A Communication Architecture for UAV Swarms

C Sai Aditya

Indian Institute of Technology Kanpur

achundi@iitk.ac.in

May 31, 2019

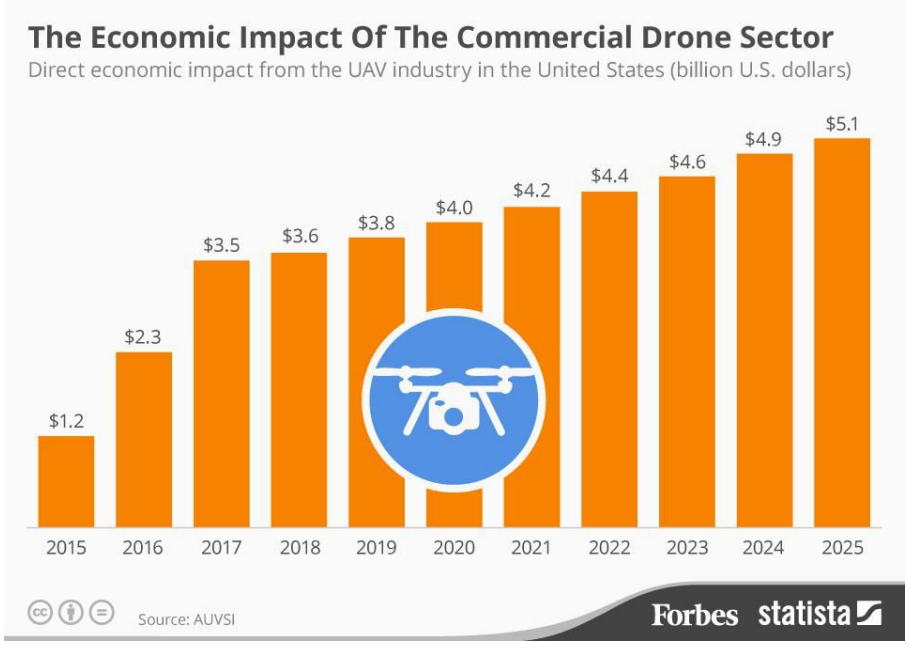
Overview

Introduction

- Unmanned Aerial Vehicles or UAVs have become ubiquitous in both research and industry.
- Have wide applications in diverse fields. Domestic Applications like
  - Package delivery
  - Monitoring and surveillance
  - Traffic Management
  - Precision agriculture
  - Disaster management
- UAVs are very appealing for military purposes as well, with applications ranging from simple reconnaissance, search and rescue missions to combat missions like targeted hits.

- Much of this rise can be attributed to associated advances in robotics and allied fields, driven by growing availbility of robust and cheap sensors and communication equipment.

Introduction



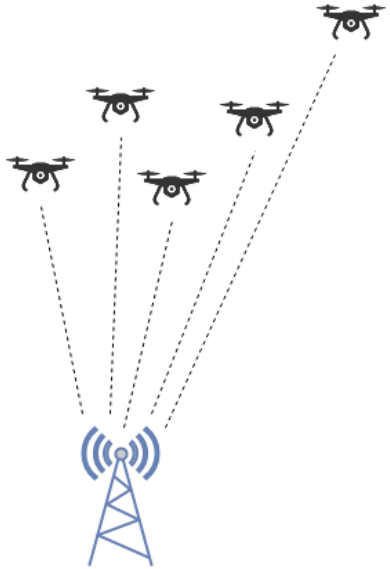
- The UAV industry is poised to grow significantly in the next decade.

- The focus is shifting from single UAV systems to multi UAV systems or UAV swarms.
- Multi UAV systems perform more efficiently than single UAV systems in certain applications.
- For instance, use of multiple UAVs for surveying an area can significantly reduce the time taken to complete the task [6].
- Many applications of single UAV systems can be extended to multi UAV systems.
  - Surveillance
  - Search and rescue missions
  - Package delivery

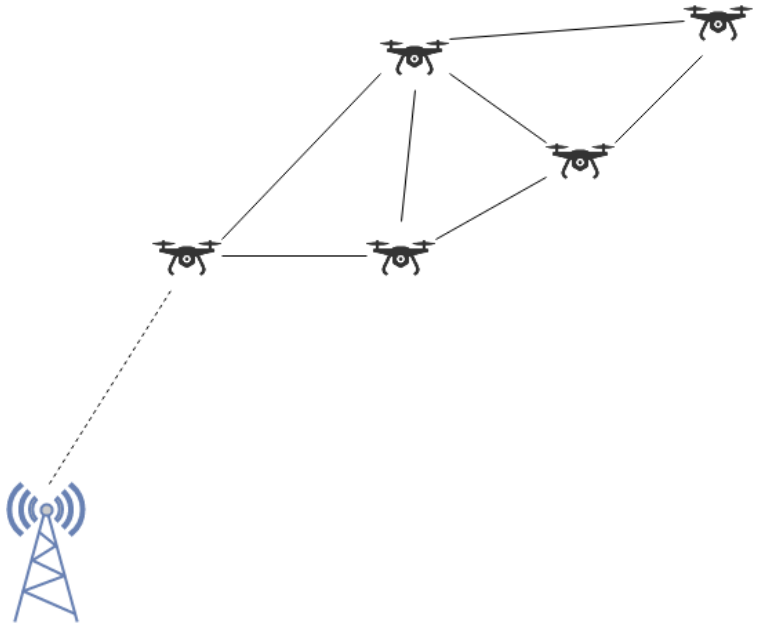
Communication in UAVs

- Communication is an important component in UAV systems. While single UAV systems do not demand much communication wise, it becomes a critical piece in UAV swarms.
- To qualify any multi robotic system as a *swarm*, it needs to have some extent of autonomy and collaborative behaviour among the individual agents.
- There are two aspects of communication in UAV swarms.
  - Communication between a UAV and the ground station.
  - Inter communication among the UAVs.
- Apart from the communication between UAVs and the ground, robust and reliable inter communication among the UAVs is essential for enabling cooperative and collaborative behaviour in UAV swarms.

Communication in UAVs



(a) Star Configuration



(b) Mesh Configuration

- While there have been several proposed architectures for communication in multi UAV systems, we show two popular configurations here.
- In star configuration, all the UAVs are connected to the ground station and all the inter UAV communication happens through the ground station.
- In the second configuration, a wireless adhoc network is established among the UAVs. This configuration has several advantages over the first one.

