Autonomous Navigation Systems (ANS)

- Unmanned Autonomous Systems

Course Objectives

- Practical advances in Artificial Intelligence.
- drones, robotics and driverless vehicles are enabled by artificial intelligence will used to transport goods, perform surveillance, transport people in efficient and safe way.
- This course is to understand the concepts and algorithms that underlie an autonomous vehicle’s understanding of itself and the world around it.

Course Out Come

1. Concepts of UAV/Drones, UGV/Robotics and Autonomous vehicles.
2. Overall understanding of significant elements of UAV/UGV and Autonomous vehicles, the basics of flight, navigation, guidance, sensors, design concepts etc., and various AI algorithms useful in building the Autonomous systems.
3. Build/integrate micro/small to medium size UAV/UGV.
4. Concepts of various sensor elements, data collection, processing and various software tools.

- Understanding of AI algorithms for autonomous systems, followed by design and development of autonomous systems as a case study system.

Chapters/Units

- **Unit – I:**

- **Introduction to Autonomous systems:** Definition, Characteristics, differences between non autonomous Vs autonomous, Types of vehicles, Introduction to navigation and communication.

- **Unit – II**

- **Basics of navigation (Aerial and Ground):** Different types of flight vehicles; Components and functions of an airplane; Forces acting on Airplane; Physical properties and structure of the atmosphere; Aerodynamics – aerofoil nomenclature, aerofoil characteristics, Angle of attack, Mach number, Lift and Drag, Propulsion and airplane structures.

- **Unit – III**

- **UAV / UGV Elements:** Introduction to UAV and UGV**,** DGCA Classification of UAVs; Types and Characteristics of Drones: Fixed, Multi-rotor, Flight controller Software, MAVLINK protocol, Robot Arm Kinematics and Dynamics, Manipulator Trajectory planning and Motion Control, Robot Sensing, Robotic Operating System, Robotic Programming Languages.

- **Unit – IV**

- **Navigation and guidance:** Data Link; Sensors and Payloads: GPS, IMU, Light Detection and Ranging (LiDAR), Imaging cameras, Classification of payload based on applications; Hyper-spectral sensors; Laser Detection and Range (LiDAR); cameras; ultra-sonic detectors; Introduction to navigation systems and types of guidance; Mission Planning and Control, Case studies: Autonomous Obstacle avoidance - Vision, Sonar and LiDAR; Drone Swarms.

https://droneshowsoftware.com/

Introduction to Autonomous Systems

- Definition
- Characteristics
- Levels of Autonomy
- Types of Autonomous Vehicles
- Introduction to Navigation and Communication

Definition: Autonomous / Unmanned Autonomous Systems

- **Unmanned Systems are the machines/vehicles operated remotely without human on board.**
- **Unmanned autonomous systems (UAS) are intelligent machines capable of traveling by air, land, or sea without a human crew on board.**
- **In simplest form autonomy is the ability of a machine to perform task without human input.**
- **Autonomous system is machine, whether hardware or software , that once activated, performs some task or function on its own.**

Unmanned System - NIST Definition

- An Electro-mechanical system [platform]; with no human operator aboard, that is able to exert its power designed missions. May be mobile or stationary. Includes categories unmanned ground vehicles (UGVs), UAVs, Unmanned Under water vehicles (UUVs), Unmanned Surface Vehicles (USVs), Unattended Munitions (Ums), and unattended ground sensors (UGSs). Missiles, rockets, and their sub-munitions and artillery are not considered unmanned systems.

Autonomous systems – concepts and relation to AI

- The concept of autonomy is key to vision of next generation human less vehicles development.
- Promising technology with integration of smart services and systems minimizing human intervention.
- Autonomous vehicles focuses almost exclusively on AI and learning techniques with emphasis on autonomous system design issues.
- Autonomous systems involve **agents and objects coordinated** in some common environment, so that, their collective behavior meets a set of global goals.
- The main characteristics of autonomous systems is their ability to handle knowledge and adaptively respond to environment changes.
- Autonomous systems should be associated with functionality and not with a specific technique.
- Machine Learning is essential for autonomy although it can meet only a small portion of the needs implied by autonomous system design.

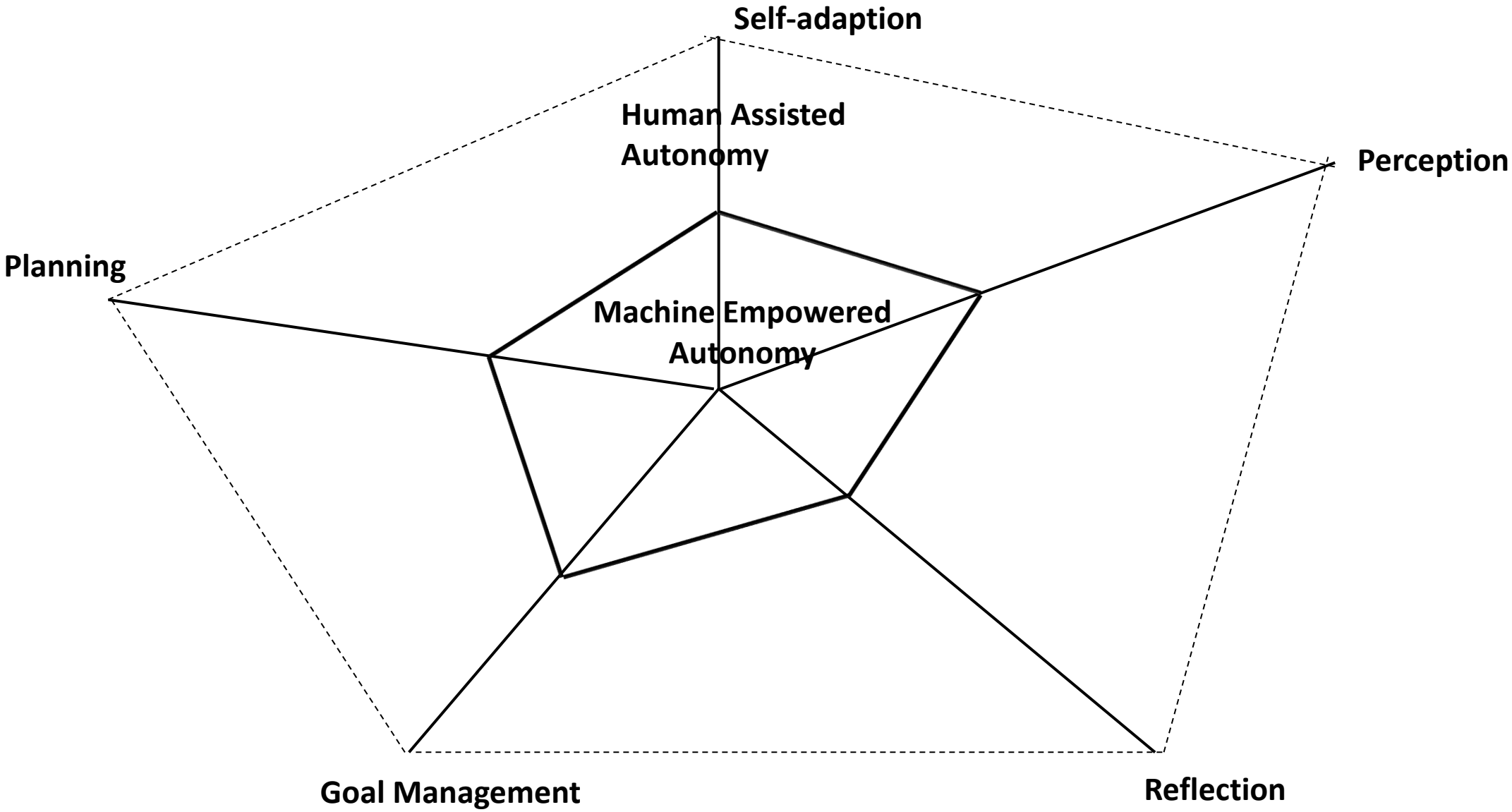
Characteristics – Autonomous Systems

- Five major aspects / characteristics of AS

1. **Perception :** Removing ambiguity / vagueness from complex input data and determining relevant information.
2. **Reflection:** Building/updating a faithful environment run time model.
3. **Goal Management:** Choosing among possible goals the most appropriate ones for a given configuration of the environment model.
4. **Planning:** To achieve chosen goals.
5. **Self adaptation:** the ability to adjust behavior through learning and reasoning and to change dynamically the goal management and planning process.

All the 5 aspects are mutually independent each other. The first two aspects deal with "understanding" the situation environment. The third and forth aspect deal with autonomy of decision, The 5th aspect ensures adequacy of decisions with respect to the environment situation.

Human Assisted Vs Machine Empowered Autonomy



Levels of Vehicle Autonomy – In a nutshell

- Total 6 Levels
- Level 0 – No Driving Automation i.e. fully manual
- Level 1 – Driver Assistance
- Level 2 – Partial Driving Automation
- Level 3 – Conditional Driving Automation
- Level 4 – High Driving Automation
- Level 5 – Fully Driving Automation

Autonomy Levels – In Brief

Level 0	No automation
Level 1	Driver assistance required (“hands on”)
	The driver still needs to maintain full situational awareness and control of the vehicle e.g. cruise control
Level 2	Partial automation (“hands off”)
	Auto pilot manages both speed and steering under certain conditions, e.g. highway driving.
Level 3	Conditional Automation (“eyes off”)
	The car, rather than the driver, takes over actively monitoring the environment when the system is engaged. However, human drivers must be prepared to respond a “request to intervene”.
Level 4	High automation (“mind off”)
	Self driving is supported only in limited areas (geofenced) or under special circumstances, like traffic jams
Level 5	Full automation (“steering wheel optional”)
	No human intervention is required e.g. a robotic taxi

Levels of Driving Automation – In detail

0
NO AUTOMATION Manual Control. The human performs all driving tasks (steering, acceleration, breaking etc)

1
DRIVER ASSISTANCE The vehicle features a single automated system (e.g. it monitors speed through cruise control)

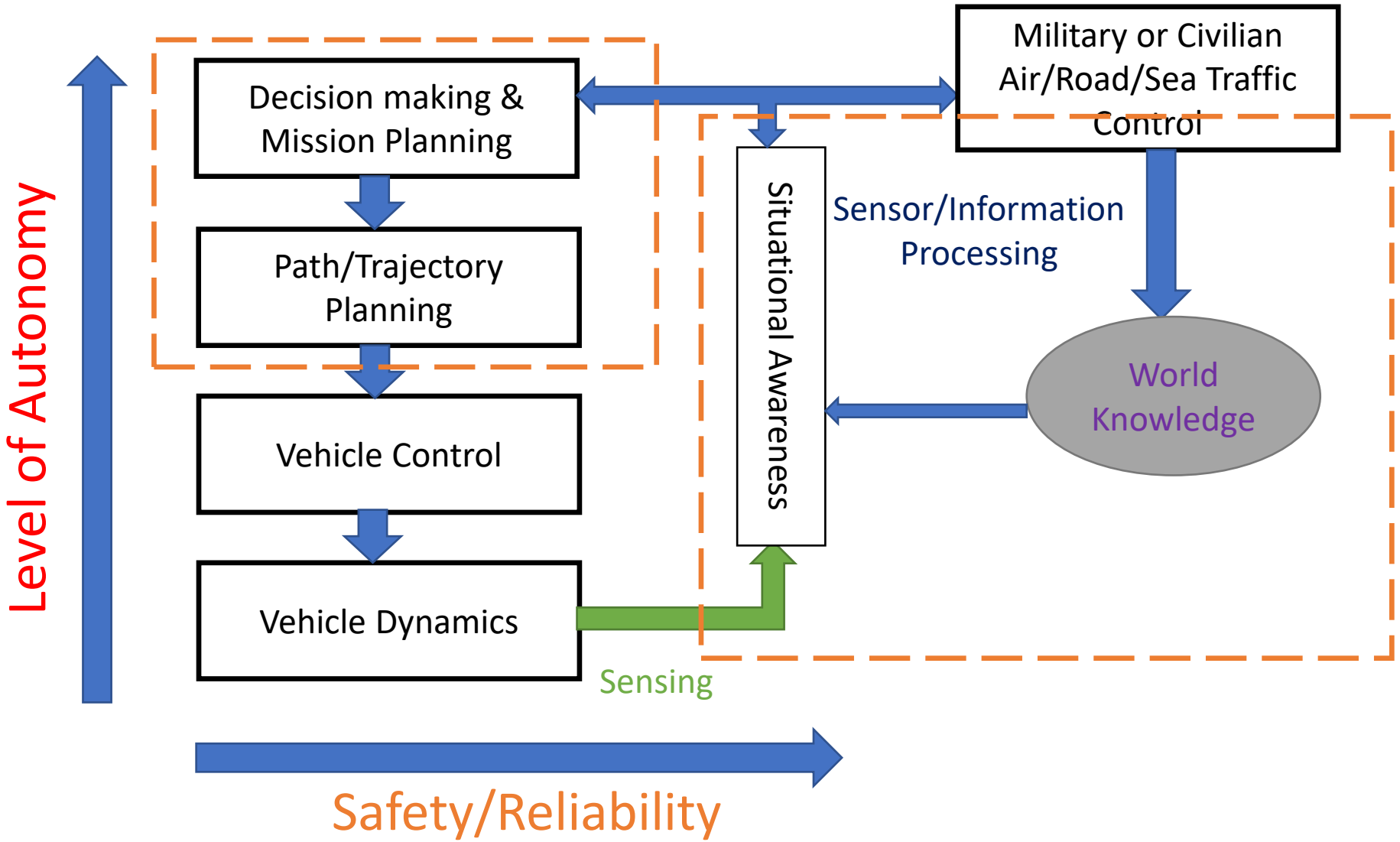
2
PARTIAL AUTOMATION The vehicle can perform steering and acceleration. The human still monitors all tasks and can take control at any time.

3
CONDITIONAL AUTOMATION Environmental detection capabilities. The vehicle can perform most driving tasks, but human override is still required.

4
HIGH AUTOMATION The vehicle performs all driving tasks under specific circumstances. Geo referencing is required. Human override is still an option.

5
FULL AUTOMATION The vehicle performs all driving tasks under all conditions. Zero human attention or interaction is required

Autonomous System - Components



Autonomy or Autonomous - Characteristics

- **Level of independence** :
  - human grant a system to execute a given task.
- **Condition or quality of being self-governing**
  - to achieve an assigned task based on the systems' own situational awareness (integrated, perceiving, analyzing), planning and decision making.
- **Autonomy refers**
  - spectrum of automation in which independent decision making can be tailored for a specific mission, level of risk, and degree of human-machine learning
- **Levels of autonomy**
  - range from remotely controlled (non autonomous), operator assistance, Partial Automation, Conditional Automation, High Automation or Full Automation.
- **Challenges**
  - Advanced Computing, sensing and electronic technologies
  - Mechanical designs, Maneuvers

Operating modes of Autonomous

- Being completely able to carry out missions/tasks with minimum human interaction is an ultimate goal for unmanned systems.
- Different levels of autonomy shall be achieved towards the goal depending on the complexity of tasks : fully autonomous or not.
- NIST (National Institute of Standards and Technology) defines
  - Fully Autonomous
  - Semi – Autonomous
  - Teleoperated
  - Remotely Operated

Fully Autonomous

- Carry out the delegated task/mission without human interaction where all decisions are made onboard based on sensors observations adapting to operational and environmental changes.

Semi Autonomous

A human operator is needed for high level mission planning and for interaction during the movement when some decisions are needed. The Vehicle can maintain autonomous operation in between the interactions.

- For Example : An operator can provide a list of way points to guide the vehicle where it can manage to move safely towards these positions with obstacle avoidance capability.

Teleoperated

- The remote operator relies on feedback from onboard sensors to move the vehicle either by directly sending control commands or intermediate goals with no obstacle avoidance capabilities. This mode is used in Beyond-Line-Of-Sight (BLOS) applications.

Remotely Controlled

- A remote pilot is needed manually control the vehicle without sensors feedback which can be used in Line-of-Sight (LOS) applications.

Types of Vehicles - Unmanned Autonomous Systems (UAS)

- **Unmanned Systems(US) or Vehicle (UV)** can be defined as an “electro-mechanical system with no human operator on board, that is able to exert its power to perform designed missions.
- **Unmanned Autonomous Systems (UAS)** are high tech, intelligent machines capable of travelling by air, land or sea without a human crew on board.
- UAS are built with autonomous decision making (mission planning / task allocation), situational awareness, guidance(path planning) and controlled algorithms
- **Examples are :**
  - Unmanned Aerial Vehicles (UAV)’s – Drone
  - Unmanned Ground Vehicles (UGV) – Robots and Driver less cars/vehicles
  - Unmanned Water Vehicles (UWV) – Under water drones/Robots

Types of Vehicles : Autonomous Contd…

- UAV : Unmanned Aerial Vehicles
  - Evolved greatly over recent decades with prevalent use in military and civilian applications such as search and rescue, wireless sensor networks and Internet of Things (IoT), Remote Sensing, Surveillance and Monitoring, 3D mapping, object grasping, aerial manipulation, underground mine exploration and tunnel inspection etc.
- UGV: Unmanned Ground Vehicles
- UWV: Unmanned Water Vehicles

Towards fully Autonomous Operations

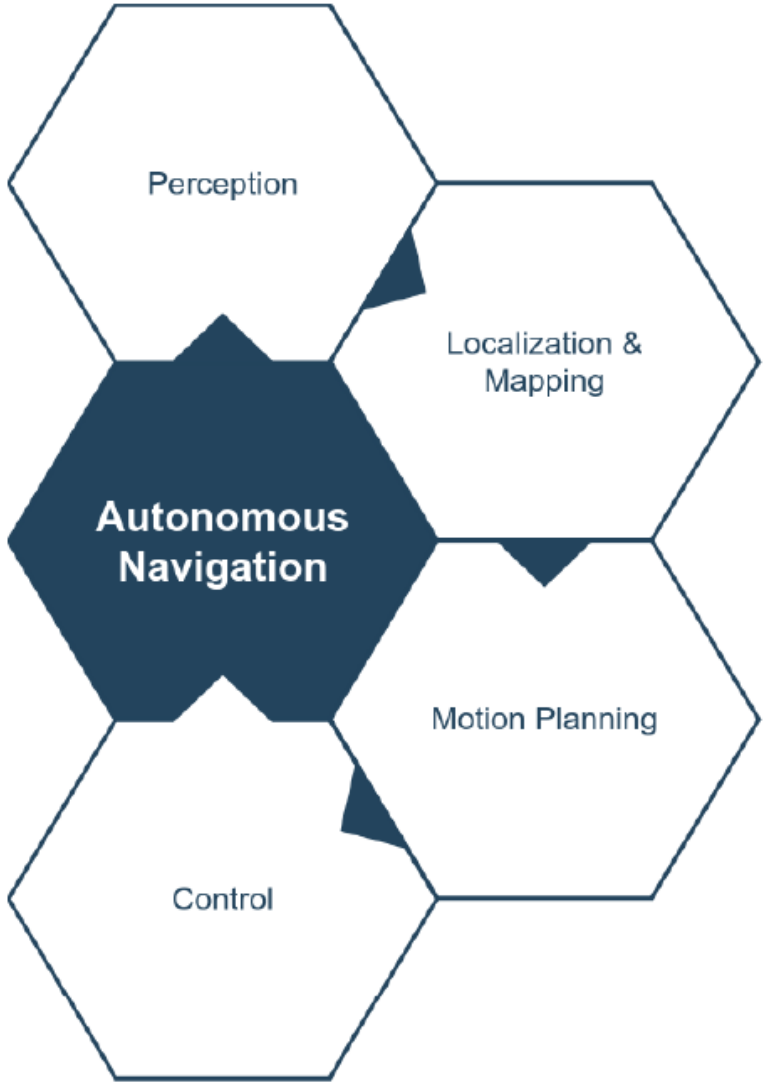
Computational model for autonomous systems

Introducing : Autonomous Navigation and Communication

- Autonomous Navigation problem can generally be defined as the vehicle's ability to reach a goal location while avoiding collisions with surroundings without human interaction.
- This is very challenging problem as it is important to achieve safe navigation to avoid causing damage or injuries.
- More Complex to the development of autonomous navigation methods in order to ensure reliability and robustness.
- The challenges to overcome are :
  - Sensing Capabilities
  - Allowed payload capacity
  - Flight time (Endurance)
  - Energy Consumption
  - Communication
  - Actuation and control effort

Autonomous Navigation

- Mobility-related modules are the core components needed to ensure collision-free navigation in all applications.
- By considering only the mobility-related components, a popular modular structure for autonomous navigation is adopted in the literature which consists of the following modules/subsystems



- Perception;
- Localization and Mapping
- Motion Planning and Obstacle Avoidance;
- Control.

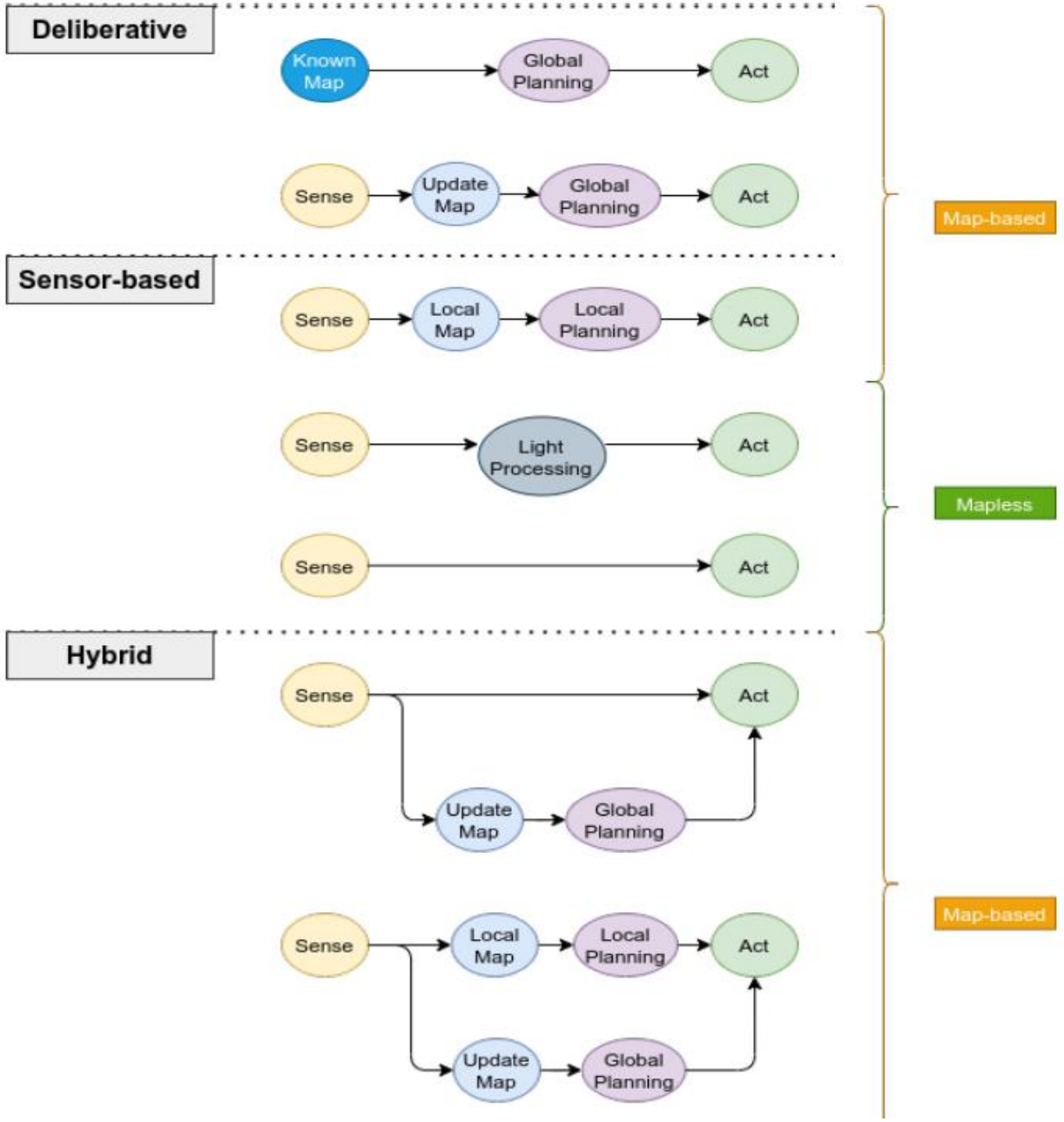
Navigation Techniques

- A crucial part of autonomous navigation is to ensure that the vehicle can move while avoiding collisions with its surroundings.
- This is a general problem in robotics which can be addressed by motion planning or reactive control.
- Generally, the motion planning problem can roughly be described as trying to find collision-free trajectories between initial and final configurations while satisfying some kinematic and dynamic constraints. A configuration in this case refers to the position and orientation of a mobile robot where a configuration
- Space is the set of all possible configurations. The dimension of the configuration space equals the number of controllable degrees of freedom. For example, planning motions for quadrotors can be carried out in a space of their 3D position coordinates and heading (yaw) angle while motions for omnidirectional (fully actuated) UAVs can be planned considering all translational and rotational states (6DOF).

Navigation Paradigms

Existing navigation techniques for autonomous mobile robots in general can be classified into deliberative (global planning), sensor-based (local planning) or hybrid
Deliberative approaches require a complete knowledge of the environment represented as a map. Global path planning methods can then be used to search for safe and optimal paths.
Classical path planning algorithms can be categorized into:

Search-based methods (ex. Dijkstra, A, D, etc.);
- Potential field methods (ex. navigation function, wavefront planner, etc.);
- Geometric methods (ex. cell decomposition, generalized Voronoi diagrams, visibility
graphs, etc.);
- Sampling-based methods (ex. PRM, RRT, RRT*, FMT, BIT, etc.);
- Optimization-based methods (PSO, genetic algorithms, etc.).



Introduction to Drones/UAV/UAS

- Miniature plane piloted remotely, capable of flying and gathering information with minimal human intervention and without a person on board.
- Often the words (Aerial Vehicle), Unmanned Aerial System (UAS) and Drones are used interchangeably.

Benefits/Advantages

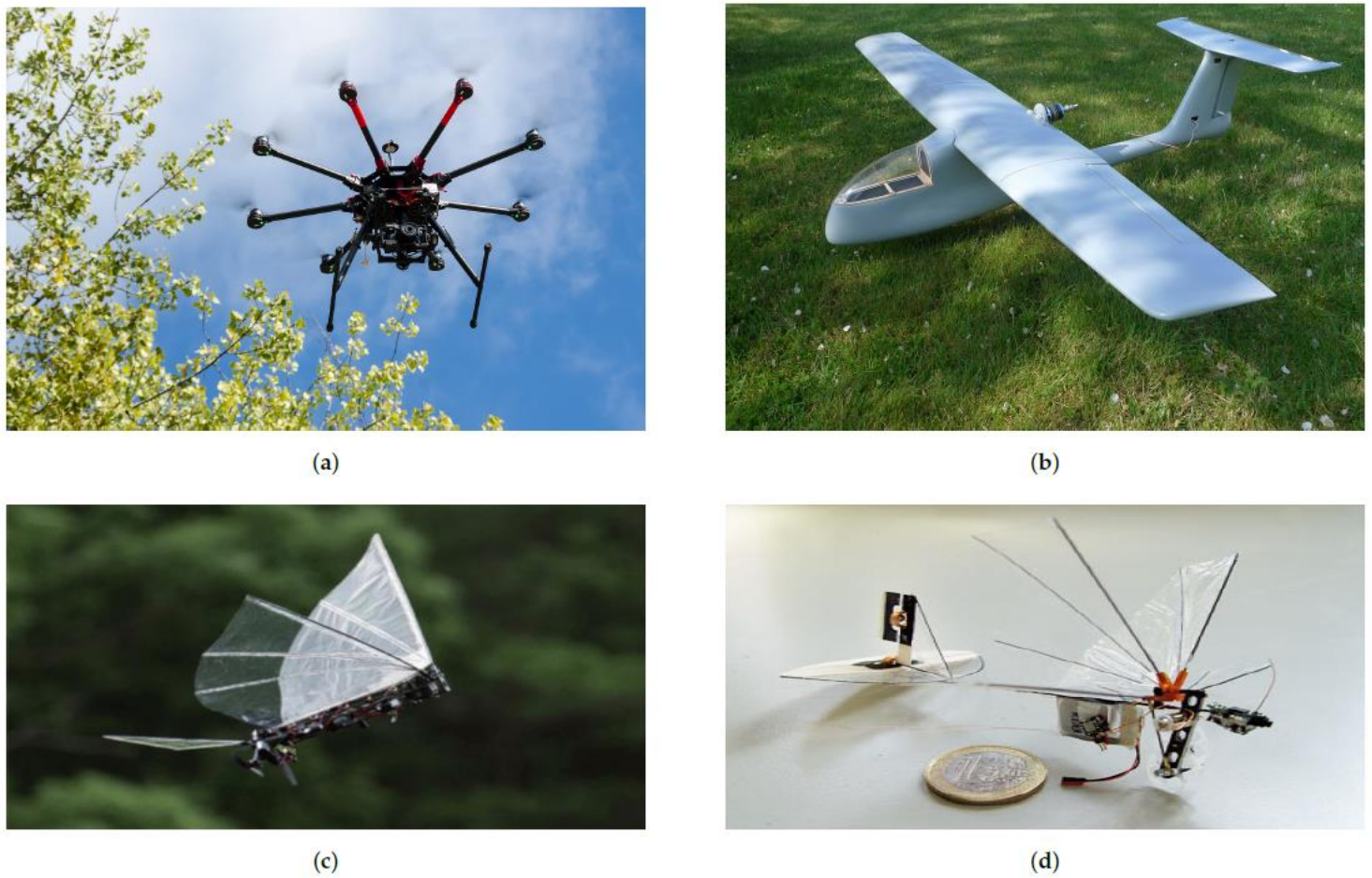
- Controlled and Owned by users themselves, unlike more expensive satellite and aerial imagery.
- Provides opportunities to produce **timely, high quality and cloud free very high resolution imagery** for mapping applications and thus allowing quick decision making.

Drones

- Definition:
- Types with images of drones: Fixed Wing, Multi Rotor
- A Video of drone flying and typical images(data acquired from drones)

UAVs' can be categorized into

- single-rotor : helicopter;
- multi-rotor : tricopter, quadrotor, hexacopter, etc.;
- fixed-wing;
- hybrid ;
- flapping wings: Ornithopters and Entomopters.



(a) (b) (c) (d)

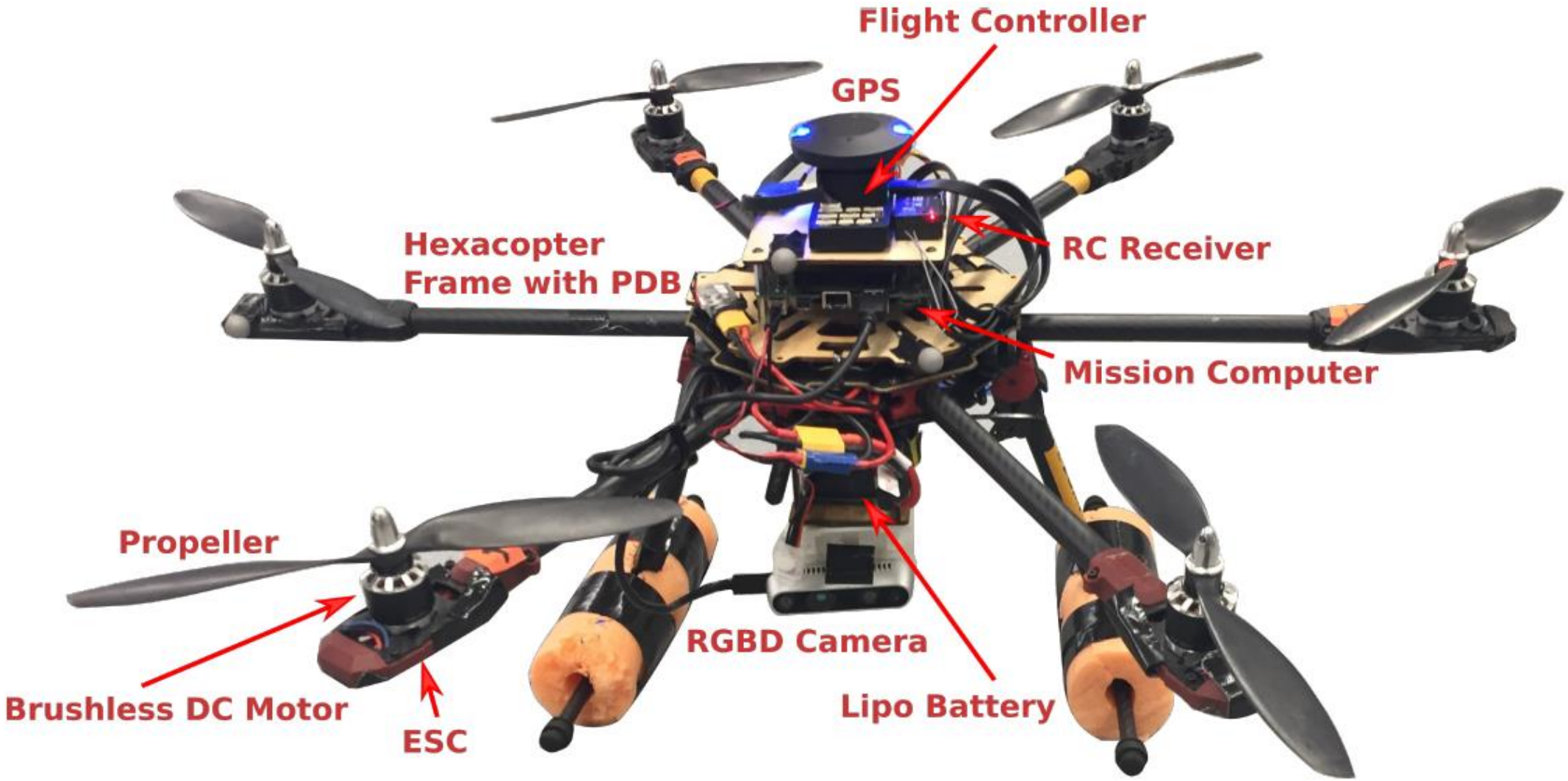
Figure 1. Different UAV types based on control configurations. (**a**) Multirotor (Hexacopter), (**b**) Fixed-Wing, (**c**) Ornithopter flapping-wing UAV (Robo Raven) [37], (**d**) Entomopter flapping-wing UAV (DelFly Micro).

Types of UAS/Drone

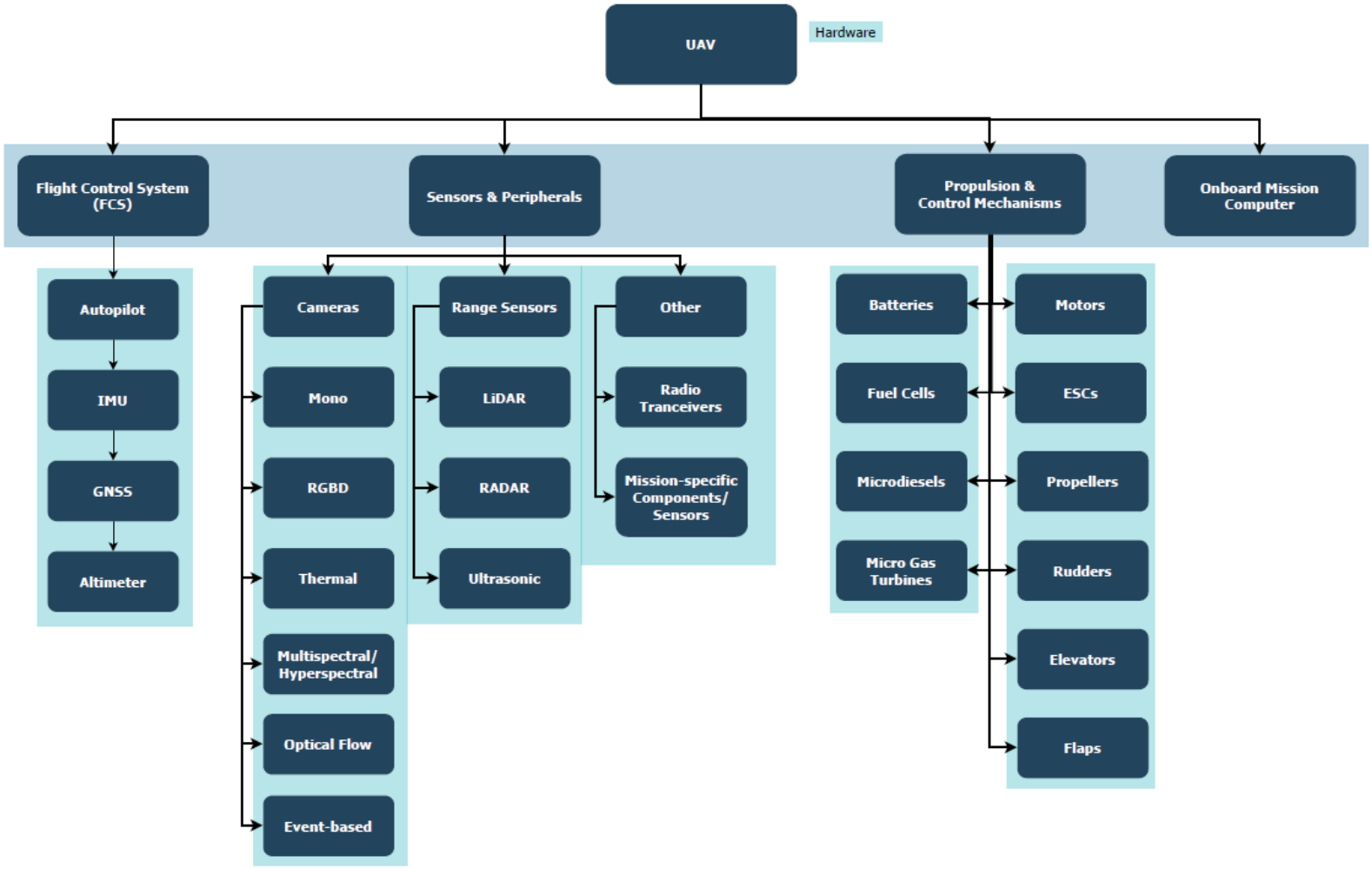
- **Multi rotor Copter**
  - **Quad :**
    - **Four - arms, propellers, motors**
  - **Hexa**
    - **Six – arms, propellers, motors**
  - **Octa**
    - **Eight – arms, propellers, motors**

- **VTOL(Vertical Take Off and Landing)**
  - **Fixed Wing drone and able to take of vertical**
  - **Longer flight time**

Drone – Anatomy



Drone - Hardware Components commonly used



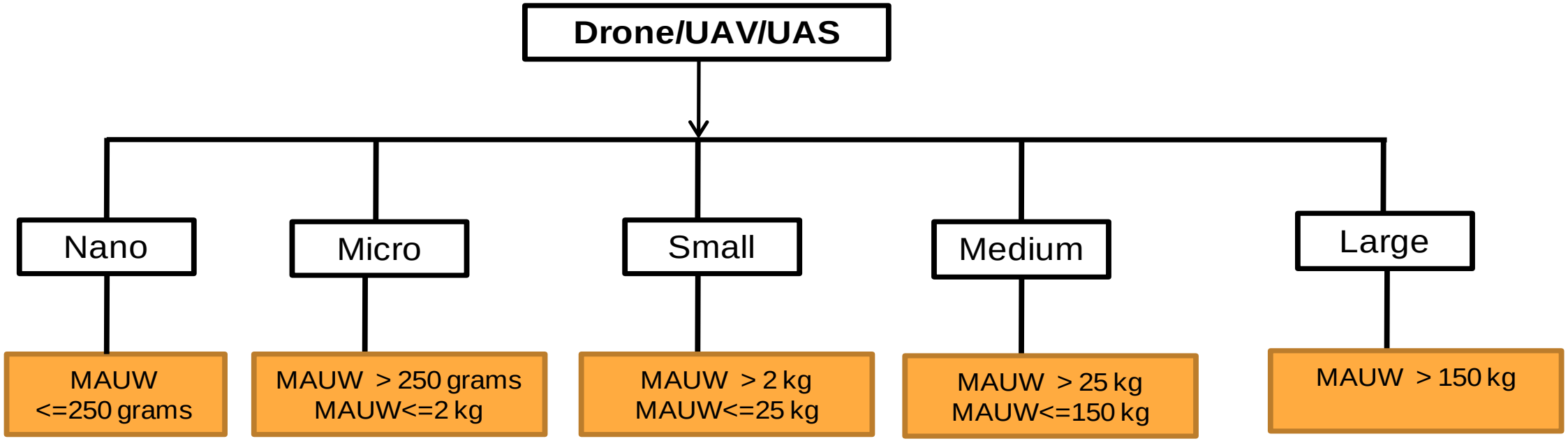
Drone Applications

- Corridor Mapping
- Surveillance
- High scale mapping
- Sporadic Event Monitoring and Management
- Ad Hoc Communication network
- Entertainment: Movies, Hobbies [some pictures]
- Swam Flying : [Rock shows etc]

Threats

- Security
- Privacy
- …

Drone Categories/classification as per Civil Aviation, India



MAUW – **M**aximum **A**ll **U**p **W**eight

Drone Licensing (operator and flight) is essential to operate in India and will be effective in India from December 2018 , as per Drone Regulations, DGCA, August 2018.

UAV Types

UAVs can be classified based on several factors such as size, mean takeoff weight, control configuration, autonomy level, etc. For example, classifications of UAVs based on size according to the Australian Civil Aviation Safety Authority (CASA) are:

- Micro: less than 250 g;
- Very Small: 0.25–2 kg;
- Small: 2–25 kg;
- Medium: 25–150 kg;
- Large: More than 150 kg.

