 which will give X_0 as a cartesian coordinate value. The UAV is now ready to take flight. All the missions will now be described keeping in mind that the preflight process has been completed.

With the dynamic autonomous build as the centre point of this report, there are few processes which we need consistently in all the missions, these are :-

Function Ω (Movement Between two Coordinates) : This function will take in 3 values as its arguments namely the previous orientation X_0, Current Location Coordinate and Coordinate to which the UAV has to travel. The location coordinates will be first used to determine the distance between them from algorithm 1.2 and stored. Then these location coordinates will be converted to cartesian coordinates using the algorithm 1.4 for further calculations. Now to determine the new orientation towards which the UAV has to face, we use the cartesian values of both the location coordinates in the section formula algorithm 1.4 . This will give us the cartesian value of the new orientation towards which we have to rotate the UAV. So to calculate the angle of rotation θ from diagram 2.2, we first calculate the distance between X_0 and X_1 using distance formula as these are cartesian coordinates. Then we use this distance value in the equation 1.4 to get the value of **α**. Hence the rotation angle θ will be given by 1.5. On checking the direction of rotation, it returns whether the UAV has to rotate clockwise or counterclockwise. With this function, we get the distance to be travelled between two location coordinates and the angle by which the UAV has to rotate at its current position in order to face the destination. This angle is passed as a signal to the Flight Control from Jetson. Then the signal to move forward is passed to the flight control which makes the UAV move towards the destination. Upon arriving at the destination coordinate, the UAV calculates its current orientation X_0 by the section formula equation 1.3 simply by reversing the values of ‘m’ and ‘n’ according to the path. This X_0 is used in the next movement of the UAV as its previous orientation value.

Flightpath Correction :-

For flight path correction we have to make sure that the UAV knows : Its Velocity, distance travelled, roll, pitch, yaw angles & Position at time "t" then it will be a bit easier to relocate the UAV if it has lost the direction due to wind or any other factor.

Optical Flow : With the help of a monocular camera and using optical flow can give a promising solution and also help us with minimizing the number of sensors ultimately reducing our overall load. The pattern of motion of objects is known as Optic Flow or Optical Flow. Objects like the surface and the edges of the viewing scene. The pattern of movement of these objects with respect to the observer is known as optical flow. This information can be used to identify the distance between the objects that are in a visual scene as well as the speed by which the object or eye (camera) is moving. Some Equations for Optical Flow Field :

The projection of the 3d velocity field in an image plane is known as the optical field. Considering P =

[X, Y, Z]> as a 3d point with respect to camera frame. Z-axis is the optical axis, focal length is given by f and the origin has the projection center. On the image plane the P's pixel coordinates are given by

$$p = f\frac{P}{Z} \qquad (1)$$

As we know the focal length f is equal to the distance of origin and the image plane, the coordinate p which is our third coordinate is constant p = [x, y, f]>. The camera's and P's relative motion is then given by

$$V = -T - \omega * P \qquad (2)$$

Here the angular velocity is ω & translational component is T. when the equation (1) is derivative with respect to time, we get the relation between the velocity of P in the camera reference frame and the velocity or the flow of p in the image plane on both the sides

$$\frac{flow}{\Delta time} = v = f\frac{ZV - V_z P}{Z^2} \qquad (3)$$

As written in components and using equation(2) the optical flow field is defined as

$$V_x = \frac{T_z x - T_x f}{Z} - \omega_y f + \omega_z y + \frac{\omega_x xy - \omega_y x^2}{f} \qquad (4)$$

$$V_y = \frac{T_z y - T_y f}{Z} - \omega_x f + \omega_z y + \frac{\omega_x y^2 - \omega_y xy}{f} \qquad (5)$$

The sum of translational and rotational parts gives the components of the optical flow field. The Angular velocity does not contain the information about depth of scene as the Z index is independent of rotational parts. In the Equation (4 & 5) translational component is scaled with current distance Z and the focal length to the scene. The translational velocity can be turned to metric scale if we need special attentiveness on the translational velocity for example: in such condition where rotational velocity is constant or known

$$V_{m,trans} = v\frac{Z}{f} = v\frac{b}{d} \qquad (6)$$

The inter axial distance between d and the cameras is b. On combining depth estimation and the values of optical flow gives us the measurement of translational velocity per pixel in metric scale.