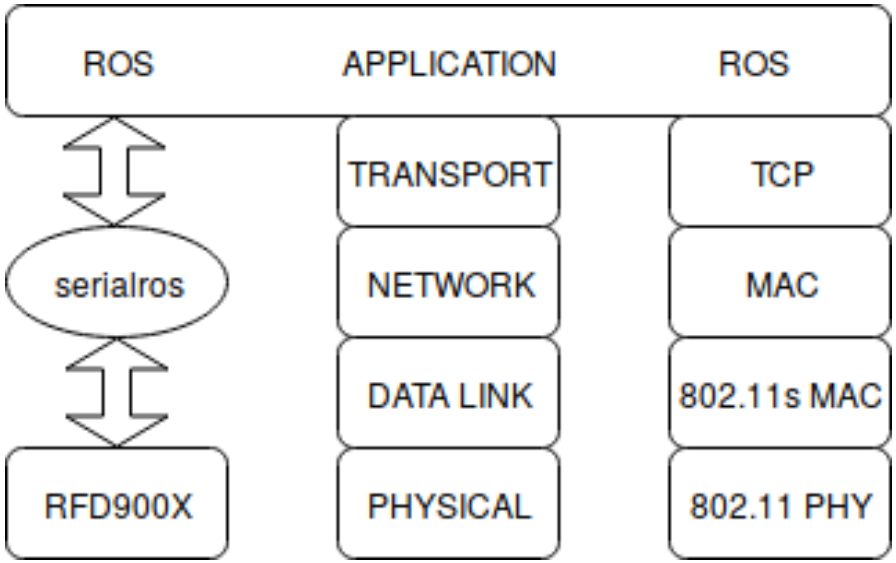
Problem Motivation

- Inter UAV communication and the communication between UAVs and the ground station have different requirements.
- For instance, the range over which they need to support the communication as well as the data rate they are very different.
- The communication architecture between the UAVs and the ground station needs to be a long range one, while the inter UAV communication architecture may tolerate relatively shorter ranges.
- Furthermore, the architecture for inter communication among the UAVs needs to support higher data rates for distributed applications, enabling UAV swarms.
- On the other hand, in swarm applications, much of the decision making happens on board the UAV, and the ground station is mostly used for monitoring telemetry, which does not demand high data rates.

Problem Motivation

- For inter UAV communication, the consensus in research community is to create a wireless adhoc network among the UAVs.
- While there are previous works [5, 2], which implemented wireless adhoc networks in UAVs, they make the ground station a part of the adhoc network, which restricts the area the UAVs can cover.
- On the other hand, the long range radios which allow the UAVs to cover longer distances are not suitable for inter UAV communication.
- Our primary aim in this work was to design and implement a communication architecture which integrates both long range and short range radios.
- Furthermore, We integrate our entire architecture into Robot Operating System(ROS), which makes writing distributed application fairly easy.

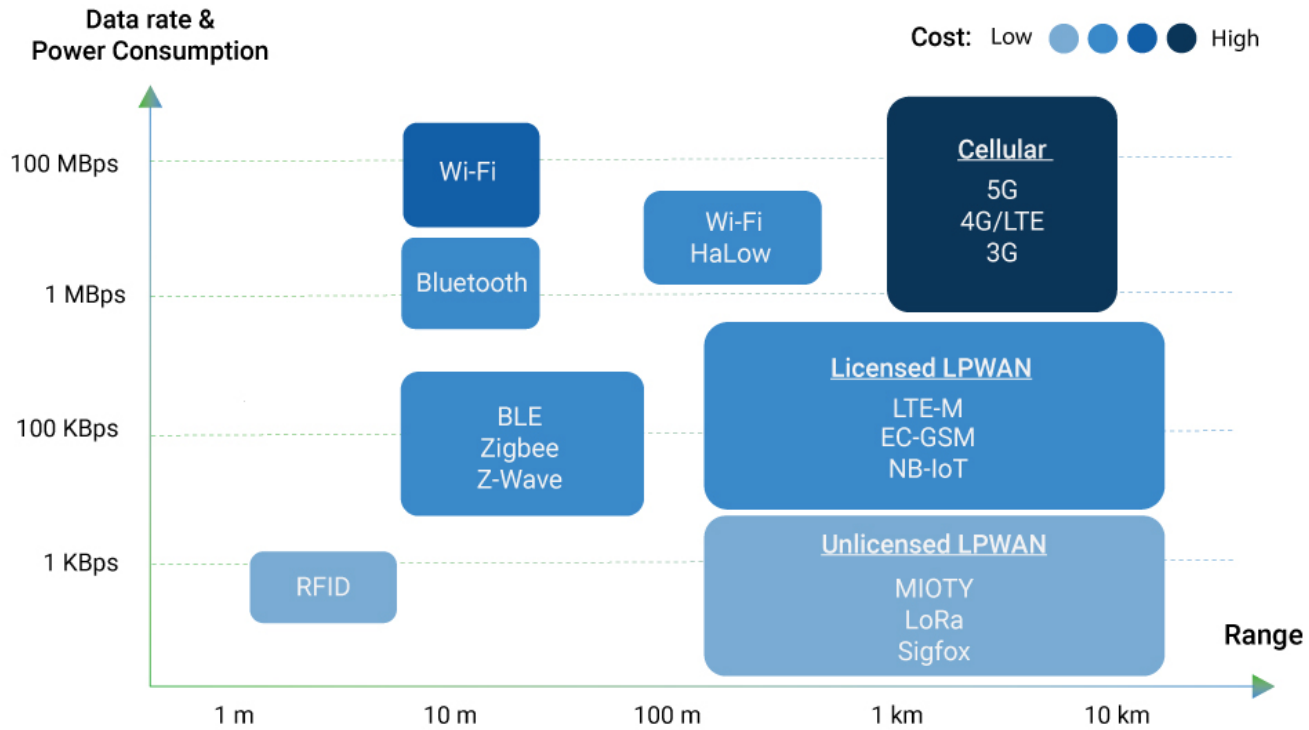
Brief Outline



- Figure gives a high level overview of our architecture with reference to the TCP 5 layer model of networking.
- ROS sits in the application layer of the model. It supports TCP, which makes running multiple ROS applications on multiple machines over a network fairly easy.

- We use mesh networks based on the IEEE standard 802.11s for short range inter communication among the UAVs.
- Integration with ROS is straightforward since the standard supports TCP protocol stack.
- For long range communication between the UAVs and the ground station, we use a popular 900Mhz radio RFD900x.
- The radios do not have a TCP protocol stack and offer a serial interface. We have written our own interfaces for integrating these radios into ROS.

Mesh Networks



- With the advent of IOT, there have been many mesh networking standards like 802.15 [1] and Bluetooth mesh [3].
- None of these standards beat the data rate or range offered by the standard Wifi.

IEEE 802.11 - Wifi

- IEEE 802.11(Wifi) offers a promising solution for short range communication among the UAVs with reasonably high data rates.
- Every node of the network is defined as a *Station*, STA.
- One of the *stations* in the network would be an *Access Point*, AP.
- It is a centralized architecture, where all the STAs in the network are connected to the AP.
- The AP coordinates all the STAs in the network and all the traffic between any two STAs would be routed through the AP.

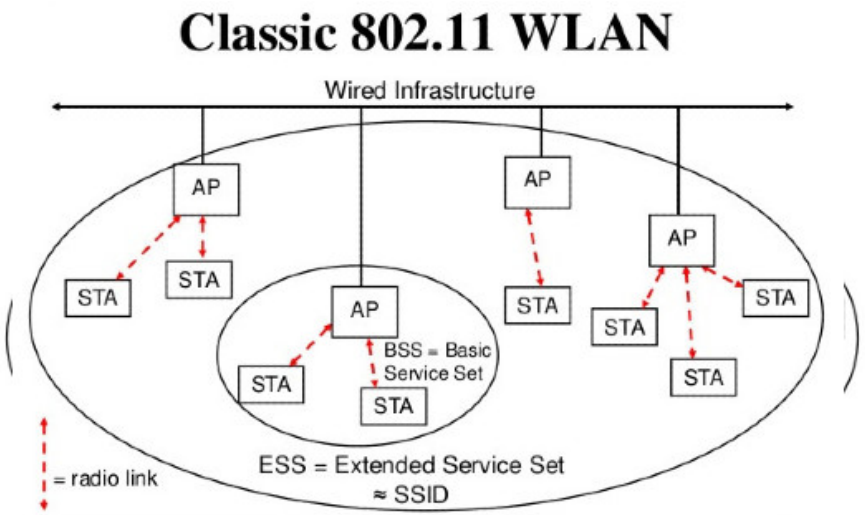


Figure: Standard Wifi architecture

IEEE 802.11s - Wifi Mesh

- With growing demand for wireless mesh networks, a mesh network standard IEEE 802.11s emerged as an extension of IEEE 802.11
- IEEE 802.11s extends the 802.11 MAC layer enabling a MAC based multi hop network.
- The standard enables building self-configuring and self-healing networks with legacy wifi hardware.
- The class of networks that can be built come under Mobile Adhoc Networks(MANETs), whose topology is dynamic.