Large UAVs are mainly used in tactical missions and military applications; for more detailed classifications related to military use.

Drone Regulations – Civil Aviation Requirements (CAR) for Remotely Piloted Aircraft System(RPAS)

- **Civil Aviation Requirements(CAR)**
  - Unique Identification Number (UIN)
  - Unmanned Aircraft Operator Permit (UAOP)
  - Operational requirements of Civil Remotely Piloted Aircraft Systems (RPAS)
- **Application Process**
  - Shall obtain Equipment Type Approval (ETA) from WPC (Wireless Planning and Coordination Wing), DOT, for operating in de-licensed frequency band(s).
  - All models except Nano category should apply to DGCA for import clearance. Based upon the import clearance issued by DGCA, DGFT shall issue license for import of RPAS.
  - Locally purchased RPAS shall have ETA from WPC wing, DOT operating in de-licensed frequency band(s).
- **License application and Vehicle Registration**
  - Upon receipt of import license, the applicant shall apply to DGCA for UIN/UOAP, as applicable.
  - Applications are processed on case-to-case basis through **“Digital Sky Platform”.**

Hive Mind Swarming Autonomous Drones

Robot Boat Swarm in Action

Practical – Lab

1. Assemble, integrate and demonstrate the Quad copter with all necessary parts.
2. Calibration of UAV(Quadcopter) using Ardupilot Mission planner and demonstrate the calibrated IMU parameters
3. Write a program to read Telemetry parameters using Serial Port using TTC device.
4. Write a program to read GPS coordinates on Raspberry Pi and Arduino micro controller
5. Write a program to send MAVLINK commands to Pixhawk version of Flight Controller
6. Write Software in the Loop (SITL) / MAVproxy
7. Write a program to connect Dronekit for communication and testing various commands
8. Write a program for Copter SITL/MavProxy to the commands such as Flight
9. Use Mission planner for flight path panning and demonstrate the transfer of planning transects to flight controller
10. Arm, take off, change of flight mode parameters, flying machine and geofence
11. Write object avoidance program using the following sensors and test them on UGV(Robot)
    a. Sonar
    b. LiDAR
    c. Camera
12. Write a Program to communicate UAV/UGV using BLE/WiFi/UHF/Cellular devices
13. Write a program for navigation in GPS denied Environments

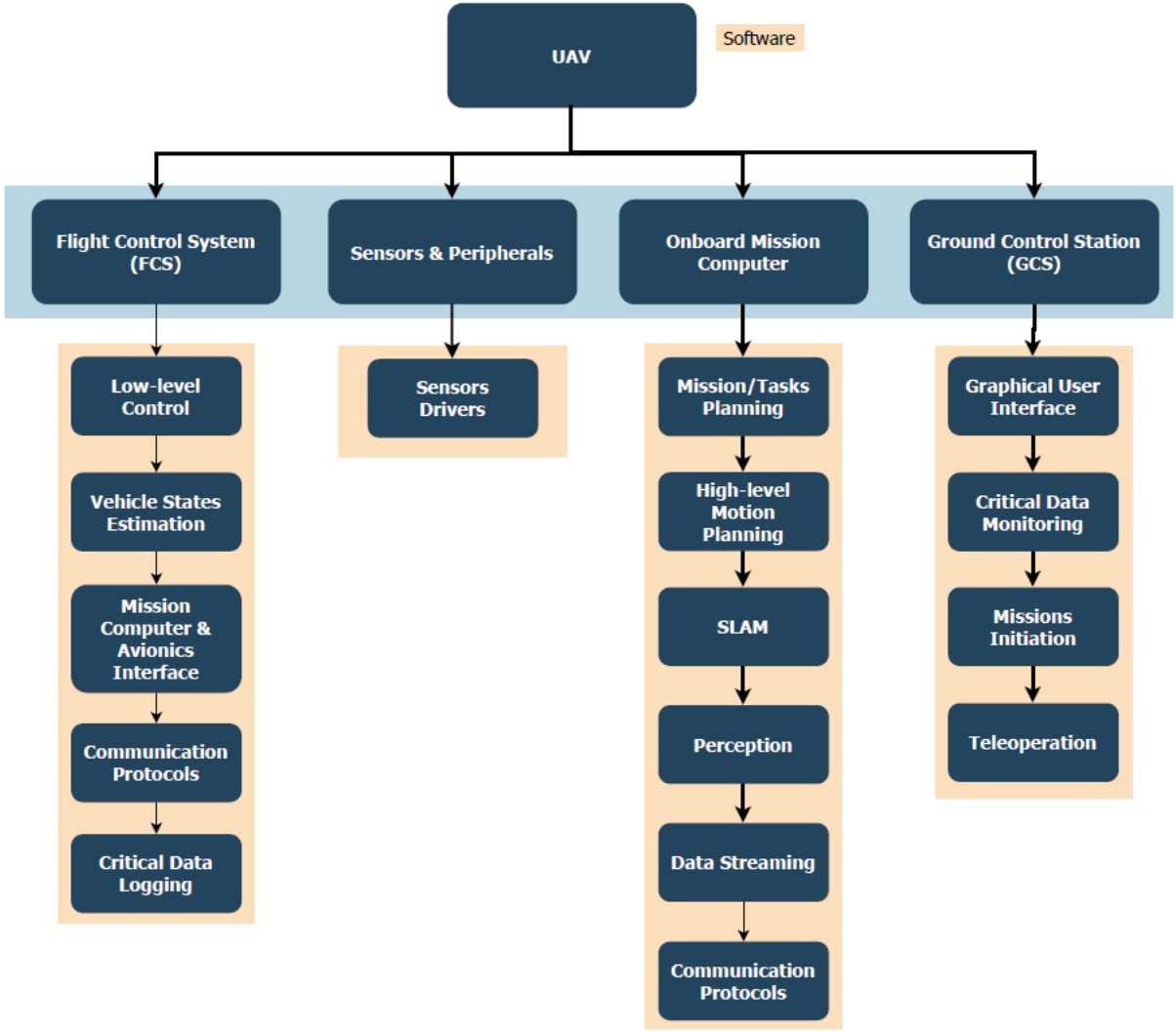
Programming Languages

- Python
- C

Software Tools

- Ardupilot Mission Planner
- SITL / MAVproxy
- ROS : Robotic Operating System
- ChibiOS or latest Embedded RTOS

Software Components commonly used



Demo – Drone Anatomy

Applications of UAV

They have a variety of functionalities in different industries. A few of the most popular industries that use them are:

- Military
- Space Research
- Product Delivery
- Agriculture
- Search and Rescue
- Internet Carrier

Applications of Drones/ UAVs in Military

- The first drones were used during wars to drop bombs without the requirement of a pilot to carry them into the enemy zone. It is also used for surveillance, reconnaissance, aerial photography, specific target tracking, and counter air threats.
- It reduces the risk factor involved during such missions due to minimal involvement of human personnel.

Applications of Drones/ UAVs in Space Research

- The United States and the United Kingdom have been the pioneers in the use of drones for space research. NASA’s most popular drone X-37B has made multiple visits to space for a number of classified missions.
- Following NASA’s route, other space agencies from across the globe are also using drones for their space researches.

Applications of Drones/ UAVs in Food and Product Delivery

- The food and product delivery industry is one of the most booming sectors today.
- The most challenging factor that is considered in this industry is the problem of delivering food through roads and delivery agents.
- Drones can be a direct profit-making solution for this. In fact, a few countries have started using this technology.

Applications of Drones/ UAVs in Agriculture

- One of the most strenuous jobs in the world is agriculture. Growing crops in the vast lands is a tedious job without a doubt. If this could be done in a manner with reduced human efforts, it is one of the best things that can be explored in this century.
- Drones have become really popular in this sector. A few of the functions for which drones are used are spraying water, pesticides, and doing video surveillance in the vast field.

Applications of Drones/ UAVs in Search & Rescue (SAR)

- Drones is very useful in Search & Rescue (SAR) operations as they can even go in areas with difficult access.
- They can also patrol larger areas as compared to an individual.
- Further, with the addition of thermal imaging devices on it, the identification of people in distress is much more faster and reliable.

Applications of Drones/ UAVs as Internet Carriers

- As stated earlier, Google and Facebook are already working on concept of some giant drones that can carry signal to remote locations for direct internet access.

Advantages of Drones/ UAVs

- Quality Imaging.
- Accessibility to hard to reach areas.
- Reduced human effort and risks.
- Need minimal launch time.
- Precise Operation.
- Reliable.
- Saves time.

Disadvantages of Drones/ UAVs

- Privacy becomes vulnerable if wrongly used.
- Safety of people/ equipment depends on operator skills
- Laws for Drones are not so specific yet. Still Evolving.
- Shorter Lifespan
- Can be easily hacked

Unit - II

- **Basics of navigation (Aerial and Ground):**
- Different types of flight vehicles; Components and functions of an airplane;
- Forces acting on Airplane; Physical properties and structure of the atmosphere;
- Aerodynamics – aerofoil nomenclature, aerofoil characteristics,
- Angle of attack, Mach number, Lift and Drag, Propulsion and airplane structures.

History of Drones/UAVs

1849, Air Balloons

Austrians used balloons to drop bombs during attack on city of Venice

1935, Queen Bee

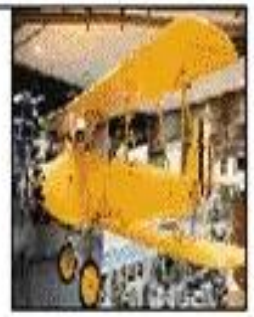
Created in UK, this drone was used by military for moving target practice

1918, Kettering Bug

Designed to drop bombs on targets during WWI. The war ends before the Bug was used.

1937, Curtiss N2C-2

Used by US Navy as radio controlled aircraft



1941, Radio Plane by Reginald Denny

During WWII, Reginald Denny from US created first remote controlled aircraft called Radioplane.

1973, Mastiv UAV & IAA Scout

Israel developed both unpiloted surveillance machines.

1964 - 1969 The Lightening Bug

It was created for surveilance during Cold War by US

1982, Battlefield UAVs

A major milestone. Israel changed the way world was seeing Drones. Destroyed many Syrian aircrafts with minimal loss using UAVs.

History of Drones/UAVs

1986, Reconnaaissance Drones

A joint venture between US and Israel produced RQ2 Pioneer, a medium size reconnaissance drones.

2003 - Present Commercial Drones

Commercial Drones gain popolarity in construction, real estate, search and rescue, etc.

2001, Predator

Designed in US, this drone is used for surveillance

2014, Product Delivery by Drones

Amazon CEO announces the drone delivery plan, opening the door for product delivery use.

Use of Drones for Providing Internet by Facebook

Different Types of vehicles

- Based on the type of aerial platform used, there are 4 major types of UAVs.

1. Multi Rotor Drone
2. Fixed Wing Drone
3. Single Rotor Drone
4. Fixed Wing Hybrid VTOL

Multi Rotor Drone

- Multi Rotor Drone uses multiple propellers to navigate and fly. They are utilized for common uses such as photography and video surveillance.
- They are the most common of all UAVs.

They can be further classified in four most popular categories based on the number of propellers being used. They are:

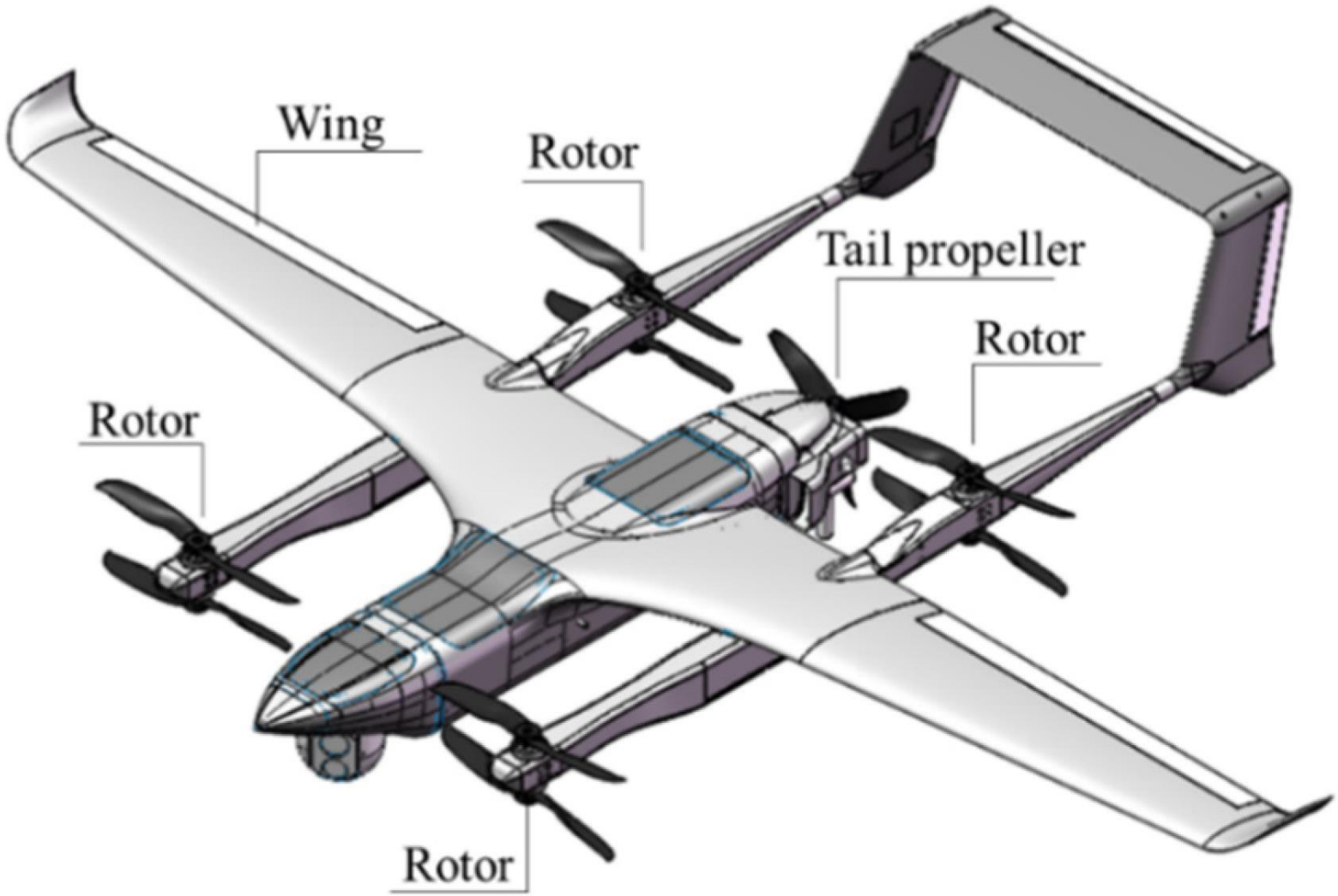
Multi Rotor Drone contd…

- **Tricopter** – Three Propeller Drones
- **Quadcopter** – Four Propeller Drones
- **Hexacopter** – Six Propeller Drones
- **Octocopter** – Eight Propeller Drones

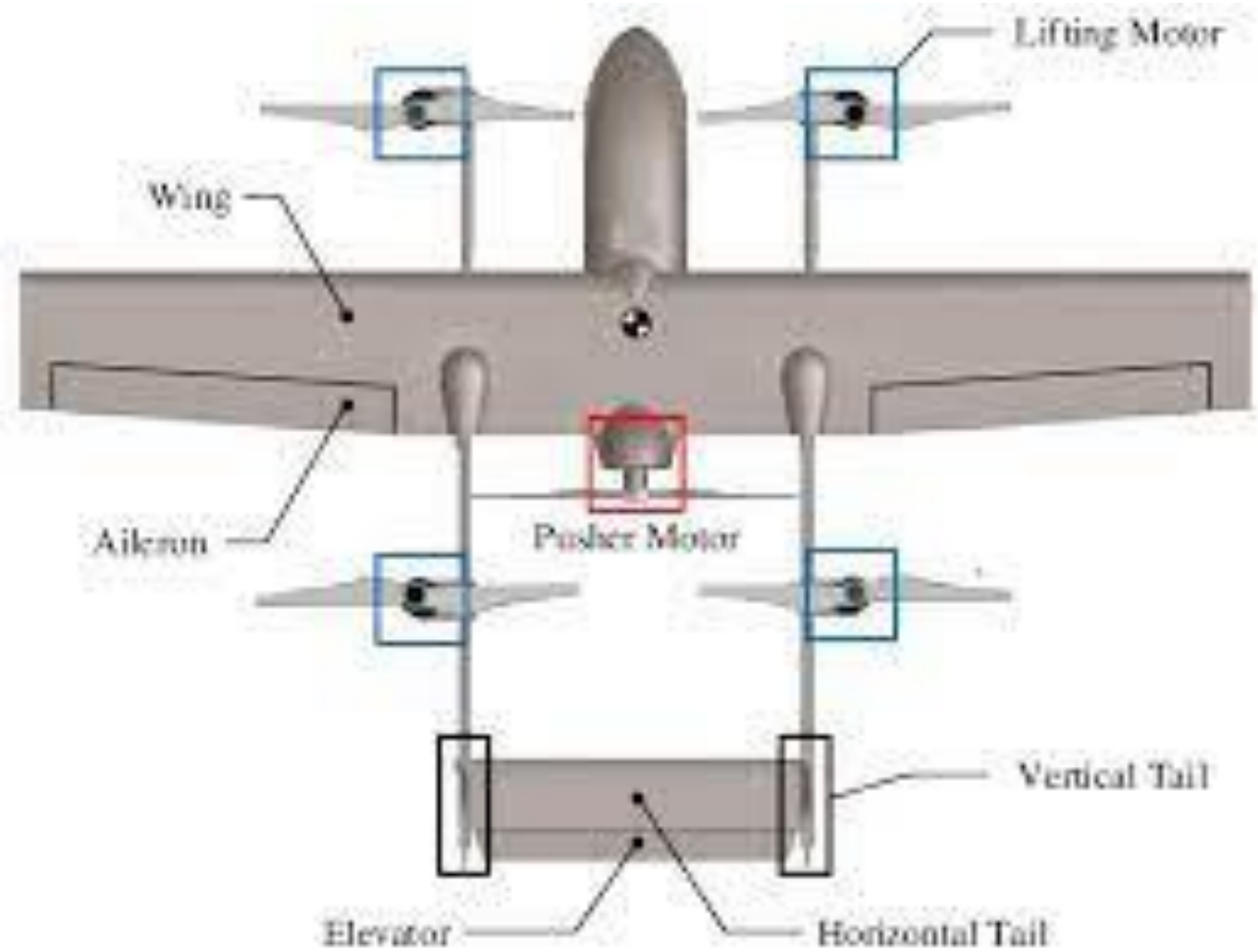
Fixed Wing

- Fixed Wing drones have wings in place of propellers just like an airplane.
- They cannot hover at one place. They fly on the set course till their energy source is functional.
- Further categorized into : VTOL and Landing Strip Take of and Landing

Fixed Wing



Fixed Wing


Lifting Motor
Wing
Aileron
Pusher Motor
Vertical Tail
Elevator
Horizontal Tail

Single Rotor Drone

- As the name suggests, a Single rotor drone has only one rotor and a small tail to control it direction. They resemble more like a helicopter and are very energy efficient.

Single-rotor drone types are strong and durable. They look similar to actual helicopters in structure and design. A single-rotor has just one rotor, which is like one big spinning wing, plus a tail rotor to control direction and stability. Single rotor drone being used for agriculture in Australia.

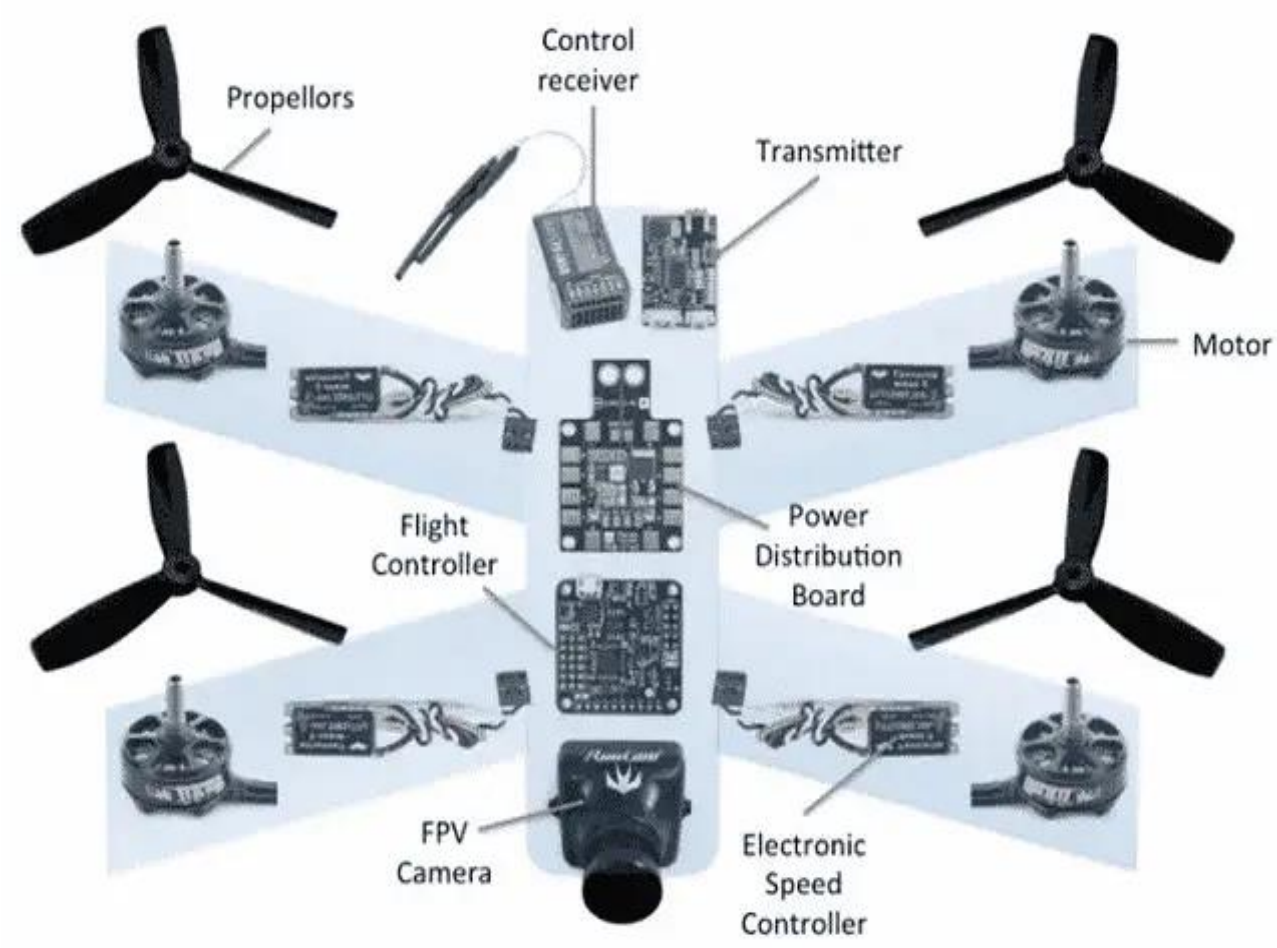
Fixed Wing Hybrid VTOL

- VTOL stands for Vertical take Off & Landing. Fixed Wing Hybrid VTOLs uses propeller(s) to lift off and wings for gliding.

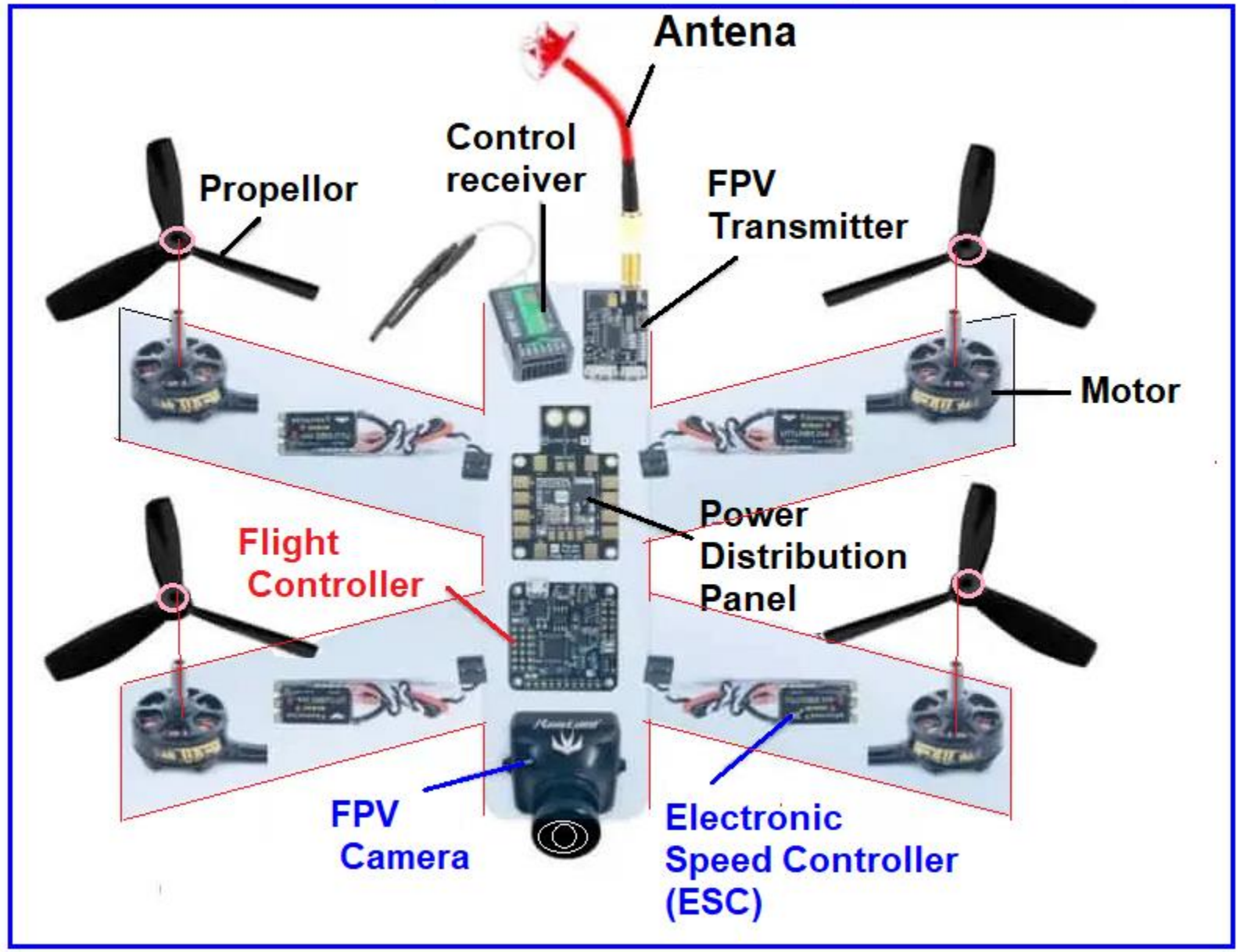
A fixed-wing drone has one rigid wing that is designed to look and work like an aeroplane, providing the lift rather than vertical lift rotors. Hence, this drone type only needs the energy to move forward and not to hold itself in the air. This makes them energy-efficient.

Components and functions of airplane

1. Propellers/ Wings
2. Chassis
3. DC motors
4. Flight Controller
5. Electronic Speed Controllers
6. Landing Gear
7. Transmitter
8. Receiver
9. GPS Module
10. Battery
11. Camera



Parts of Drone



1.Propellers/ Wings

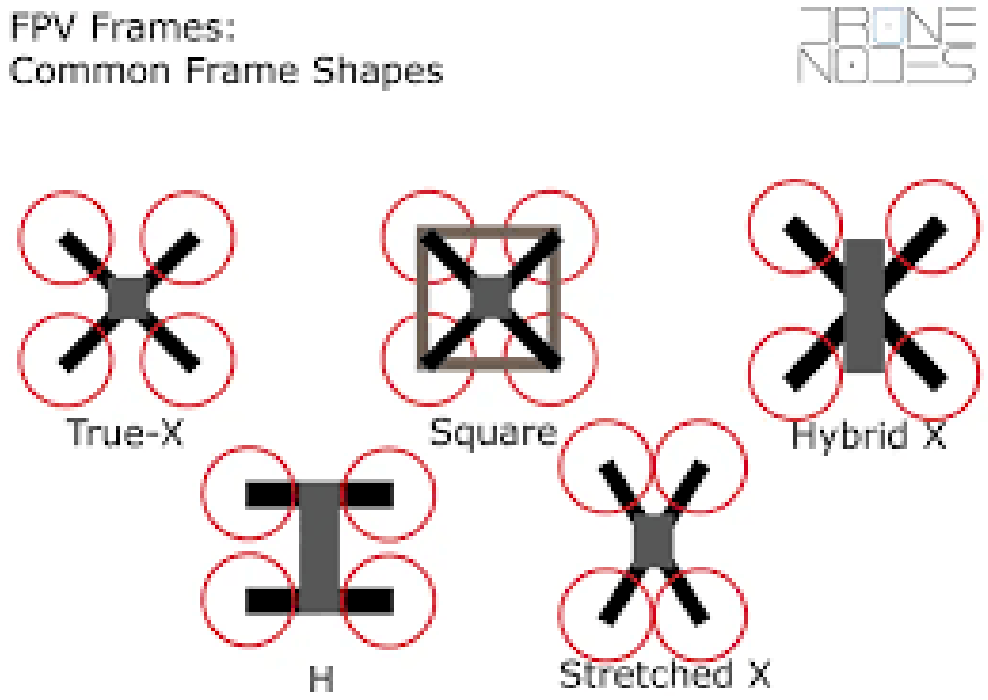
- The Drones/ UAVs use either propellers/ wings or both (depending on the availability) to direct them.
- Propeller driven Drones have two types of propellers onboard for direction and thrust.
- These are
  - Standard Propellers
  - Pusher Propellers

Propellers Contd…

- **Standard Propellers**
  - These are located in the front of Quadcopter. These propellers provide direction to the Drone.
- **Pusher Propellers**
  - These are located in the back of Quadcopter. These propellers provide forward and backward thrust to the Drone.

2. Chassis

- This is the main body of Quadcopter which provides the housing facility to all other components.



3.DC motors

- To ensure that the drone is airborne for a good amount of time, we need high torque motors.
- The high torque also helps to change the speed of the propellers. Brushless DC motors are preferred as they are lighter than the brushed ones.

DC motor – Understanding Kv Rating

- What is Kv Rating of motor

“Kv” refers to the constant velocity of a motor (not to be confused with “kV,” the abbreviation for kilovolt). It is measured by the number of revolutions per minute (rpm) that a motor turns when 1V (one volt) is applied with no load attached to that motor. The Kv rating of a brushless motor is the ratio of the motor’s unloaded rpm to the peak voltage on the wires connected to the coils.

DC motor Contd…

- Knowing the Kv rating of a motor will help you determine how fast that motor will rotate when a given voltage is applied to it. For example, a 980Kv motor powered by an 11.1V battery would spin at 10,878 rpm (980 x 11.1) with no load.
- A change in voltage will change the rpm and will require changing the propeller to avoid overloading the motor. Kv allows you to get a handle on the torque that can be expected from a particular motor. Torque is determined by the number of winds on the armature and the strength of the magnets.
- A low Kv motor has more winds of thinner wire—it will carry more volts at fewer amps, produce higher torque, and swing a bigger prop.
- A high Kv motor has fewer winds of thicker wire that carry more amps at fewer volts and spin a smaller prop at high revolutions.

DC motor contd…

- Knowing the Kv rating of a motor is helpful to determine which motor belongs in which aircraft. An FPV racing quad, for example, requires high rpm for high speed, so you would use a high Kv motor and a small-diameter prop.
- On the other hand, you would use a lower Kv motor in a heavy-lift multirotor because you want to turn a large prop at lower rpm and obtain high torque.

4.Flight Controller

- The flight controller is usually referred to as the brain of the drone. The flight controller controls the power supply to the electronic speed controller. It is also used to detect orientation changes in the drone. It controls the motors and ensures that the drone is in air.

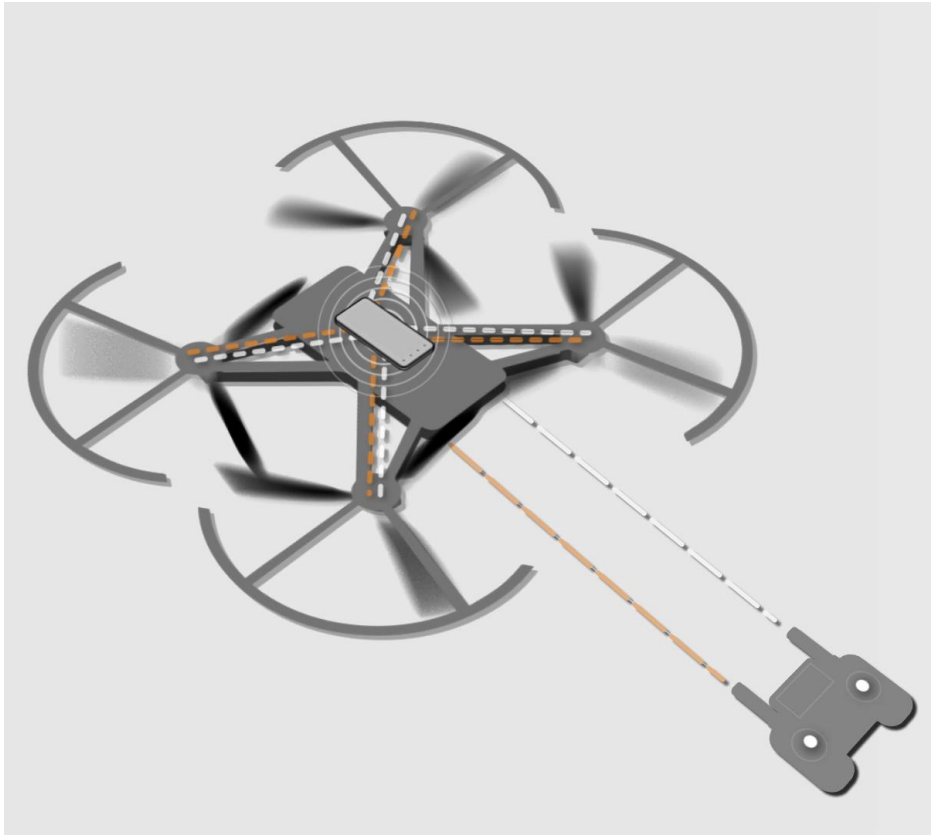
Physically, a flight controller is nothing more than a circuit board with electronic chips on them. You can compare them to the motherboard and processor in your laptop. The flight controller is the brain of a drone. A small box filled with intelligent electronics and software, which monitors and controls everything the drone does. And just like the brains of different organisms, flight controllers also vary in sizes and complexity. (picture of different flight controllers)

What does flight controller does

- everything a flight controller does can be classified within one of three categories: Sensing, controlling, and communicating.

Perception (sensing)

The flight controller is connected to a set of sensors. These sensors give the flight controller information about like its height, orientation, and speed. Common sensors include an Inertial Measurement Unit (IMU) for determining the angular speed and acceleration, a barometer for the height, and distance sensors for detecting obstacles. Just like how we perceive as humans, the drone filters a lot of this information and fuses some to get more efficient and precise information. Advanced flight controllers can sense more precisely and detect differences more quickly.



FC contd…

- **Controlling**
- Aside from sensing what's going on, a flight controller… unsurprisingly controls the motion of the drone. The drone can rotate and accelerate by creating speed differences between each of its four motors. The flight controller uses the data gathered by the sensors to calculate the desired speed for each of the four motors. The flight controller sends this desired speed to the Electronic Speed Controllers (ESC's), which translates this desired speed into a signal that the motors can understand.
- Calculating the movements, fusing and filtering the sensory information, and estimating the safety and durability of a flight is all done by an algorithm. A fancy word that is used a lot nowadays which in essence nothing more than a set of strict rules that every microchip on the board has to apply to. The most commonly used flight control algorithm is called PID control: Proportional Integral Derivative control. Within this area, there is a lot of research going on, which resulted in INDI: Incremental Nonlinear Dynamic Inversion. This algorithm reads out and reacts to incoming information way faster, therefore making the drone flight more stable.

FC Contd…

- **Communicating**
- A key part of a flight controller is communication. A part of the sensor's job is to give out information that needs to be translated clearly for a pilot to read, which means efficiently. An obvious thing to communicate is its battery level, which can decide if a pilot wants to fly further or return to the charge.
- But communication goes further than from flight controller to human pilot; with the entrance of auto-pilot programs in the drone industry, flight controllers need to communicate with other computer systems about its flight destination and how to get there. Communication is mostly done with wi-fi and radio frequencies right now, but cellular solutions are also already in use.

FC Contd…

- **What kind of flight controllers are there?**

There are a lot of different flight controllers on the market. They range from very basic to expensive systems. To make it a bit more comprehensible, I made four categories based on their users.

•**FC's for hobbyists/builders –** Easy to install and perfect for people that do not want to spend large amounts of money from the get-go.

•**Racing FC's –** Designed to be very lightweight, precise, and responsive. Most of them are in the €50,- or less range.

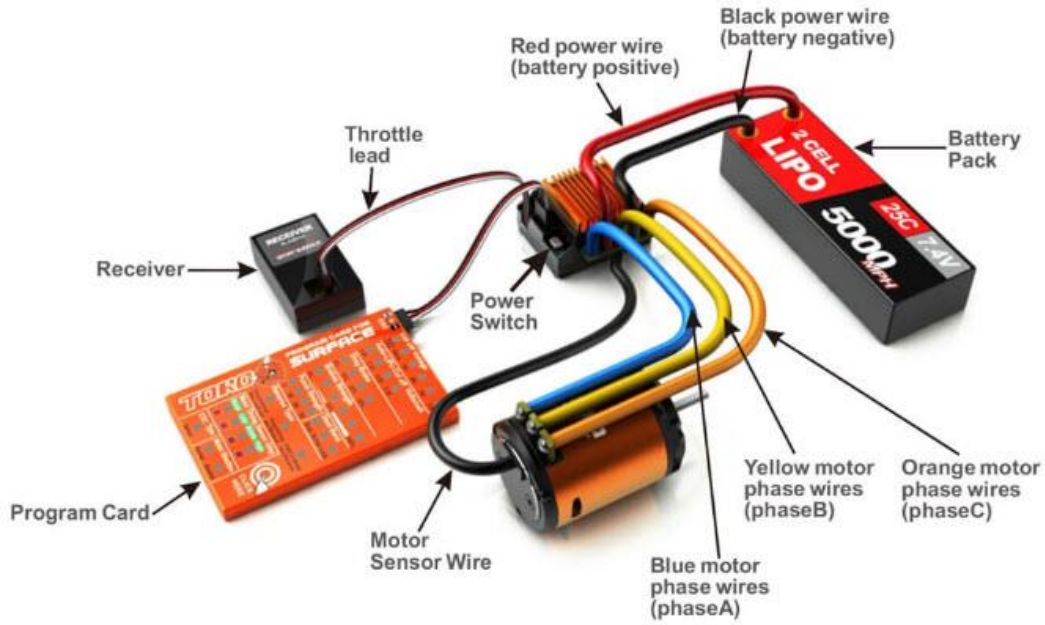
•**FC's for filming** – Although mostly bought included in a drone with a camera, these flight controllers are more focussed on creating fluent shots and accessible handling for a pilot. Within this segment, Chinese company Dà-Jiāng Innovations (better known as DJI) is a household name.

•**Commercial FC's** – The latest segment to evolve in the previous years. These are for the most advanced drones, capable of safe flying and transporting high-value cargo. The biggest players in this field are DJI and Pixhawk, but new flight controllers like Auterion's Skynode and Fusion Engineering's Fusion Reflex are also promising flight controllers in the industry.

5.Electronic Speed Controller

- Electronic speed controllers (ESC) are the electronic circuits that regulate the speed of the DC motors. It also provides dynamic braking and reversing options.

The term ESC stands for “electronic speed control [is an electronic circuit](#) used to change the speed of an electric motor, its route, and also to perform as a dynamic brake. These are frequently used on radio-controlled models which are electrically powered, with the change most frequently used for brushless motors providing an **electronically produced 3-phase electric power** low voltage source of energy for the motor. An ESC can be a separate unit that lumps into the throttle receiver control channel or united into the receiver itself, as is the situation in most toy-grade R/C vehicles. Some R/C producers that connect exclusive hobbyist electronics in their entry-level vehicles, containers, or aircraft use involved electronics that combine the two on a sole circuit board.



6. Landing Gear

- Landing gears are not required for small drones. However, bigger drones need a landing gear to avoid any damage while landing. The requirement of landing gear varies with functionality of the drone. For example – Delivery drones which carry parcels require a spacious landing gear as they need space to hold the items.

7. Transmitter

- The transmitter send signals from controller to the drone to generate command of direction and thrust.

8.Receiver

- The receiver receives the signals sent by the transmitter and passes it to Flight Controller PCB.

9.GPS Module

- The GPS module provides the navigational data (longitude, latitude and elevation) to the Controller. This module assists the controller in recognizing the taken path and safely return to the initial point in case of lost connection.

