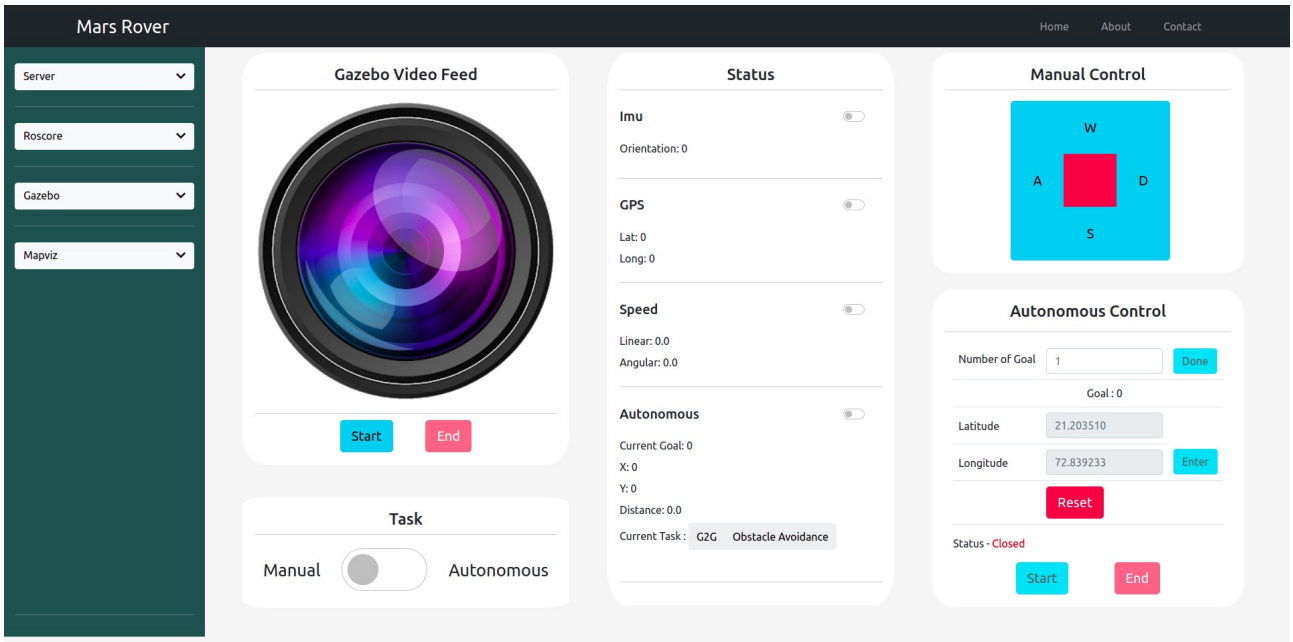
1. Webgui:



Innovation provides an interface to operate ROS (Robot Operating System) based Bot from Website. Interface provides two operating mode Autonomous and Manual Control along with status of Bot in form of Visual Feed and Sensor Data.

SetUp - ROS

*Tested on Ubuntu 20.04 and Ros-Noetic.

Install Ros version according to your Ubuntu Version.
Ros-Melodic for Ubuntu 18.04 & Ros-Noetic for Ubuntu 20.04

Install ROS Packages

- sudo apt-get install ros-noetic-rosbridge-server
- sudo apt-get install ros-noetic-rosbridge-suite

Install ROS packages inside the ROS workspace

- git clone https://github.com/RobotWebTools/mjpeg_server.git
  (Change CV_IMWRITE_JPEG_QUALITY to cv::IMWRITE_JPEG_QUALITY if get error while catkin_make)
- git clone https://github.com/SachinVekariya/Autonomous_Bot.git
  Dont't forget to add "source ~/Your_WorkSpace_Name/devel/setup.bash" in ~/.bashrc

SetUp - Web

- Install Nodejs (Version > 15.0.0)
- $ git clone https://github.com/SachinVekariya/Mars_Rover-WebInterface.git
- $ cd Mars_Rover-WebInterface/WebGUI/
- $ npm i
- $ node app.js #where app.js is present

Open your favourite broswer and search http://localhost:3000/ and Now your Web Interface is ready to control Bot.

For Camera feed :

Open workspace (usb_test) -> source devel/setup.bash

roslaunch usb_cam usb_cam.launch

rosrun web_video_server web_video_server

2. AI-ML interface with ros:

https://nu-msr.github.io/me495_site/lecture11_images.html

Cameras On Linux:

- Cameras on Linux are viewed as files (like any other device).
- Cameras on Linux are usually called /dev/videoX, where X is a number
- Currently some cameras create two video devices, the lower numbered one is the one to use
- v4l2-ctl is a command line tool for interfacing with cameras. See man v4l2-ctl
- v4l2-ctl --list-formats-ext prints useful information about the formats supported by cameras connected to your computer
    - You often need to specify these parameters to the ROS camera node, as parameters
- v4l2-ctl --all prints all information about your cameras
- v4l2-ctl --list-devices Lists all the usb cameras. Some cameras have multiple /dev/video* devices so it is useful to see what goes with what
- When working with cameras you should use udev rules to give your cameras the proper permissions and a persistent name.
- More information can be found here: V4l2

Using usb_cam

1. The camera is one of the `/dev/videoX` devices
2. Run the `usb_cam` node. The basic parameters for most webcams are: `rosrun usb_cam usb_cam_node _pixel_format:=yuyv`
3. `rosrun image_view image_view image:=/usb_cam/raw_image` to view
4. You can also view the image in `rviz` or `rqt_image_view`

Using cv_camera

1. The camera is one of the `/dev/videoX` devices
2. Run the `cv_camera_node`. The basic parameters for most webcams are: `rosrun cv_camera cv_camera_node`

2. Wifi network setup with raspberrypi:

- connect ethernet cable

- coomand - ifconfig (for ubuntu)/ ipconfig (for windows) --> IP address
- ssh username@IP # username= name of the system (raspberrypi)

  or
- ssh username@user # user= username of the system (raspberrypi)
- nmcli