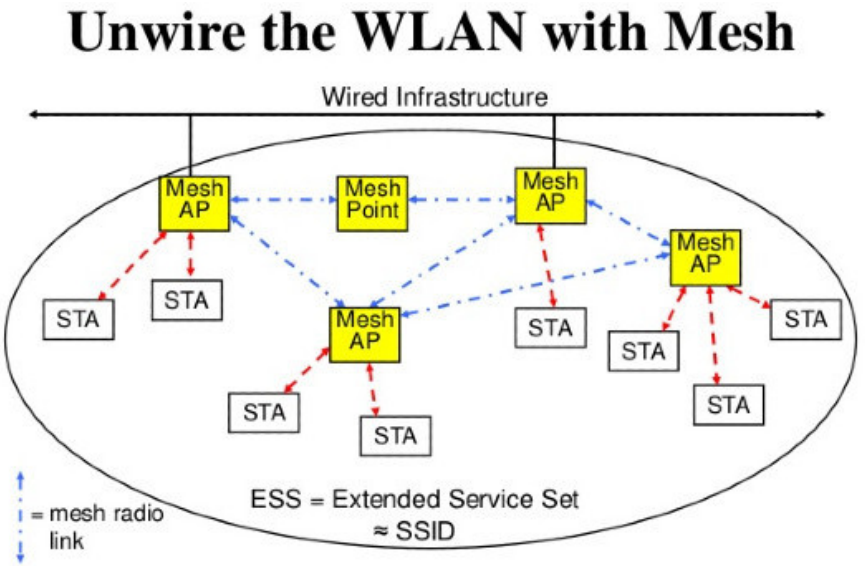


Figure: 802.11s Mesh architecture

Routing in 802.11s

- Routing in wireless adhoc networks is in itself a vast area of research. Many routing protocols have been proposed over the years [4].
- Can be broadly classified into two categories.
- **Proactive routing protocols**
  - These protocols keep track of the topology of the network by storing all the routes in table. The tables are kept updated as the topology changes.
  - Little to no latency since the next hop in a route can be found instantly.
  - Bandwidth inefficient since a the network is flooded with messages to track the topology.
  - Examples include *Optimized Link State Routing* (OLSR) , *Destination Sequenced Distance Vector* (DSDV) and *Better Approach To Mobile Adhoc Networking* (BATMAN)

Routing in 802.11s

- **Reactive routing protocols**
  - On-demand protocols, where the route between two nodes is found only when there is data to transferred.
  - Bandwidth efficient since there is no flooding of messages, meant to keep track of routes.
  - Finding the route before sending a packet introduces latency into the system.
  - Examples include *Dynamic Source Routing* (DSR) and *Adhoc On Demand Distance Vector* (AODV).
- There are other protocols apart from these. Hybrid protocols like *Zone Routing Protocol*(ZRP) and TORA, and *Geographic Routing Protocols*.
- The 802.11s standard defines HWMP(*Hybrid Wireless Mesh Protocol*), a default routing protocol to implement. HWMP is a hybrid protocol designed by combining aspects of both proactive and reactive protocols.
- However HWMP is optional and can be replaced by any other routing protocol.

open80211s

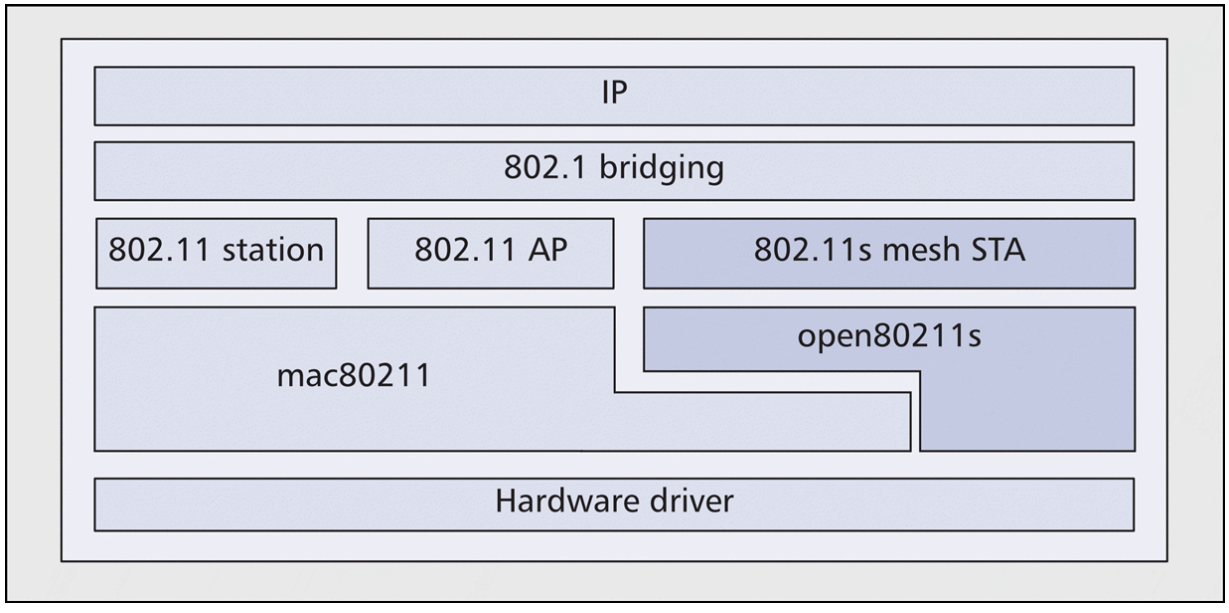


Figure: open8011s stack on Linux kernel

- open80211s is an open source implementation of the 802.11s standard on the Linux kernel.
- It is based on the mac80211 stack which implements the standard 802.11 on Linux kernel.
- 802.11s makes minimal changes to 802.11 MAC layer and consequently, open80211s is supported on all legacy hardware that supports mac80211.

OpenWrt

- Typically, wireless routers come with default firmware from the manufacturer, which limit the abilities of the router.
- OpenWrt is an embedded operating system based on Linux kernel, which can be flashed on commercial wireless routers.
- It provides much flexibility in configuring the routers, since it is based on Linux kernel.
- In many wireless routers which are capable of 802.11s mesh networks, the feature is not supported by the factory firmware which they come with.
- OpenWrt cannot be flashed on all the routers. A list of routers which support it can be found on their website.
- Once OpenWrt is flashed on the router, it is just a matter of configuring the router for it to enable the mesh network.