10.Battery

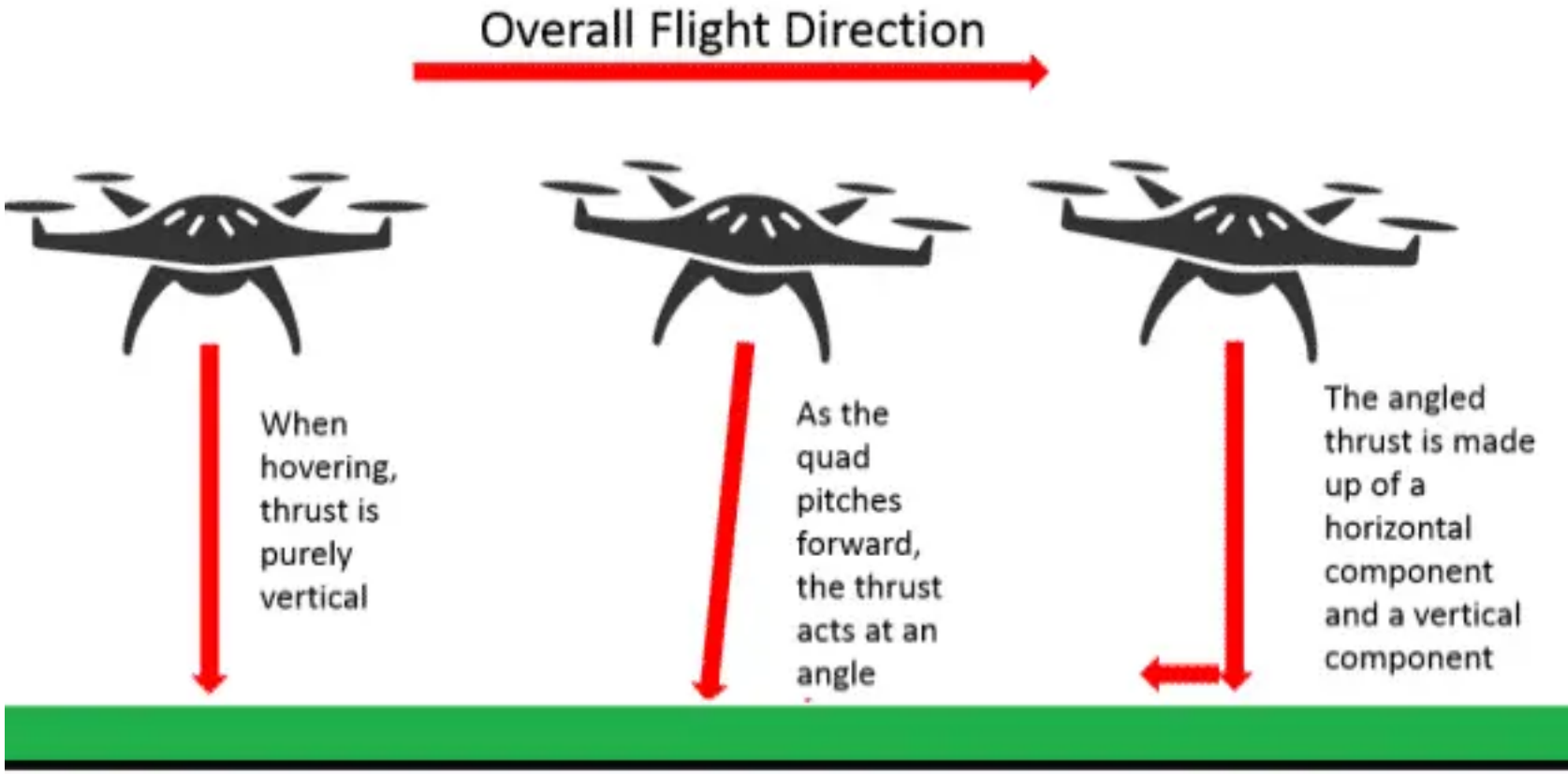
- It provides power to the drone. Generally, rechargeable battery is used in drone.

11.Camera

- There is normally an attached inbuilt camera with drones. If the drone is not provided with inbuilt camera, then it will have a provision of detachable camera.

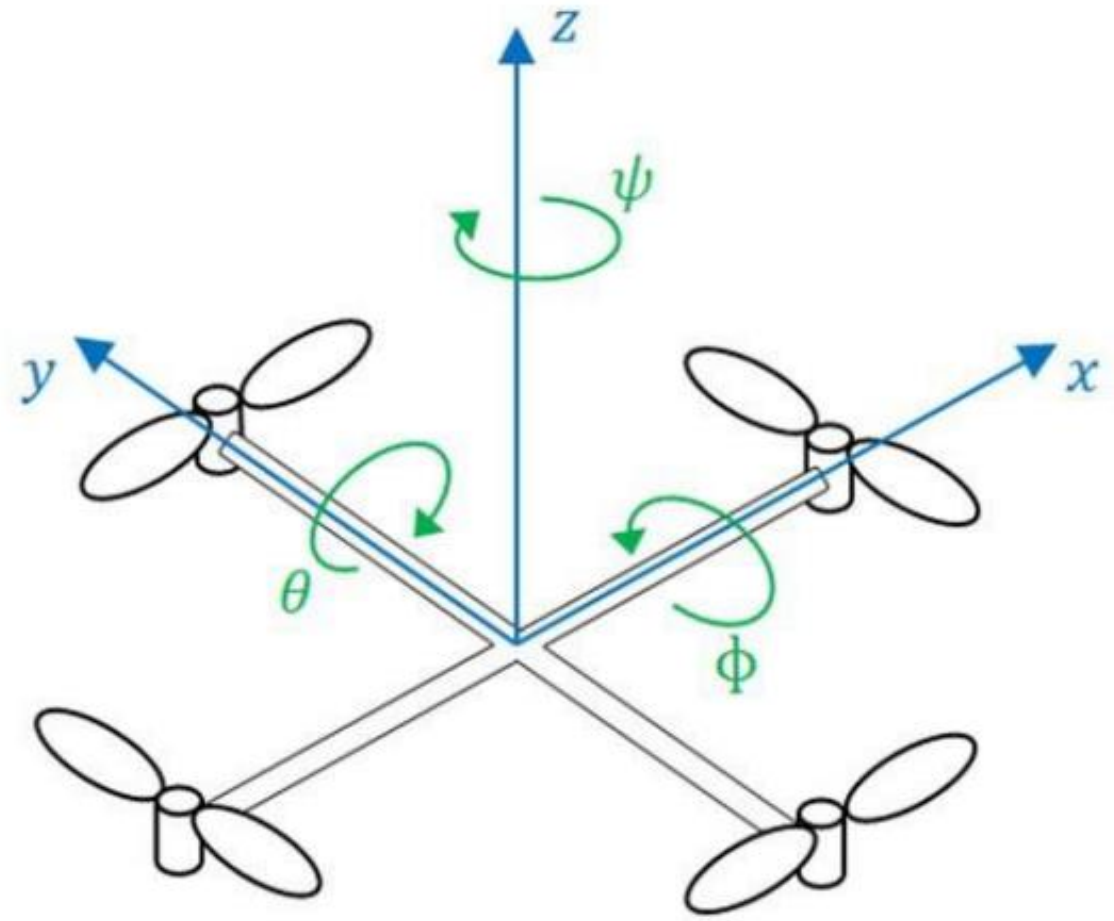
How Drones/UAVs Work

- When the controller provide the lift off instruction to the receiver of drone, it forwards it to the Flight Controller Board, which in turn actuates the propeller. The propellers provide the lift off thrust and the drone becomes airborne. It is these propellers, which provide the direction as well as speed to the drone.



The Drones have a complex multi propeller system that assists in reduction of failures. The propellers work in group and even if any one of them fails, the drone is capable to get support form its available propellers and navigate/ land safely. The GPS module helps the drone to identify its travelled path and get real time data such as location, speed and elevation.

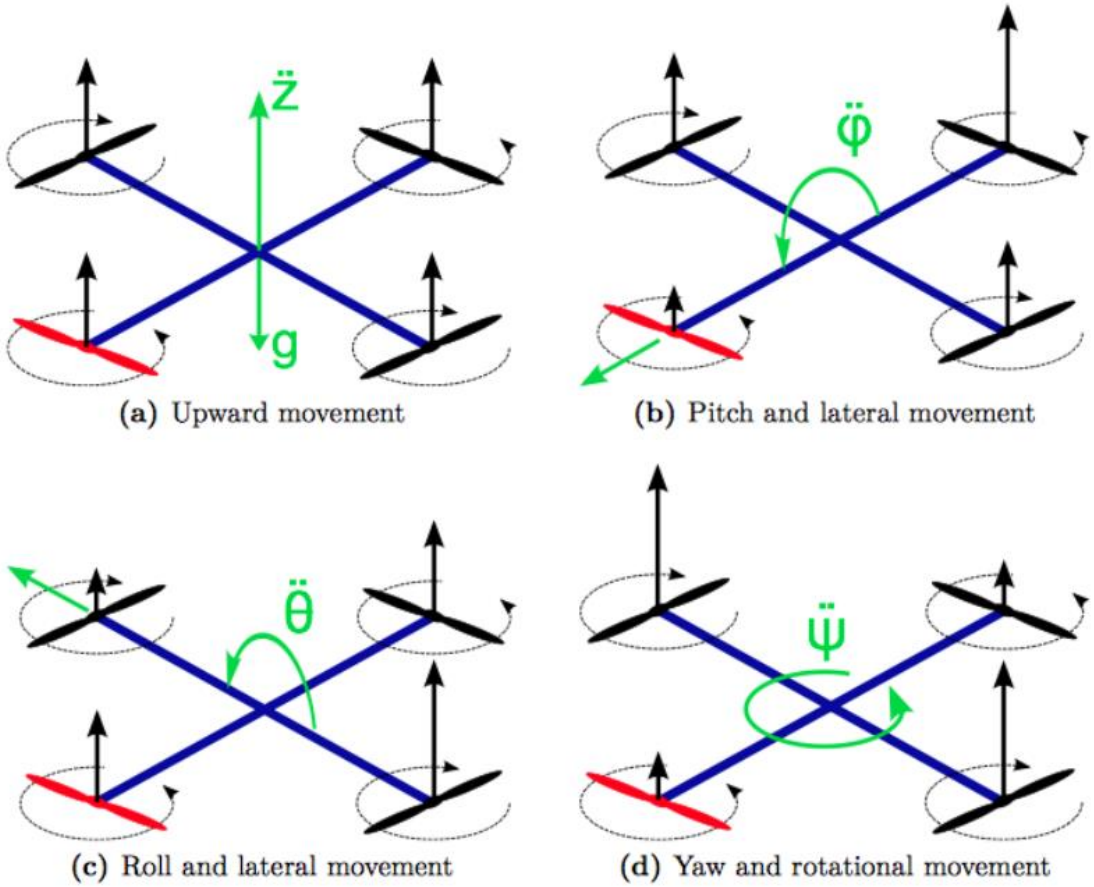
For a multirotor aircraft in motion, it is appropriate to classify the three rotation axes that can alter the craft's attitude as 1) Roll: Rotation of x-axis, 2) Pitch: Rotation of y-axis, 3) Yaw, rotation of z-axis. These axes of rotation are shown in Figure



The coordinate system of a multirotor aircraft.

Movements of Multi Rotors

- A common multirotor propeller configuration usually consists of four propellers or better known as a quadcopter. Other known multirotor configurations include tricopter, hexacopter and octocopter.
- Different configurations are for the different area of application and have performance variation in terms of
  - flight time, agility, stability and payload capacity.
- The propeller's main task is to generate enough lift force to bring the aircraft up to the sky and also deliver adequate thrust to move its current position to forward, backward, left and right.
- The attitude of multirotors is actuated by the difference in rotation speed of each propeller.
- Figure shows how the aircraft's attitude change and the corresponding body movement when the rotation speed of certain propellers is manipulated.
- There is four basic movements for multirotors



(a) Upward movement (b) Pitch and lateral movement

(c) Roll and lateral movement (d) Yaw and rotational movement

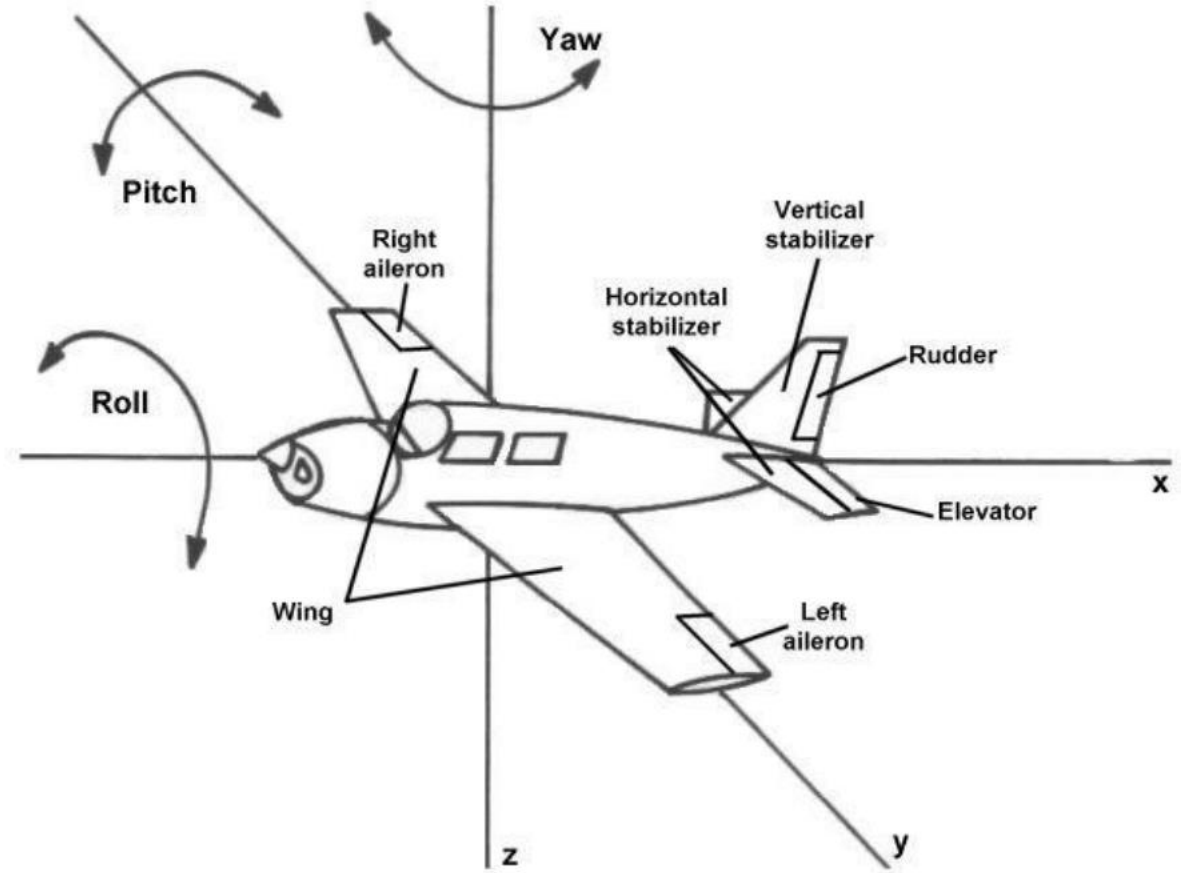
Attitude and Body Movement Change due to rotation speed change in the individual propeller

Movements of Multi Rotors contd…

- The multirotor must be able to achieve a reasonably stable flight during hover before attempting to manoeuvre it in its pitch, roll and yaw axes.
- Stable flight requires that the pitch and roll axes must be first stabilized.
- The multirotor will be at risk of tipping over and crash if these axes are not controlled adequately.
- The pitch and roll attitudes are determined using the gyroscope and accelerometer which is built onboard the Flight Controller Unit (FCU).
- As for the yaw axis, it is less critical to stabilizing the axis provided that the multirotor is still controllable.
- Unstable yaw axis would slightly drift the multirotor body in that axis but can be balanced by using the radio controller.

Fixed Wing movements

- For a fixed-wing aeroplane in motion, the coordinate system is similar to that of the multirotor.
- A clear difference is that aeroplanes use propellants that are aligned to its fuselage for forwarding propulsion as depicted in Figure.
- Lift in this case is generated by the aerodynamic pressure difference between the upper and lower surface of the wing.
- The attitude of the aircraft is controlled using the control surfaces attached to the main wing, horizontal tail wing and vertical tail wing.
- The three control surfaces are called 1) Elevator: Control the pitch axis, 2) Aileron: Control the roll axis, 3) Rudder: Control the yaw axis.



Coordinate System of a fixed-wing aircraft

Similar to multirotors, to fly fairly stable while in straight-and-level flight, the roll and pitch axes must be stabilized properly. In full-scale manned aircraft, this can be done by directly trimming the ailerons and elevators. In the case of small-scale unmanned fixed-wing aircraft, the roll and pitch axes can be stabilized by using the attitude sensors such as the accelerometer and gyroscope.

Forces acting on Airplane…. How things fly?

- These same four forces help an airplane fly. The four forces are **lift, thrust, drag, and weight**.

Have you ever thrown a Frisbee®? It flies because of four forces. These same four forces help an airplane fly. The four forces are lift, thrust, drag, and weight. As a Frisbee flies through the air, lift holds it up. You gave the Frisbee thrust with your arm. Drag from the air made the Frisbee slow down. Its weight brings the Frisbee back to Earth again.

You see them everyday: airplanes, jets, and helicopters, soaring, zooming, and even roaring through the skies. We may take flight for granted; yet, knowing the science behind it gives us a better understanding of the marvels of air travel.

Forces acting Airplane contd…

- Wings keep an airplane up in the air, but the four forces are what make this happen. They push a plane up, down, forward, or slow it down.

•Thrust is a force that moves an aircraft in the direction of the motion. It is created with a propeller, jet engine, or rocket. Air is pulled in and then pushed out in an opposite direction. One example is a household fan.

• Drag is the force that acts opposite to the direction of motion. It tends to slow an object. Drag is caused by friction and differences in air pressure. An example is putting your hand out of a moving car window and feeling it pull back.

•Weight is the force caused by gravity.

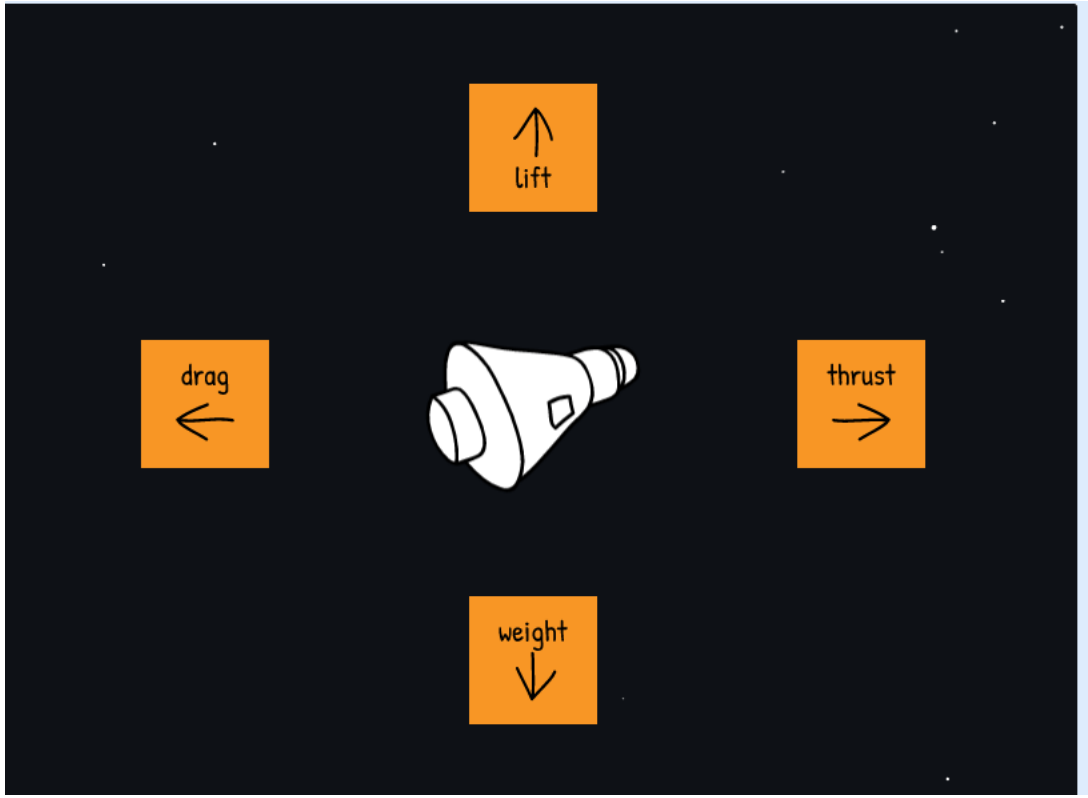
•Lift is the force that holds an airplane in the air. The wings create most of the lift used by airplanes.

Forces acting on Airplane…

- The way the four forces act on the airplane make the plane do different things. Each force has an opposite force that works against it. Lift works opposite of weight.
- Thrust works opposite of drag. When the forces are balanced, a plane flies in a level direction. The plane goes up if the forces of lift and thrust are more than gravity and drag.
- If gravity and drag are bigger than lift and thrust, the plane goes down.
- Just as drag holds something back as a response to wind flow, lift pushes something up. The air pressure is higher on the bottom side of a wing, so it is pushed upward.

Forces contd…

- **ALL FOUR FORCES ACT ON AN AIRPLANE**
- When an airplane is flying straight and level at a constant speed, the lift it produces balances its weight, and the thrust it produces balances its drag. However, this balance of forces changes as the airplane rises and descends, as it speeds up and slows down, and as it turns.



In summary

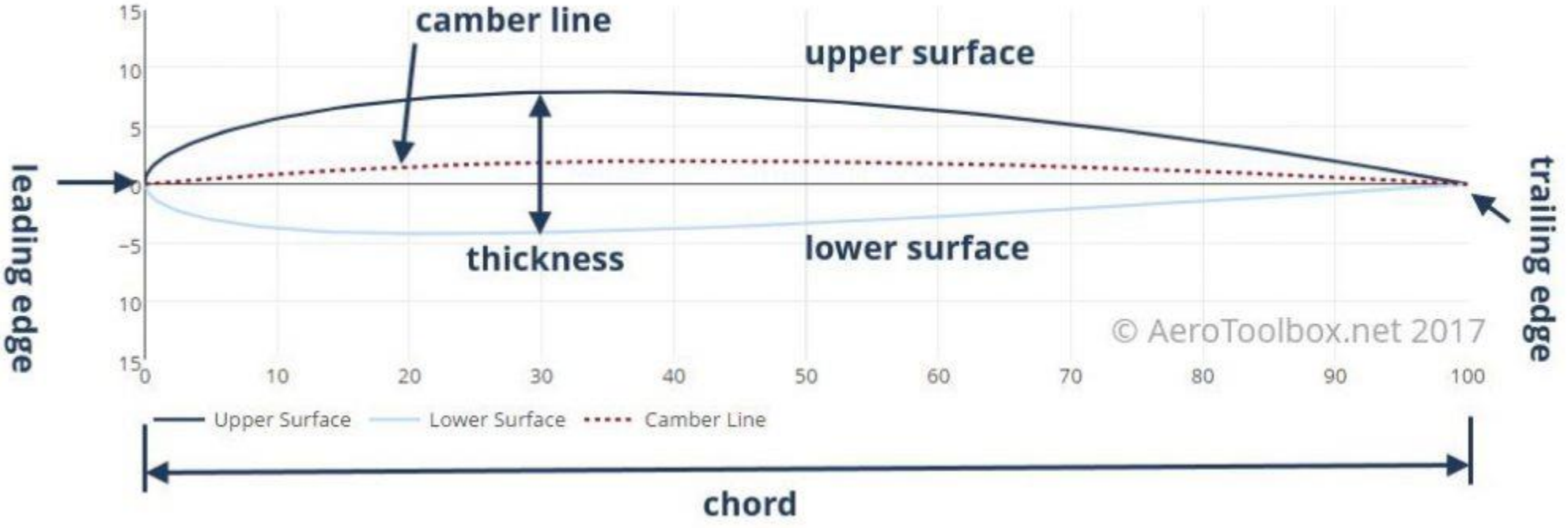
1. **Weight** is the force of gravity. It acts in a downward direction—toward the center of the Earth.
2. **Lift** is the force that acts at a right angle to the direction of motion through the air. Lift is created by differences in air pressure.
3. **Thrust** is the force that propels a flying machine in the direction of motion. Engines produce thrust.
4. **Drag** is the force that acts opposite to the direction of motion. Drag is caused by friction and differences in air pressure.

Physical properties and structure of the atmosphere

Aerodynamics – aerofoil nomenclature, aerofoil characteristics

Aerofoil nomenclature

- he image below of a cross-section through a typical wing, shows a number of fundamental definitions associated with airfoil (aerofoil) nomenclature.

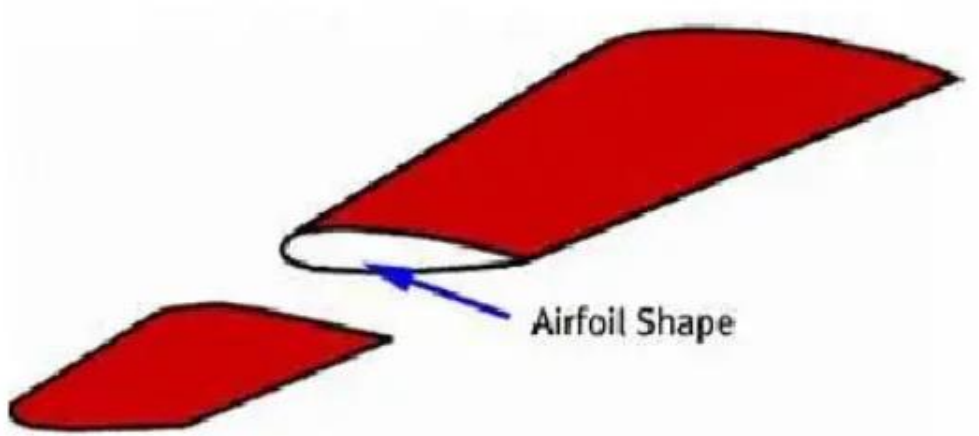


Aerofoil nomenclature….

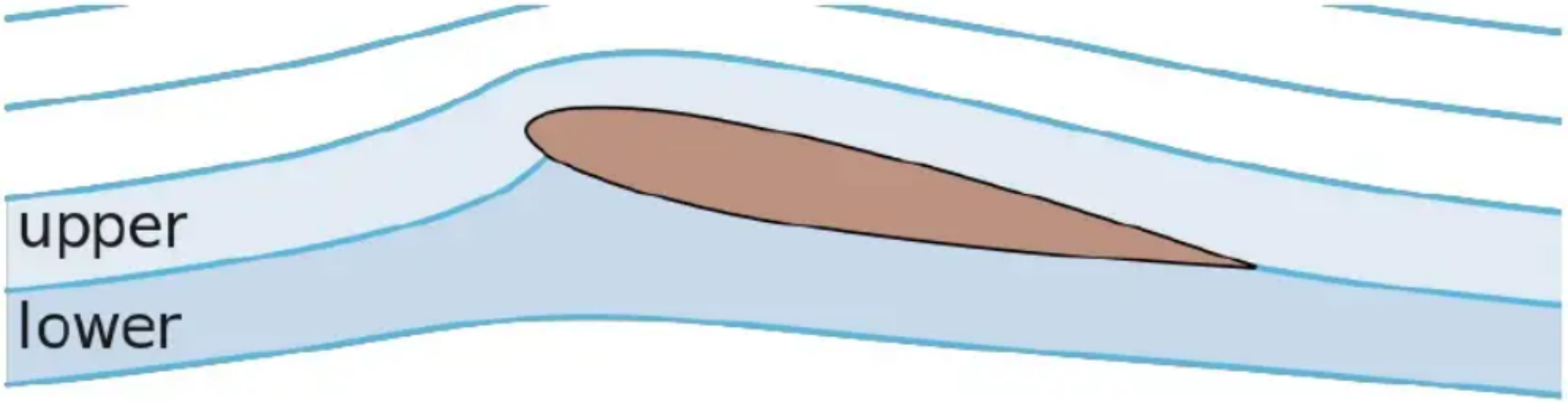
- The forward section of the airfoil is named the **leading edge** and the rear the **trailing edge**. The airfoil upper and lower surfaces meet at the leading and trailing edges.
- The length of the airfoil from leading to trailing edge is known as the **airfoil chord**. This often varies down the span of the wing as the wing tapers from the root to the tip.
- The thickness of the airfoil is a very important design parameter and as always expressed as a percentage of the total chord. The airfoil plotted above has a **thickness-to-chord ratio** of 12%. This means that the thickest section has a height equal to 12% of the total chord.

INTRODUCTION

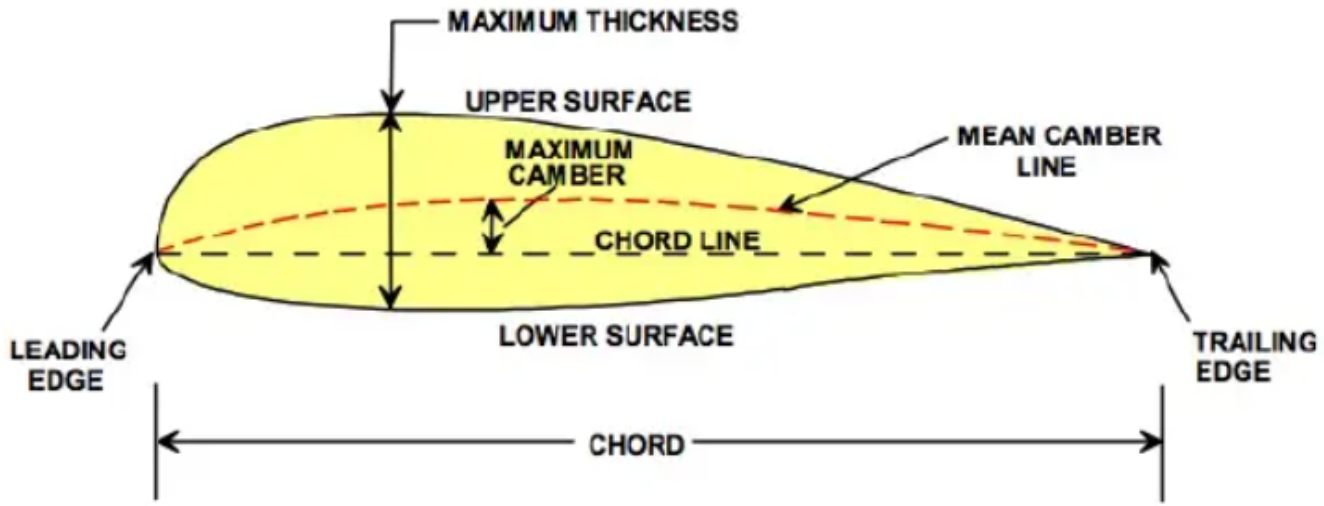
- An **aerofoil** is the cross-sectional shape of a wing
- An aerofoil-shaped body moved through a fluid produces an aerodynamic force.



TERMINOLOGY



- The suction surface (a.k.a. upper surface) is generally associated with higher velocity and lower static pressure.
- The pressure surface (a.k.a. lower surface) has a comparatively higher static pressure than the suction surface.

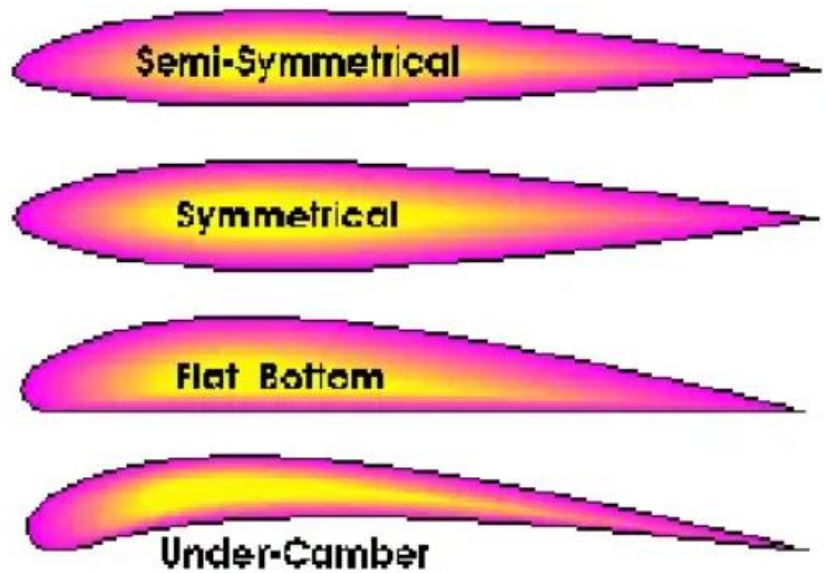


- Mean camber line : The locus of points halfway between the upper and lower surfaces as measured perpendicular to the mean camber line.
- Leading & trailing edges: The most forward and rearward points of the mean camber line.
- Chord line: The straight line connecting the leading and trailing edges

- Chord C : The distance from the leading to trailing edge measured along the chord line.
- Camber : The maximum distance between the mean camber line and the chord line.
- Leading edge radius and its shape through the leading
- The thickness distribution: The distance from the upper surface to the lower surface, measured perpendicular to chord line

TYPES OF AEROFOILS

- The symmetrical aerofoil is distinguished by having identical upper and lower surfaces. The mean camber line and chord line are the same on a symmetrical aerofoil
- The non symmetrical aerofoil has different upper and lower surfaces, with a greater curvature of the aerofoil above the chord line than below. The mean camber line and chord line are different.

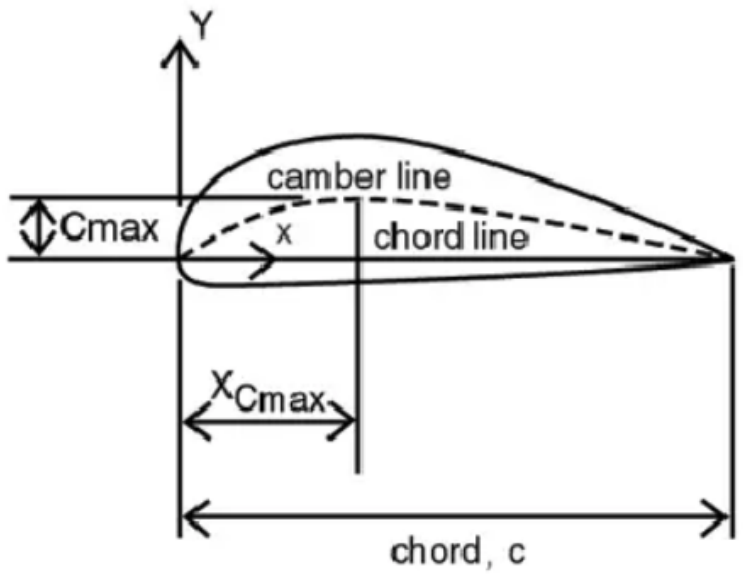


SERIES

- NACA (National Advisory Committee for Aeronautics) or NASA (National Aeronautics and Space administration) identified different airfoil shapes with a logical numbering system.
- NACA Airfoil Series
    1- NACA 4-digit series
    2- NACA 5-digit series
    3- NACA 1-series or 16-series
    4- NACA 6- series
    5- NACA 7- series
    6- NACA 8- series

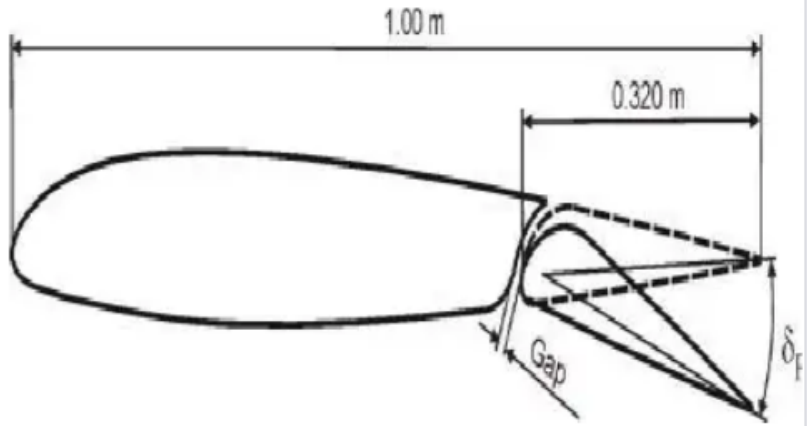
NACA 4 DIGIT SERIES

- The NACA four-digit wing sections define the profile by
- First digit describing maximum camber as percentage of the chord.
- Second digit describing the distance of maximum camber from the aerofoil leading edge in tens of percent of the chord.
- Last two digits describing maximum thickness of the aerofoil as percent of the chord.
- Example: NACA 2412



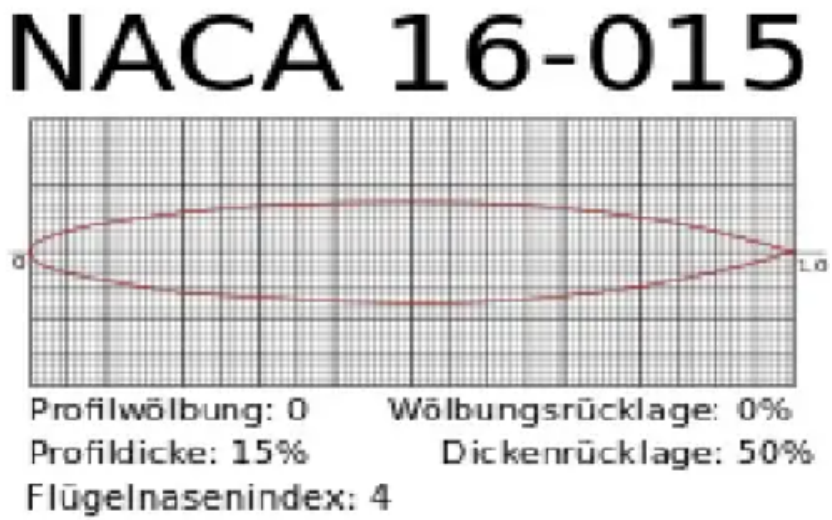
NACA 5 DIGIT SERIES

- The NACA five-digit series describes more complex aerofoil shapes.
- The first digit when multiplied by 0.15, gives the designed coefficient of lift
- The second and third digit when multiplied by 2 give p, the distance of maximum camber from leading edge as percent chord
- Fourth and fifth gives the maximum thickness of aerofoil as percent of chord
- Example: NACA 23012



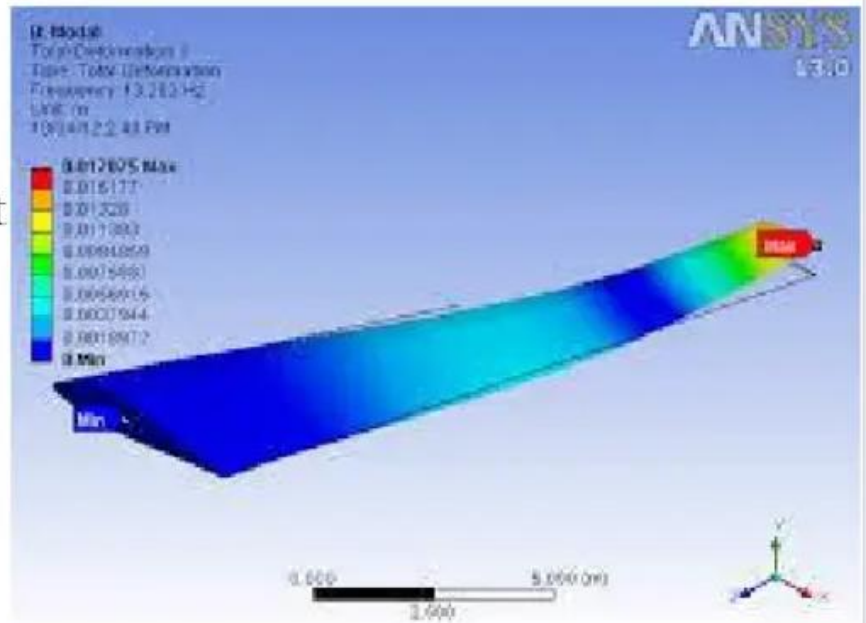
NACA 1 SERIES

- The 1-series aerofoils are described by five digits in the following sequence:
- The number "1" indicating the series
- One digit describing the distance of the minimum pressure area in tens of percent of chord.
- A hyphen.
- One digit describing the lift coefficient in tenths.
- Two digits describing the maximum thickness in percent of chord.
- For example, the NACA 16-015



NACA 6 SERIES

- The aerofoil is described using six digits in the following sequence:
- The number "6" indicating the series.
- One digit describing the distance of the minimum pressure area in tens of percent of chord.
- The subscript digit gives the range of lift coefficient in tenths above and below the design lift coefficient in which favourable pressure gradients exist on both surfaces
- A hyphen.
- One digit describing the design lift coefficient in tenths.
- Two digits describing the maximum thickness as percent of chord.
- For example, the NACA 61_2-315



