OpenCV : https://youtu.be/xmSiehFDsW8

Publisher & subscriber node- https://youtu.be/5gN1JbhvQuI

Turtle Sim- https://youtu.be/zE50kjDfg4U

GitHub- https://github.com/nuhb008/wek2-altair

Theoretical Part:

Task-1:

LoRa and generic RF module have advantages in different factors while using for communication. According to the given factors:

Range:

LoRa module is famous for long range communication as it is named after Long-Range (LoRa). It can cover up to 100 km. But a generic RF module cannot be used for long distant communication. Its normal range is below 50 m. It can cover more hundred meters according to frequency and power.

Power consumption:

LoRa module is specially made for low power consumption and long-range communication. A RF module usually consumes higher power than a LoRa module. So LoRa can be used for longer duration than a RF module.

Data Transmission rate:

LoRa is useful for long-range communication but data receiving and sending rate is very low. It sends data at rate maximum 50 Kbps. So it

cannot send videos. RF modules have higher data transmission like 1.3 Gbps or more.

Connectivity:

For challenging environment, LoRa is more efficient as it has long-range data transmission ability in low power and longer duration with less interference. But RF modules have more frequency interference and higher power consumption in small range.

From the above points, I prefer to use LoRa as the rover has to be controlled in a vast and remote area. Though it has low data transmission rate, it has less risk to lost the rover in remote area.

Task-2:

Pose Graph Optimization is a method used in an autonomous vehicle to achieve SLAM in an unknown indoor environment. This helps to get probable positions and orientations from relative pose measurements.

A pose graph consists of nodes connected by edges. Each node is linked by edge constraints to the graph which define the node's relative position and uncertainty of measurement. When a new edge between two non-sequential nodes is added, it forces a **loop closure**. Several loop closures give a closer look in the map. But the bad ones can't be undone. So, we need to check the environment. The environment model can be in several ways. One of them is **binary occupancy grid** which is broken into grid of cells. When the cell is occupied, the value is 1 otherwise it is zero. We have to assume entire environment empty then fill the cell which are obstacles. The sensor detects the obstacles when get the value 1. With this method, the vehicle can achieve SLAM by detecting the obstacles and can operate where it wants to go.

Task-3:

To make a smart autonomous rover using Pixhawk, we have to go through rover platform selection, hardware selection, sensor connection, avoiding obstacles, testing, telemetry etc.

We discuss in brief in below:

1. First we have to select required Pixhawk then select necessary hardwires like GPS modules, gyroscopes, magnetometer, telemetry radio and sensors like LIDAR.

2. After that, a suitable platform has to be designed to assemble the components including wheels, tracks, motors, power supplier etc.

3. The sensors (gyroscopes, magnetometer, GPS etc.) have to be connected to Pixhawk using I2C or SPI protocol.

4. Have to add a communication module like Wi-fi, radio or Bluetooth.

5. Then we have to install the related software for providing the machine instructions on Pixhawk like ArduPilot.

6. Have to insert an algorithm which will guide the rover autonomously from its current location to destination using GPS sensors along with other sensors.

7. Rover shares information with ground station using telemetry.

8. To check the rover whether the components work properly or not, we have to test it on field and carefully assess the performance of the sensors and other components.

This is how we can build a smart autonomous rover using Pixhawk.