Hardware - Tp Link WR902AC

Figure: Tp-Link WR902AC

- The router has a 10/100 Mbps ethernet port, a 2.0 USB port. It needs external power support of 5V/2A via the micro USB port.
- The factory firmware supports IEEE 802.11ac/n/a @ 5GHz and IEEE 802.11b/g/n @ 2.4 GHz, making it a dual band router. However, the openwrt port for the device does not support operation @ 5 GHz.
- The router can support speeds upto 300 Mbps @ 2.4GHz and 433 Mbps @ 5GHz, with the factory firmware. Transmit power $<$20dBm.

Hardware - Alfa Tube 2H

Figure: Alfa Tube 2H

- The router has a 10/100 Mbps ethernet port, an N-type male antenna port. It supports *Power Over Ethernet*(POE) and is powered by the ethernet port.
- The factory firmware supports IEEE 802.11b/g/n @ 2.4 GHz. The router can support speeds upto 150 Mbps. Transmit power can go upto 500mW 27dBm.
- Compared to Tp-Link routers, these routers have higher transmit power and also support external antennas. This makes them have longer range than the Tp-Link routers.

OpenWrt Network Configuration

- After the routers are flashed with openwrt and are booted up, they come up with a default IP address of 192.168.1.1/24.
- One can log into a shell of the openwrt by sshing into the router from a host PC connected to the router via the ethernet.
- The environment and file structure of openwrt is quite similar to standard linux based distributions like ubuntu.
- Configuration of various sub systems is done by modifying the files in the folder */etc/config/*.
- To connect the router as a *client* to a wifi *Access Point*(AP)
  - Create a wireless device interface in the */etc/config/wireless* file.
  - Specify the SSID of the AP and authentication, if any in the wireless config file.
  - Define a network interface in */etc/config/network* file, for the above created device interface.
  - Assign IP address to the created interface and restart the network service.
- OpenWrt also provides LUCI, a web browser based GUI for configuring network. Once can open LUCI by typing http://192.168.1.1 in their browser.
- Unfortunately, LUCI cannot be used to configure 802.11s mesh networks.
- Openwrt, by default does not support 802.11s mesh networks and we need to install some packages and remove others, using the systems package manager, opkg.

OpenWrt Network Configuration

- Replace wpad-mini with wpad-mesh.
- Install kmod-batman-adv and batctl for BATMAN routing protocol
- Configure the routers as mesh points using the above configuration files.
- Bridge both the ethernet and wireless interfaces into a single logical interface, to which we assign a static IP of the form, 192.168.1.xxx/24.

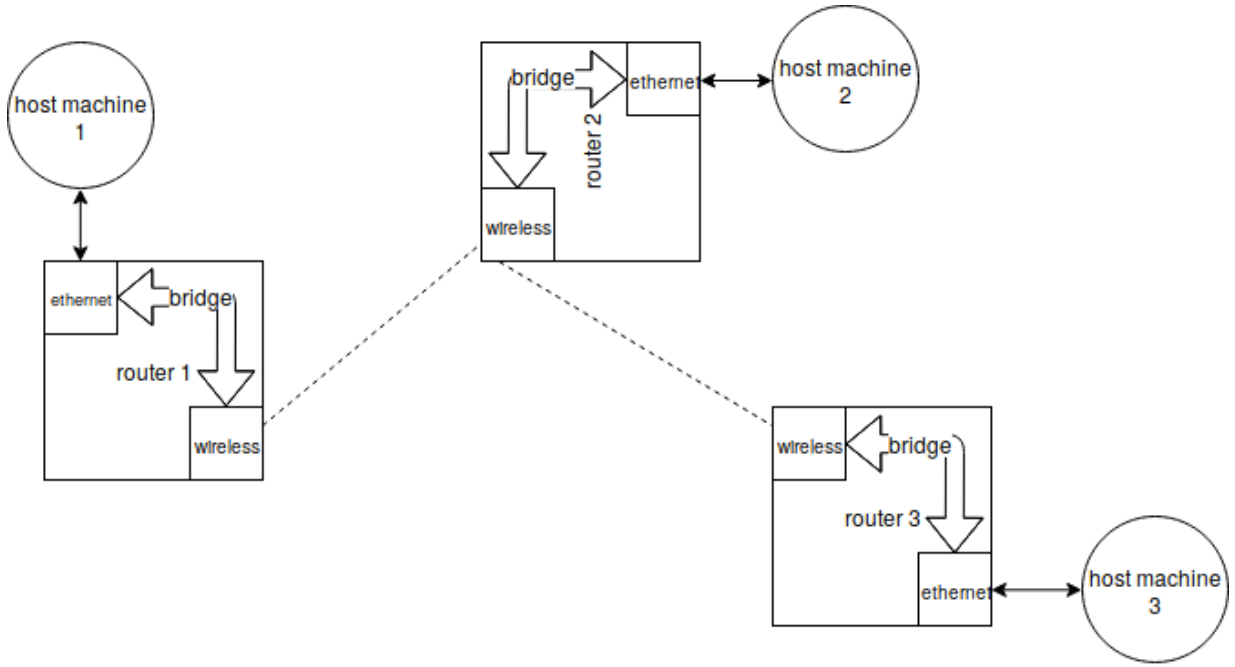


Figure: Mesh network with routers and host PCs

- After all the routers are configured similarly and are powered on, they would establish a mesh network among themselves.
- Since we bridged the ethernet and wireless interfaces in all the routers, by connecting a host PC via the ethernet to the router, the host PCs themselves should be connected in the network.