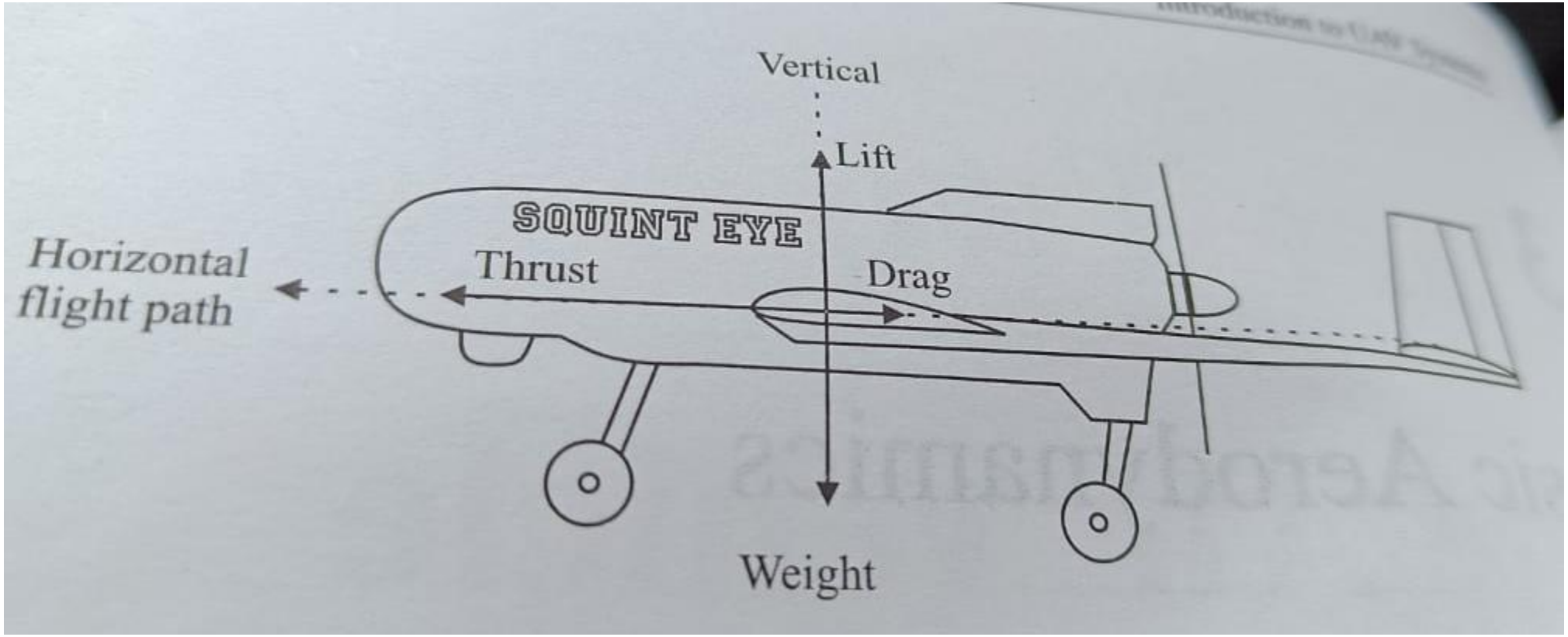
NACA 7 & 8 SERIES

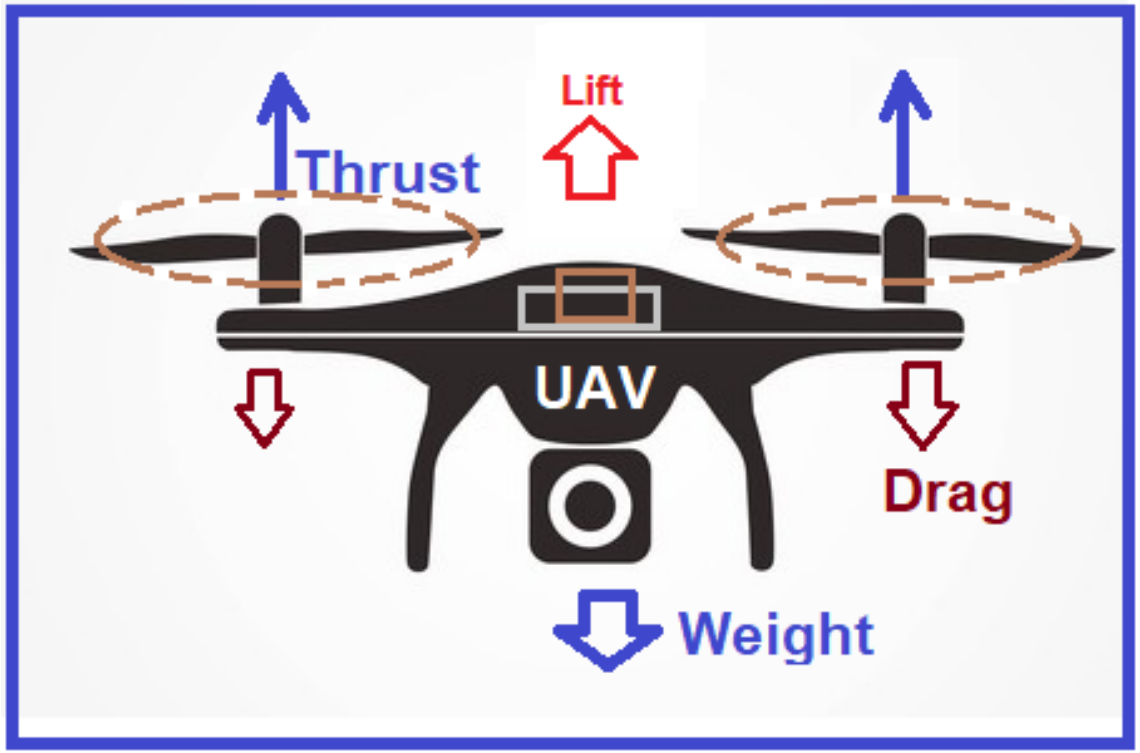
- **NACA 7 SERIES:** The aerofoil is described by seven digits in the following sequence:
- The number "7" indicating the series.
- One digit describing the distance of the minimum pressure area on the upper surface in tens of percent of chord.
- One digit describing the distance of the minimum pressure area on the lower surface in tens of percent of chord.
- One letter referring to a standard profile from the earlier NACA series.
- One digit describing the lift coefficient in tenths.
- Two digits describing the maximum thickness as percent of chord.
- "a=" followed by a decimal number describing the fraction of chord over which laminar flow is maintained. a=1 is the default if no value is given.
- For example, the NACA 712A315

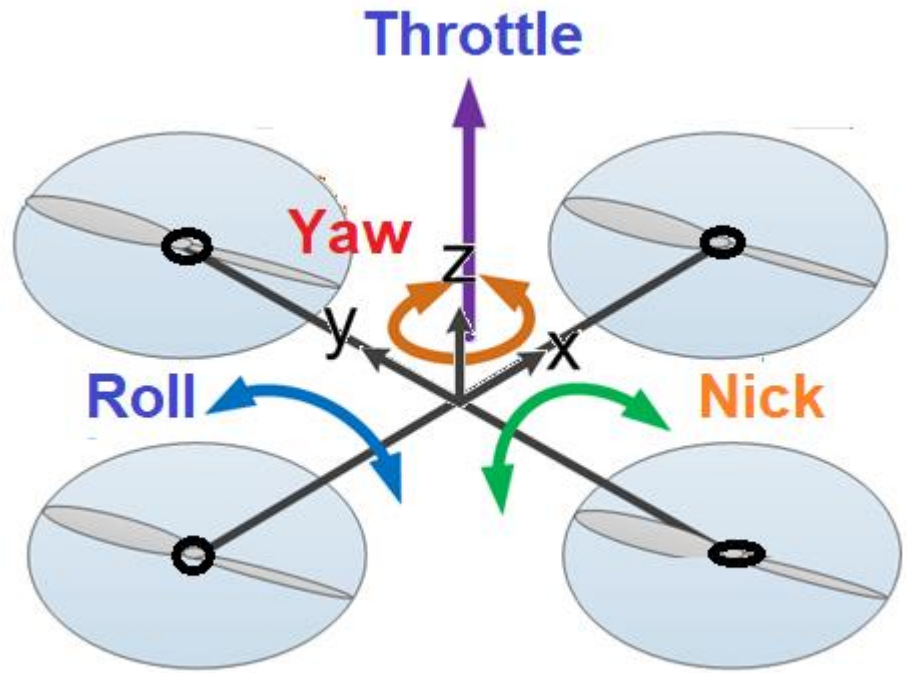
- **NACA 8 SERIES:** The numbering is identical to the 7-series aerofoils except that the sequence begins with an "8" to identify the series.

Basic Aerodynamics

- The primary forces that act on vehicles are thrust, lift, drag and gravity (or weight).




Lift
Thrust
UAV
Drag
Weight


Throttle
Yaw
z
y
x
Roll
Nick

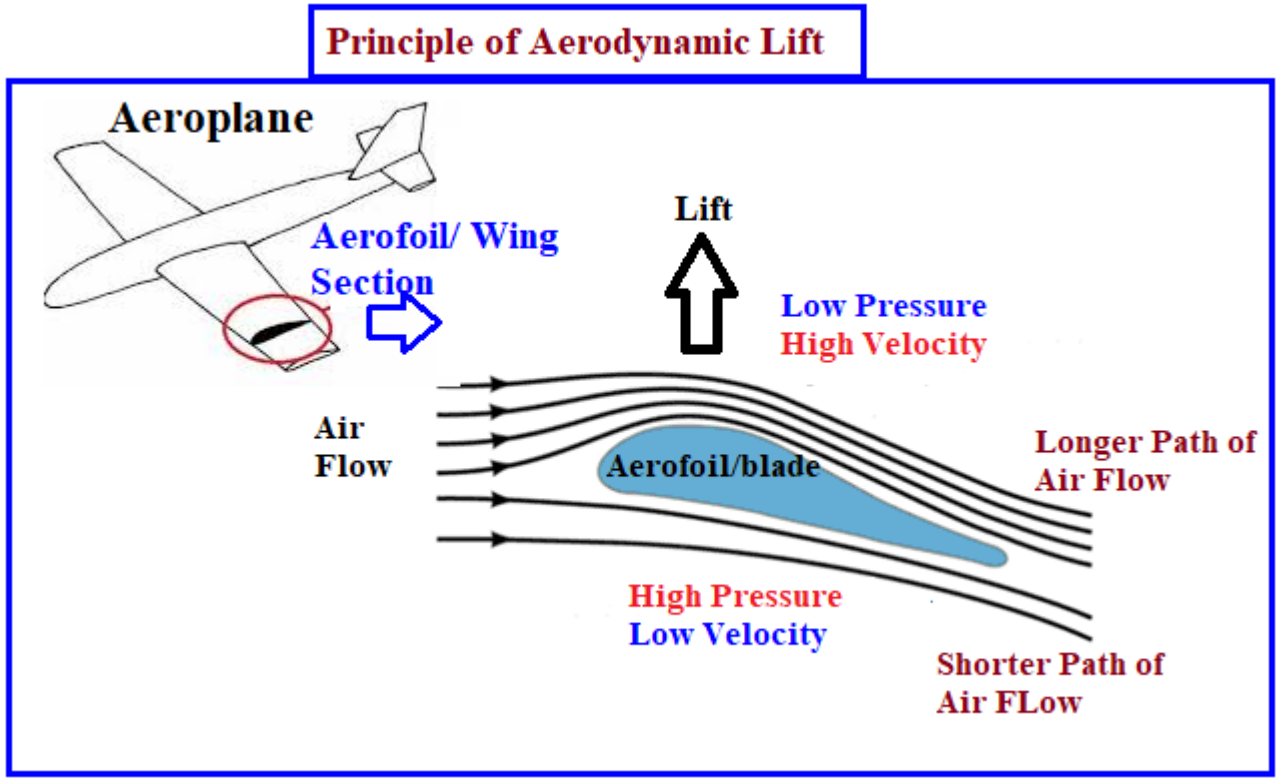
Understanding and development of drones depend on many subjects. The design of drone for a particular application comprises many factors like the aerodynamic shape of propellors, strength and weight of drone parts, electric motor, electric speed controller, radio transmitter or receiver, and software interface on mobile or computer for monitoring and data analysis.

- **Fluid Dynamics or Aerodynamics:**
  - Fluid dynamics plays an important role to decide the forces acting on the body of a drone
  - The shape, size, and speed of the propeller and drone depending on the aerodynamics of propellers or blades
  - Computational Fluid Dynamics (CFD) modeling helps for flow dynamics of airflow over drones
  - CFD modeling of turbo-machinery) is essential to decide the amount of thrust generated by propellors
  - Wind tunnel testing of the aerofoil blade of the drone is still important for testing CFD results

- **Mechanical Design**
  - Rigid body dynamics to study the motion and forces acting on drones
  - Strength of materials
  - Low weight and rigid materials are selected for drone
- **Electronics and Electrical Components:**
  - Electric motor with and without brush is required to drive the propellors
  - Electronic Speed Controller
  - Flight controller unit and computer processors
- **Radio Communication:** transmitter and receiver for radio signals
- **Battery:** Low weight and high-power wattage battery is important
- **Software-based interface:** data collection and analysis using mobile or computer

Working principles of Drone and Flow Pattern

•The subject of Fluid dynamics plays a significant role in the design and development of aircraft and drones. This subject consists of the working principle of the aerodynamics of aircraft.
•A sufficient amount of upward force is required to lift the vehicle against gravity which is named Lift.
•A force created to move the vehicle or body in motion is called thrust. These forces can be studied using the kinematic laws of fluid flows



•When air flows over an aerofoil and pressure, viscous and drag force act on the profiles
•Force is directly proportional to the velocity of air at the inlet

Force:
related to change in fluid momentum (m V) with time (t).

F = Force
V = Velocity
ρ = density
m = mass
t = time
A = Area

$$F = \text{Constant } (mV)/t$$

Re-write equation: $F = \text{Constant } (m/t)\ V$

Use definition of mass flow rate (m/t): $m/t = \rho V A$

$$F = \text{Constant } (\rho V A)\ V$$

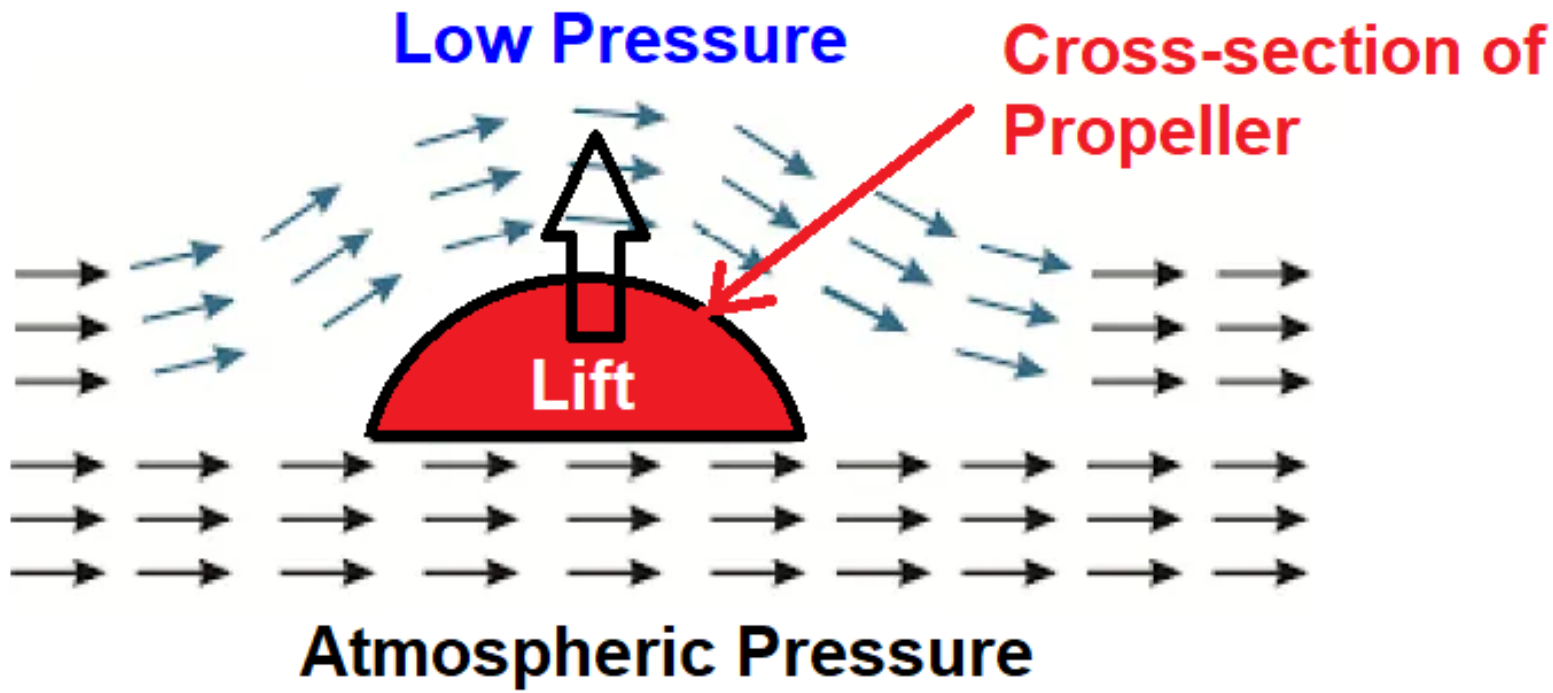
Change the constant to include A: $F = \text{Constant } \rho\ V\ V$

$$F = \text{Constant} \times \rho \times V^2$$

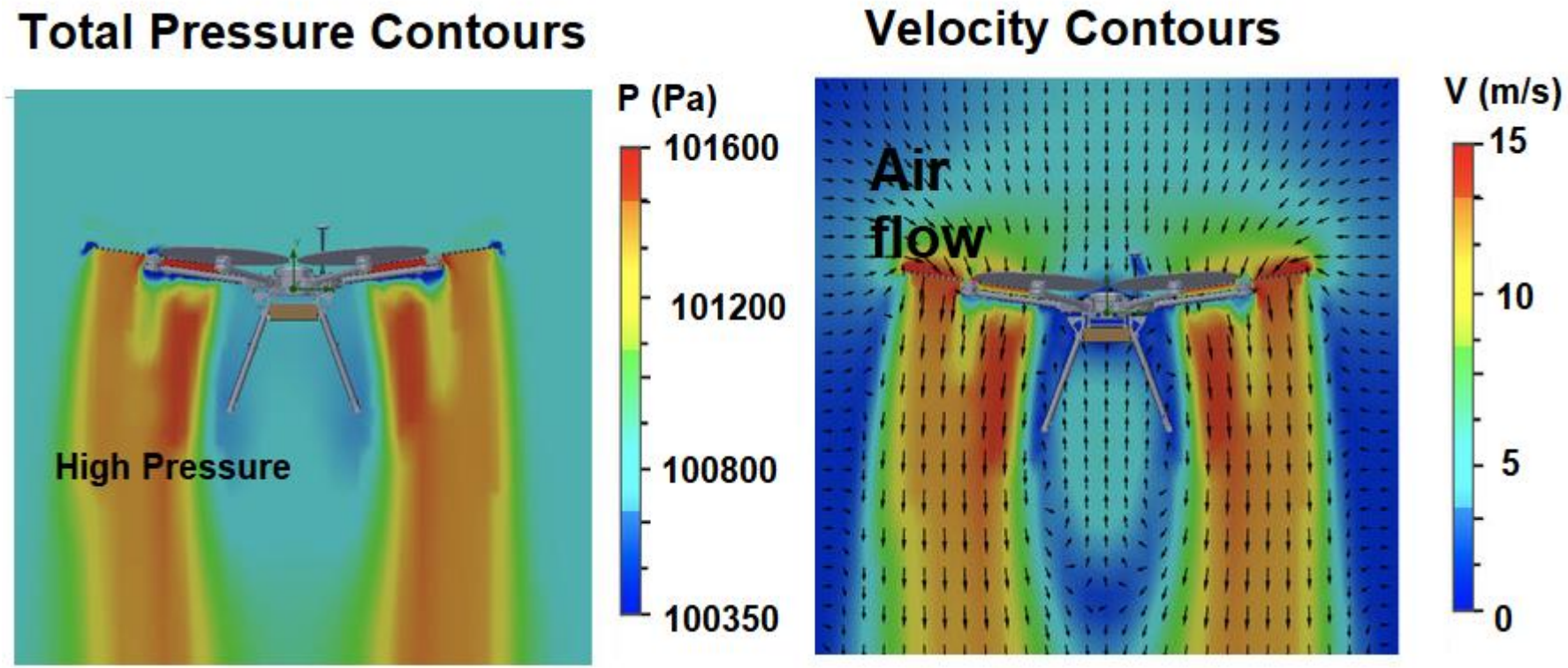
Double the Velocity --> Quadruple the Force

Half the Density --> Half the Force

•The flow pattern around the cross-section of the aerofoil or propeller is shown below. High fluid pressure at the bottom and low pressure at the top of the propeller causes an upward force which is called a lift. This force is responsible for lifting the weight of an aero-plane or drone.
•The amount of lift force depends on the angle of inclination of the aerofoil or propeller.

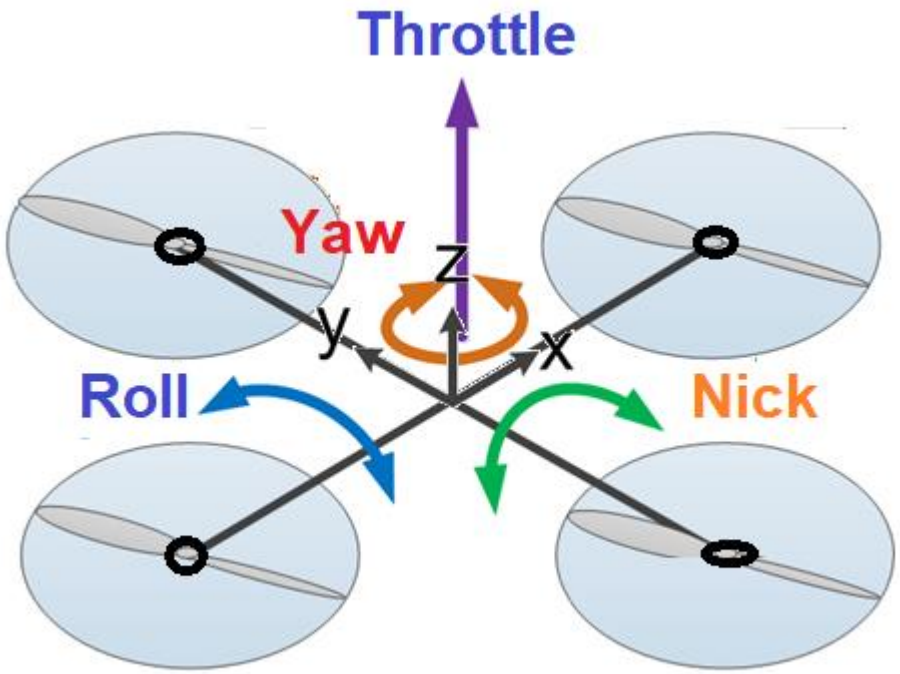


•Based on the **principle** of conservation of energy in fluid flow (**Bernoulli's principle**, the sum of all forms of energy in a fluid is constant along the **streamline**
•When air flows over an aerofoil or wing, its velocity increases at the top portion. But the pressure of air decreases.
•In contrast, the air velocity decreases and pressure increase at the bottom side of the blade. The next pressure difference across the aerofoil results in an upward force which is called a lift
•CFD modeling of flow over an aerofoil has been important in many vehicular and aerospace industries



Quadcopter Dynamics

•The movement of drone are classified into four types based on the relation motion between four propellors: 1) throttle, 2) Pitch, 3) Roll, and 4) Yaw
•The details of quadcopter dynamics are explained in many references



•**Throttle/ Hover:** up and down movement of the drone is called throttle

- If all four propellors run at normal speed, then the drone will move down
- If all four propellors run at a higher speed, then the drone will move up. This is called the hovering of a drone

•**Pitch:** movement of a drone about a lateral axis (either forward or backward) is called pitching motion

- If two rear propellors run at high speed, then the drone will move in a forwarding direction
- If two front propellors run at high speed, then the drone will move in the backward direction

•**Roll**: movement of a drone about the longitudinal axis is called rolling motion

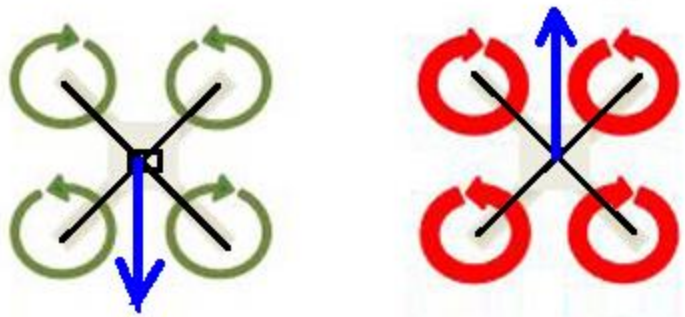
- If two right propellors run at high speed, then the drone will move in the left direction
- If two left propellors run at high speed, then the drone will move in the right direction

•**Yaw:** the rotation of the head of the drone about the vertical axis (either the left or right) is called Yawning motion

- If two propellors of a right diagonal run at high speed, then the drone will rotate in an anti-clockwise direction
- If two propellors of a left diagonal run at high speed, then the drone will rotate in a clockwise direction

How to Fly a Drone: Controls of Quadcopter

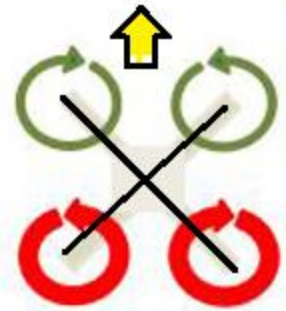
Throttle



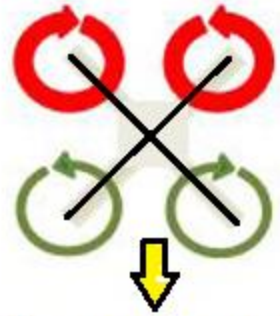
Move down

Move up

Pitch

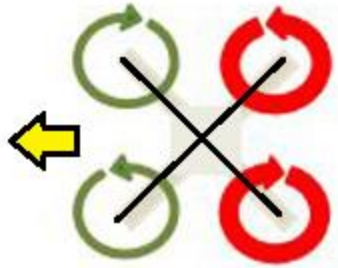


Move forward

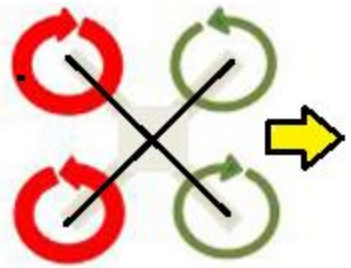


Move backward

Roll

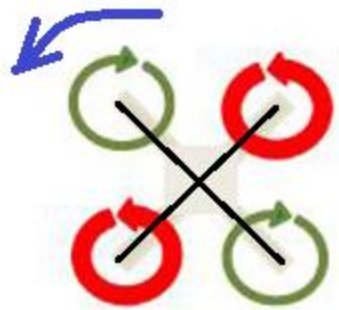


Bend left

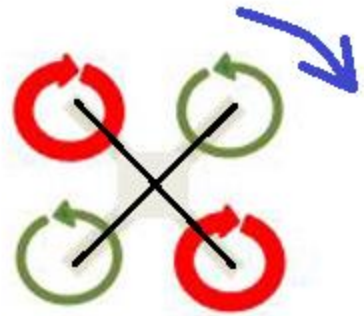


Bend Right

Yaw



Rotate left



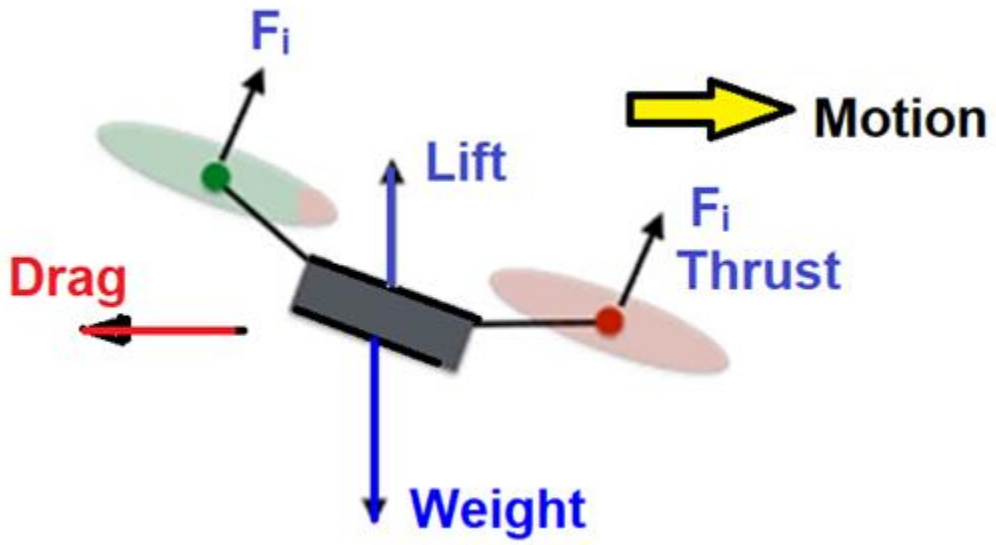
Rotate right

Normal speed High Speed

Forces and Moments Acting on a Drone

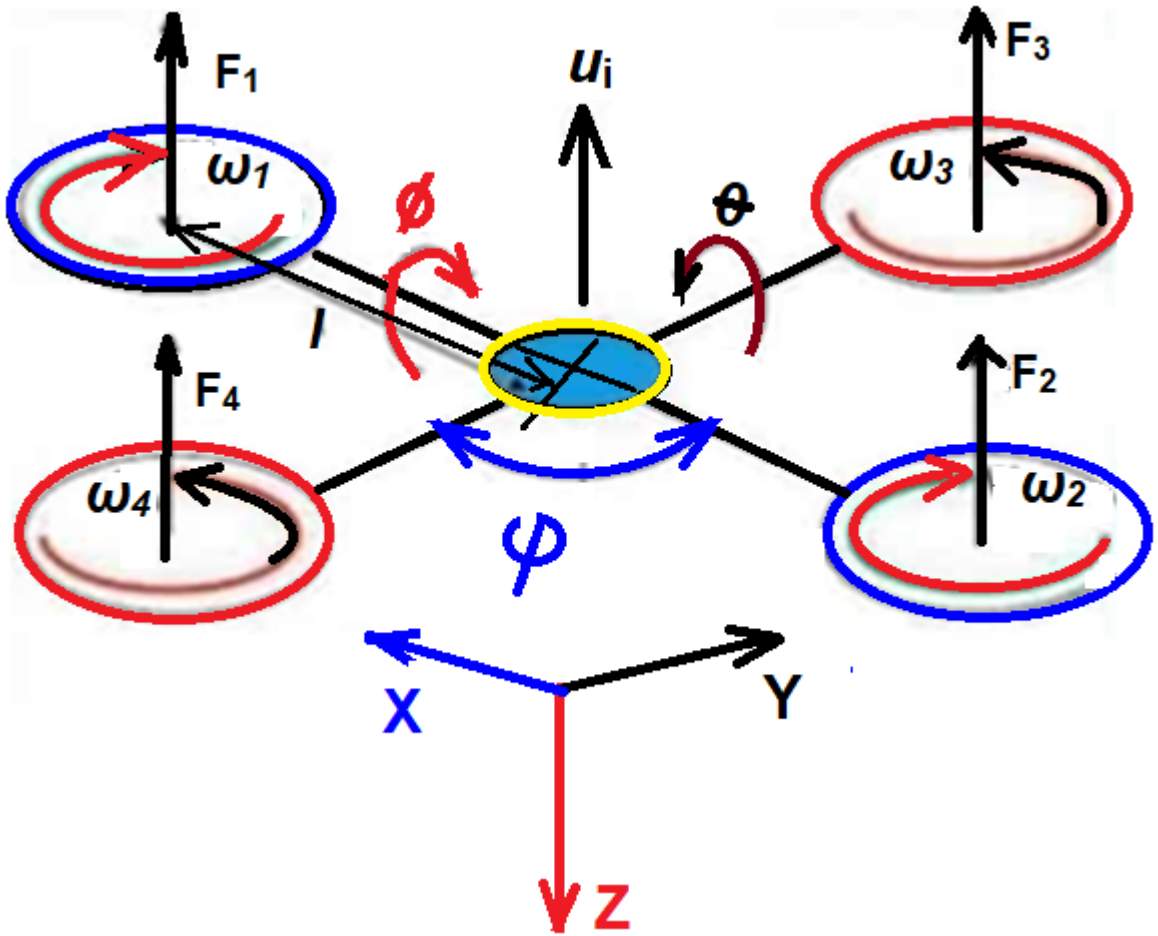
Major forces acting on a Drone

When a drone moves in the air, various forces act on it. The resultant force will decide its movement. There are major forces acting on a drone



- **Weight**
  - Due to the mass of the drone, the body mass force always acts in the direction of gravity
  - Higher the weight of the drone, more power is required to lift and move the drone
  - Weight of drone = mass of drone × acceleration due to gravity
- **Lift**:
  - The vertical force acting on the drone is called lift
  - This force is due to pressure differences across the drone (in the vertical direction). Hence, the speed, size, and shape of the propeller blade decide the amount of lift force
  - Lift is essential to lift the body against the gravity
  - To create this force, all four propellors run at high speed to lift the drone
- **Thrust**
  - The force acting on the drone in the direction of motion is called thrust. However, for drone dynamics, it is normal to the rotor plane.
  - During hovering, the thrust is purely vertical. If thrust is inclined then the drone will tilt forward or backward.
  - This force is essential to move the drone in the desired direction at equal speed
  - To get desired motion, two propellors have been given high speed
- **Drag**
  - The force acting on the drone in the opposite direction of motion due to air resistance is called drag
  - This may be because of pressure difference and viscosity of air
  - To reduce the drag, the aerodynamic shape of the drone is selected

Kinematic for Quad-copter



•The thrust produced by each propeller is perpendicular to the plane of rotation of propellors. It is directly proportional to the square of the angular velocity of the propeller

$$F_i = k_f \times \omega_i^2$$

•If L is defined as the distance between two motors or propellors for any diagonal of the drone, then the reaction moments about the X-axis and Y-axis

$$Mx = (F_3 - F_4) \times L$$

$$My = (F_1 - F_2) \times L$$

•**Newton's second law of motion**

- For linear motion: Force = mass × linear acceleration
- For rotational motion: Torque = inertia × angular acceleration

Hovering Motion

- **Equilibrium Conditions for hovering**

$mg = F_1 + F_2 + F_3 + F_4$

All moments = 0

- **Equation of motion**

$m = F_1 + F_2 + F_3 + F_4 - mg$

$m = o$

Rise or Fall Motion (Throttle up)

- **Conditions for hovering (rise)**

$mg < F_1 + F_2 + F_3 + F_4$

All moments = 0

- **Conditions for Fall**

$mg > F_1 + F_2 + F_3 + F_4$

All moments = 0

- **Equation of motion**

$m = F_1 + F_2 + F_3 + F_4 - mg$

$m > o$

Yaw Motion

- **Conditions for hovering**

$mg = F_1 + F_2 + F_3 + F_4$

All moments ≠ 0

- **Equation of motion**

$mass^* linear\ acceleration = F_1 + F_2 + F_3 + F_4 - mg$

$I_{zz}\ ^*angular\ acceleration@\ Z\text{-}axis = M_1 + M_2 + M_3 + M_4$

Pitch and Roll Motion

- **Conditions for hovering**

$mg < F_1 + F_2 + F_3 + F_4$

All moments ≠ 0

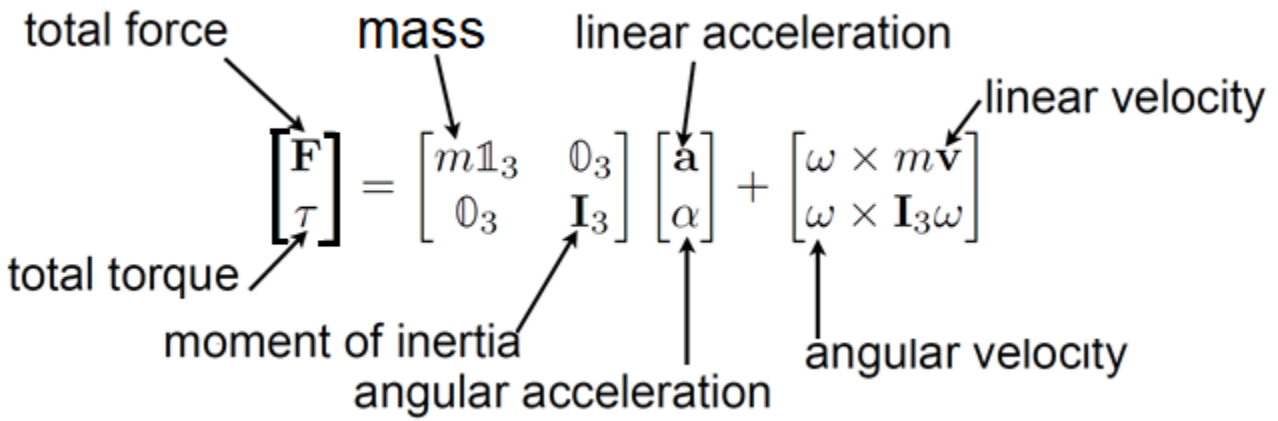
- **Equation of motion**

$mass^* linear\ acceleration = F_1 + F_2 + F_3 + F_4 - mg$

$I_{xx}\ ^*\ angular\ acceleration\ @\ x\text{-}axis = (F_3 - F_4) \times L$

Rigid-body dynamics

•To calculate individual speeds and forces acting on drones, the three-dimensional rigid-body dynamics should be modeled
•The first step is to identify the reference coordinates, the direction of rotor speed and forces acting the drones
•For the rigid body, we have to consider the effect of aerodynamic, inertial, gravitational, and gyroscope
•**Aerodynamic Forces:** rotation of the propellors in air causes various forces such as friction and drag
•**Secondary aerodynamic effects**: blade flapping, ground effect, and local flow fields
•**Inertial counter torques**: gravitational forces acting at the center of drone affect the rotation of propellors
•**Gyroscopic effects**: change in the orientation of drone body and plane rotation



•propellors.

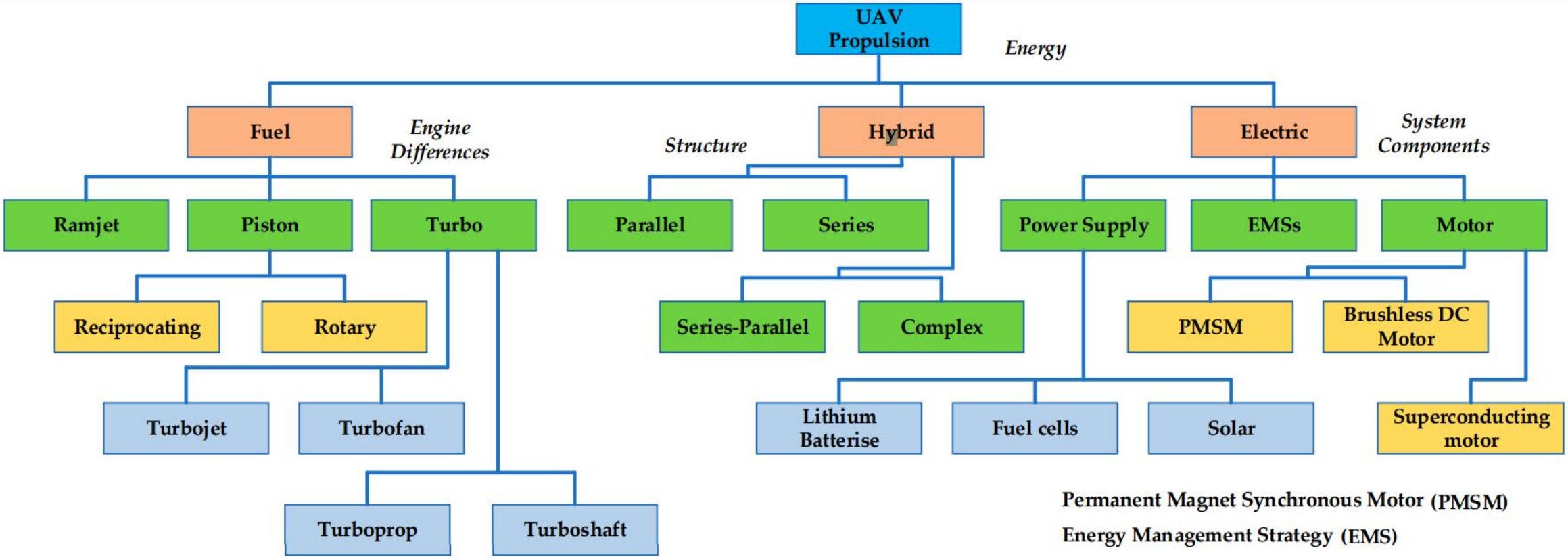
•Based on Newton-Euler equations, all forces and moments acting on a quadcopter are combined and result in a complete model of the drone dynamics
•This physical model is useful to control the desired motion of the quadcopter

Angle of attack

Mach number

Propulsion and airplane structures.