Methodology : All the actions performed by the UAV (movement & motors input) will be stored in a dictionary in pair with the state (that is the position of the UAV along with angles and time) , as the UAVs moves ahead by time t the new values will be appended in it. For eg :

State = (Latitude, Longitude ,roll ,pitch ,yaw ,distance ,time ,velocity)
Action = (M1 , M2 , M3 , M4)
Dictionary = {'Pair1' : State : Action}

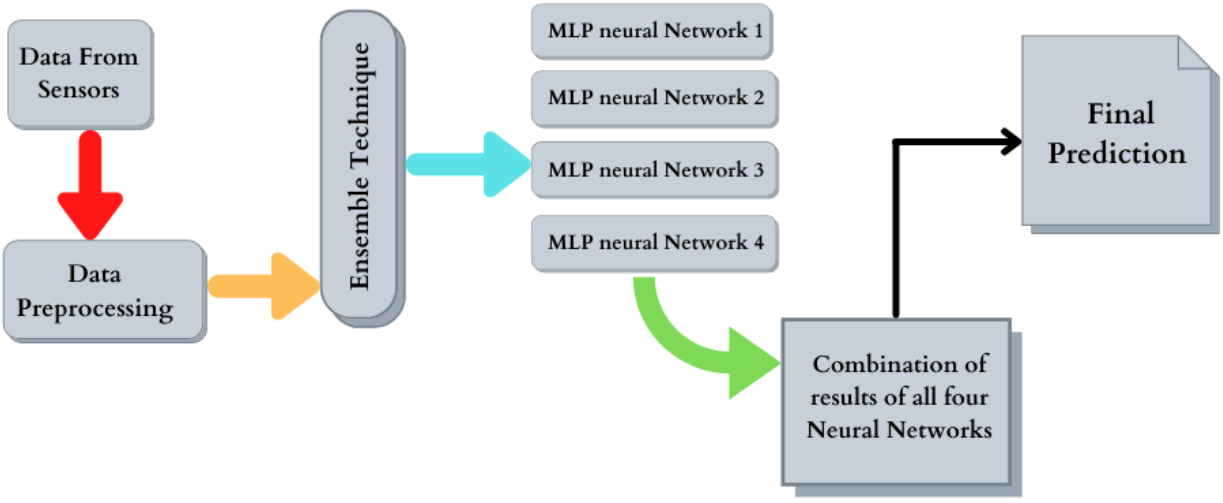
In “State” variable we will include position features like Latitude, Longitude, roll, pitch, yaw ,distance ,time ,velocity and in “ Action “ variable the features that are included are the inputs given to the four ESC's to control the motors (throttling).Time taken to complete one round of computation is 't'. So all the values of position, angle, speed, distance will be measured for every such time 't', the reason for making measurements after time 't' is that it reduces the computational load if we perform it for every second it's possible that there may be errors which may lead to incorrect flight of our UAV.
The advantage of measuring the position and other parameters is that it cannot be lost until and unless the sensors are damaged because we aren't using any signal operated devices.The state action pairs will be helpful to keep in check the position of the UAV so that in case there is some computational error due to which the UAV motion is stuck or lost it can take help of the state action pair and accordingly check the previous actions to get back to the track. In case of gusts of wind when the UAV will be displaced or change orientation and position these changes will be measured by the sensors but the input to the ESC required to overcome the displacement won't be recorded so to tackle this issue we will be making a DeepLearning model which will be trained on Mars.The performance of deep learning algorithms is better than machine learning algorithms in cases where there are multiple inputs and multiple outputs.

Data Collection : For this deep learning model input features will be roll angle ,pitch angle , yaw angle, distance ,direction ,velocity and output feature will be the input given to ESC for motors to control the propellers. The UAV will be made to fly in MARS environment and for collecting data, the angular changes along the x,y and z axis will be measured by the gyroscope and the optical flow method will take care of the velocity & distance. The output features will be stored by Jetson Nano in a micro SD card as it is directly connected to our Flight controller.The flight data can also be collected and stored in Pixhawk flight controller as it also has a micro SD card slot.

Data Preprocessing : The data which we obtain from sensors will be in its raw form, it won’t be standardised, this directly affects the ability of our model to learn so it becomes necessary to preprocess our data to ensure that it is of high quality. We can expect noisy data from our sensors because of the martian environment which has a distributed magnetic field all around ,cold temperature, low air density and other factors, so to tackle this issue we will make use of binning and outlier analysis. Binning is a method in which we segregate the data by dividing it into smaller intervals, each data inside the bins can be replaced by that bin mean or bin median. Outlier analysis is a clustering approach for dealing with noise.
After collecting the data from sensors the data will be integrated and then it needs to be in a simplified & ordered manner, for this we will make use of techniques like high variance filter, low variance filter and principal component analysis. Further we will normalize the sensor data so that it lies between a smaller range, this makes the analysis of data easier. After performing all these steps we can say that our data is ready to be fed in the model.