ROS multimaster

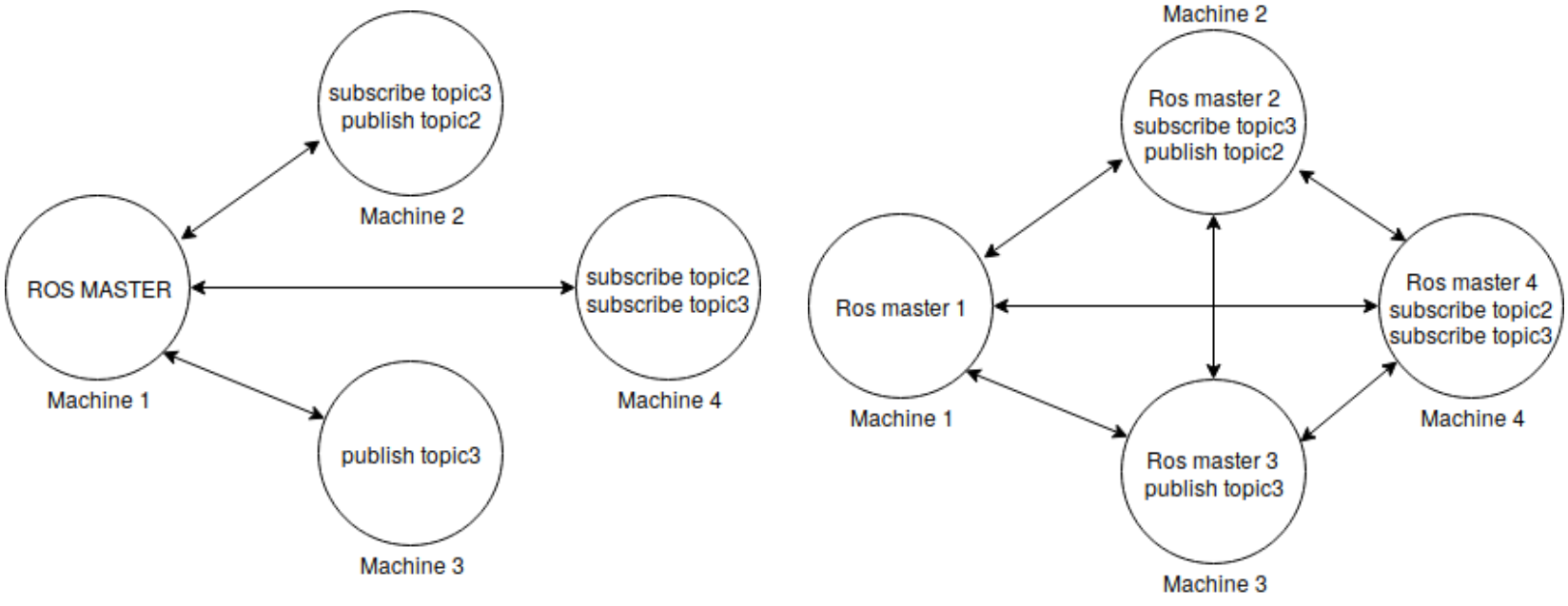


Figure: ROS single master and multimaster setups

- Normally, ROS is setup over multiple machines in a single master setup, where one of the nodes acts rosmaster and coordinates all the other nodes.
- This is unreliable because all the nodes have to keep communicating with the rosmaster. Moreover, if the rosmaster is inaccessible, it fails the entire setup.
- Instead, ROS is multimaster setup where each UAV is a rosmaster offers a much better solution. This can be setup using the ROS package *multimaster_fkie*.

Hardware - RFD900X

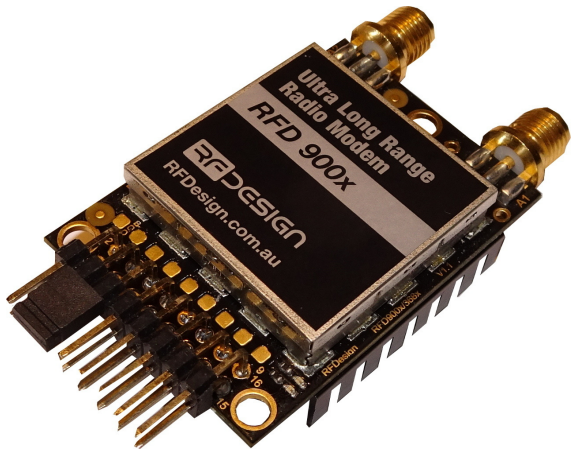


Figure: RFD900x

- Popular serial radios that operate on frequency 902-928Mhz.
- Selectable transmit power upto 30dBm, in steps of 1dBm.
- Supports multiple air data rates from 4Kbps to 500Kbps.
- Supports serial UART interface with baud rates from 9600 to 1000000.
- Advertised range of 50+Kms in Line of Sight, depending on antennas.
- Supports AT commands to configure wide range of parameters.
- The radios support both point to point and point to multipoint configurations.

Point to Point Firmware

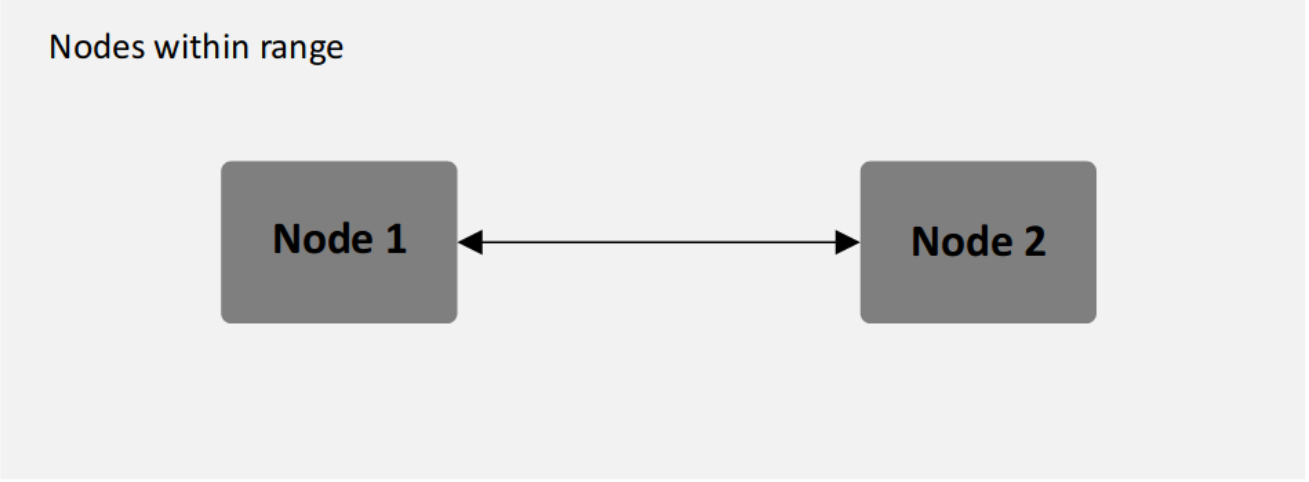


Figure: radios in point to point configuration

- The radios come with the point to point firmware by default. This is a simple configuration where just a pair of radios can communicate with each other.
- The radios should be configured to have same NETID parameter, configured via the AT commands. Once configured, any raw bytes that are input at the serial interface of one radio can be received at the other end.
- Even though you can have multiple pairs of radios with different sets, as the number of radios increase, interference also increases reducing the effective data rate.

Point to Multipoint Synchronous Firmware

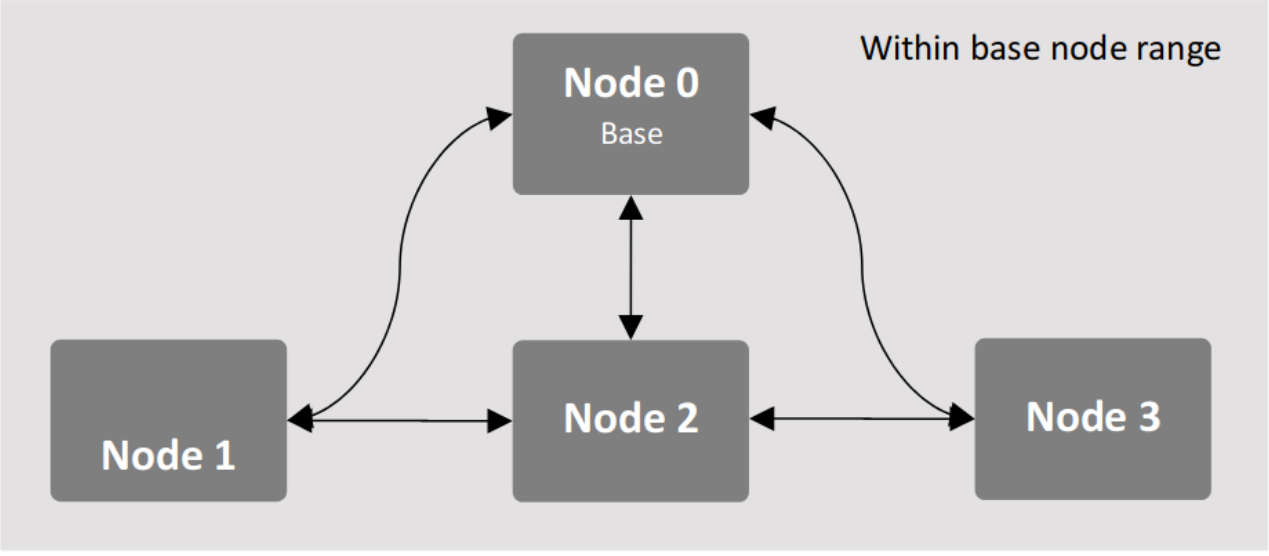


Figure: radios in point to multipoint synchronous configuration

- Apart from the NETID parameter, there is a NODEID parameter, a NODEDESTINATION parameter and a NETCOUNT parameter.
- NODEID can be any unique value between 1 to 15. If NODEDESTINATION is set as 255, it enables broadcast mode.
- One of the nodes needs to be configured as a base node, by setting its NODEID to 1. The NETCOUNT parameter also needs to be set at the base node.
- The base node assigns slots to all the other nodes to communicate. Each node transmits data only during its slot.