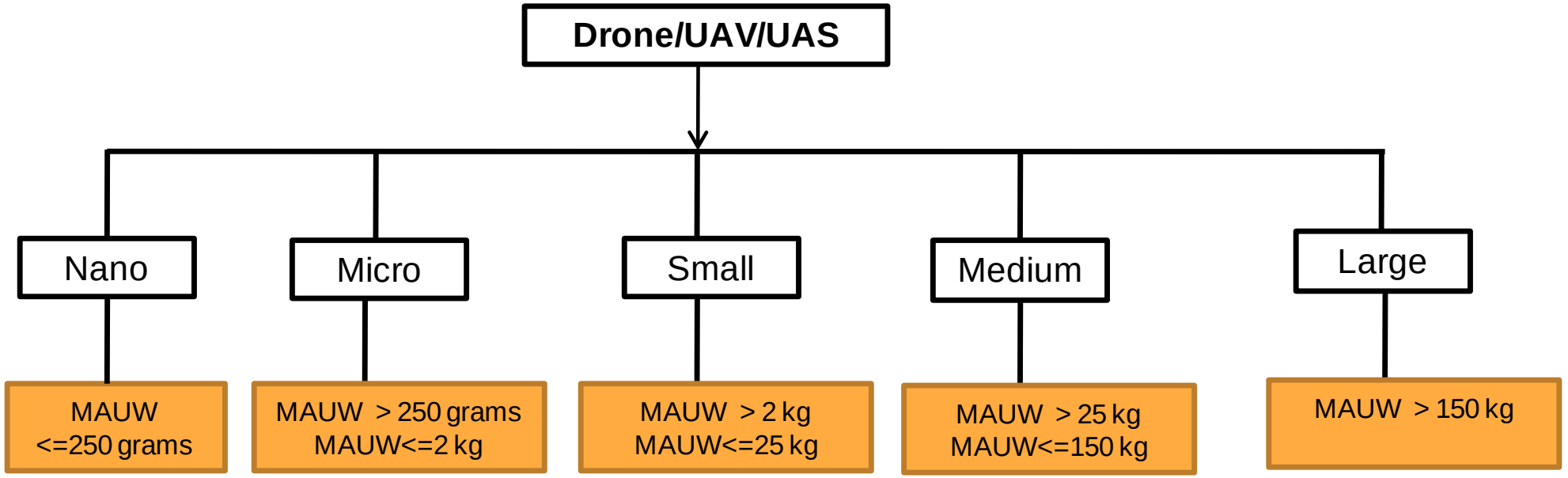
UAV Propulsion System



Permanent Magnet Synchronous Motor (PMSM)

Energy Management Strategy (EMS)

Drone Classifications - DGCA



MAUW – **M**aximum **A**ll **U**p **W**eight

Complete list of Flight Controller firm ware projects

- There are loads of different flight controller projects out there.
- Each usually focus on some specific hardware, technology or aims.
- Here, we have composed a master list of all of the flight controller firmware and software out there, along with a few comments about each of them.
- This will help you get a good picture of what is available so you can make up your mind about which you want to use or get involved in.

Full Auto Pilot Projects

- Ardupilot
  - Who should you use it?
- PX4 Flight Stack
  - Why should you use it?
- iNav
  - Why should you use it?
- LibrePilot
  - Why should you use it?
- Paparazzi
  - Why should you use it?

Contd…

- FPV and Racing Projects
  - BetaFlight
    - Why should you use it?
  - CleanFlight
    - Why should you use it?
  - dRonin
    - Why should you use it?
  - KISS
    - Why should you use it?
  - RaceFlight
    - Why should you use it?

Contd…

- Outdated/inactive projects
  - MultiWii
  - Baseflight
  - TauLabs
  - OpenPilot

Ardupilot

Ardupilot

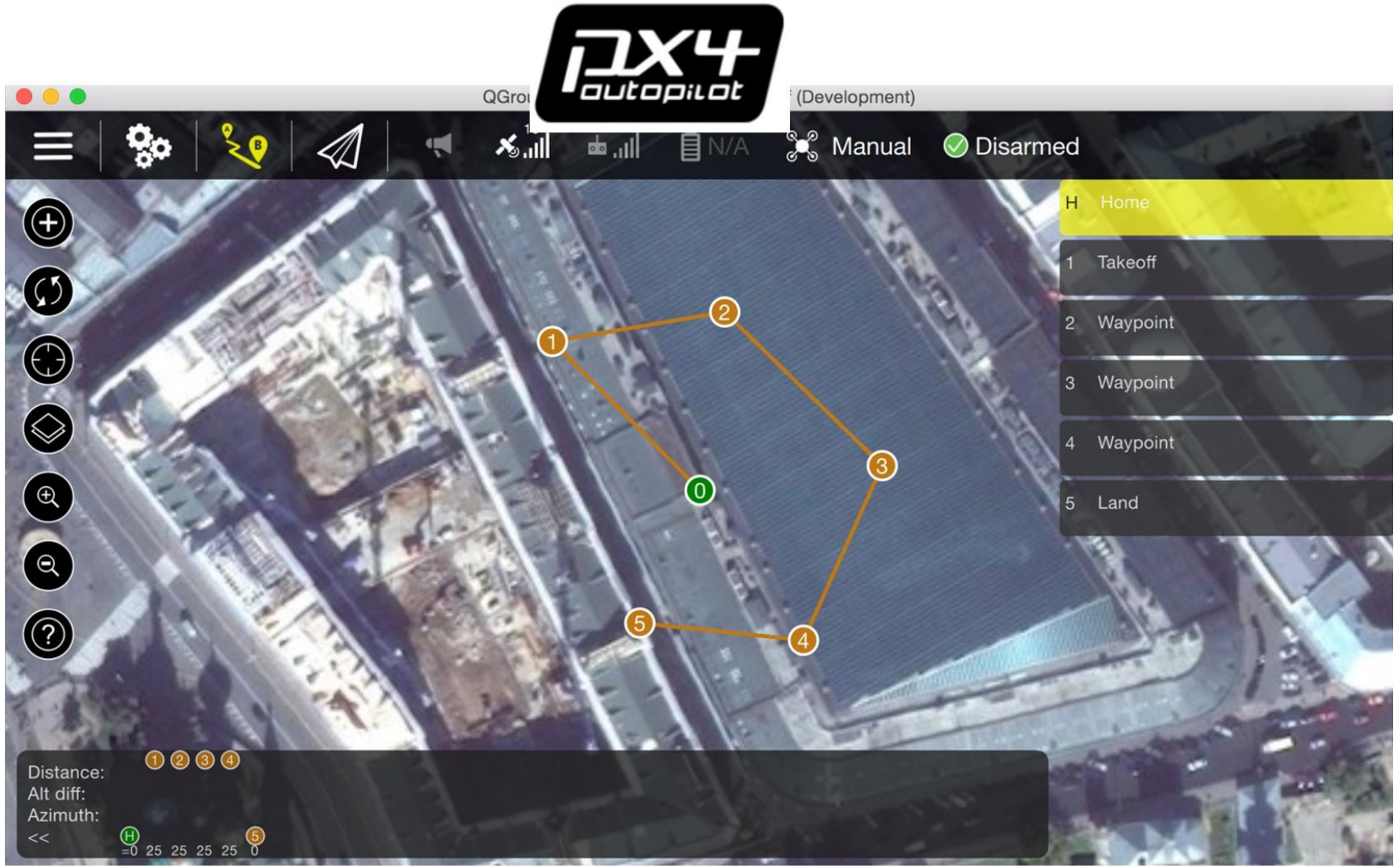
- Ardupilot is probably the most popular drone autopilot software.
- It has been around since the beginning and has an active development community.
- The software can control just about any vehicle, both fixed-wing planes, quadcopters and everything between including hybrids, ground rovers and boats.
- The firmware is divided into three branches: Ardupilot for fixed-wing aircraft, Arducopter for multirotors and helicopters, and Ardurover for ground-based vehicles.

https://ardupilot.org/

Who Should You Use It?

- If you want to have a fully featured autopilot system with 3D waypoint navigation, and a wide range of hardware and software support ardupilot is the best one out there. It also has the largest developer community and is used on many commercial systems.

PX4 Flight Stack



PX4 Flight Stack

- The PX4 flight stack is another powerful, fully featured autopilot.
- This project supports multiple vehicle types, also has an active developer community. The most famous platform that runs the PX4 flight stack is some of the drones from Yuneec.
- On the surface you can consider PX4 flight stack to be similar to ardupilot as there is alot of hardware that can run either PX4 or Ardupilot firmware.
- Similarly, since both systems use the same telemetry protocol (MAVlink) alot of software is also compatible with both such as Qgroundcontol.
- It is only when you get to very specific functions you may notice some variations between the two projects.
- However in terms of core autonomous functions PX4 and Ardupilot are very similar.

https://px4.io/

Why Should You Use It?

- There are many discussions between PX4 and Ardupilot to compare which one is better.
- However, the key differentiation is with the software license. PX4 uses BSD while ArduPilot uses GPL.
- This makes PX4 more attractive for commercial use since you don’t need to make your modifications open source.
- With ardupilot, any changes need to be shared as open source. So if you plan to develop some special function that you want to keep private or sell, PX4 is the way to go.
- This is why PX4 is more attractive for business use.

INav

https://github.com/inavFlight/inav/wiki

iNAV

- iNav is essentially a fork of Betaflight that focuses on adding autonomous drone functions rather than FPV racing features.
- This lets the software fly your drone to waypoints, or return home all on its own.
- This firmware supports most of the FPV racing flight controllers (that are much cheaper to buy than ardupilot boards). iNav also supports both multi-rotors and fixed-wing aircraft.

Why Should You Use It?

- If you already have a FPV racing flight controller (such as the SPF3) and want to have full automomous control with an easy to use/ familiar cleanflight style GUI, iNav is a great option.

LibrePilot

https://www.librepilot.org/site/index.html

Librepilot was forked from [OpenPilot]

Libre Pilot

- LibrePilot aims to be a general flight autopilot that supports both fixed and rotor wing aircraft.
- The GUI is an excellent piece of software. The firmware itself is also solid and great to work with and on, The only downside is that there are not as many features as arudpilot, but this is mainly because this project is not as popular so has fewer developers in the community.

Why Should You Use It?

- If you want to run a powerfull, but cheap autopilot platform (running CC3D boards or similiar), librepilot is a great option for you.

Paparazzi

- Paparazzi project homepage

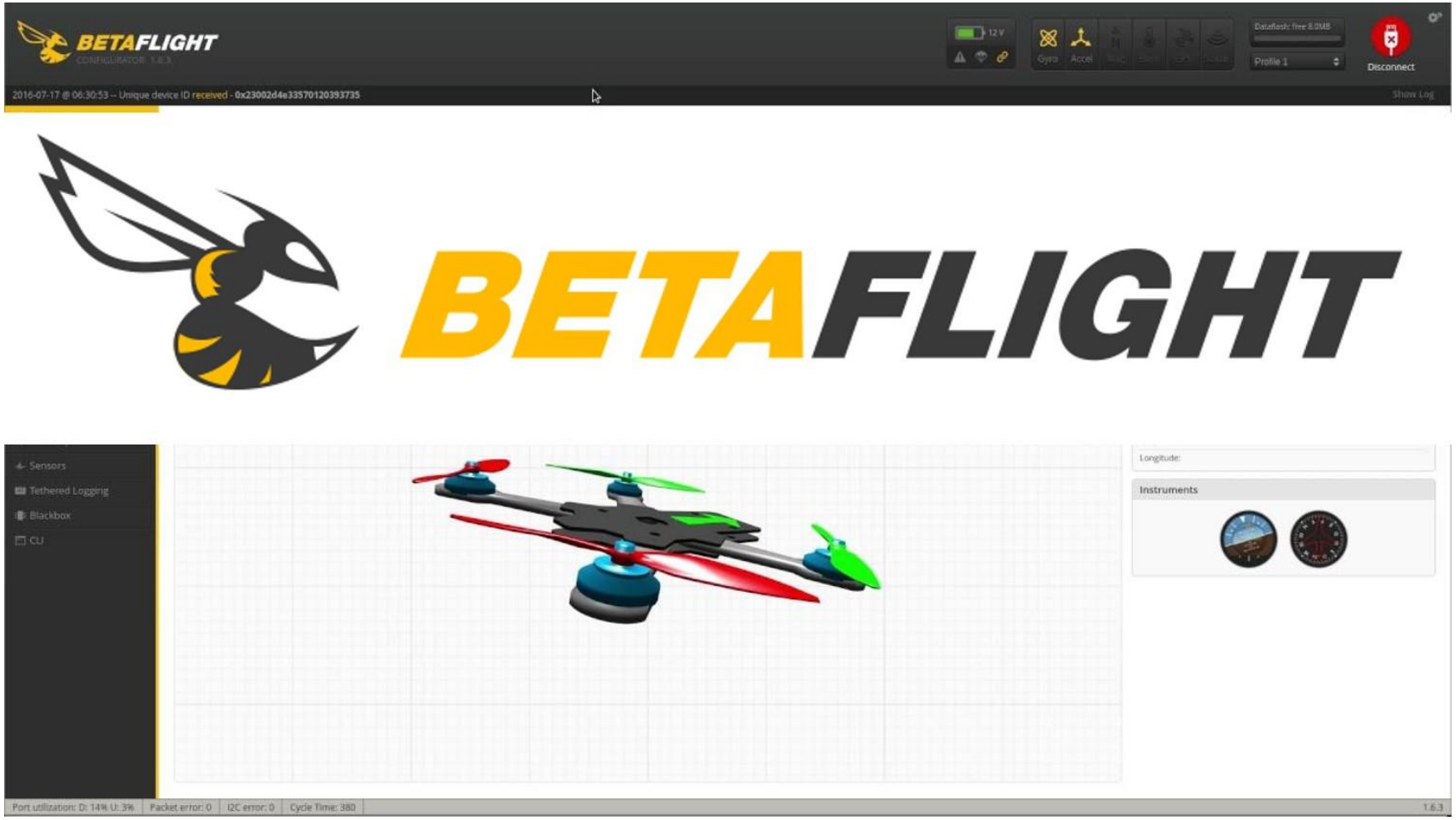
Paparazzi

- Paparazzi was the first truly open source firmware to control drones. It has not received as much adoption as other projects because it is quite technical. Also, alot of the hardware is not widely available, or fully open source which is a shame. However, this firmware is still actively developed and mainly used by universities.

FPV And Racing Projects

These projects are dedicated to FPV and FPV racing, they have modes and features dedicated to squeezing the extra performance out of your motors to make your aircraft as agile as possible.

BetaFlight


BETAFLIGHT
CONFIGURATOR
2016-07-17 @ 06:30:53 -- Unique device ID received - 0x23002d4e33570120393735
Gyro
Accel
Profile 1
Disconnect
Show Log
BETAFLIGHT
Sensors
Tethered Logging
Blackbox
CLI
Longitude:
Instruments
Port utilization: D: 14% U: 3%
Packet error: 0
I2C error: 0
Cycle Time: 380
1.6.3

Beta Flight

- BetaFlight is currently the most popular FPV firmware and is considered to be on the cutting edge of FPV flight controllers.
- Primarily used for quadcopters, it also has support for fixed wing FPV planes. Although you could use any of the full autopilot projects to power your FPV quad, betaflight developers are focused on getting the most speed and performance out of your flight controller and quadcopter, with things like reading sensor data at 32Khz and sending updates to the motor at the same insanely fast rate. Betaflight might not be able to fly your drone for you, but it will give you the best feel and peformance for FPV flying!

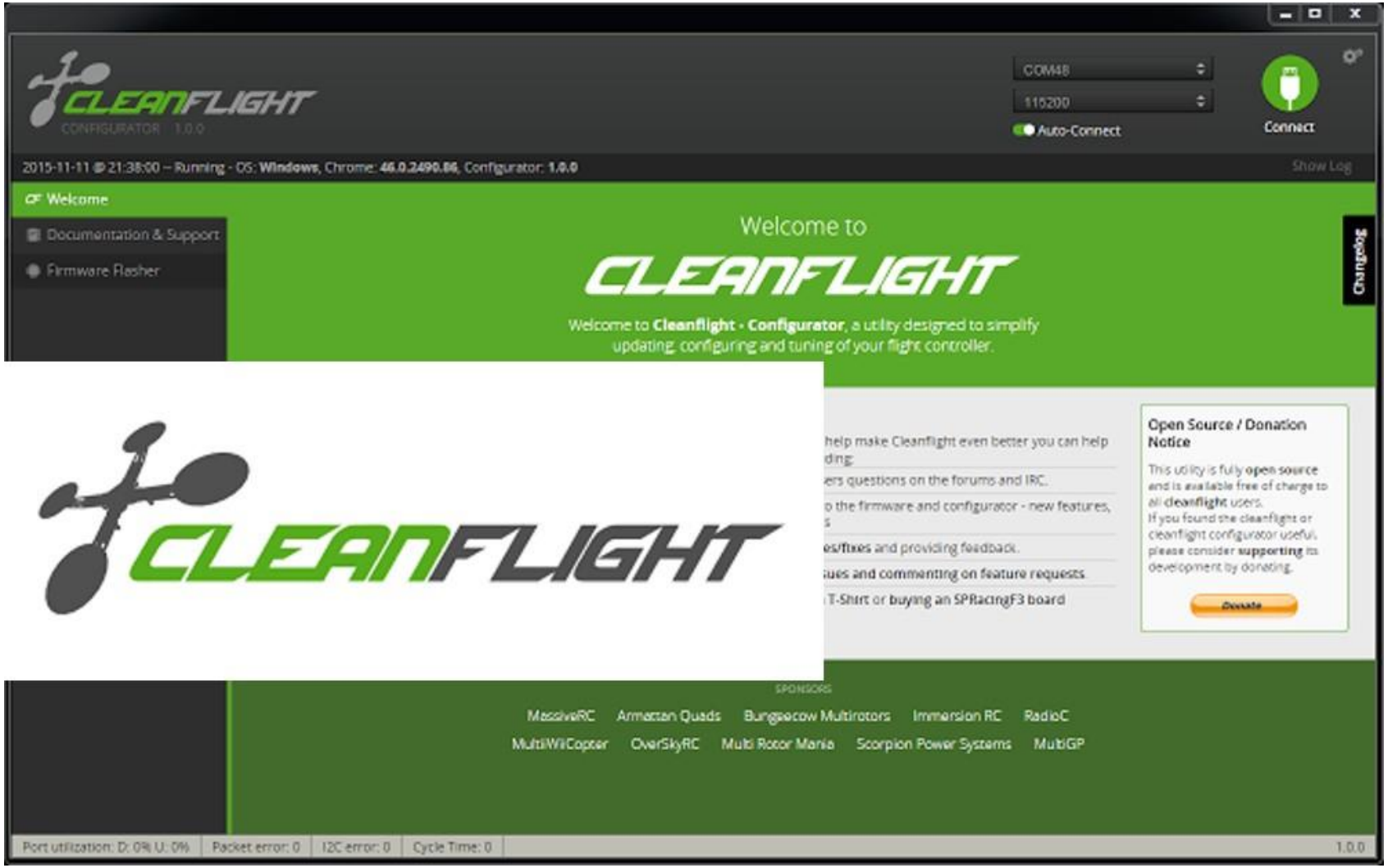
•Betaflight project homepage

Betaflight was forked from Cleanflight

Why Should You Use It?

- Right now, it is the best FPV firmware. It supports a load of flight controllers, and has an impressive list of compability with advanced FPV features like OSD control, VTx control, Blheli Dshot. So at the time of this article, if you want to fly FPV, then run betaflight firmware on your FC (if it is supported)

CleanFlight



CleanFlight

- CleanFlight was built from baseflight, but with the focus on making the code easy to work with and maintain.
- This quickly gained popularity and was the most popular firmware until betaflight came along. Recently Cleanflight has merged back with Betaflight (since v2.0) to bring it back up to date with all the cutting edge features the Betaflight developers have added.

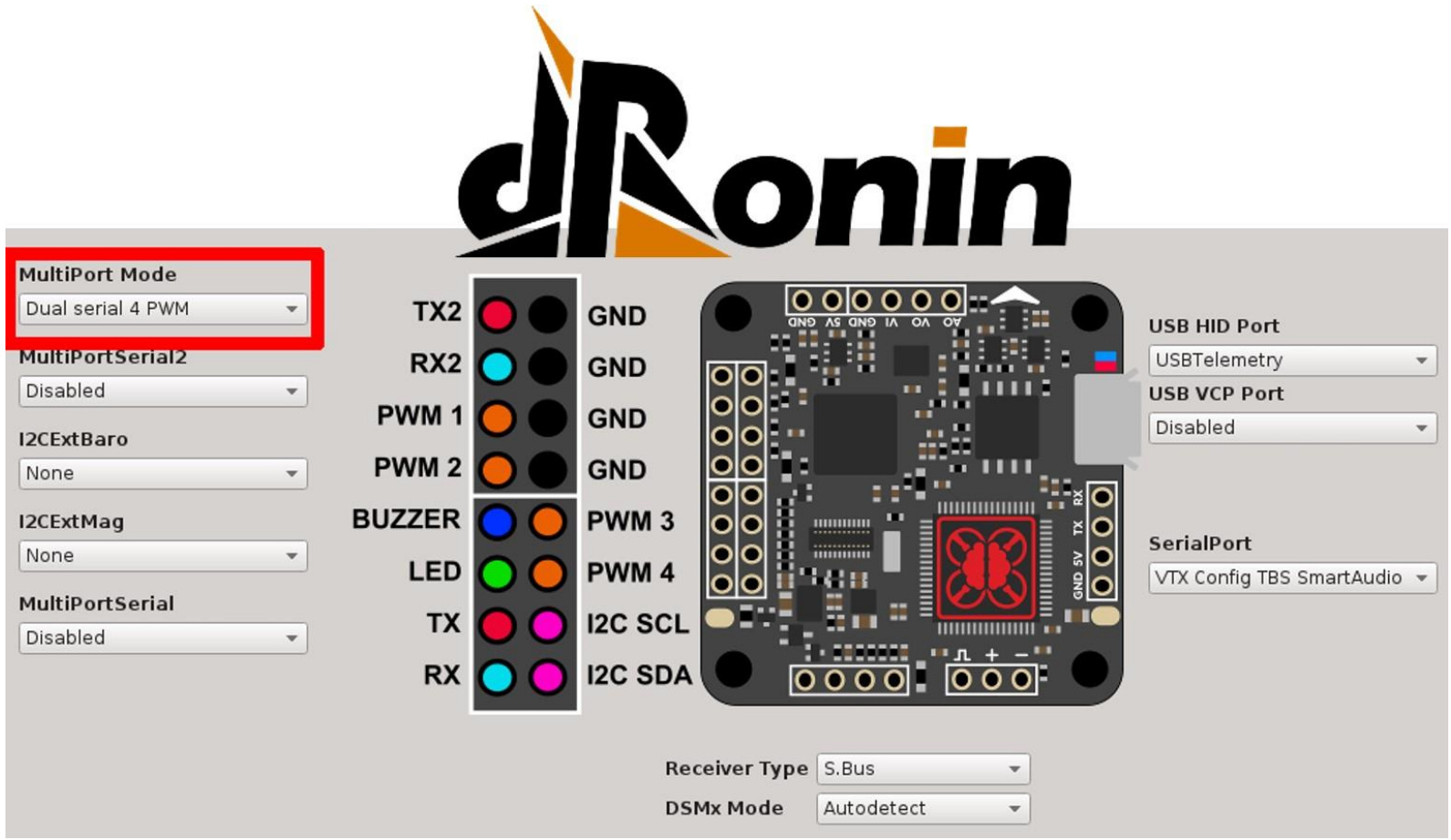
•Cleanflight project homepage

Cleanflight was forked from BaseFlight

Why Should You Use It?

- If you are running one of the official SPracing flight controllers its best to use cleanflight. But in general at the time of this article I suggest you just run betaflight for your FPV quad (assuming your board is supported by betaflight). This is beacuse betaflight is upgaded more regularly.

DRonin



- [dRonin project homepage]

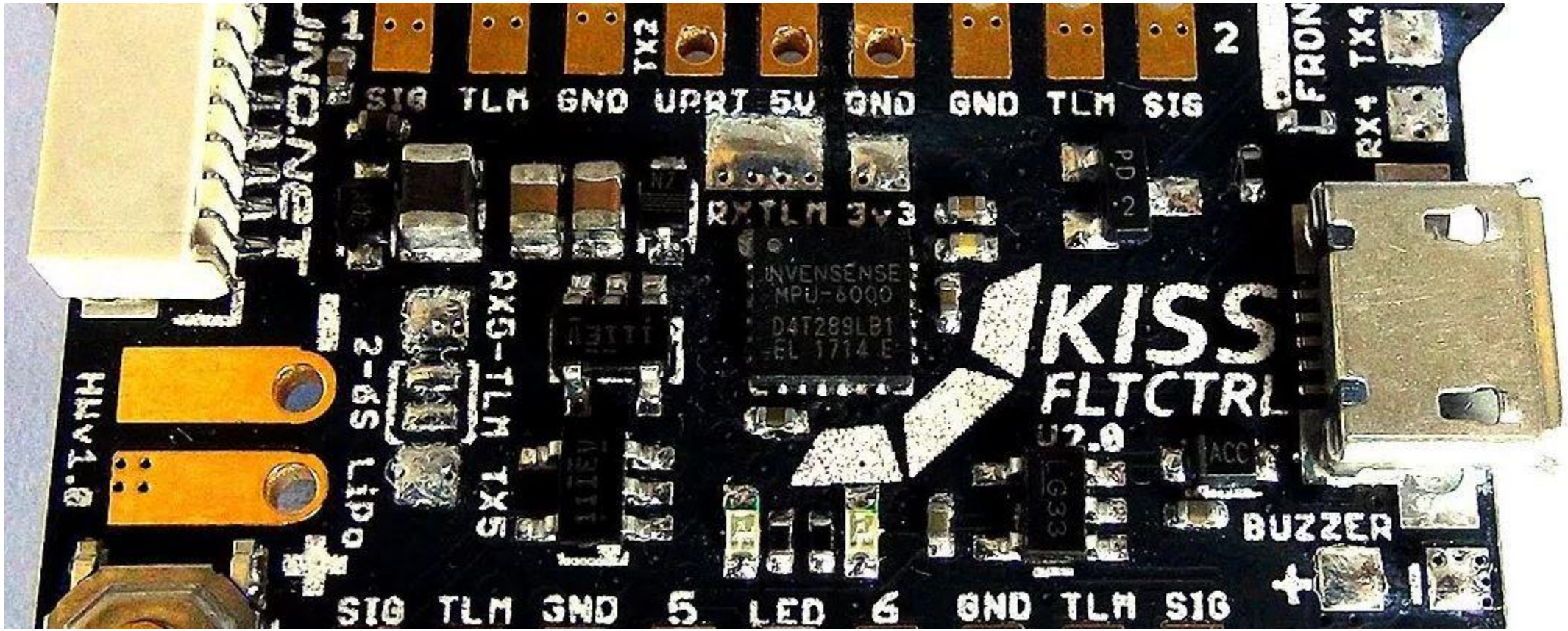
DRonin

- Another fork from openpilot, DRonin has a core focus on FPV racing using the openpilot firmware. The novel feature of dRonin is their autotune program. This will automatically fine tune the settings on your flight controller for your quad to give you the best response from your quadcopter. However, many of the other FPV specific features that come with Betaflight.

Why Should You Use It?

- If you want a sophisticated flight control firmware, along with great setup wizards on affordable CC3D based hardware.

KISS



- [KISS project homepage]

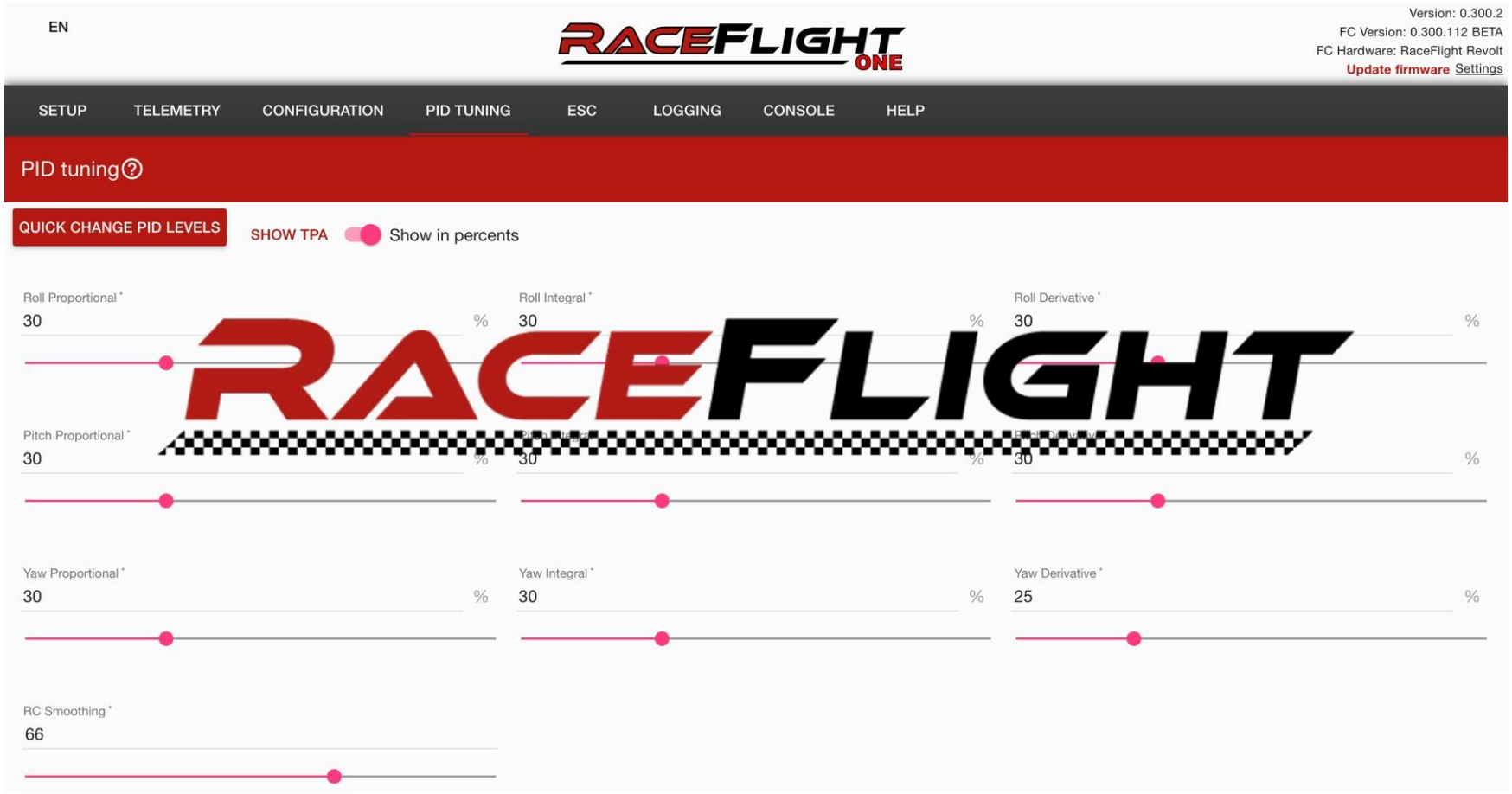
KISS

- KISS is the only closed source project on this list, and it only runs on KISS boards from FlyDuino. KISS has gained a lot of popularity as having reliable hardware that is easy to use and just works. As the name suggests, this project is aimed to Keep It Super Simple. The configuration software has a lot less flexibility when compared to betaflight, but this makes it easy to use. However, don’t let this simplicity make you think it lacks the latest features. To some extent, the KISS developers have introduced new features that forced the open source projects to play catch up for a long time.

Why Should You Use It?

- If you like apple products, then you will probably like the KISS ecosystem. Their gear plays well with echother and work very well, but the closed source nature does force you to use KISS FC, along with KISS ESC etc..

RaceFlight



- [RaceFlight project homepage]

RaceFlight

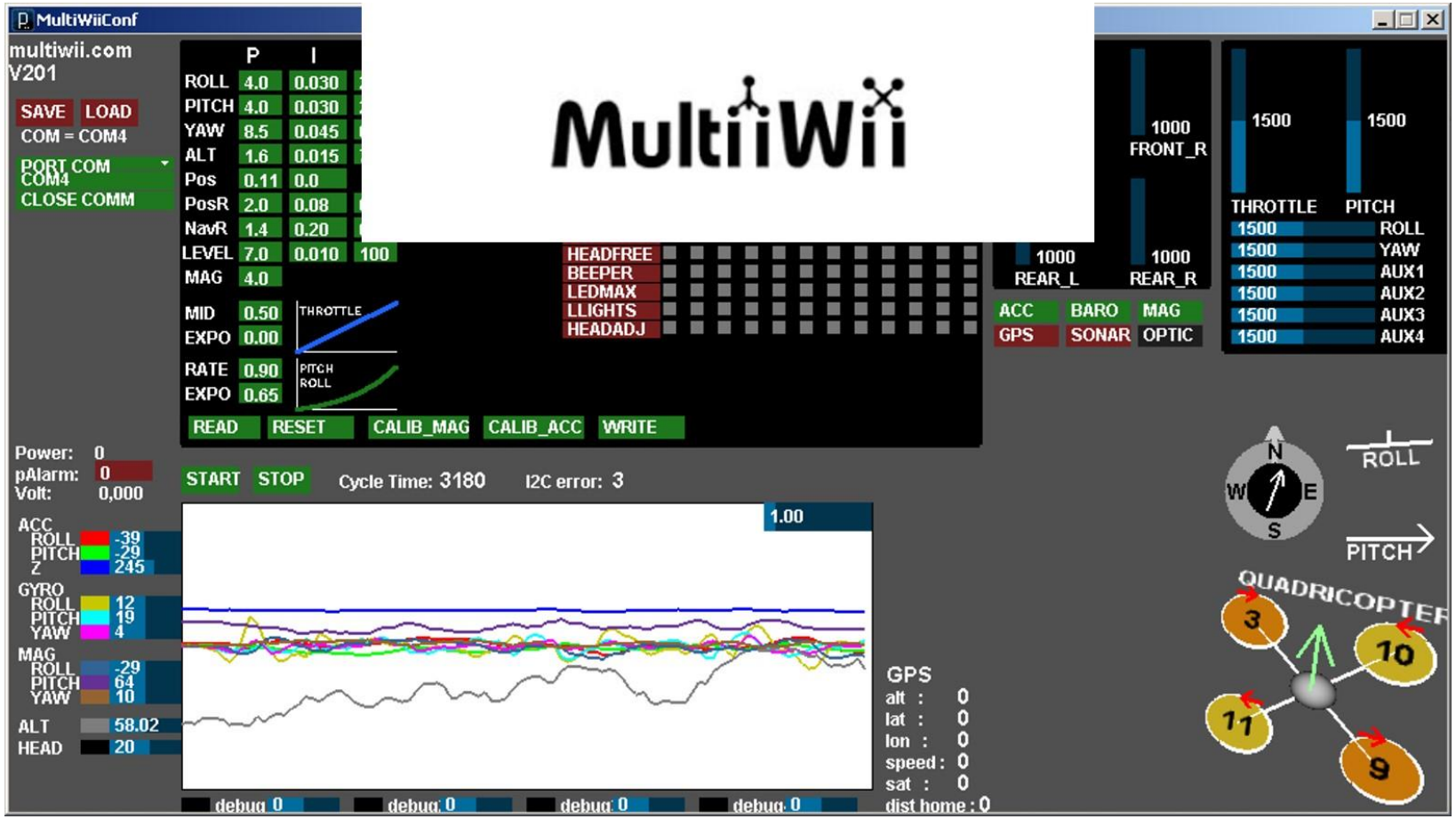
- RaceFlight started life as a fork of CleanFlight with a focus on FPV racing and F4 processors (at the time cleanflight only supported F3). However, after a while, the developers started moving the raceflight code to eventually make the entire project closed source and only work on the official raceflight boards. The justification for this is that now the developers can make some money for their efforts on pushing the project forward. This approach has seen the raceflight firmware being known for its smoothness and high response. It also has very nice software to make setting up your flight controller.

Why Should You Use It?

- Similar to KISS, the raceflight ecosystem locks you into their products. But that is not necessarily a bad thing, they have great flight controllers and ESC’s and their setup software is also very nice to use. However, it still lacks some of the new features being released with Betaflight.

Outdated/Inactive Projects

MultiWii



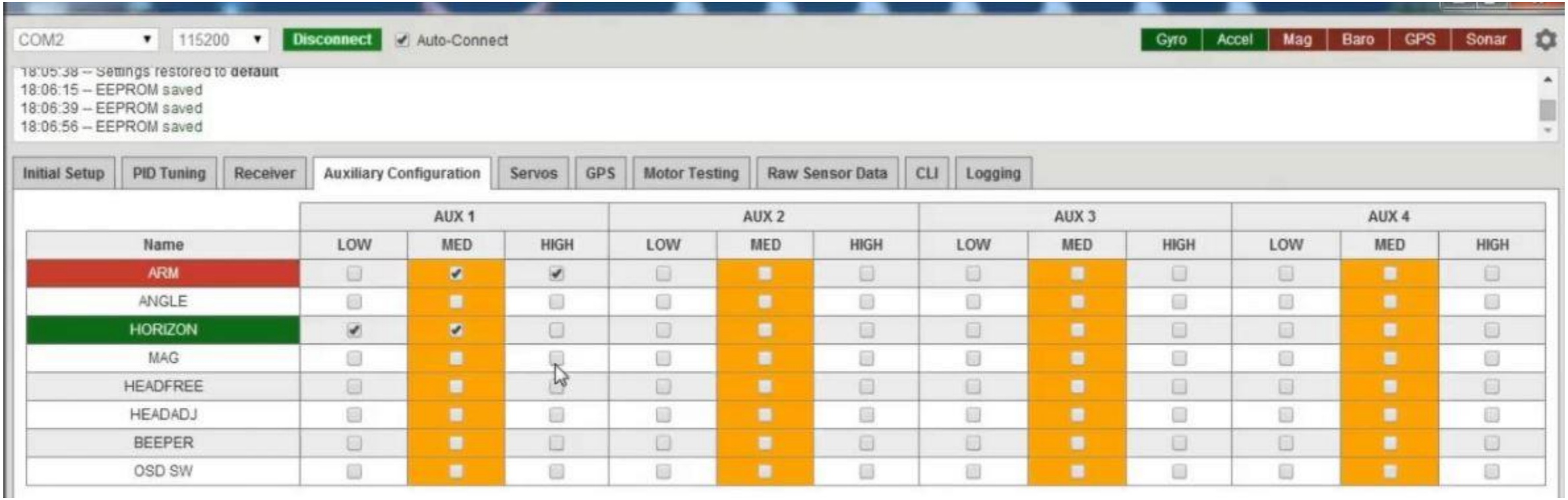
- [MultiWii github page]

MultiWii

- MultiWii was the genesis of modern quadcopter firmware. It started around the same time the Nintendo Wii was released. The Wii gave everyone access to cheap IMU sensors (from the Wiimote), so the MultiWii project was born (Multirotor + Wiimote). This project combined an Arduino 8-bit MCU and a Wiimote, and later was extended to dedicated boards. Although not in active development this firmware is worth an honourable mention for the history books.

Multiwii lead to the development of Baseflight which ultimately gave us BetaFlight which we all use today.

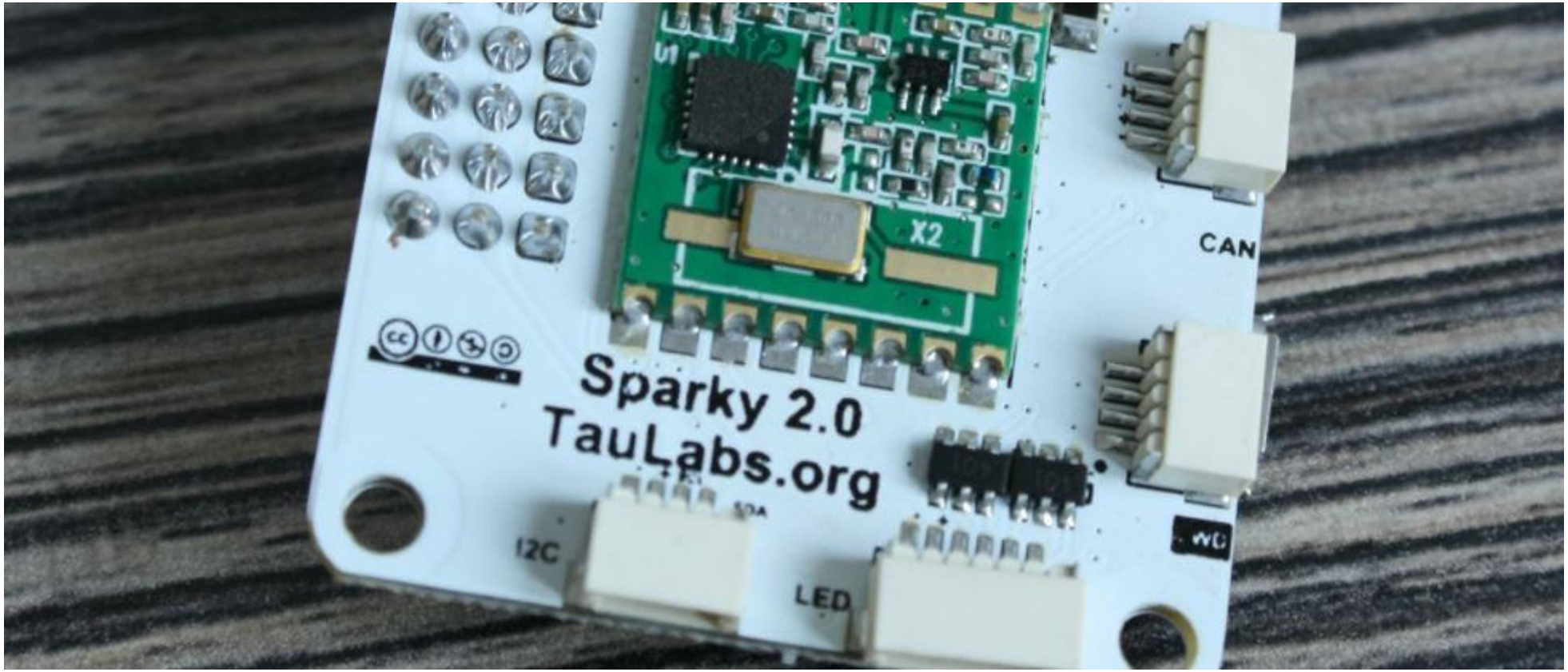
Baseflight


COM2
115200
Disconnect
Auto-Connect
Gyro
Accel
Mag
Baro
GPS
Sonar
18:05:38 -- Settings restored to default
18:06:15 -- EEPROM saved
18:06:39 -- EEPROM saved
18:06:56 -- EEPROM saved
Initial Setup
PID Tuning
Receiver
Auxiliary Configuration
Servos
GPS
Motor Testing
Raw Sensor Data
CLI
Logging
AUX 1
AUX 2
AUX 3
AUX 4
Name
LOW
MED
HIGH
ARM
ANGLE
HORIZON
MAG
HEADFREE
HEADADJ
BEEPER
OSD SW

Baseflight

- Baseflight was is the great-grandfather of betaflight, as it was the first 32-bit FPV flight controller. Started because the baseflight creator was fed up with MultiWii still using slow and outdated 8-bit firmware. So Baseflight was a complete re-write of the multiwii firmware adapted to use 32bit hardware. Although this project is not actively maintained, it is noteworthy as it paved the way for many other 32 bit FPV firmware.

