This document is the set of instructions for setting up and flying the RB5 drone safely, both manually and programmatically using ROS. This document assumes a base level knowledge and experience with ROS and ubuntu.

References:

<https://docs.modalai.com/> - manufacturer documentation

<https://forum.modalai.com/> - manufacturer forums – ask technical questions here

<http://wiki.ros.org/mavros> - general info on MAVROS

<https://docs.px4.io/main/en/> - PX4 documentation

Section 1 – Setup

This section covers the initial setup of the RB5 drone. There are 3 overall steps to get the drone working out of the box to flying with ROS. They are:

* OS / firmware setup
* PX4/QGC
* ROS setup

All three steps **must** be completed for the drone to be able to fly programmatically using ROS. **You need a Linux machine with Ubuntu 18.04+.** Virtual Machine is fine, but the steps in section 1.1 that require direct usb connection are much easier using a machine that runs Linux natively (but still doable with virtual machine).

1.1 – OS and Firmware

Steps 1 to 5 cover updating the drone’s operating system.

Steps 1 to 3: <https://docs.modalai.com/Qualcomm-Flight-RB5-system-image/#voxl-sdk-for-rb5-flight>

Steps 4 to 5: <https://docs.modalai.com/Qualcomm-Flight-RB5-voxl-sdk-upgrade-guide/>

1. Plug in RB5 Drone to computer and ensure it is detected. You can run **adb devices** and should see it. If not then configure the VM or plug it into a machine that natively runs linux.
2. Go to <https://developer.modalai.com/> to find the proper system image. It is in *protected downloads* 🡪*RB5 Flight with VOXL SDK Platform Releases*. Download that file and unzip it using the command:
   * tar -xzvf {file\_name}
3. While the RB5 drone is connected to computer, cd into the folder with the unzipped system image and execute:
   * sudo ./full-flash.sh
4. Back up camera calibration. Create a directory to store these files locally on your machine and execute adb pull to save them on your machine:
   * mkdir ~/cam\_cal\_files
   * adb pull /data/modalai/opencv\_tracking\_intrinsics.yml
   * adb pull /data/modalai/opencv\_stereo\_intrinsics.yml
   * adb pull /data/modalai/opencv\_stereo\_rear\_intrinsics.yml
   * adb pull /data/modalai/opencv\_stereo\_extrinsics.yml
   * adb pull /data/modalai/opencv\_stereo\_rear\_extrinsics.yml
5. On host computer navigate to where you downloaded the system image and install it by executing the following commands:
   * cd rb5\_platform\_1.3.1-0.8
   * ./install.sh
6. Push backup camera calibration files after installation is complete:
   * cd ~/cam\_cal\_files
   * adb push opencv\_tracking\_intrinsics.yml /data/modalai
   * adb push opencv\_stereo\_intrinsics.yml /data/modalai
   * adb push opencv\_stereo\_rear\_intrinsics.yml /data/modalai
   * adb push opencv\_stereo\_extrinsics.yml /data/modalai
   * adb push opencv\_stereo\_rear\_extrinsics.yml /data/modalai

1.2 – PX4 and QCG

Before starting the steps in section 1.2, download QGroundcontrol

Steps 1 to 5 enable wifi connectivity with the drone: <https://docs.modalai.com/voxl-wifi/>

Steps 6 to 10 enable MAVLINK and QCG communication with computer

1. Setup a wifi hotspot. I personally use my cellphone and use SSID = “hotspot123” and password = “hotspot123”.
2. Connect the drone via USB cable, ensuring that it is detected by the computer.
3. On the open a terminal computer execute:
   * voxl-wifi station <ssid> [password]
4. For example, using the wifi hotspot I setup in step 1) I would execute:
   * voxl-wifi station hotspot123 hotspot123
5. The next time you reboot the drone, it should automatically connect to that network and you can ssh into it.
6. To ssh into the drone, ensure both the computer and drone are on the same wifi network. On the computer open a terminal and execute:
   * ssh root@{DRONE\_IP}
     + password is *oelinux123*
     + you can find DRONE\_IP by going on connections 🡪 mobile hotspot and tethering 🡪 mobile hotspot 🡪 connected devices
7. configure mavlink by executing the following command and following the prompts
   * voxl-configure-mavlink-server

Text

Description automatically generated

1. enable px4 and setup QCG IP by executing the following command and responding to the prompts as follows:
   * voxl-configure-vision-px4

Text

Description automatically generated

QGC ip address is the ip address of the computer running it, you can find it by executing:

* ifconfig -a
  + this command is to be executed on a terminal on your machine, not a terminal that is ssh’d into the drone. We want your computer’s IP address here, not the drone’s IP address

1. Update param file. Go to my github repo: <https://github.com/albud187/uav_nav_ops> and download “RB5\_MAVROS\_VIO\_09-26.params”
2. Open QGC 🡪 vehicle setup 🡪 parameters 🡪 tools 🡪load from file.
3. This .param file is required to allow the drone to arm and fly **without GPS.**
4. At this point you should be able to fly the drone safely in **position mode** using virtual joystick. If not, then wait a little bit for this popup to appear on QGC:

Shape, rectangle

Description automatically generated

After this popup, you should be able to switch to position mode

Map

Description automatically generated

1.3 – ROS setup

Steps 1 to 7: <https://docs.modalai.com/setup-ros-on-voxl-2/>

Steps 8 to 14: <https://docs.modalai.com/mavros-voxl-2/>

1. Open a terminal and ssh into the drone
2. execute:
   * sudo apt-get update
   * sudo apt install -y ros-melodic-ros-base ros-melodic-image-transport
   * voxl-configure-mpa -p -f rb5-flight
3. Open ~/.bashrc by executing:
   * nano ~/.bashrc
4. On ~/.bashrc add the following line:
   * . /opt/ros/melodic/setup.sh
5. Save and close the ~/.bashrc file. Then source it by executing:
   * source ~/.bashrc
6. Now you can execute:
   * roslaunch voxl\_mpa\_to\_ros voxl\_mpa\_to\_ros.launch
7. The command in step 6) will start publishing the camera imagery as ROS topics
8. Install MAVROS by executing:
   * Sudo apt-get install ros-melodic-mavros ros-melodic-mavros-extras ros-melodic-control-toolbox
   * cd /opt/ros/melodic/lib/mavros
   * ./install\_geographiclib\_datasets.sh
9. Build mavros test by executing:
   * cd /home
   * git clone -b simple-example <https://gitlab.com/voxl-public/support/mavros_test.git>
   * cd /home/mavros\_test
   * ./build.sh
10. Configure ROS environment. Open ros\_environment.sh by executing:
    * nano /home/mavros\_test/ros\_environment.sh
11. Under configure IPs set them to the following:
    * export ROS\_MASTER\_IP=localhost
    * export ROS\_IP=localhost
    * export ROS\_MASTER\_URI=http://localhost:11311/
12. source the ROS environment on the ~/.bashrc so you don’t have to source it each time. open bashrc by executing:
    * nano ~/.bashrc
13. Add the following line to the ~/.bashrc:
    * source /home/mavros\_test/ros\_environment.sh
14. Now you can launch MAVROS by executing:
    * roslaunch mavros px4.launch fcu\_url:=udp://127.0.0.1:14551@:14551 tgt\_system:=${PX4\_SYS\_ID}

Section 2 – Operation

The drone has only been tested on two modes of flight. Position mode and offboard mode. Position mode (<https://docs.px4.io/v1.11/en/flight_modes/position_mc.html>) is "an easy-to-fly RC mode in which roll and pitch sticks control speed over ground in the left-right and forward-back directions (relative to the "front" of the vehicle), and throttle controls speed of ascent-descent. When the sticks are released/centered the vehicle will actively brake, level, and be **locked to a position in 3D space** — compensating for wind and other forces. "

Basically with position mode, the throttle is used to adjust the drone’s position in 3D space, and the drone is **automatically stabilized and fixed in place** unless throttled**.** The other flight modes (manual, acro, stabilized) are more difficult to fly. **It is recommended not to use them unless you are experienced with drones.**

Offboard mode is flight using offboard control, in this case, a ROS node publishiing position and velocity commands. The drone cannot be switched into offboard mode unless the following conditions are met:

* the MAVROS node is running. This is from executing:
  + roslaunch mavros px4.launch fcu\_url:=udp://127.0.0.1:14551@:14551 tgt\_system:=${PX4\_SYS\_ID}
* You are publishing setpoint velocity and position commands at a rate greater than 2 hz.

The ROS topics used to fly the drone programmatically are:

* Desired velocity: /mavros/setpoint\_velocity/cmd\_vel
* Desired position: /mavros/setpoint\_position/local

The drone’s estimated position and velocity are from the following topics. You can rostopic echo to display this information:

* Estimated velocity (body frame): /mavros/local\_position/velocity\_body
* Estimated velocity (drone world frame): /mavros/local\_position/velocity\_local
* Estimated position: /mavros/local\_position/pose

Publishing to /mavros/setpoint\_velocity/cmd\_vel will send control signals to drive the drone’s current velocity to the desired velocity. **/mavros/setpoint\_velocity/cmd\_vel uses the same reference frame as /mavros/local\_position/velocity\_local**

Publishing to /mavros/setpoint\_position/local causes the drone to attempt to fly to that position. Its estimated position from /mavros/setpoint\_position/local should match the one from /mavros/setpoint\_position/local or be very close when the drone is done moving to that position.

A typical sequence of steps for flying the drone using ROS is as follows.

1. Power up drone, record its IP address and open QGroundControl.
2. Ensure that you can switch it to position mode.
3. Open 2 terminals and ssh into the drone on both of them.
4. On terminal 1 execute:
   * roslaunch mavros px4.launch fcu\_url:=udp://127.0.0.1:14551@:14551 tgt\_system:=${PX4\_SYS\_ID}
5. On terminal 2 run whatever node you create that publishes to the setpoint position and velocity topics. I made a node called “position\_publisher.py” in which I can input setpoint positions for the drone to fly to and hover there until another positon is inputed:
   * cd /path\_to\_ROS\_node\_you\_made
   * python position\_publisher.py
6. Go to QGroundControl and arm the drone **with throttles all the way down and in position mode.**
7. Switch to offboard mode. The drone will now fly as per the commands from your ROS node. You should then center the throttle while in offboard mode, this will do nothing until you switch to another mode, but is very important.
8. **Do not under any circumstance exit the ROS Node while still in offboard mode.** To stop the drone go on QGroundcontrol and switch to **position mode while throttles are centered**. In position mode, if the drone is flying and **the throttles are centered** (this is why you center the throttle right away after switching to offboard mode – so that if you need to stop the drone, you just change to position mode and it will stop right away) the drone will hover in place. You can then use position mode to land the drone safely, disarm it, and then exit the ROS node.