Point to Multipoint Asynchronous Firmware

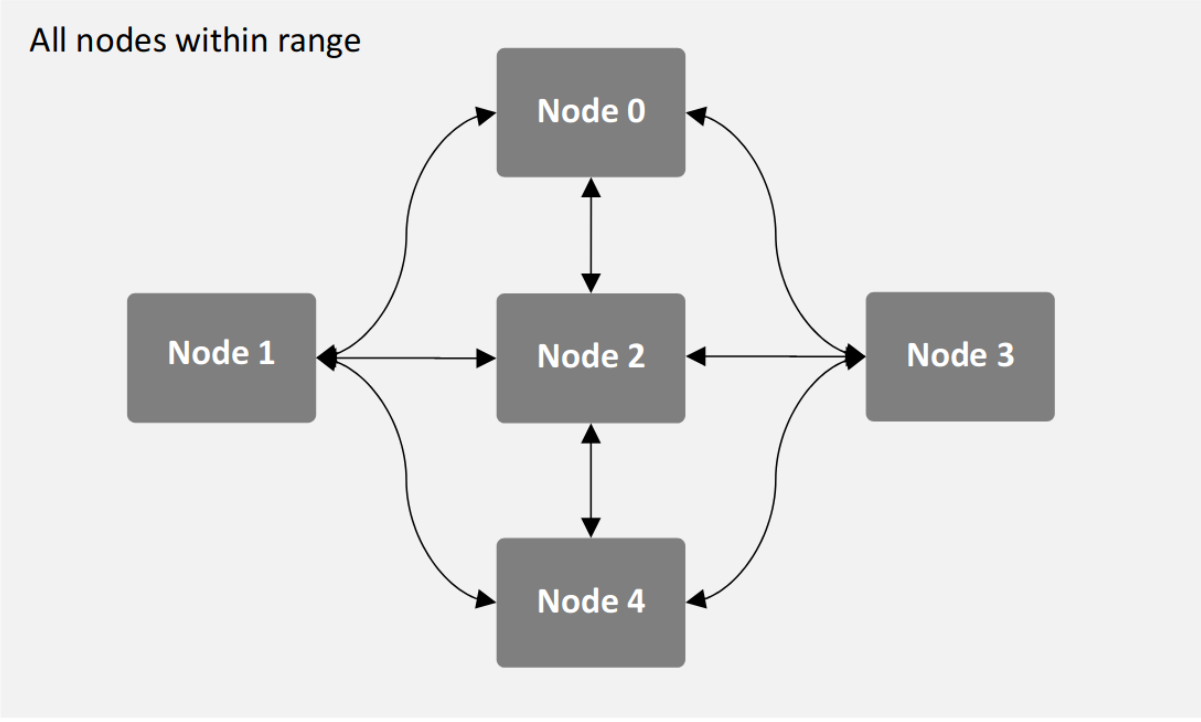


Figure: radios in point to multipoint asynchronous configuration

- Similar to synchronous firmware, but the parameter NODEDESTINATION is replaced by DESTID and there is no NETCOUNT.
- NODEID can be set to any value between 1 to 32767. If DESTID set to 65535, broadcast mode is enabled.
- All nodes transmit data asynchronously. If there is a collision, they attempt a fixed number of retries.

serialros

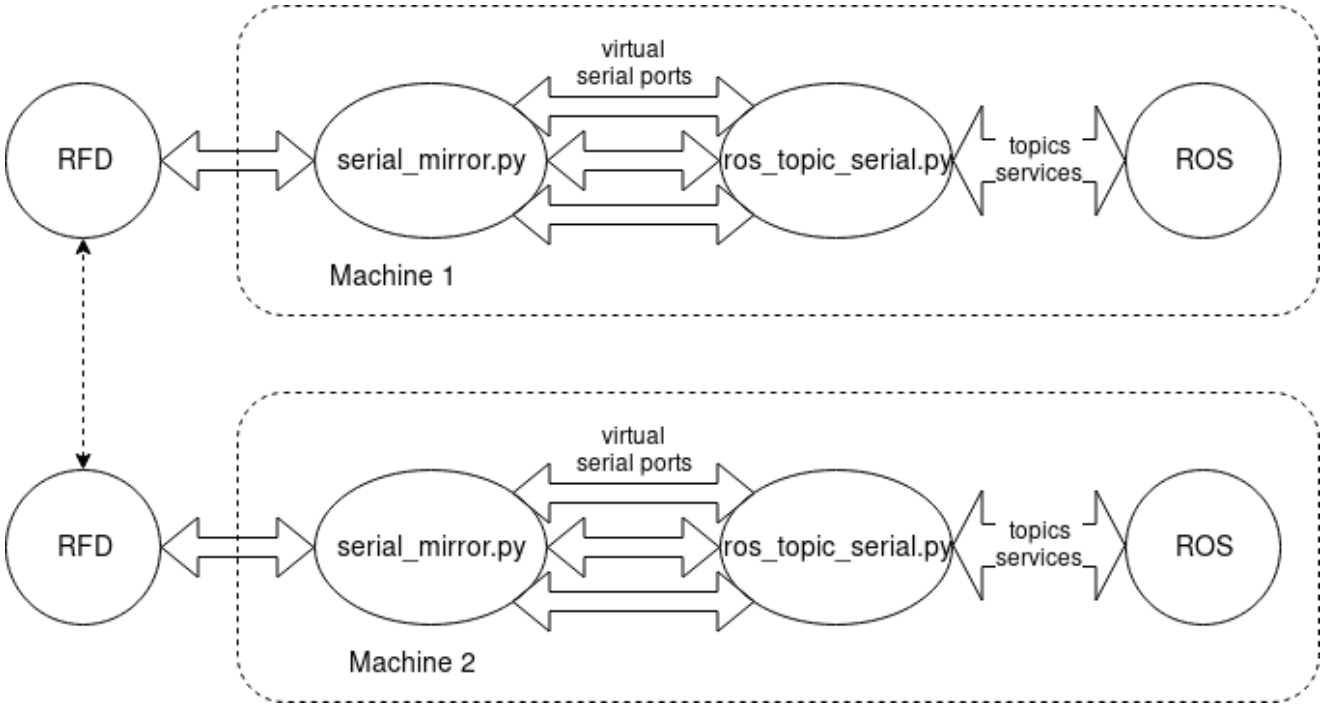


Figure: Interface between RFDs and ROS

- serial_mirror.py splits a single serial link into multiple virtual serial links.
- ros_topic_serial.py sends and receives ros topics and services via the created virtual serial links.
- The ros_topic_serial.py node takes as input, a configuration file, which specifies what ros topics to be transmitted and what ros services of the remote machine should be made accessible.