TauLabs

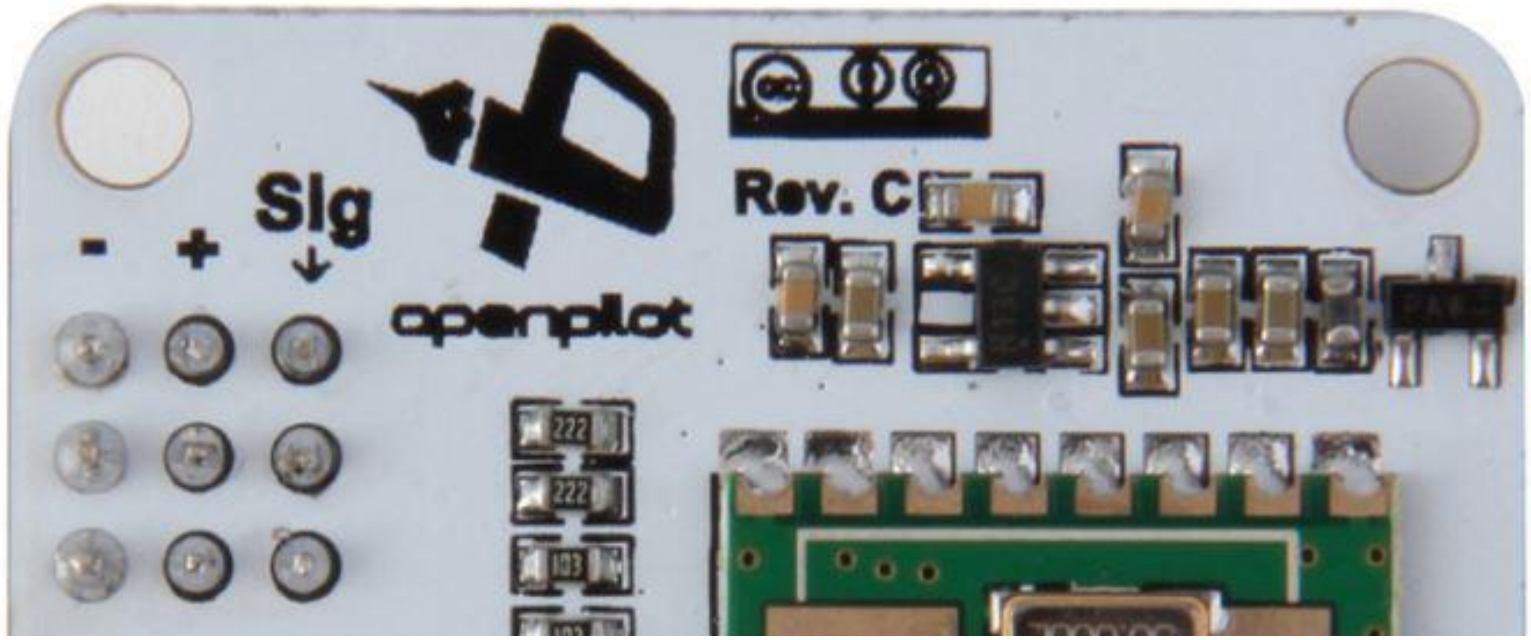


- [TauLabs github]

TauLabs

- TauLabs was one of the openpilot forks, that was aimed at professionals and researchers. The aim is/was to create high-quality code that can be re-used for professional applications, and make it easy to be used for research applications. Just like the other openpilot forks, TauLabs also suffers from less hardware support. However, the code is still available for anyone who wants to have some high-quality code to learn from.

OpenPilot



Openpilot

- Openpilot was among the first open source flight control software projects and helped shapes many of the standard features available. At the time it was one of the first to use 32bit hardware and had solid performance and features (with a professional feel). However due to some different views of the developers so Librepilot, dRonin, and TauLabs where created as a fork of openpilot both of which are still in active development.

MAVLink

- Based Ardupilot and its hardware
- Based Ground Stations over view
- MAVLink Flight modes
- MAVLink : Messages and Commands
- Based Overview of RC and Transmitters
- Sending the Drone to Commands through MAVLink
- Working on SITL : Starting Ardupilot Simulator : Simulation in the Loop
- Connecting SITL to Ground Stations
- Getting and Setting MAVLink Parameters
- Take of and Landing, Flying in Guided Mode

MAVLINK - Protocol

Introduction to MAVLINK Ardupilot, and Its common hardware

What is MAVLINK?

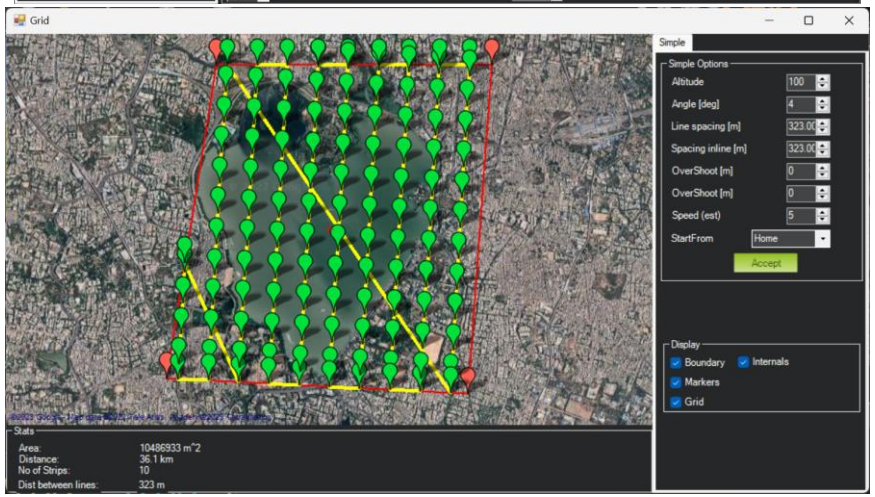
- MAVLINK or Micro Air Vehicle Link is a protocol for communicating with unmanned Vehicles. Preferably small category.
- MAVLINK was first released early 2009 by Lorenz Meier under GPL (https://ch.linkedin.com/in/meierlorenz), https://auterion.com/
- It specifies a set of messages that are exchanged between a small unmanned vehicle and a ground station.

MAVLink Protocol

Drone

MAVLink

{“time_boot_ms”;5482399,
“lat”:17.4239,”lon”:78.4738,”alt”:10987,”relative_alt”:0,”vx”:2,”vy”:6,”vz”:0,”hdg”:18309,”sysid”:0,”compid”:0,”msgid”;33}



MAVLINK
MICRO AIR VEHICLE COMMUNICATION PROTOCOL

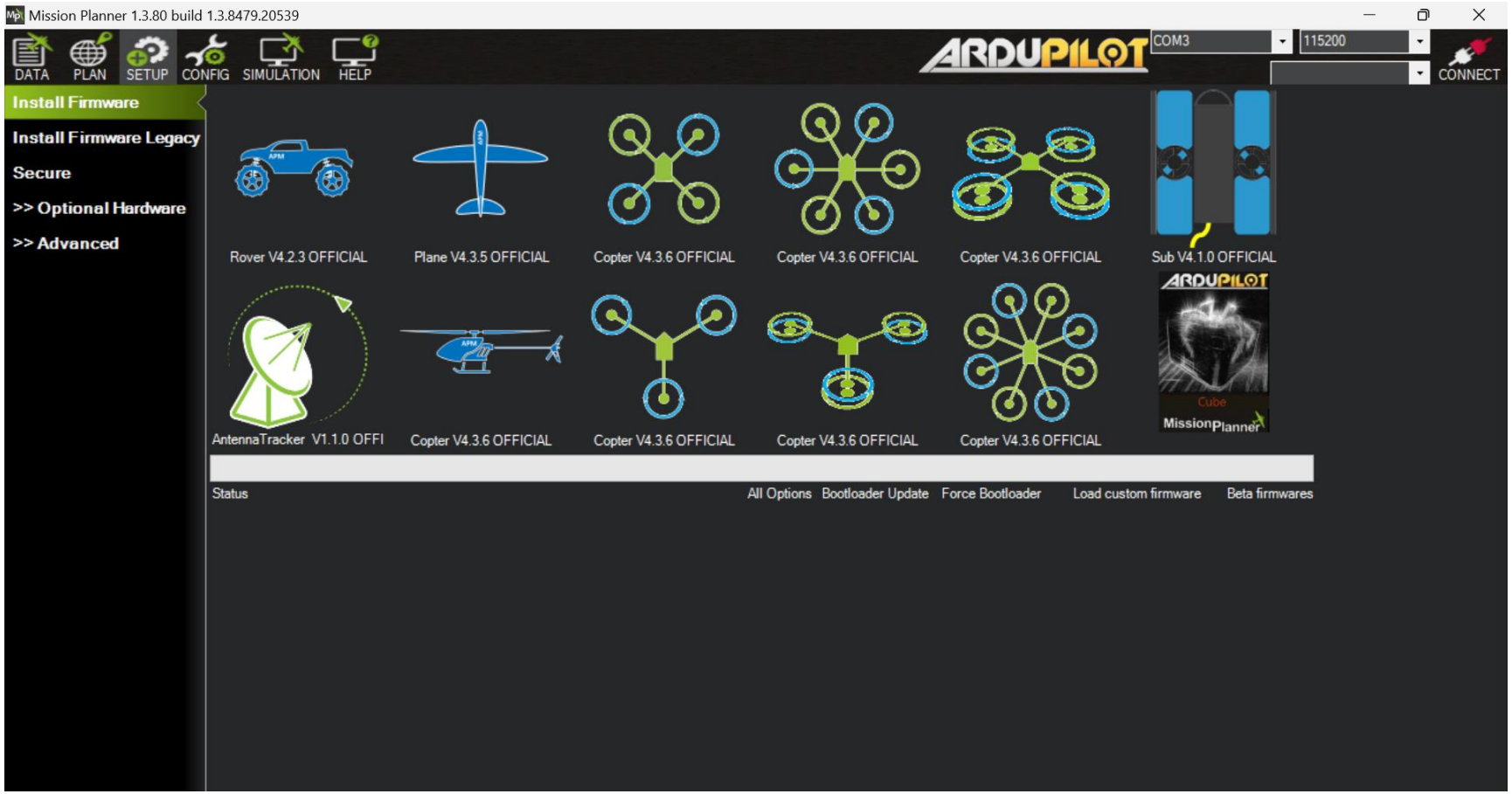
What is Ardupilot?

- Full featured and enable autopilot software
- The autopilot allows to control vehicle systems autonomously
- Developed over 5+ years by a team of diverse professional engineers and computer scientists
- Autopilot software capable of controlling any vehicle system, from airplanes, multirotors, and helicopters, to boats and even submarines.
- Supports the MAVLink protocol
- Advanced data logging, analysis and simulation tools

What is Ardupilot?

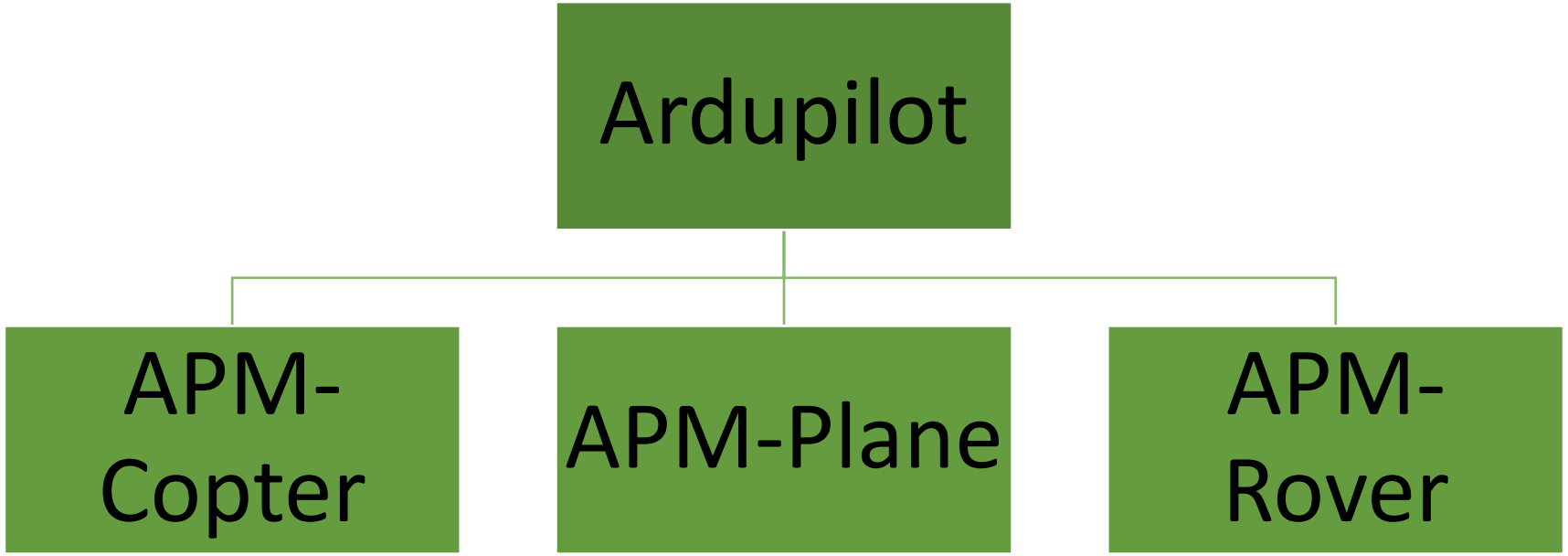
- Initially, developed for 8 bit micro controllers
- Now optimized for use with 32 bit ARM microcontrollers
- Run also under Linux and Single Board Computers

Vehicle Types



Ardupilot is configured for the vehicle

Ardupilot support Software Stack?



APM-Copter

- Open-Source Multicopter UAV Controller
- Supports Multicopters and Helicopters
- Supports different frames types of multicopters
- Won the Sparkfun 2013 and 2014 Autonomous Vehicle Competetion

MAVLINK
MICRO AIR VEHICLE COMMUNICATION PROTOCOL

APM : Plane

- For Fixed Wing Aircrafts
- Provides advanced functions such as support for
  - Hundreds of three-dimensional way points
  - Automatic take-off and landing
  - Sophisticated mission planning and camera controls

APM: Rover

- Advanced open source autopilot for guiding ground vehicles and boats
- Can run fully autonomous missions that are defined using mission planning software
- Won the 2013 and 2014 Sparkfun Autonomous Vehicle Competition.

Common Autopilot Hardware

Pixhawk

Pixhawk2

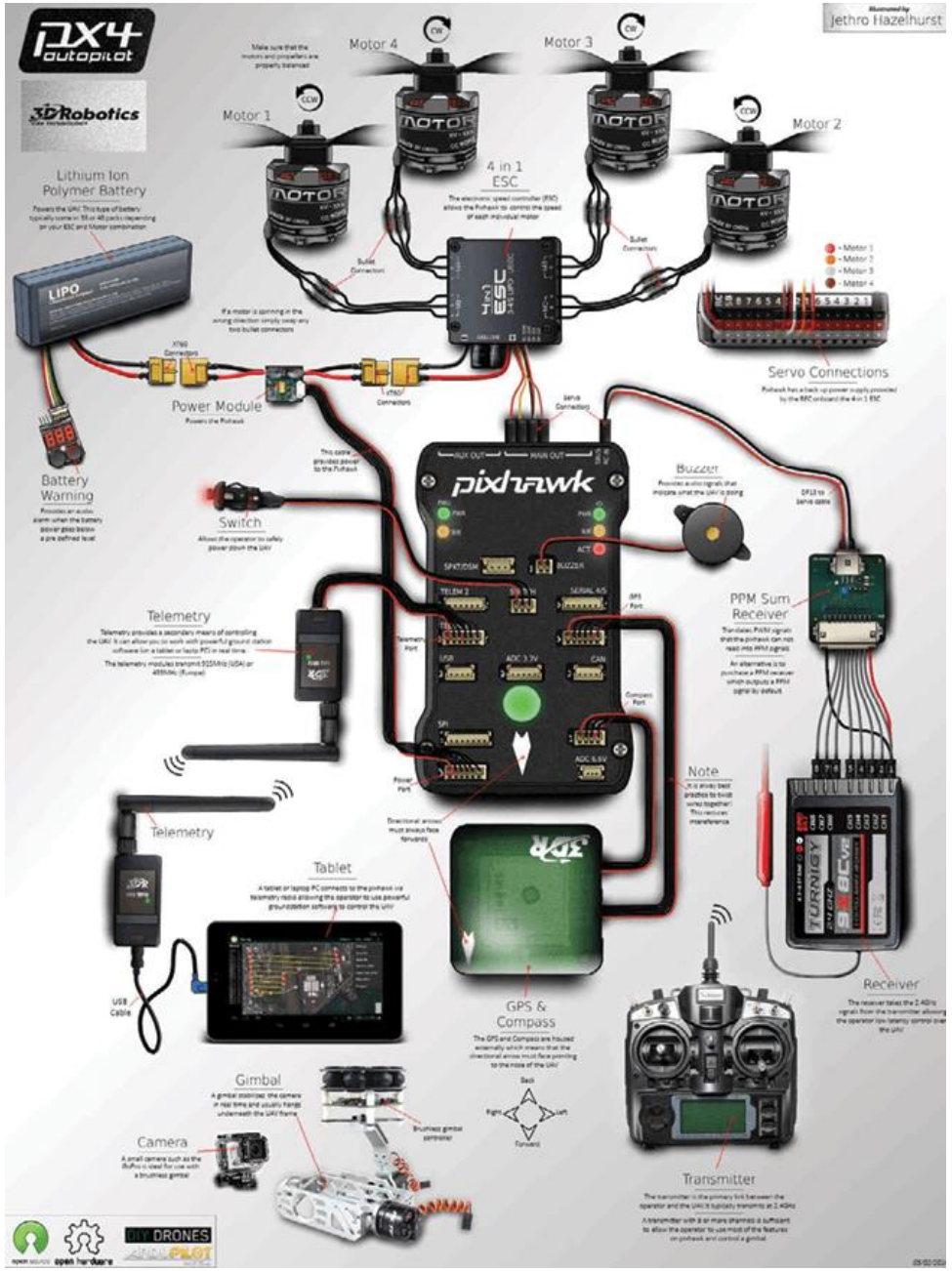
Navio2

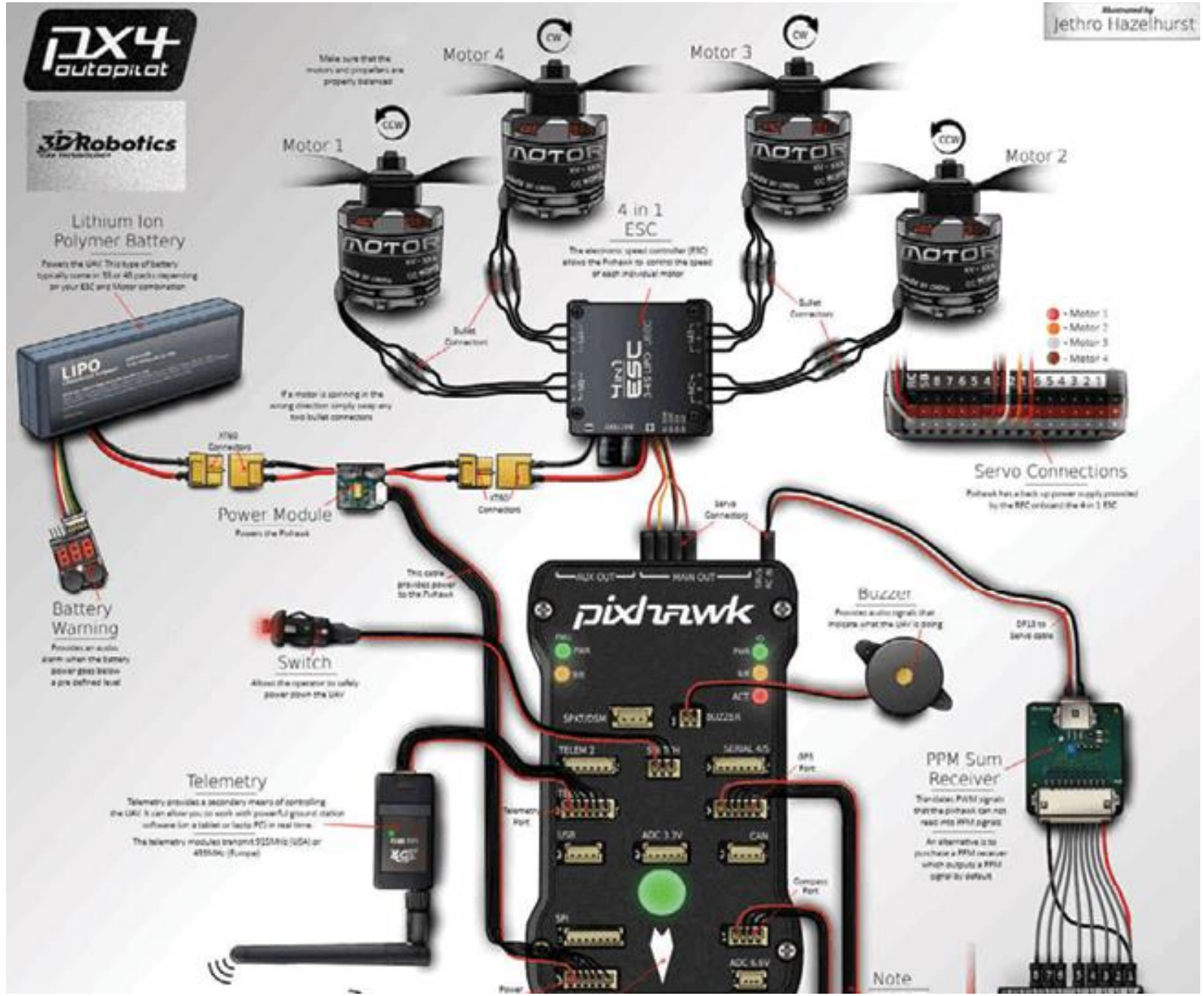
PXFmini

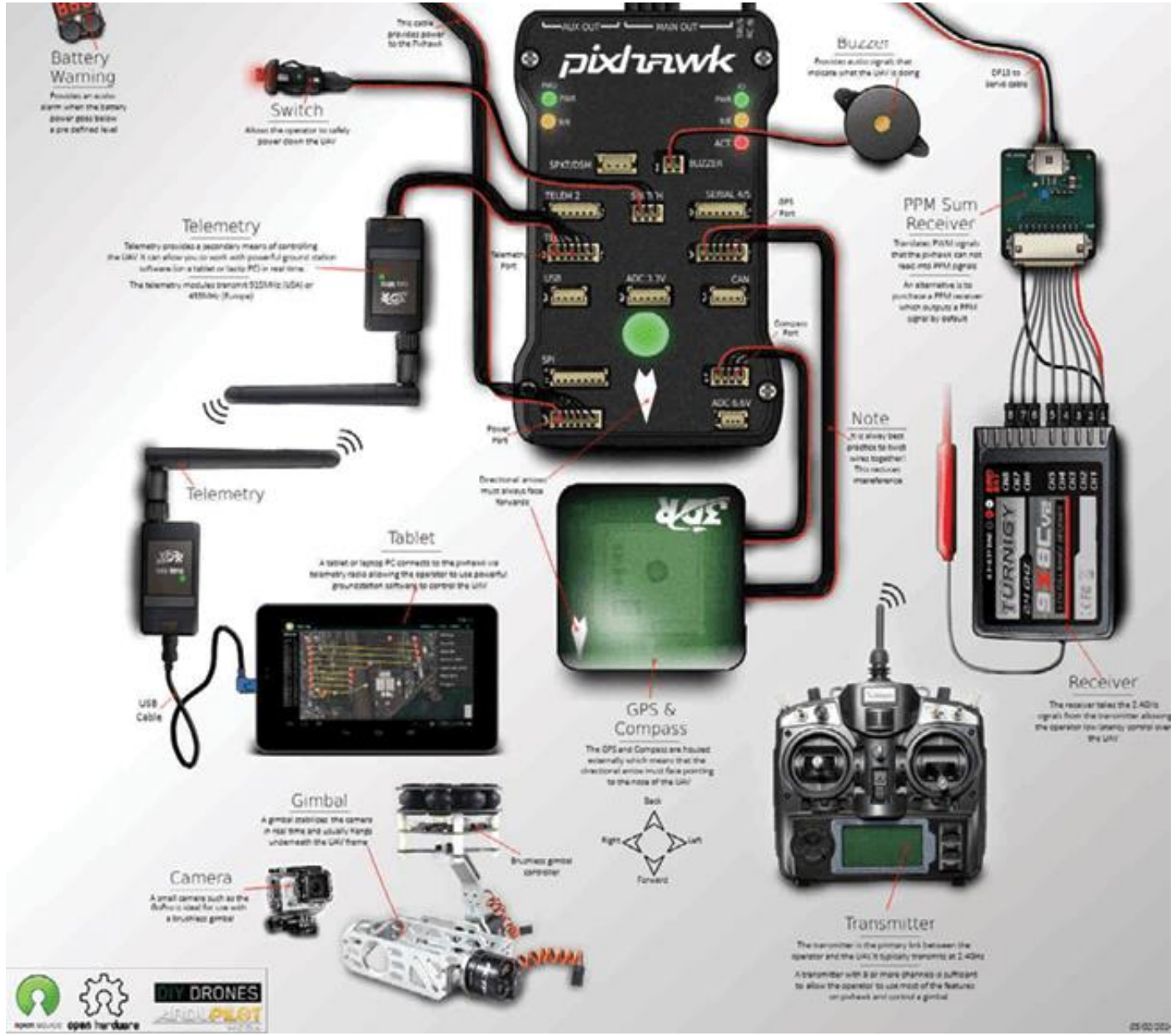
Erle Brain-2

Pixhawk 4

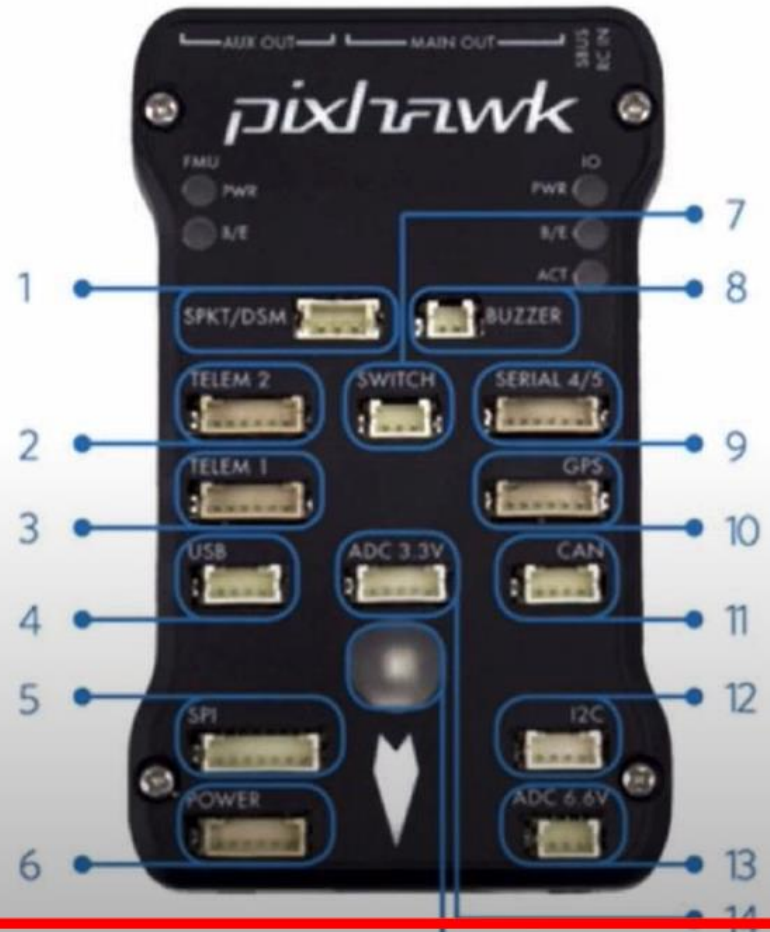
Hardware Interfaces - Ardupilot




px4 autopilot
3DRobotics
Illustrated by Jethro Hazelhurst
Motor 1
Motor 2
Motor 3
Motor 4
CW
CCW
Lithium Ion Polymer Battery
LIPO
4 in 1 ESC
Bullet Connectors
Servo Connections
Power Module
XT60 Connectors
Battery Warning
Switch
pixhawk
MAIN OUT
AUX OUT
Buzzer
PPM Sum Receiver
Telemetry
Note


Battery Warning
Switch
Telemetry
Telemetry
Tablet
USB Cable
Gimbal
Brushless gimbal controller
Camera
GPS & Compass
Buzzer
PPM Sum Receiver
Note
Receiver
Transmitter
pixhawk
DIY DRONES

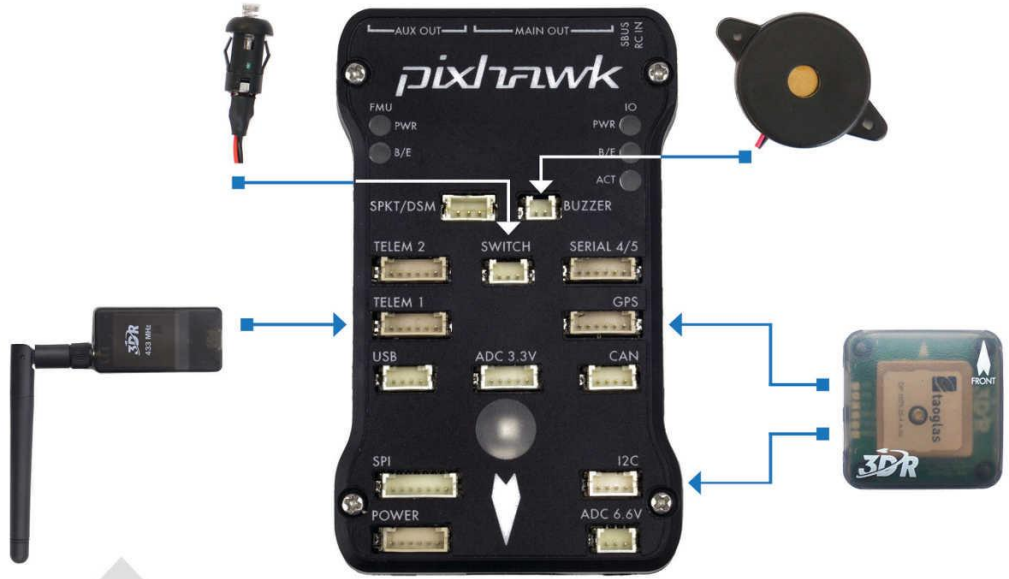
Pixhawk



1 Spektrum DSM receiver
2 Telemetry (radio telemetry)
3 Telemetry (on-screen display)
4 USB
5 SPI (serial peripheral interface) bus
6 Power module
7 Safety switch button
8 Buzzer
9 Serial
10 GPS module
11 CAN (controller area network) bus
12 I^2C splitter or compass module
13 Analog to digital converter 6.6 V
14 Analog to digital converter 3.3 V
15 LED indicator


Spectrum Satellite Receiver
Motor Safety Switch
Telemetry Radio
External USB
Sonar
Battery Power In
SPI
Front
MultiColor LED
Aux PWMs
Main ESC PWMs
SBUS
PPM
Audio
Serial UART 4 & 5
GPS
CAN Serial Port
I2C Serial & External Magnetometer
6.6 volt Analog Input
pixhawk
SPKT/DSM
TELEM 2
TELEM 1
USB
SWITCH
BUZZER
SERIAL 4/5
ADC 3.3V
SPI
POWER
GPS
CAN
I2C
ADC 6.6V

(Required) Connect the buzzer and safety switch.

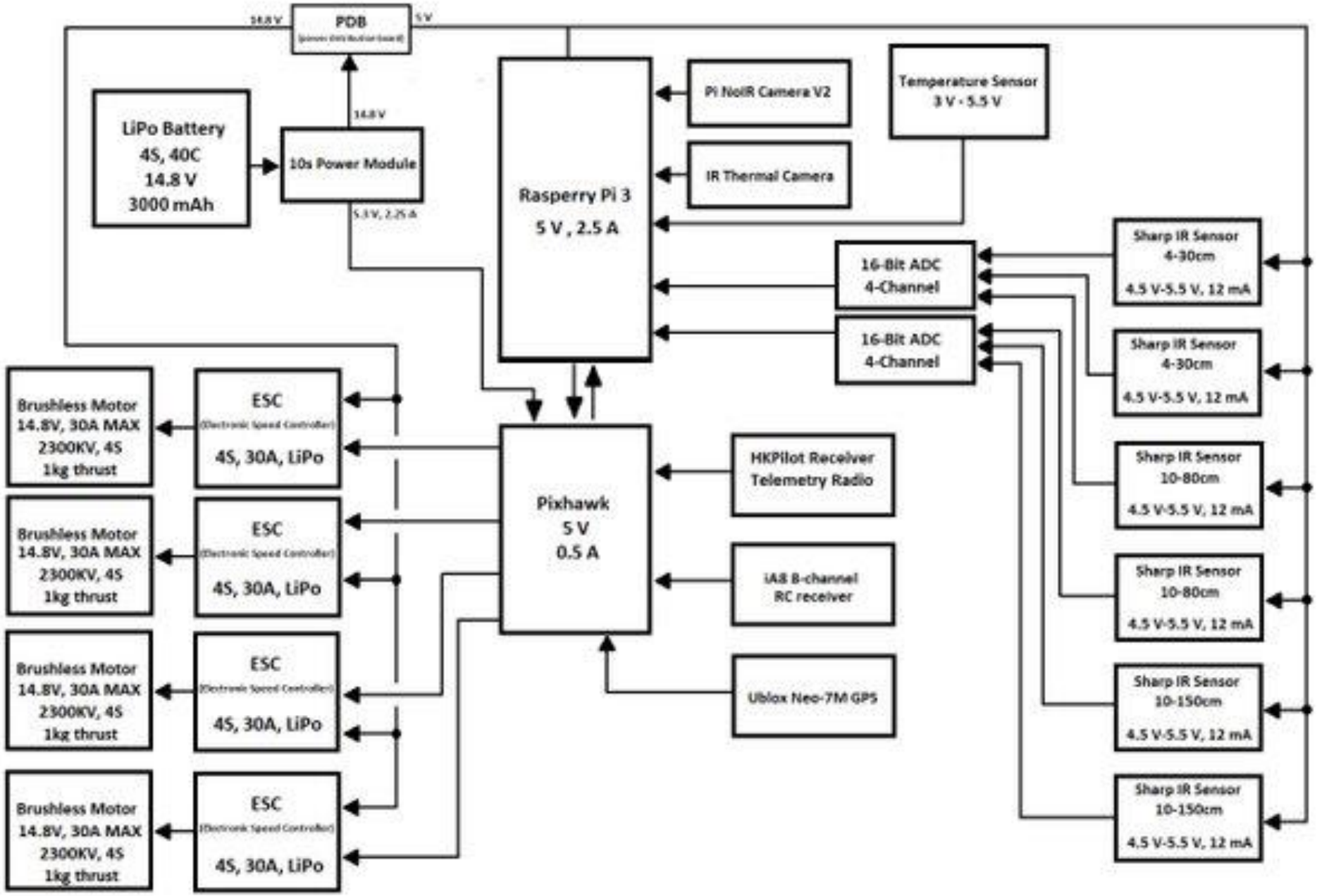


(Optional) Connect a 3DR Radio to Pixhawk's Telem port using the 6-wire cable provided with your 3DR Radio Kit to receive data and communicate with the autopilot in flight.

(Required) Connect a 3DR GPS+Compass to provide the autopilot with positioning data during flight. This kit includes a 6-wire cable to connect the GPS ports. Connect the MAG to the I²C port using the 4-wire cable provided with the 3DR GPS+Compass.

(Required) Connect the 3DR Power Module to the Power port using the 6-wire cable to direct power from your lithium polymer (LiPo) battery to the autopilot.

(Optional) The I²C splitter expands the I²C port to allow up to four additional peripherals to connect to Pixhawk. Use the 4-wire cable to connect the I²C splitter and add a compass module, external LED, digital airspeed sensor, or other peripherals to your vehicle.

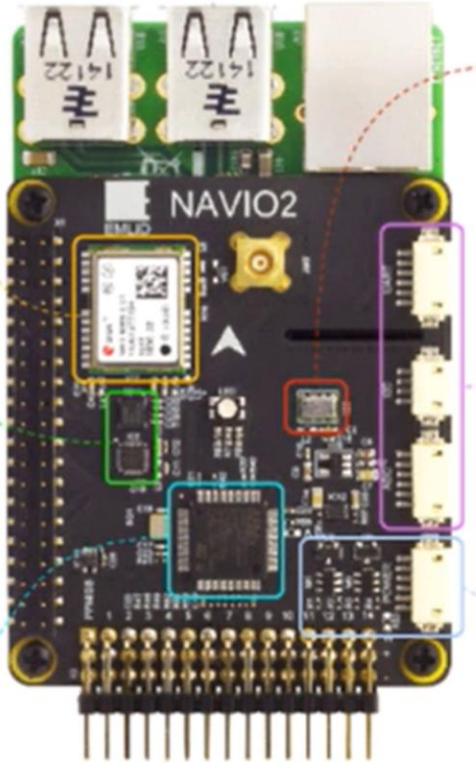

14.8 V
PDB
5 V
LiPo Battery
4S, 40C
14.8 V
3000 mAh
14.8 V
10s Power Module
5.3 V, 2.25 A
Rasperry Pi 3
5 V , 2.5 A
Pi NoIR Camera V2
IR Thermal Camera
Temperature Sensor
3 V - 5.5 V
16-Bit ADC
4-Channel
16-Bit ADC
4-Channel
Sharp IR Sensor
4-30cm
4.5 V-5.5 V, 12 mA
Sharp IR Sensor
4-30cm
4.5 V-5.5 V, 12 mA
Sharp IR Sensor
10-80cm
4.5 V-5.5 V, 12 mA
Sharp IR Sensor
10-80cm
4.5 V-5.5 V, 12 mA
Sharp IR Sensor
10-150cm
4.5 V-5.5 V, 12 mA
Sharp IR Sensor
10-150cm
4.5 V-5.5 V, 12 mA
Brushless Motor
14.8V, 30A MAX
2300KV, 4S
1kg thrust
ESC
(Electronic Speed Controller)
4S, 30A, LiPo
Brushless Motor
14.8V, 30A MAX
2300KV, 4S
1kg thrust
ESC
(Electronic Speed Controller)
4S, 30A, LiPo
Brushless Motor
14.8V, 30A MAX
2300KV, 4S
1kg thrust
ESC
(Electronic Speed Controller)
4S, 30A, LiPo
Brushless Motor
14.8V, 30A MAX
2300KV, 4S
1kg thrust
ESC
(Electronic Speed Controller)
4S, 30A, LiPo
Pixhawk
5 V
0.5 A
HKPilot Receiver
Telemetry Radio
iA8 8-channel
RC receiver
Ublox Neo-7M GPS

NAVIO2

runs real-time Linux and APM flight stack.

GNSS RECEIVER
with GPS\GLONASS\Beidou and external antenna port.

HIGH RESOLUTION BAROMETER
senses altitude with 10 cm resolution.



DUAL IMU
for unmatched flight performance and redundancy.

EXTENSION PORTS
expose ADC, I2C and UART interfaces for sensors and radios.

RC I/O CO-PROCESSOR
accepts PPM\SBUS input and provides 14 PWM output channels with variable frequencies.

TRIPLE REDUNDANT POWER SUPPLY
with overvoltage protection and power module port for voltage and current sensing.