Model Selection : As we identified our model problem to be Multiple Input Multiple Output (MIMO) we have decided to go with Multi Layer Perceptron (MLP) as it has the ability to generalize better by learning hidden relationships for data with high volatility and non-constant variance between features which helps it for better predictions. Also it has no restrictions on distribution of input variables and

performs well with large datasets. The number of neurons in the hidden layer determines the approximation capacity of the network. This may also lead to overfitting of our model because of high variance therefore we will estimate the number of neurons by cross validation method which will reduce the chances of overfitting. Further we have decided to use four MLP models and apply ensemble technique to reduce variance of predictions and generalization errors. Thus by reducing the variance we ensure that the ensemble method will give us better prediction than any other single model. After training and testing then when we receive good results we can save the model in .h5 file in micro SD card and use it during operations .



PID Tuning : PID tuning plays a very important role as it determines the stability of our UAV and how it will react to disturbances. For the UAV to operate on Mars the values of P, I and D will be set according to the Mars environment, if we set the values of P, I and D according to Earth then our UAV won't be able to take a proper flight and will most probably crash it. Taking an example of " I " gain (which is responsible for holding the UAVs position against external forces such as wind) if we set its value according to Earth then we may notice some stiffness in our UAV because of the fact that the atmosphere on Mars is 100 times thinner than Earth's atmosphere and also it contains 95% of CO2 which is different from Earth, also due to increase in stiffness our UAV will become unresponsive.

As we know about the environmental differences between Earth and Mars such as gravity , atmospheric composition, density, volume etc, because of studies conducted by different Space organisations we can also tune values by taking into account comparison of all the factors we know i.e by training it on simulation through ROS & Gazebo or any other software, but the problem with this approach is that we might not know about some external hidden factors which may affect our UAV.

So to make sure that our UAV achieves a good flight the best way is to tune PID values on Mars itself, this method will be the most accurate as we don't have to worry about some hidden external factors affecting our sensitivity.

Reconnaissance Mission: After alignment, the flight control will receive the signal to lift off the ground and reach a defined altitude. The function Ω will be executed by supplying the arguments to it which are: the X_0 calculated from the initial alignment, the coordinate locations of the start point and the coordinate locations of the recon point. (Reference to electronics and design section for recon mission). After completion, the UAV will return to its start position by executing the function Ω, this time the

arguments supplied will be: the X_0 from previous movement, the coordinate locations of the recon point and the coordinate locations of the start point.

Logistics Mission: After alignment, the flight control will receive the signal to lift off the ground and reach a defined altitude. The function Ω will be executed by supplying the arguments to it which are: the X_0 calculated from the initial alignment, the coordinate locations of the start point and the coordinate locations of Point A (Package Pickup Location). (Reference to electronics and design section for pick up). After pickup, the UAV has to move to Point B(Package Delivery Location). Hence the function Ω is executed again with the following arguments: the X_0 from the previous movement, the coordinate locations of Point A(Package Pickup Location) and Point B(Package Delivery Location). (Reference to electronics and design section for package placement). After the delivery of the package is complete, the UAV has to return to its launch position. So the function Ω is used with the following arguments: the X_0 calculated from previous movement, the coordinate locations of Point B(Package Delivery Location) and the coordinate locations of Start Point/ Launch Point.

Atmospheric Analysis Mission: After alignment, the flight control will receive the signal to lift off the ground and reach a defined altitude. The function Ω will be executed by supplying the arguments to it which are: the X_0 calculated from the initial alignment, the coordinate locations of the start point and the coordinate location of the atmosphere analysis point. (Reference to electronics and design section for atmosphere analysis). After completion, the UAV will return to its start position by executing the function Ω, this time the arguments supplied will be: the X_0 from previous movement, the coordinate locations of the point and the coordinate locations of the start point.

References :-

1- https://www.geogebra.org/geometry for precision drawings.
2- https://www.google.com/intl/en_in/earth/ for knowledge on mars surface and testing the feasibility of the algorithms and equations used in the operations.
3-