Static Leader

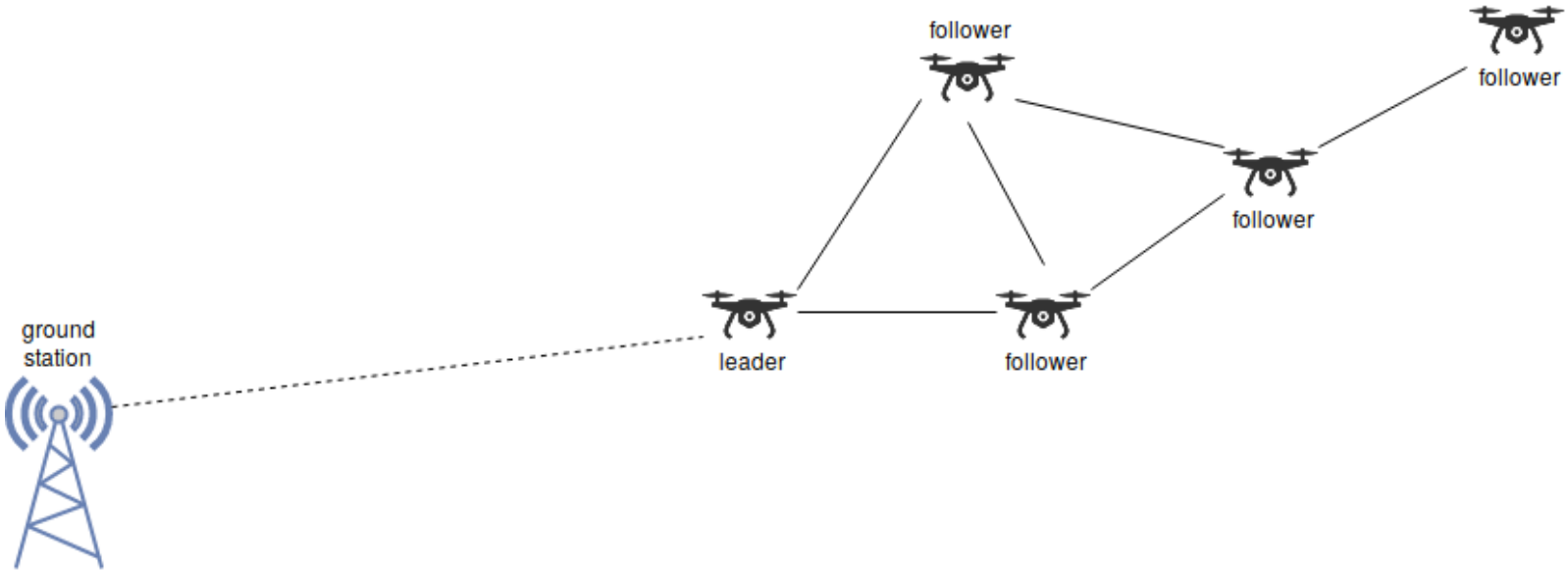


Figure: Static Leader Configuration

- *leader* is decided and fixed apriori and remains unchanged.
- The *leader* launches the serialros node, with a configuration file, which specifies all the required ros topics from all the UAVs.
- The *leader* itself, transmits all the required data of all the UAVs to the ground, since it has access to all the ros topics and services of all the remaining UAVs(*followers*) via the wifi mesh.

Dynamic Leader

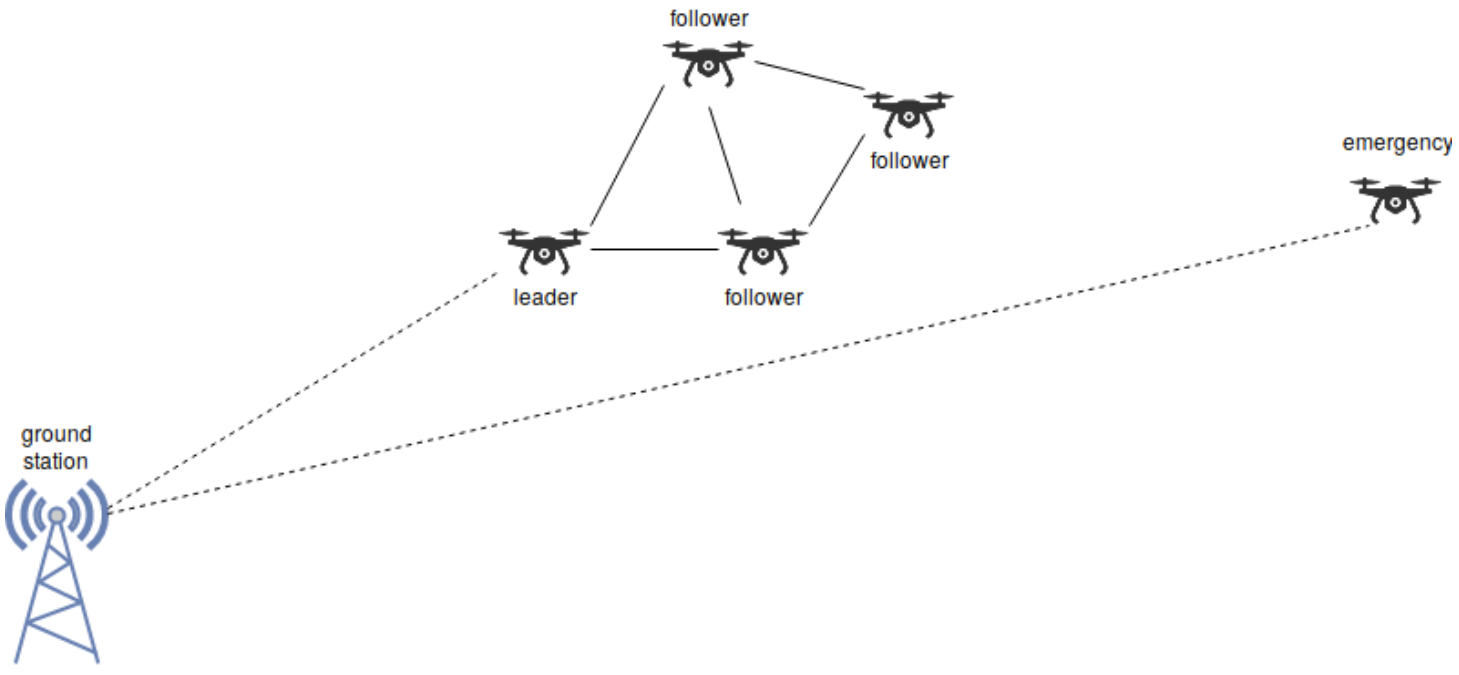
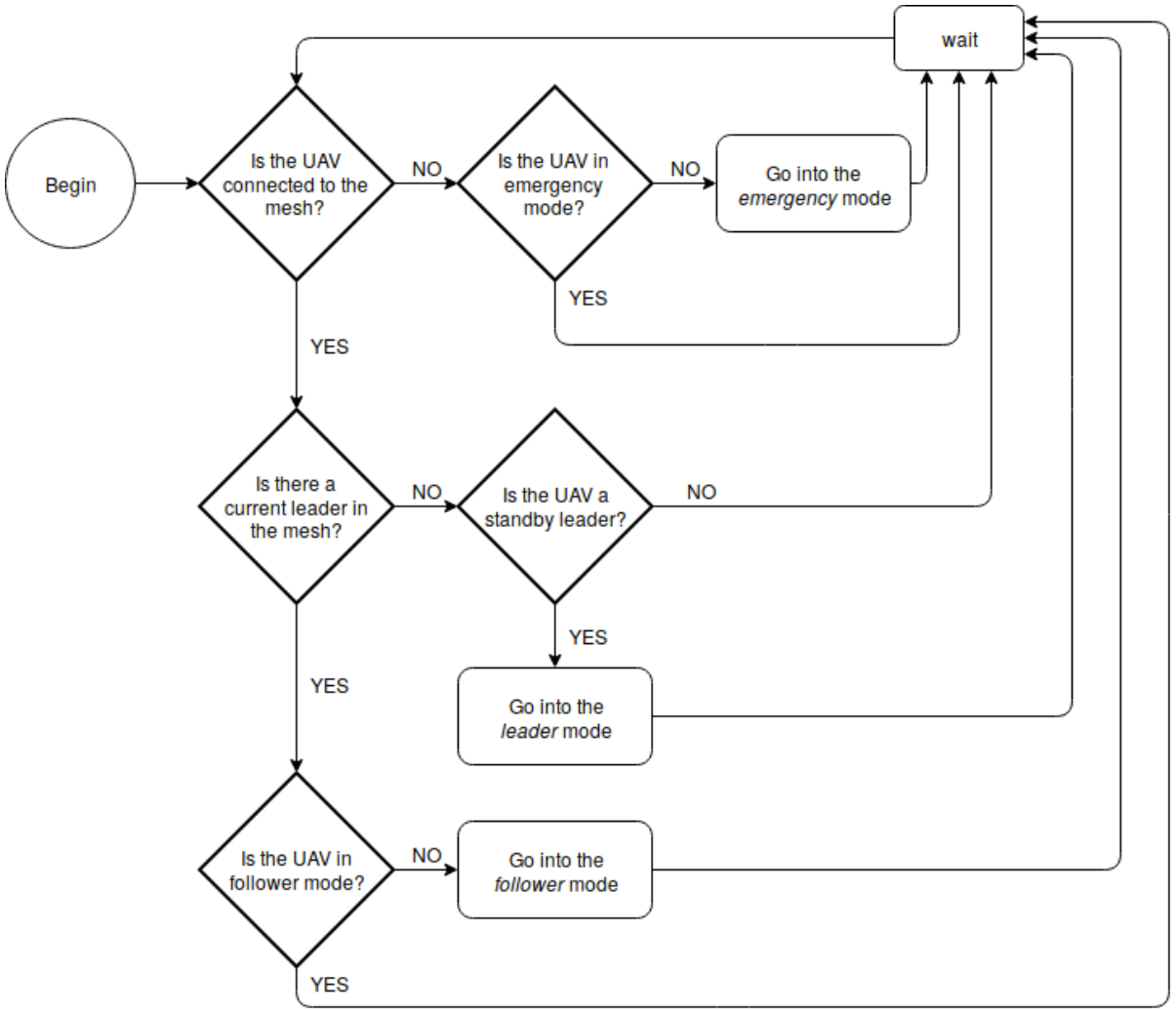