MAVLink : Ground Stations Overview

Ground Stations

Ground Station

- A Ground station is a software that communicates with the micro-vehicle through a serial or network interface by exchanging MAVLink messages.
- It allows to control the micro-vehicle and monitor its status in real time
- There are several ground stations for different platforms
- More about ground stations

https://ardupilot.org/copter/docs/common-choosing-a-ground-station.html

QGroundControl (QGC)

- Full Vehicle setup support for Atdupilot and PX4 pro powered vehicles
- Developed in C++
- Mission planning for autonomous flight
- Flight map display showing vehicle position, flight track, way points and vehicle instruments
- Video streaming with instrument display overlays
- Flight support for any MAVLink capable vehicle
- Runs on windows, OS X and Linux platform as well as iOS and Android devices.

http://qgroundcontrol.com/

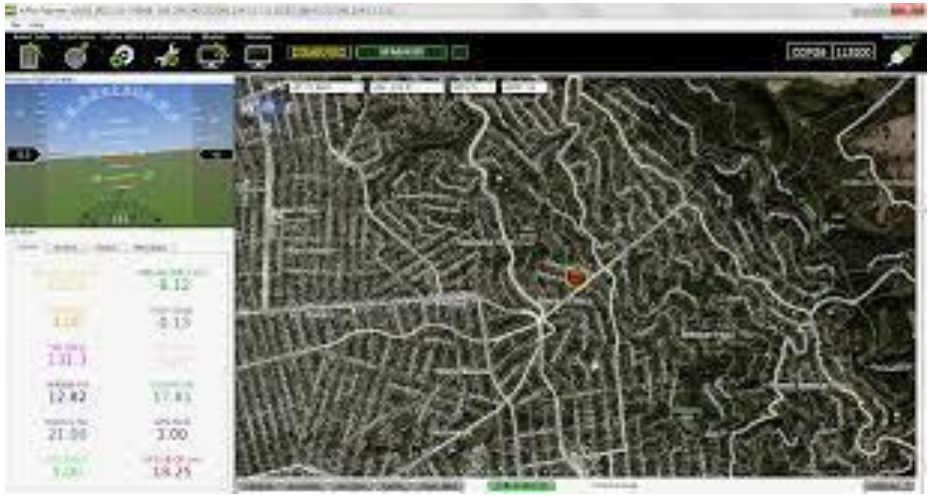
Mission Planner

- The mission planner, created by Michael Cborne
- Platform : windows
- Point and click way point entry, using Google Maps / Bing/Open Street maps/custom WMS.
- Select mission commands from drop down menus
- Download mission log files and analyze them
- Configure APM settings for the airframe
- Interface with a PC flight simulator to create a full hardware in the loop UAV simulator
- See the output from APMs serial terminal.

https://ardupilot.org/planner/

APM Planner 2

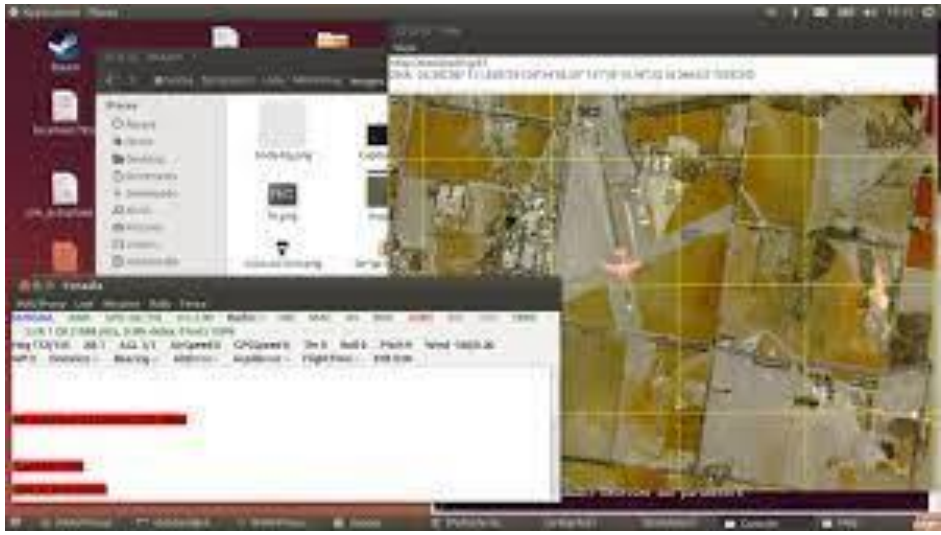
- The best ground station for use on MAC and Linux Platforms
- Platform : Windows, Mac OS X, and Linux
- It has smaller user based and reduced feature set when compared with mission planner
- Intended for developers and enthusiasts.



https://ardupilot.org/planner2/

MAVProxy

- Primarily a command line interface with graphical modules for map and mission planning.
- Platform : Linux and windows
- Written in python and extensible python modules
- It is a command line console based app.
- There are plugins included in MAVProxy to provide a basic GUI.
- Can be networked and run over any number of computers.



https://ardupilot.org/mavproxy/#:~:text=MAVP
%20powerful%20command,provide%20a%20gr
%20interface.

Droid Planner (Tower)

- Platform : Android
- Powerful, streamlined tools for planning autonomous missions
- Parameters settings
- Create missions on the fly
- Quick Actions Buttons

https://github.com/DroidPlanner/

MAVLink messages and commands

MAVLink


```
{"time_boot_ms":5482399,
    "lat":-353632599,
    "lon":1491652293,
      "alt":584030,
    "relative_alt":0,
  "vx":2,"vy":6,"vz":0,
      "hdg":18309,
"sysid":0,"compid":0,"msgid":33}
```

MAVLink messages and commands

MAVLink is a message protocol, designed to exchange information between a UAV and a GCS or a subsystem (ex:GPS,Compass,IMU,Gimbal), using a serial communication channel.

MAVLink

- MAVLink is a very light weight messaging protocol for communicating with drones (and between onboard drone components)
- MAVLink follows a modern hybrid publish-subscribe and point-to-point design pattern
- Data Streams are sent / published as topics while configuration sub protocols such as the mission protocol or parameter protocol are point to point with retransmission.
- Message are defined within XML files.
- Each XML file defines the message set supported by a particular MAVLink system, also referred to as “dialect”.
- The reference message set that is implemented by most ground control stations and autopilots defined in common.xml (most dialects build on top of this definition).

Code generators

- Code generators create software libraries for specific programming languages from these XML message definitions, which can then be used by drones, ground control stations, and other MAVLink systems to communicate.
- C reference implementation is a header only library that is highly optimized for resource constrained systems with limited RAM and flash memory. It is field proven and deployed in many products where it serves as interoperability interface between components of different manufacturers.

Key Features

- **Very efficient.** MAVLink 1 has just 8 bytes overhead per packet, including start sign and packet drop detection. MAVLink 2 has just 14 bytes of overhead (but is a much more secure and extensible protocol). Because MAVLink doesn't require any additional framing it is very well suited for applications with very limited communication bandwidth.
- **Very reliable.** MAVLink has been used since 2009 to communicate between many different vehicles, ground stations (and other nodes) over varied and challenging communication channels (high latency/noise). It provides methods for detecting packet drops, corruption, and for packet authentication.

Key Features contd…

- **Many different programming languages** can be used, running on numerous microcontrollers/operating systems (including ARM7, ATMega, dsPic, STM32 and Windows, Linux, MacOS, Android and iOS).
- Allows up to 255 concurrent systems on the network (vehicles, ground stations, etc.)
- Enables both offboard and onboard communications (e.g. between a GCS and drone, and between drone autopilot and MAVLink enabled drone camera).

Language/Generator List

- The sections below lists MAVLink generators and their associated programming languages.
- MAVLink Project Generators/Languages
- MAVLink organization provides (and supports) the mavgen, mavgenerate and rust-mavlink

Language	Generator	MAVLink v1	MAVLink 2	Signing	Notes
C	mavgen	✓	✓	✓	This is the MAVLink project reference implementation. Generated libraries are also published for both protocol versions.
C++11	mavgen	✓	✓	✓	
Python (2.7+, 3.3+)	mavgen	✓	✓	✓	Python bindings. Library also available on PyPi: pymavlink.
C#	mavgen	✓	✓		
Objective C	mavgen	✓			
Java	mavgen	✓	✓		Dronefleet offers a more idiomatic generated library

JavaScript (Stable)	mavgen	✓	✓	X	Old mavgen JavaScript binding (has known bugs and no test suite).
JavaScript (NextGen)	mavgen	✓	✓	✓	New mavgen JavaScript library. Full test suite, resulting library produces binary compatible output compared to C bindings. Slightly incompatible with previous version, but not hard to migrate.
TypeScript/JavaScript	mavgen	✓	✓	X	TypeScript classes which can be used with node-mavlink.
Lua	mavgen	✓	✓	X	Lua library. Does not support zero trimming of MAVLink 2 messages.
WLua (Wireshark Lua bindings)	mavgen	✓	✓	NA	Allow MAVLink-aware packet inspection in Wireshark. Generated lua scripts should be copied to the Wireshark plugin directory (e.g. **wireshark/plugins/mavlink.lua**).
Swift	mavgen	✓			
Rust	rust-mavlink	✓	✓		Rust MAVLink generated code. Has tests and docs.

Prebuilt MAVLink C Libraries

C MAVLink Source Files (only) are auto-generated for the latest versions of all message specifications/dialects (for both MAVLink 1 and 2):

- c_library_v2 (MAVLink 2)
- c_library_v1 (MAVLink 1)

Using C Libraries explains how to use these libraries.

Packet Format

MAVLink 2 Packet Format
Below is the over-the-wire format for a MAVLink 2 packet (the in-memory representation might differ).

MAVLink v2 Frame(12 - 280)

Byte Index	C version	Content	Value	Explanation
0	`uint8_t magic`	Packet start marker	0xFD	Protocol-specific start-of-text (STX) marker used to indicate the beginning of a new packet. Any system that does not understand protocol version will skip the packet.
1	`uint8_t len`	Payload length	0 - 255	Indicates length of the following `payload` section. This may be affected by payload truncation.
2	`uint8_t incompat_flags`	Incompatibility Flags		Flags that must be understood for MAVLink compatibility (implementation discards packet if it does not understand flag).
3	`uint8_t compat_flags`	Compatibility Flags		Flags that can be ignored if not understood (implementation can still handle packet even if it does not understand flag).
		Packet		

4	`uint8_t seq`	Packet sequence number	0 - 255	Used to detect packet loss. Components increment value for each message sent.
5	`uint8_t sysid`	System ID (sender)	1 - 255	ID of *system* (vehicle) sending the message. Used to differentiate systems on network. Note that the broadcast address 0 may not be used in this field as it is an invalid *source* address.
6	`uint8_t compid`	Component ID (sender)	1 - 255	ID of *component* sending the message. Used to differentiate *components* in a *system* (e.g. autopilot and a camera). Use appropriate values in MAV_COMPONENT. Note that the broadcast address `MAV_COMP_ID_ALL` may not be used in this field as it is an invalid *source* address.
7 to 9	`uint32_t msgid:24`	Message ID (low, middle, high bytes)	0 - 16777215	ID of *message type* in payload. Used to decode data back into message object.
For *n*-byte payload: `n=0` : NA.	`uint8_t`			Message data. Depends on message type (i.e.

For *n*-byte payload: `n=0` : NA, `n=1` : 10, `n>=2` : 10 to (9+n)	`uint8_t payload[max 255]`	Payload		Message data. Depends on message type (i.e. Message ID) and contents.
(n+10) to (n+11)	`uint16_t checksum`	Checksum (low byte, high byte)		CRC-16/MCRF4XX for message (excluding `magic` byte). Includes CRC_EXTRA byte.
(n+12) to (n+25)	`uint8_t signature[13]`	Signature		(Optional) Signature to ensure the link is tamper-proof.

- The minimum packet length is 12 bytes for acknowledgment packets without payload.
- The maximum packet length is 280 bytes for a signed message that uses the whole payload.

•The minimum packet length is 12 bytes for acknowledgment packets without payload.
•The maximum packet length is 280 bytes for a signed message that uses the whole payload.

MAVLink 1 Packet Format

Below is the over-the-wire format for a MAVLink 1 packet (the in-memory representation might differ).

Byte Index	C version	Content	Value	Explanation
0	`uint8_t magic`	Packet start marker	0xFE	Protocol-specific start-of-text (STX) marker used to indicate the beginning of a new packet. Any system that does not understand protocol version will skip the packet.
1	`uint8_t len`	Payload length	0 - 255	Indicates length of the following `payload` section (fixed for a particular message).
2	`uint8_t seq`	Packet sequence number	0 - 255	Used to detect packet loss. Components increment value for each message sent.
3	`uint8_t sysid`	System ID	1 - 255	ID of *system* (vehicle) sending the message. Used to differentiate systems on network. Note that the broadcast address 0 may not be used in this field as it is an invalid *source* address.
4	`uint8_t`	Component	1 - 255	ID of *component* sending the message. Used to differentiate components in a *system* (e.g. autopilot and a camera). Use appropriate values in MAV_COMPONENT.

5	`uint8_t msgid`	Message ID	0 - 255	ID of *message type* in payload. Used to decode data back into message object.
For *n*-byte payload: `n=0` : NA, `n=1` : 6, `n>=2` : 6 to (5+n)	`uint8_t payload[max 255]`	Payload data		Message data. Content depends on message type (i.e. Message ID).
(n+6) to (n+7)	`uint16_t checksum`	Checksum (low byte, high byte)		CRC-16/MCRF4XX for message (excluding `magic` byte). Includes CRC_EXTRA byte.

- The minimum packet length is 8 bytes for acknowledgment packets without payload.
- The maximum packet length is 263 bytes for full payload.

Incompatibility Flags (MAVLink 2)

Incompatibility flags are used to indicate features that a MAVLink library must support in order to be able to handle the packet. This includes any feature that affects the packet format/ordering.

MAVLink does not include information about the message structure in the payload itself (in order to reduce overhead)! Instead the sender and receiver must share a common understanding of the meaning, order and size of message fields in the over-the-wire format.

Messages are encoded within the MAVLink packet:

- The `msgid` (message id) field identifies the specific message encoded in the packet.
- The `payload` field contains the message data.
  - MAVLink reorders the message fields in the payload for over-the-wire transmission (from the order in the original XML Message Definitions).
  - MAVLink 2 truncates any zero-filled bytes at the end of the payload before the message is sent and sets the packet `len` field appropriately (MAVLink 1 always sends all bytes).
- The `len` field contains the length of the payload data.
- A CRC_EXTRA byte is added to the message checksum. A receiver can use this to confirm that it is compatible with the payload message format/definition.

A MAVLink library should notify a bad CRC during decoding if a message specification is incompatible (e.g. the C library mavlink_parse_char() gives a status `MAVLINK_FRAMING_BAD_CRC`).

List of the software's

- Install ubuntu if you don’t have – latest version
- Already windows
  - Virtual box
  - On top of virtual box run ubuntu
- Enable USB in the virtual box
- Make a directory in the ubuntu - from github clone the below link
  - https://github.com/mavlink/c_uart_interface_example
- make clean
- make mavlink_control

UGV : Unmanned Ground Vehicles

- Robot Arm Kinematics and Dynamics
- Manipulator Trajectory planning and Motion Control
- Robot Sensing
- Robotic Operating System
- Robotic Programming Languages

UGV

- An **unmanned ground vehicle** (**UGV**) is a vehicle that operates while in contact with the ground and without an onboard human presence.
- UGVs can be used for many applications where it may be inconvenient, dangerous, or impossible to have a human operator present.
- Generally, the vehicle will have a set of sensors to observe the environment, and will either autonomously make decisions about its behavior or pass the information to a human operator at a different location who will control the vehicle through teleoperation.

UGV

- The UGV is the land-based counterpart to unmanned aerial vehicles and unmanned underwater vehicles.
- Unmanned robotics are being actively developed for both civilian and military use to perform a variety of dull, dirty, and dangerous activities.

UGV

DRDO Daksh ROV - Remotely Operated Vehicle developed by DRDO

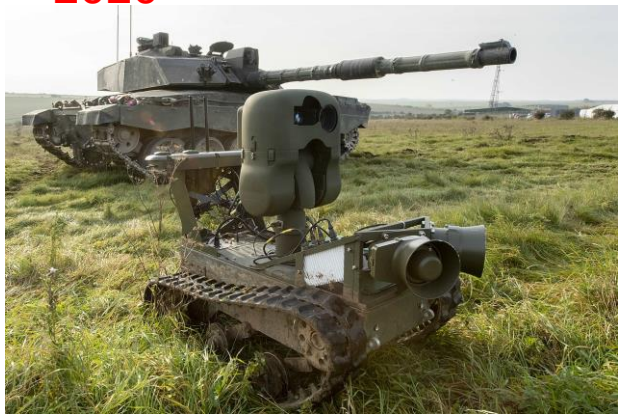
Unmanned hybrid tank

Vasavi Autonomous Tank

Foster-Miller TALON SWORDS units equipped with various weaponry

A US Army XM1219 Armed Robotic Vehicle. Canceled in 2011.

British Army trials of X-2 with existing systems in 2020



NAVYA autonomous bus being trialed on road in Western Australia during 2016

UGV

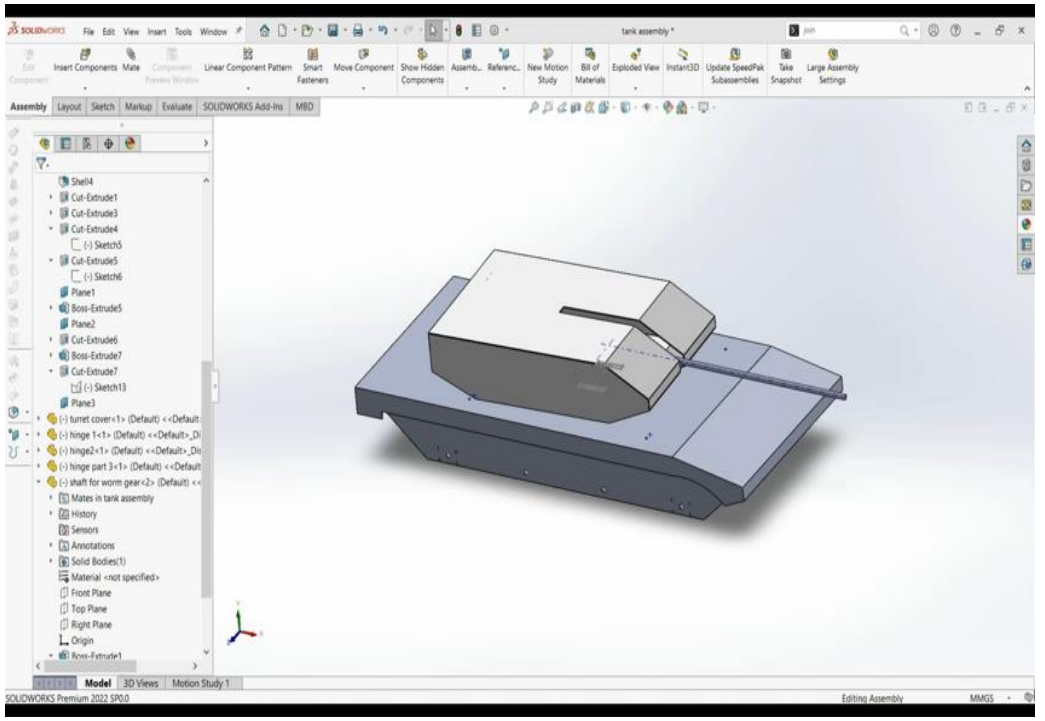
BigDog, a quadruped robot, was being developed as a mule that can traverse difficult terrain.

A Gladiator Tactical Unmanned Ground Vehicle

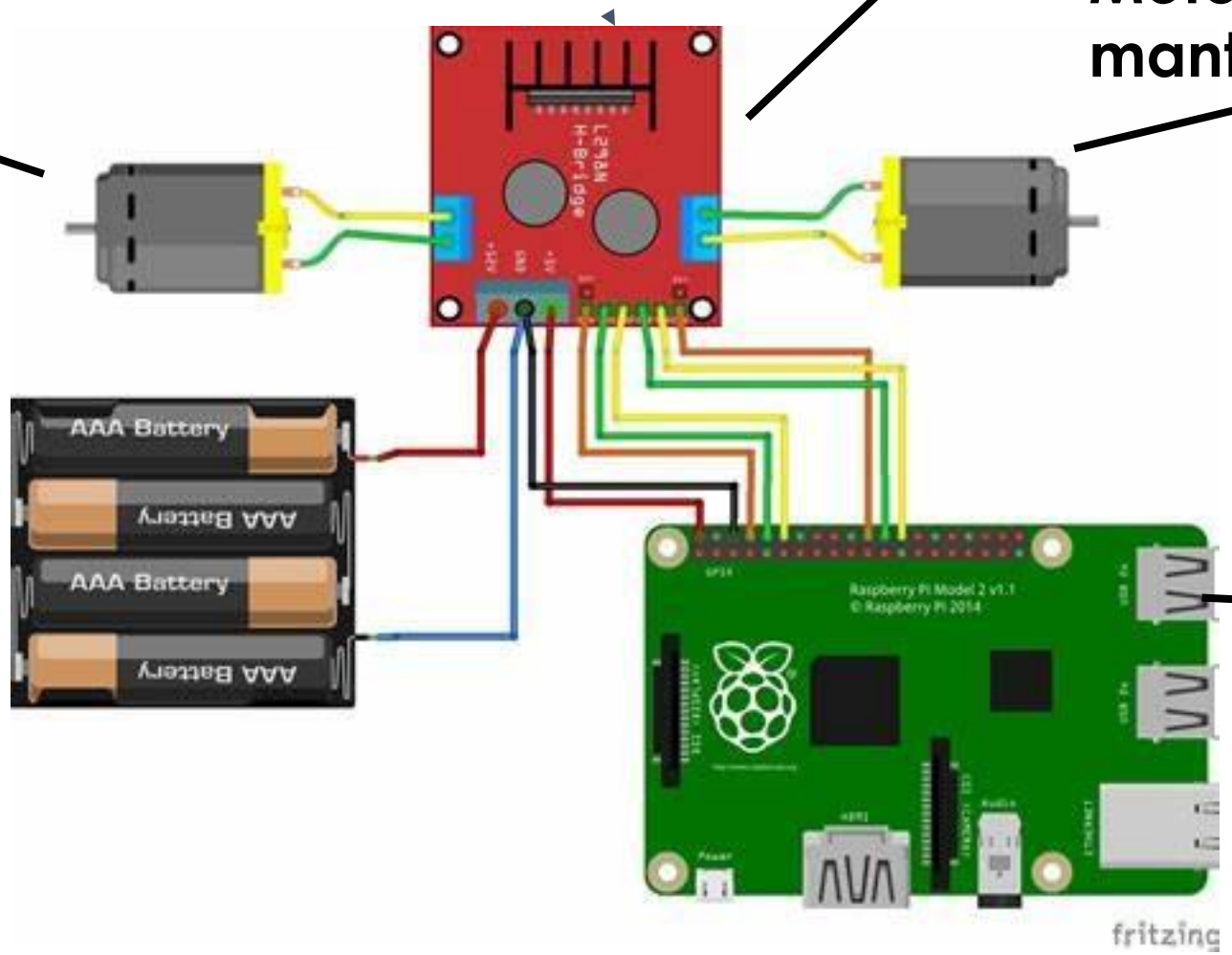
Guardium used by the Israel Defense Forces to operate as part of the border security operations

EuroLink Systems Leopardo B

UGV – Unmanned Autonomous Tank : Vasavi



Architecture and Design



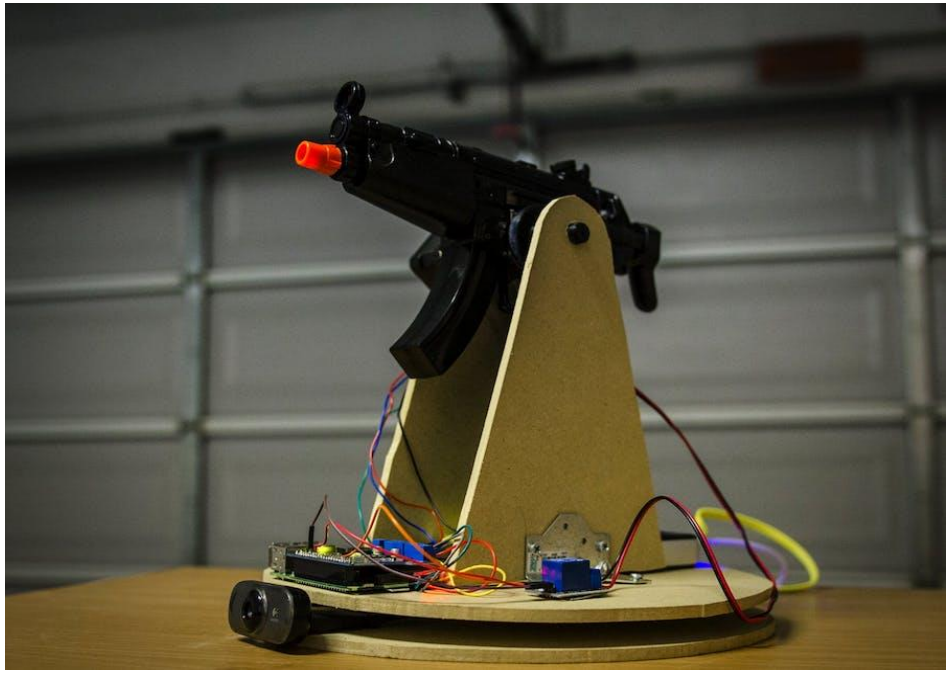
Mantlet Positioning:

Hardware Required:

- Stepper Motor x 2
- Camera Module
- Raspberry Pi
- Anti-Drone Gun

Software Required:

- Open CV
- Trained DeepLearning model for Object Tracking



- One stepper motor to move the Turret to track the object.
- One stepper motor to turn the mantlet.

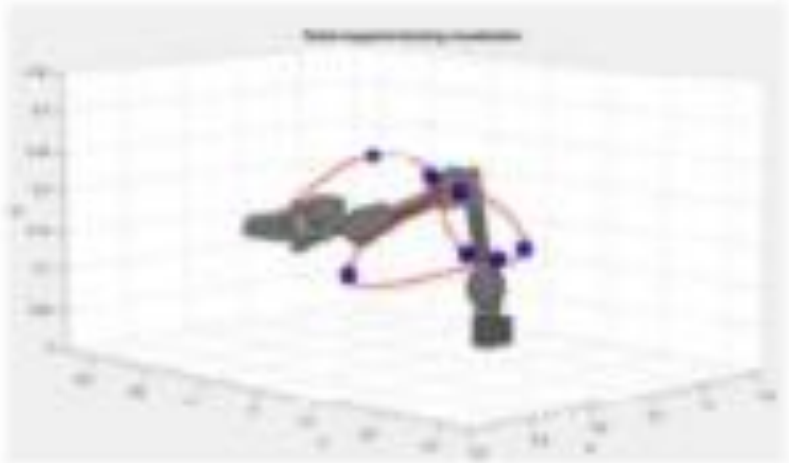
- Using OpenCV to obtain the footage from camera port and giving it as input to the Deep Learning Model. The Output will enable raspberry Pi to operate the Stepper motors and track the object.

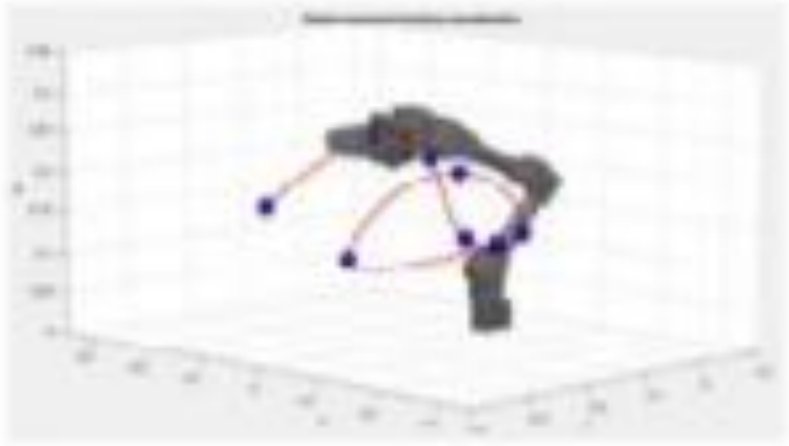
Links:
https://github.com/MertArduino/RaspberryPi-Mertracking/blob/master/mertracking.py
https://github.com/HackerShackOfficial/Tracking-Turret/blob/master/turret.py#L81

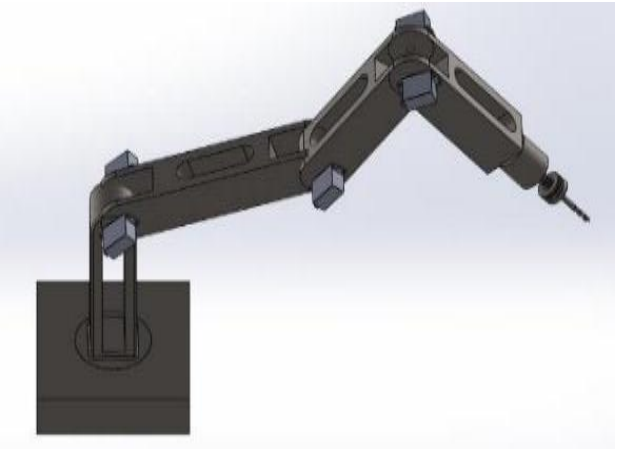
Robotic Arm

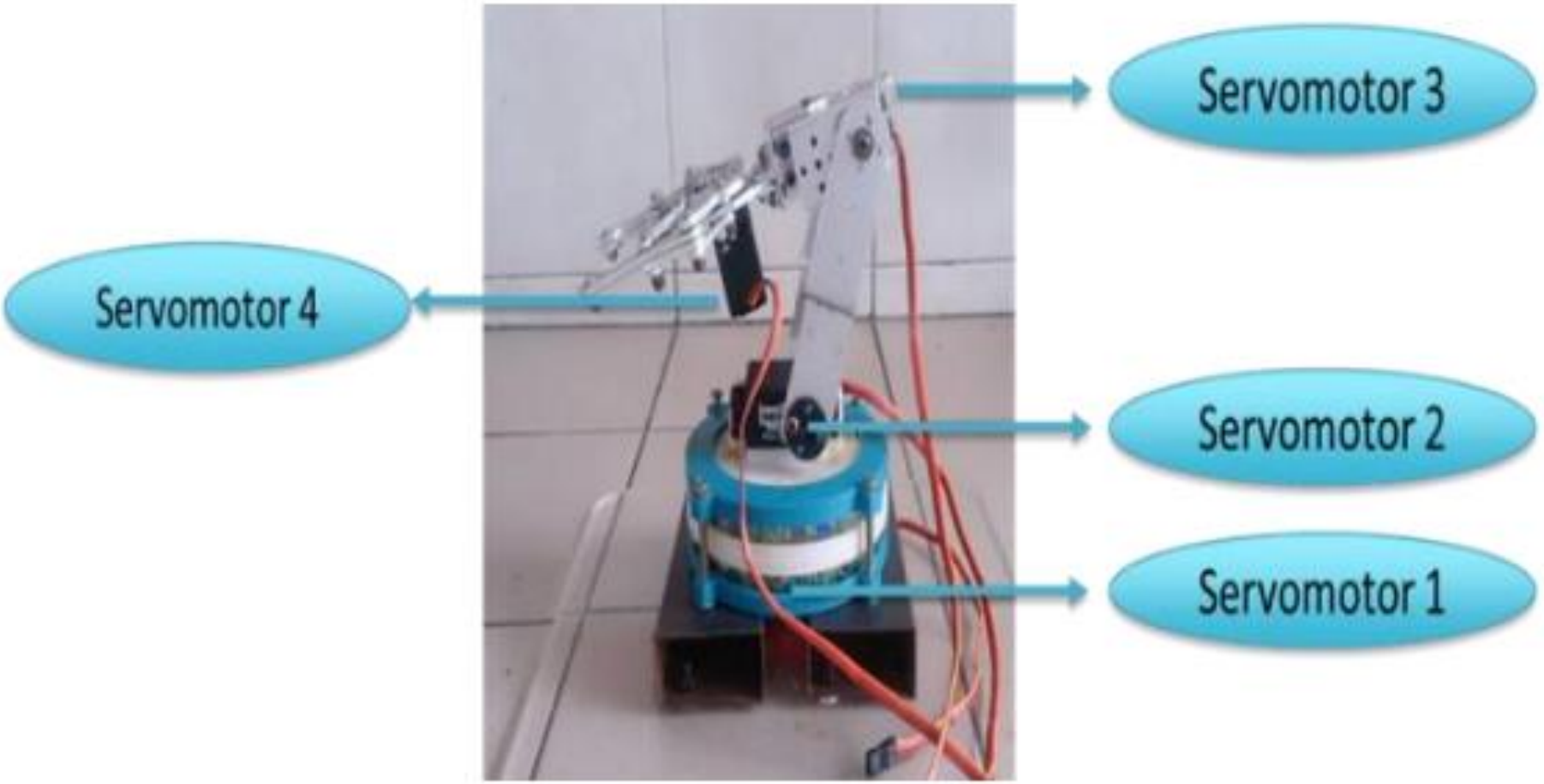
- A robotic manipulator is an electronically controlled mechanism, consisting of multiple segments, which are also commonly referred to as robotic arms.
- These robotic arms perform tasks by interacting with the environment and are extensively used in the industrial manufacturing sector, also have many other specialized applications such as space shuttles to manipulate payloads.
- These Manipulators are emerging trend to use for various industry applications with the revolution of Industry 4.0.
- However, designing of such Manipulator aka robotic arms for variety of industrial tasks is very complex in nature as it involves the mechanical structural design followed by electronic and software programmable components to operate all the components effectively in tandem.

Sample Robotic Arms









Location of each RC servo motor on the robotic arm

Robot Arm Kinematics and Dynamics

- In robotics, **robot kinematics** applies geometry to the study of the movement of multi-degree of freedom kinematic chains that form the structure of robotic systems.
- The emphasis on geometry means that the links of the robot are modeled as rigid bodies and its joints are assumed to provide pure rotation or translation.
- Robot kinematics studies the relationship between the dimensions and connectivity of kinematic chains and the position, velocity and acceleration of each of the links in the robotic system, in order to plan and control movement and to compute actuator forces and torques.
- The relationship between mass and inertia properties, motion, and the associated forces and torques is studied as part of robot dynamics.

Kinematic Equations

- A fundamental tool in robot kinematics is the kinematics equations of the kinematic chains that form the robot. These non-linear equations are used to map the joint parameters to the configuration of the robot system.

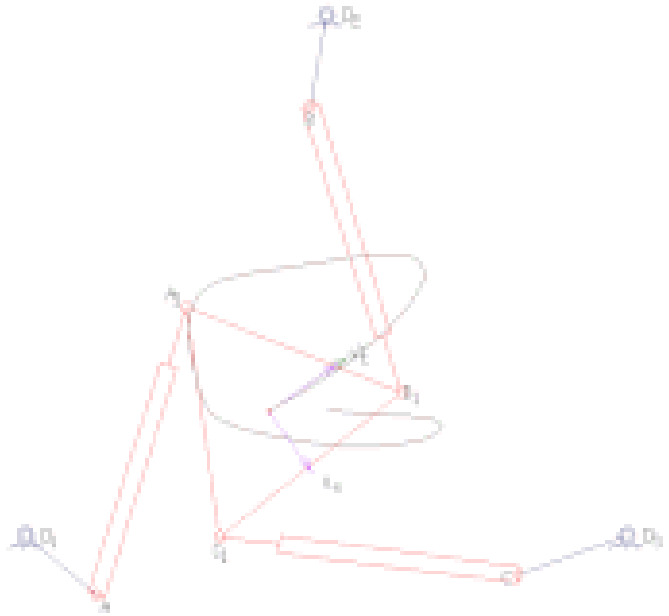
- Forward kinematics uses the kinematic equations of a robot to compute the position of the end-effector from specified values for the joint parameters. The reverse process that computes the joint parameters that achieve a specified position of the end-effector is known as inverse kinematics. The dimensions of the robot and its kinematics equations define the volume of space reachable by the robot, known as its workspace.

Classes of Robots

- There are two broad classes of robots and associated kinematics equations: serial manipulators and parallel manipulators. Other types of systems with specialized kinematics equations are air, land, and submersible mobile robots, hyper-redundant, or snake, robots and humanoid robots.

Forward kinematics[

- Forward kinematics of an over-actuated planar parallel manipulator done with MeKin2D.
- Forward kinematics specifies the joint parameters and computes the configuration of the chain. For serial manipulators this is achieved by direct substitution of the joint parameters into the forward kinematics equations for the serial chain. For parallel manipulators substitution of the joint parameters into the kinematics equations requires solution of the a set of polynomial constraints to determine the set of possible end-effector locations.



Inverse kinematics

Inverse kinematics specifies the end-effector location and computes the associated joint angles. For serial manipulators this requires solution of a set of polynomials obtained from the kinematics equations and yields multiple configurations for the chain. The case of a general 6R serial manipulator (a serial chain with six revolute joints) yields sixteen different inverse kinematics solutions, which are solutions of a sixteenth degree polynomial. For parallel manipulators, the specification of the end-effector location simplifies the kinematics equations, which yields formulas for the joint parameters.

Robot Jacobian

- The time derivative of the kinematics equations yields the Jacobian of the robot, which relates the joint rates to the linear and angular velocity of the end-effector. The principle of virtual work shows that the Jacobian also provides a relationship between joint torques and the resultant force and torque applied by the end-effector. Singular configurations of the robot are identified by studying its Jacobian.
- **Velocity kinematics**
- The robot Jacobian results in a set of linear equations that relate the joint rates to the six-vector formed from the angular and linear velocity of the end-effector, known as a twist. Specifying the joint rates yields the end-effector twist directly.

The **inverse velocity** problem seeks the joint rates that provide a specified end-effector twist. This is solved by inverting the Jacobian matrix. It can happen that the robot is in a configuration where the Jacobian does not have an inverse. These are termed singular configurations of the robot.

Static force analysis

The principle of virtual work yields a set of linear equations that relate the resultant force-torque six vector, called a wrench, that acts on the end-effector to the joint torques of the robot. If the end-effector wrench is known, then a direct calculation yields the joint torques.

The **inverse statics** problem seeks the end-effector wrench associated with a given set of joint torques, and requires the inverse of the Jacobian matrix. As in the case of inverse velocity analysis, at singular configurations this problem cannot be solved. However, near singularities small actuator torques result in a large end-effector wrench. Thus near singularity configurations robots have large mechanical advantage.

Robotic Operating System (ROS)

- The Robot Operating System (ROS) is a set of software libraries and tools that help you build robot applications. From drivers to state-of-the-art algorithms, and with powerful developer tools, ROS has what you need for your next robotics project. And it's all open source.

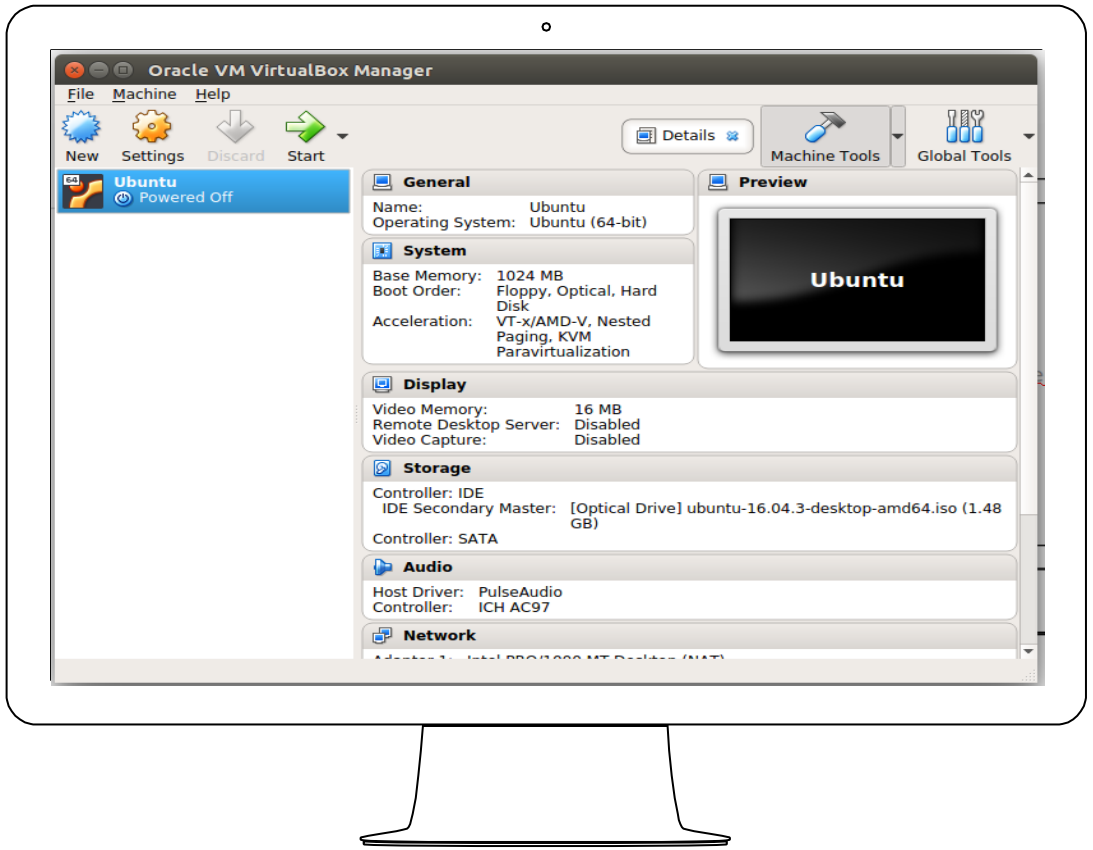
Quick Introduction to ROS

1 Installing Ubuntu 14.04

Quick and painless with Virtualbox

Open Virtualbox

- Install Virtualbox
- Download Ubuntu 14.04
- Install to a USB drive



Time for a demo!

Let's install Ubuntu on a USB drive. I've uploaded a video of this to Youtube here: https://youtu.be/UGl0x2ZT_cI

NOTE: use Ubuntu 14.04 NOT 16.04 as in video!!!!!!!!

Installing ROS Indigo

Full details here - See Chapter 4.1, Installing ROS

https://github.com/StevenShiChina/books/blob/master/ros%20by%20example%20vol%201%20indigo.pdf

2 What is ROS?

Getting started with the concepts

ROS is huge

ROS is an open-source, meta-operating system for humanoid robots

What can ROS do?

- Why ROS?
- Research development:
  - Fast prototyping easier in a simulated world.
  - Lots of packages for sensing, movement, path planning etc.
  - Error free environment for debugging
  - Break bits, not real hardware
- Transferring from simulated robot to real robot takes a bit of effort, but works!

📌 What can ROS do?

- Hardware abstraction
- Low-level device control
- Message passing between nodes
- Sophisticated build environment
- Libraries
- Debugging and Visualization Tools

📌 What are the major concepts?

- ROS packages
- ROS messages
- ROS nodes
- ROS services
- ROS action servers
- ROS topics
- ...and many more!

Packages

- ROS software is organized into **packages**
  - Each package contains some combination of code, data, and documentation

package_name/

package.xml	← describes the package and it's dependencies
CMakeLists.txt	← Finds other required packages and messages/services/actions
src/	← C++, Python code (includes in **include/** folder)
scripts/	← Python scripts for your node
msg/	← ROS messages defined for your node (for topics)
srv/	← ROS services defined for your node (for services)
launch/	← folder contains .launch files for this package

📌 Building/Running

- **Catkin** is the official build system of ROS
  - Catkin combines Cmake macros and Python scripts to provide some functionality on top of Cmake's normal workflow
- Run ROS code
  -
  -

```
$ rosrun <package_name> <script>
$ roslaunch <package_name> <launch_file>
```

📌 ROS Nodes

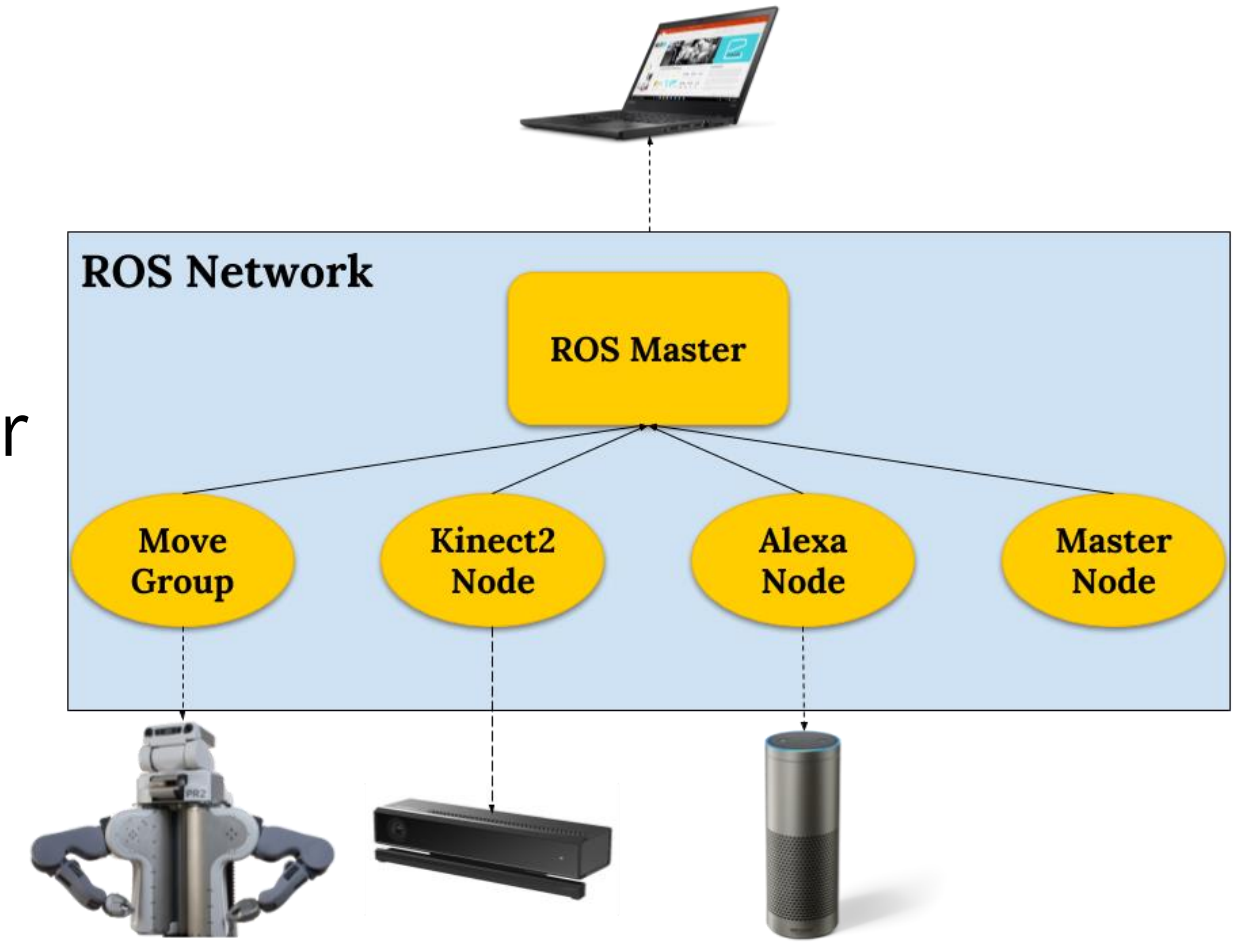
- The ROS framework is component oriented
- Each component is called a node
  - A node is a process
  - Nodes communicate through **topics**, **services**, and **actions**

📌 So what does this mean?

- Hardware talks to drivers, which then talk to nodes, which then talks to ROS
- Nodes can run any software you want as long as it is a language ROS supports

📌 ROS as a framework

- ROS Master sends/receives
- Several nodes at once
- Whole network on your computer



📌 Topics

- Each node can listen on or publish messages to topics
  - Built in message types (std_msgs)
  - User defined messages
  -

Complex.msg

float32 real

float32

imaginary

Example: Rospy Tutorials Talker_Listener

- http://wiki.ros.org/rospy_tutorials/Tutorials/WritingPublisherSubscriber

📌 Services

- A node can provide **services –** synchronous remote procedure calls
  - Request
  - Response

Example: Rospy Tutorials: Server/Client – Add 2 integers, return result

- http://wiki.ros.org/rospy_tutorials/Tutorials/WritingServiceClient

📌 Launch Files

- Automate the launching of collections of ROS nodes via XML files and **roslaunch**
- example.launch:

```
<launch>
  <node name="talker" pkg="rospy_tutorials"
        type="talker.py" output="screen" />
  <node name="listener" pkg="rospy_tutorials"
        type="listener.py" output="screen" />
</launch>

$ roslaunch rospy_tutorials example.launch
```

📌 Launch Files

- You can also pass parameters via launch files
-

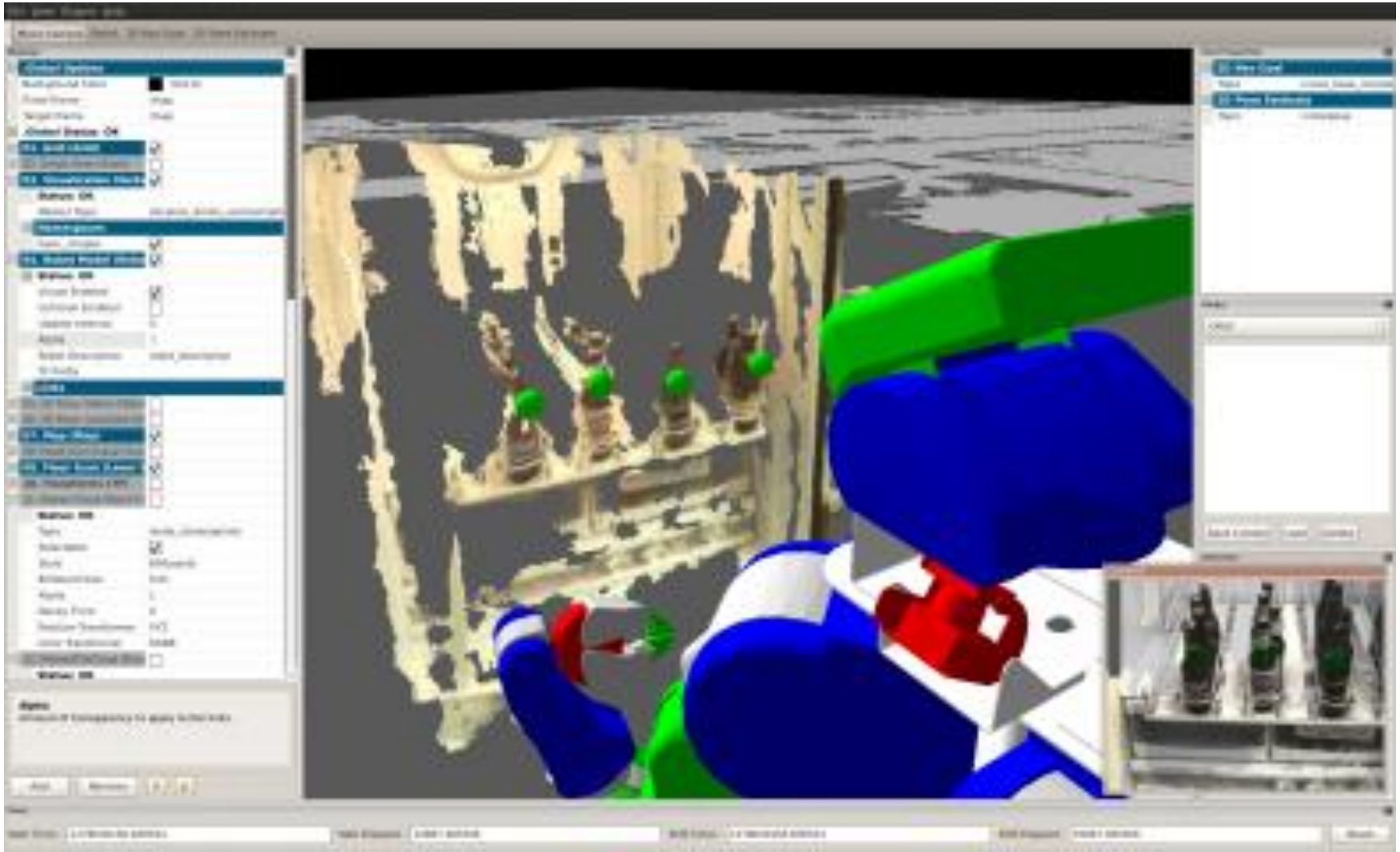
```
<launch>
  <arg name="gui" default="true"/>
  <param name="/use_sim_time" value="true" />
  <include file="$(find gazebo_ros)/launch/
  empty_world.launch">
    <arg name="world_name" value="worlds/willowgarage.world"
    />
    <arg name="gui" value="$(arg gui)" />
    </include>
<include file="$(find pr2_gazebo)/launch/pr2.launch"/>
  <node name="spawn_table" pkg="gazebo_ros" type="
  spawn_model"
   args="-urdf -file $(find humanoids_robots)/
   pr2_gazebo_pick_object/scenario/objects/table.urdf
   -model table -x 2.15 -y 0.5"
   respawn="false" output="screen" />
</launch>
```

📌 Command Line Tools

-

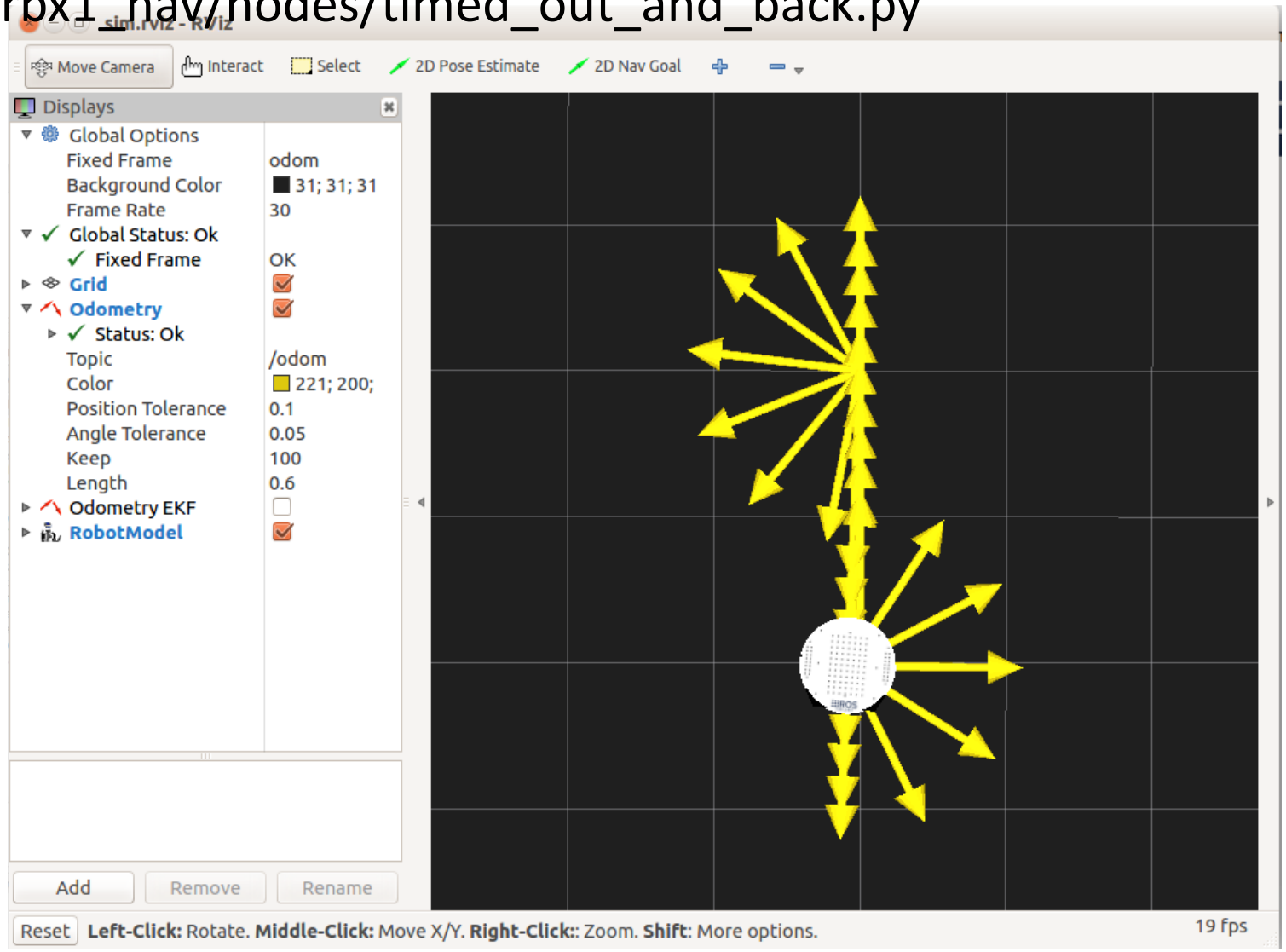
```
$ rosnode list
$ rostopic list
$ rostopic echo
$ rosmsg show
$ rosservice
$ tf viewframes
```

📌 Rviz: Robot Visualization



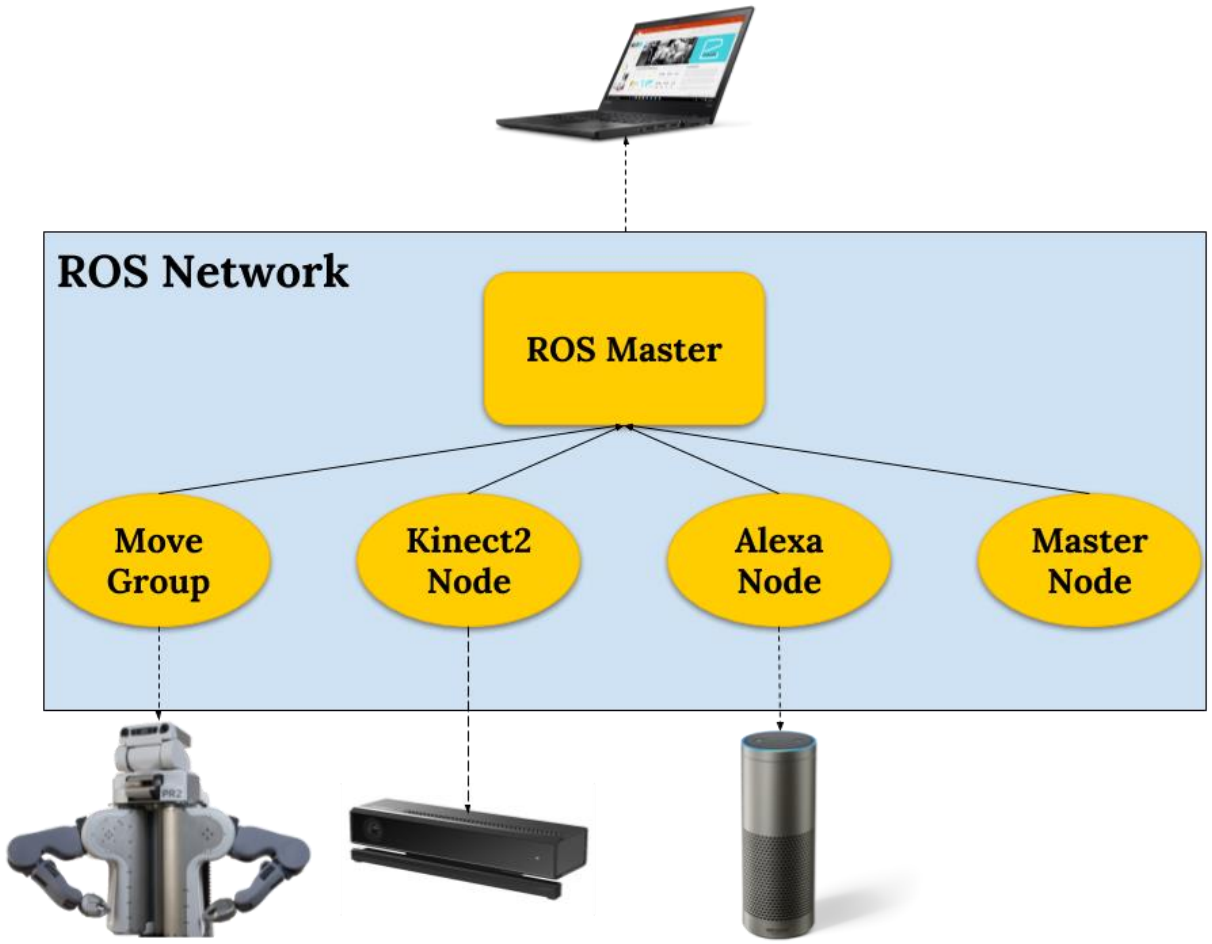
Arbotix Simulator (Homework 1): Turtlebot Simulation

https://github.com/pirobot/rbx1/blob/indigo-devel/rbx1_nav/nodes/timed_out_and_back.py



📌 ROS as a framework cont.

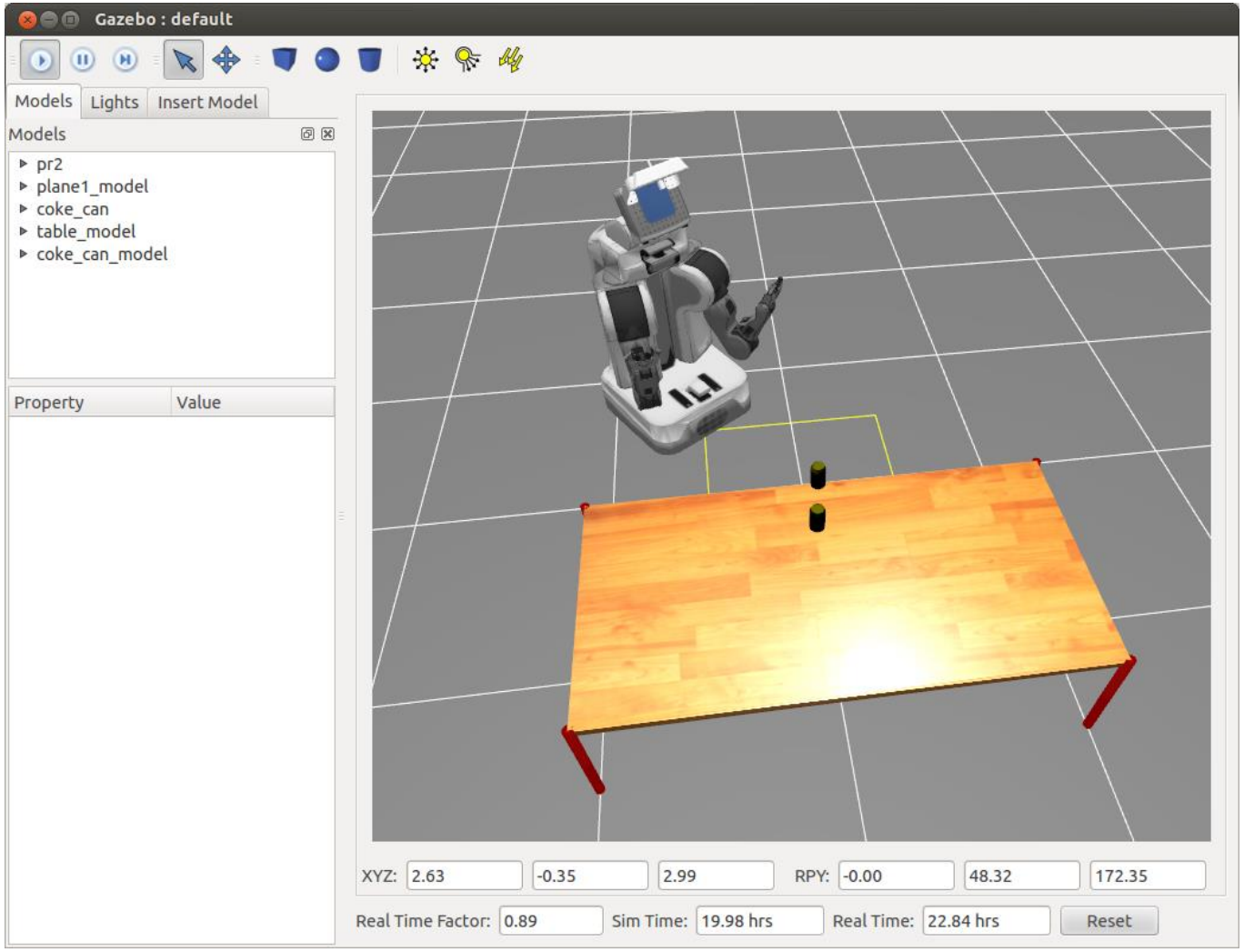
- Kinect2 → /kinect2/images
- Publishes image messages
- What are messages?



📌 Robots in the wild - Problems

- I don’t have a Robot in front of me
- I want to try something that may break my Robot
- Setting up the Robot takes too much time, I want to test changes to my code quickly

Gazebo Simulator



📌 Gazebo Simulator

- Same interface as real robot
- Add/remove 3D items in environment
- Physics engine to simulate effects of motor commands
- Collision detection
- Updated sensor feedback
- Debugging info

📌 Gazebo Demo

- Add object to world
  - Physics
  - Simulated Sensor Output Topics

```
$ roslaunch fetch_gazebo playground.launch
```

Unit – IV

- **Navigation and guidance:** Data Link; Sensors and Payloads: GPS, IMU, Light Detection and Ranging (LiDAR), Imaging cameras, Classification of payload based on applications; Hyper-spectral sensors; Laser Detection and Range (LiDAR); cameras; ultra-sonic detectors; Introduction to navigation systems and types of guidance; Mission Planning and Control, Case studies: Autonomous Obstacle avoidance - Vision, Sonar and LiDAR; Drone Swarms.

Sensors and System for Vehicle Navigation

Autonomous Navigation

- In recent years, vehicle navigation, in particular autonomous navigation, has been at the center of several major developments, both in civilian and defense applications.
- New technologies, such as multisensory data fusion, big data processing, or deep learning, are changing the quality of areas of applications, improving the sensors and systems used.
- Recently, the influence of artificial intelligence on sensor data processing and understanding has emerged. Radar, LiDAR, visual sensors, sonar systems, and other sensors are mounted onboard smart and flexible platforms and on several types of unmanned vehicles in all types of environments.
- These technologies focusing on vehicle navigation may encounter many common scientific challenges. Particularly interesting is autonomous navigation for non-GNSS applications, such as underwater and indoor vehicle navigation.

Sensors and Payloads

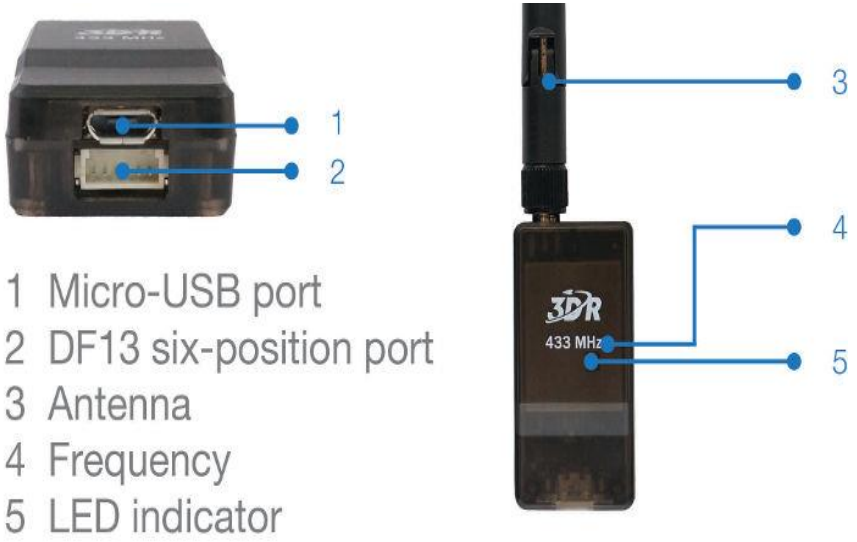
Navigation, Guidance and Control

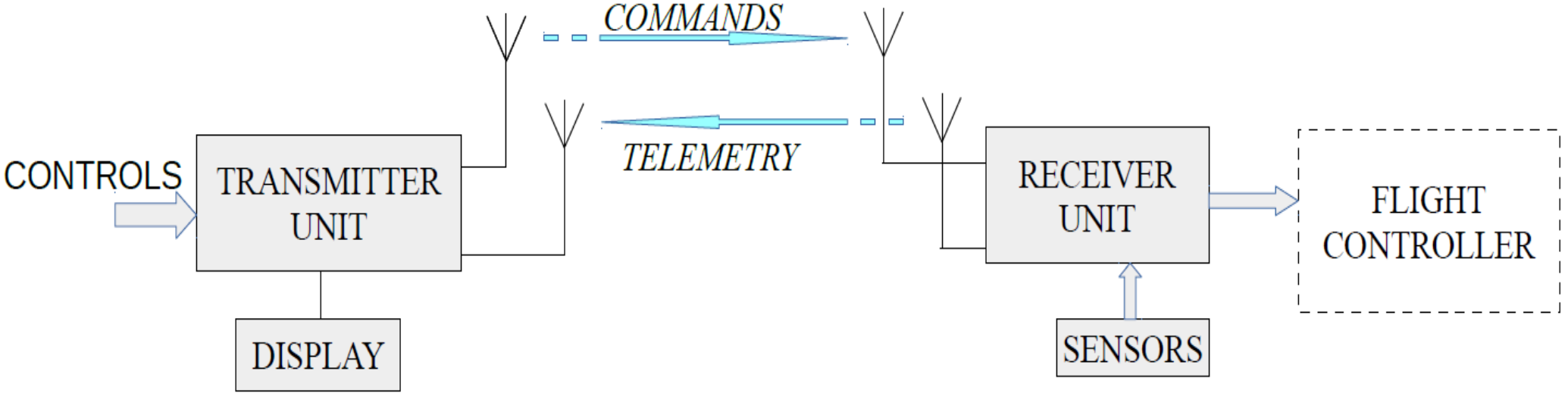
Data Link System

Provides Communication among Air and Ground Modules

Types of communication:

- UHF/VHF
- C Band
- Ku band
- LTE





Navigation and Guidance System

Navigation **Navigation sensor provides the position information to the drone, which is very essential for the drone to operate in auto flight mode.**

Compass **Compass offers the direction to the flight and other IMU sensors like gyro and barometer provides the movement and air pressures to the flight. IMU sensors are part of the Flight controller and GPS and compass is interfaced to the flight controller as an external interface.**

IMU **Inertia Measurement Unit(IMU) sensors like gyro and barometer provides the movement and air pressures to the flight. IMU sensors are part of the Flight controller and GPS and compass is interfaced to the flight controller as an external interface.**

Navigation System - UAV

- A **navigation system** is a computing system that helps to navigate/move.
- Navigation systems are in general on board UAV systems making use of radio or other signal transmission for controlling

Navigation systems – What for used?

Navigation systems may be capable of one or more of:

- Containing maps, which may be displayed in human-readable format via text or in a graphical format
- Determining a vehicle or vessel's location via sensors, maps, or information from external sources
- Providing suggested directions to a human in charge of a vehicle or vessel via text or speech
- Providing directions directly to an autonomous vehicle such as a robotic probe or guided missile
- Providing information on nearby vehicles or vessels, or other hazards or obstacles
- Providing information on traffic conditions and suggesting alternative directions
- Simultaneous localization and mapping
- Acoustic positioning for underwater navigation

Types of Navigation Systems

- **Satellite navigation system**
  - **Global Positioning System, a group of satellites and computers that can provide information on any person, vessel, or vehicle's location via a GPS receiver**
    - **GPS navigation device, a device that can receive GPS signals for the purpose of determining the device's location and possibly to suggest or give directions**
  - **GLONASS, satellite navigation system run by Russia**
  - **Galileo global navigation satellite system**
  - **IRNSS, regional satellite system run by India.**
- **Automotive navigation system - Uses satellite navigation system**
- **Marine navigation systems using sonar (** Sonar is a technique that uses sound propagation to navigate, measure distances, communicate with or detect objects on or under the surface of the water)
- **Inertial guidance system, a system which continuously determines the position, orientation, and velocity (direction and speed of movement) of a moving object without the need for external reference**
- **Robotic mapping, the methods and equipment by which an autonomous robot is able to construct (or use) a map or floor plan and to localize itself within it**
- **XNAV for deep space navigation**

https://www.gps.gov/systems/gnss/#:~:text=Global%20navigation%20satellite%20system%20(GNSS,a%20global%20or%20regional%20basis.

UAV / Drone Navigation – Satellite based

•A group of satellites and computers that can provide information on any person, vessel, or vehicle's location via receiver: a device that can receive signals for the purpose of determining the device's location and possibly to suggest or give directions

- Global Positioning System (GPS): Run by USA
- GLONASS, satellite navigation system run by Russia
- Galileo global navigation satellite system by Europe
- BeiDou: China’s Satellite Navigation System
- IRNSS, Regional satellite system run by India.
- GAGAN : GPS Aided Navigation system by India.
- **Quasi-Zenith Satellite System (QZSS)**

Drone/UAV use : GPS

The Global Positioning System (GPS) is a U.S.-owned utility that provides users with positioning, navigation, and timing (PNT) services. This system consists of three segments: the space segment, the control segment, and the user segment. The U.S. Space Force develops, maintains, and operates the space and control segments.

Space Segment consists of a nominal constellation of 24 operating satellites that transmit one way signals that give the current GPS satellite position and time

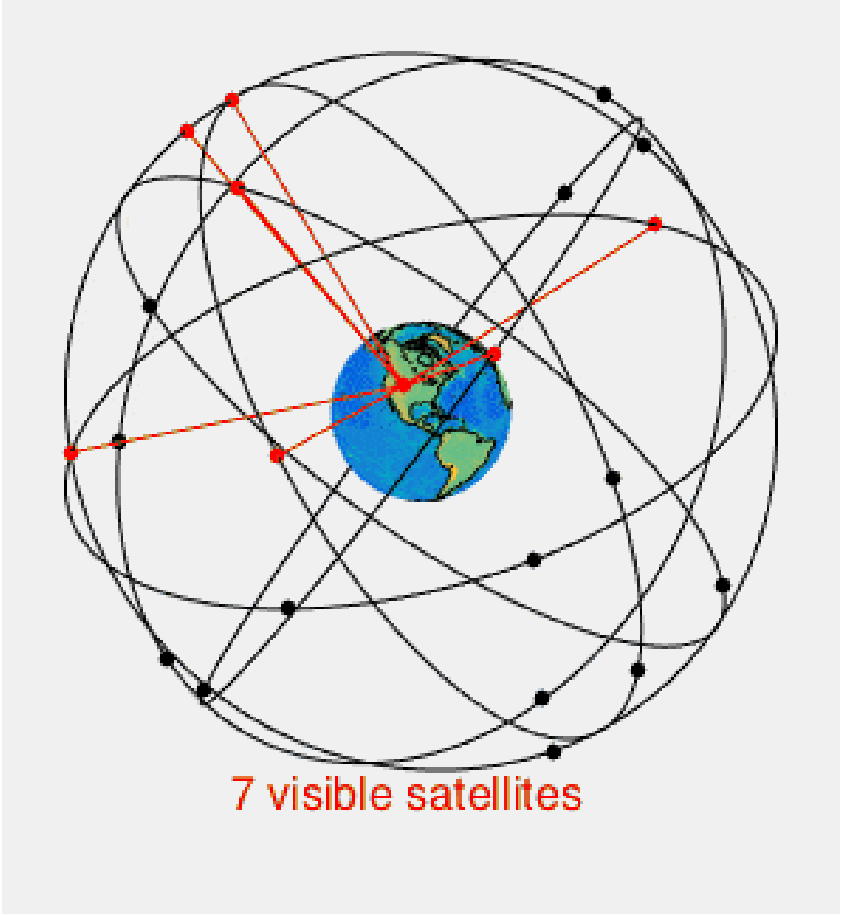
Control segment: consists of world wide monitor and control stations that maintain the satellites in their proper orbits through occasional command maneuvers, and adjust the satellite clocks. It tracks the GPS satellites, uploads the updated navigational data, and maintains health and status of satellite constellation.

User segment: consists the GPS receiver equipment, which receives the signals from the GPS satellites and uses the transmitted information to calculate the users' three dimensional position and time.

GPS Constellation

A GPS receiver calculates its position by precisely timing the signals sent by GPS satellites high above the Earth. Each satellite continually transmits messages that include the time the message was transmitted and the satellite position at the time of message transmission.

The receiver uses the messages it receives to determine the transit time of each message and computes the distance to each satellite using the velocity of light. This measured distance is called pseudorange. Because of errors in receiver clock, the pseudorange has many errors. Since this error is equal for all observations, the effect of that can be destroyed. The calculation of distance should be done for at least four satellites.

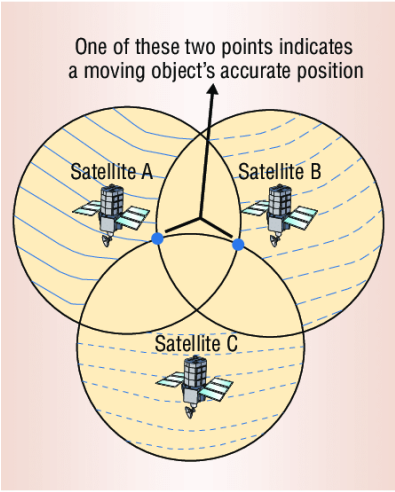


How to find position using GPS

Orbiting the Earth are a number of Global Positioning System (GPS) satellites that can help determine your location on the planet. The concepts behind GPS positioning are very simple, but the application and implementation require amazing precision.

GPS positioning works on two basic mathematical concepts. The first is called trilateration, which literally means positioning from three distances. The second concept is the relationship between distance traveled, rate (speed) of travel and amount of time spent traveling, or:

Distance = Rate × Time



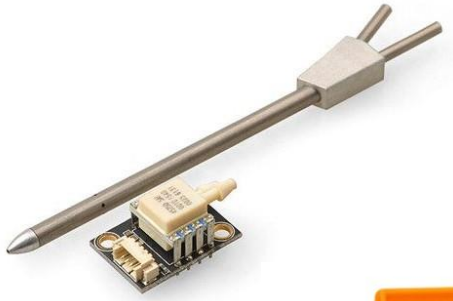
UAV : GPS Receivers

https://www.indexpro.net/en/Category/772

IMU Sensors

GPS & Compass

Airspeed



Distance

Tachometer

https://docs.px4.io/v1.12/en/getting_started/sensor_selection.html

Payload sensors

Cameras – RGB, Thermal and Hyper Spectral

- Drones uses several imaging cameras for data collection
- **RGB :** Red, Green and Blue Filter Cameras to capture Multi spectral Data
- **Thermal Cameras:** for Day and Night Operations
- **Hyper Spectral Cameras:** For Detailed Study and analysis with narrow spectral bandwidth range.

https://www.flir.in/browse/home-amp-outdoor/drone-cameras/

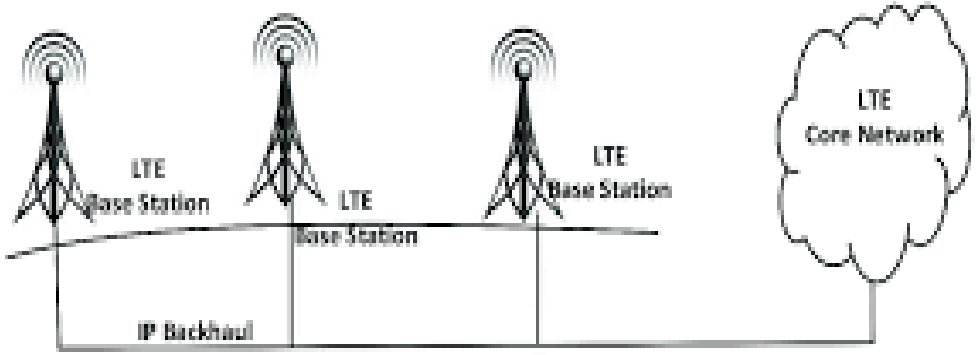
Communication – UAV and Ground

- **Telemetry and Tele Commanding (TTC)**
  - Both Uplink and Downlink
  - Joy stick
- **Payload**
  - Video Transmission
- **Communication**
  - 2.4 GHZ/ UHF / LTE
- **Security**
  - Stream / Block Ciphers (AES – 128 or 256 bit)

LTE Communication

- 4G LTE is an alternative to the radio control system that can eliminate the short distances limitation of a drone.
- In terms of using a drone for search and rescue missions, the 4G network allows a drone to fly several miles away from the controller or pilot.

- The distance improvement, in turn, improves the efficiency of drone applications.
- Replacing the standard radio control feature of drones with a 4G LTE network connection has several advantages in terms of long range communication using cellular networks, higher bandwidth etc.
- Also, the standard security protocols for communication can be used similar to Adhoc communication.

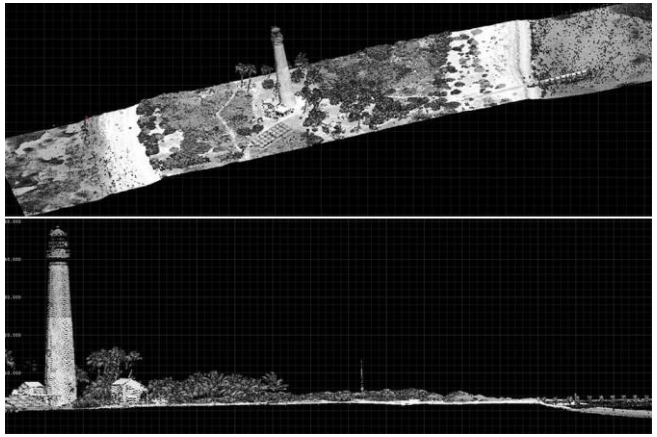


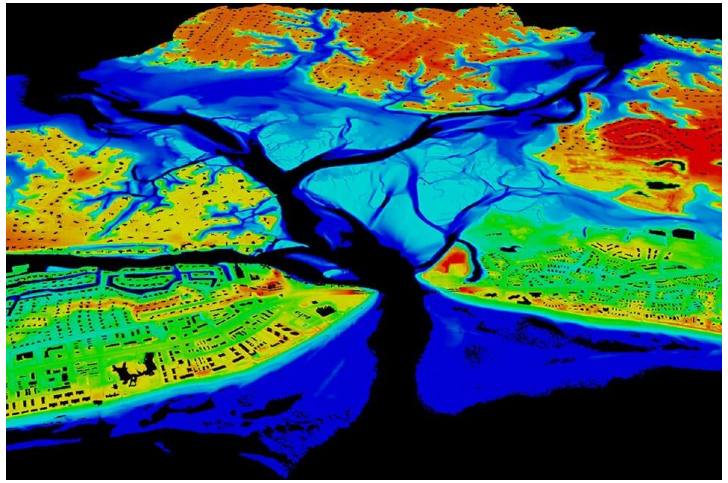
Communication contd…

- Communication systems play a major role in the control of drone with the ground based systems.
- UHF provides a stable radio communication however due to its radio spectrum features it offers short and medium range
- LTE is planned to use as an alternative communication system in the place of UHF due to its cellular features
- SATCOM plays a major communication system for most of the long range Tactical UAV systems
- In all the Drone communication systems security is to be implemented/provided to secure the air and ground based systems.

LiDAR (Light Detection And Ranging)

- Lidar — Light Detection and Ranging — is **a remote sensing method** used to examine the surface of the Earth.





Lidar, which stands for *Light Detection and Ranging,* is a remote sensing method that uses light in the form of a pulsed laser to measure ranges (variable distances) to the Earth. These light pulses—combined with other data recorded by the airborne system — generate precise, three-dimensional information about the shape of the Earth and its surface characteristics.

A lidar instrument principally consists of a laser, a scanner, and a specialized GPS receiver. Airplanes and helicopters are the most commonly used platforms for acquiring lidar data over broad areas. Two types of lidar are topographic and bathymetric. Topographic lidar typically uses a near-infrared laser to map the land, while bathymetric lidar uses water-penetrating green light to also measure seafloor and riverbed elevations.

THANK YOU