Figure: Dynamic Leader Configuration

- Unlike the Static Leader configuration, the leader is set and changed dynamically by the UAVs themselves.
- Each UAV will be in one of three modes: *leader*, *follower* or *emergency*.
- In each mode, the UAV launches the serialros node, with a configuration file specific to the mode; *leader.conf*, *follower.conf* and *emergency.conf* for the three modes respectively.

RFD modes of UAVs

- *leader.conf* configuration file contains topics of all the UAVs along with the leader's. Thus, the *leader* tranmsits all the required data of all the UAVs to the ground.
- *follower.conf* configuration file is empty. Thus, a UAV in *follower* mode does not transmit any data to the ground.
- *emergency.conf* configuration file contains only the topics of the UAV, itself. In this case, the UAV only transmits its own data to the ground.

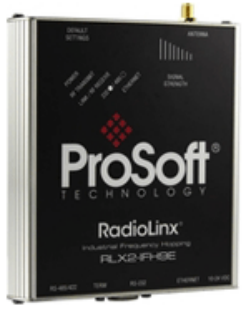


Summary

- We presented a communication architecture for UAV swarms, integrating existing long range and short range communication hardware.
- In short range, we establish a mesh network based on the IEEE standard 802.11s, for inter communication among the UAVs.
- Since the standard supports a TCP protocol stack, integration into ROS is straightforward. We use ROS in a multimaster ROS setup rather than the standard single master setup.
- For the long range communication, we used the popular 900Mhz radio, RFD900x. We have written our own ROS package *serialros*, for interfacing the radios with ROS. The package is available at `https://github.com/saiadityachundi/serialros`.
- We presented two architectures, where one of the UAVs acts as a *leader* and establishes communication with the ground station on behalf of all the UAVs.
  - In *Static Leader*, the *leader* remains unchanged. The RFD is mounted only on the leader.
  - In *Dynamic Leader*, the *leader* is set dynamically. We described the three modes of UAVs in this architecture, *leader*, *follower* and *emergency*.
- While the *Static Leader* can be implemented trivially using *serialros*, we have written another ROS package, *swarmBaba* to implement the *Dynamic Leader*. Source code can be found at `https://github.com/saiadityachundi/swarmBaba`.

Future Work

- Extensive outdoor testing is needed to assess the robustness and range of the architecture.
- Apart from the BATMAN routing protocol that we used in the 802.11s mesh networks, there are other routing protocols that can be explored.
- There are 900Mhz radios which support TCP protocol stack instead. These can be pursued for easier integration into ROS.
- Use of directional antennas can enhance the range of Wifi networks. In particular antenna tracking systems can be used which may extend the range of mesh networks to longer distances.



(a) Prosoft RLX2-IFH9E

(b) Esteem Horizon 900

References I

Ian F. Akyildiz, Xudong Wang, and Weilin Wang. "Wireless mesh networks: a survey". In: *Computer Networks* 47.4 (2005), pp. 445–487. ISSN: 1389-1286. DOI: https://doi.org/10.1016/j.comnet.2004.12.001. URL: http://www.sciencedirect.com/science/article/pii/S1389128604003457.

Timothy Brown et al. "Ad Hoc UAV Ground Network (AUGNet)". In: *AIAA 3rd "Unmanned Unlimited" Technical Conference, Workshop and Exhibit* (Sept. 2004). DOI: doi:10.2514/6.2004-6321..

Seyed Mahdi Darroudi and Carles Gomez. "Bluetooth Low Energy Mesh Networks: A Survey". In: *Sensors* 17.7 (2017). ISSN: 1424-8220. DOI: 10.3390/s17071467. URL: http://www.mdpi.com/1424-8220/17/7/1467.

L. Gupta, R. Jain, and G. Vaszkun. "Survey of Important Issues in UAV Communication Networks". In: *IEEE Communications Surveys Tutorials* 18.2 (Secondquarter 2016), pp. 1123–1152. ISSN: 1553-877X. DOI: 10.1109/COMST.2015.2495297.

S. Morgenthaler et al. "UAVNet: A mobile wireless mesh network using Unmanned Aerial Vehicles". In: *2012 IEEE Globecom Workshops*. Dec. 2012, pp. 1603–1608. DOI: 10.1109/GLOCOMW.2012.6477825.

Minai AA. Yang Y Polycarpou MM. "Multi-UAV Cooperative Search Using an Opportunistic Learning Method.". In: *Journal of Dynamic Systems, Measuremnet, and Control.* (2007), pp. 716–728. DOI: doi:10.1115/1.2764515..

Thank You