1. Give the types of drones

Drones can be categorized into the following six types based on their mission:

- **Combat**: Combat drones are used for attacking in the high-risk missions. They are also known as **Unnamed Combat Aerial Vehicles** (**UCAV**). They carry missiles for the missions. Combat drones are much like planes. The following is a picture of a combat drone:

- **Logistics**: Logistics drones are used for delivering goods or cargo. There are a number of famous companies, such as Amazon and Domino's, which deliver goods and pizzas via drones. It is easier to ship cargo with drones when there is a lot of traffic on the streets, or the route is not easy to drive. The following diagram shows a logistic drone:

- **Civil**: Civil drones are for general usage, such as monitoring the agriculture fields, data collection, and aerial photography. The following picture is of an aerial photography drone:

- **Reconnaissance**: These kinds of drones are also known as mission-control drones. A drone is assigned to do a task and it does it automatically, and usually returns to the base by itself, so they are used to get information from the enemy on the battlefield. These kinds of drones are supposed to be small and easy to hide. The following diagram is a reconnaissance drone for your reference, they may vary depending on the usage:

- **Target and decoy**: These kinds of drones are like combat drones, but the difference is, the combat drone provides the attack capabilities for the high-risk mission and the target and decoy drones provide the ground and aerial gunnery with a target that simulates the missile or enemy aircrafts. You can look at the following figure to get an idea what a target and decoy drone looks like:

- **Research and development**: These types of drones are used for collecting data from the air. For example, some drones are used for collecting weather data or for providing internet.

We can also classify drones by their wing types. There are three types of drones depending on their wings or flying mechanism:

- **Fixed wing**: A fixed wing drone has a rigid wing. They look like airplanes. These types of drones have a very good battery life, as they use only one motor (or less than the multiwing). They can fly at a high altitude. They can carry more weight because they can float on air for the wings. There are also some disadvantages of fixed wing drones. They are expensive and

require a good knowledge of aerodynamics. They break a lot and training is required to fly them. The launching of the drone is hard and the landing of these types of drones is difficult. The most important thing you should know about the fixed wing drones is they can only move forward. To change the directions to left or right, we need to create air pressure from the wing. We will build one fixed wing drone in this book. I hope you would like to fly one.

- **Single rotor**: Single rotor drones are simply like helicopter. They are strong and the propeller is designed in a way that it helps to both hover and change directions. Remember, the single rotor drones can only hover vertically in the air. They are good with battery power as they consume less power than a multirotor. The payload capacity of a single rotor is good. However, they are difficult to fly. Their wing or the propeller can be dangerous if it loosens.
- **Multirotor**: Multirotor drones are the most common among the drones. They are classified depending on the number of wings they have, such as tricopter (three propellers or rotors), quadcopter (four rotors), hexacopter (six rotors), and octocopter (eight rotors). The most common multirotor is the quadcopter. The multirotors are easy to control. They are good with payload delivery. They can take off and land vertically, almost anywhere. The flight is more stable than the single rotor and the fixed wing. One of the disadvantages of the multirotor is power consumption. As they have a number of motors, they consume a lot of power.

2. How can we differentiate between drones?

We can also classify multirotor drones by their body structure. They can be known by the number of propellers used on them. Some drones have three propellers. They are called tricopters. If there are four propellers or rotors, they are called quadcopters. There are hexacopters and octacopters with six and eight propellers, respectively.

The gliding drones or fixed wings do not have a structure like copters. They look like the airplane. The shapes and sizes of the drones vary from purpose to purpose. If you need a spy drone, you will not make a big octacopter right? If you need to deliver a cargo to your friend's house, you can use a multirotor or a single rotor:

- The **Ready to Fly** (**RTF**) drones do not require any assembly of the parts after buying. You can fly them just after buying them. RTF drones are great for the beginners. They require no complex setup or programming knowledge.
- The **Bind N Fly** (**BNF**) drones do not come with a transmitter. This means, if you have bought a transmitter for your other drone, you can bind it with this type of drone and fly. The problem is that an old model of transmitter might not work with them and the BNF drones are for experienced flyers who have already flown drones with safety, and had the transmitter to test with other drones.
- The **Almost Ready to Fly** (**ARF**) drones come with everything needed to fly, but a few parts might be missing that might keep it from flying properly. Just kidding! They come with all the parts, but you have to assemble them together before flying. You might lose one or two things while assembling. So be careful if you buy ARF drones. I always lose screws or spare small parts of the drones while I assemble. From the name of these types of drones, you can imagine why they are called by this name. The ARF drones require a lot of patience to assemble and bind to fly. Just be calm while assembling.

3. Explain the drone frames

Basically, the drone frame is the most important to build a drone. It helps to mount the motors, battery, and other parts on it. If you want to build a copter or a glide, you first need to decide what frame you will buy or build. For example, if you choose a tricopter, your drone will be smaller, the number of motors will be three, the number of propellers will be three, the number of ESC will be three, and so on. If you choose a quadcopter it will require four of each of the earlier specifications. For the gliding drone, the number of parts will vary. So, choosing a frame is important as the target of making the drone depends on the body of the drone. And a drone's body skeleton is the frame. In this book, we will build a quadcopter, as it is a medium size drone and we can implement all the things we want on it.

If you want to buy the drone frame, there are lots of online shops who sell ready-made drone frames. Make sure you read the specification before buying the frames. While buying frames, always double check the motor mount and the other screw mountings. If you cannot mount your motors firmly, you will lose the stability of the drone in the air. About the aerodynamics of the drone flying, we will discuss them soon. The following figure shows a number of drone frames. All of them are pre-made and do not need any calculation to assemble.

You will be given a manual which is really easy to follow:

You should also choose a material which light but strong. My personal choice is carbon fiber. But if you want to save some money, you can buy strong plastic frames. You can also buy acrylic frames. When you buy the frame, you will get all the parts of the frame unassembled, as mentioned earlier.

The following picture shows how the frame will be shipped to you, if you buy from the online shop:

If you want to build your own frame, you will require a lot of calculations and knowledge about the materials. You need to focus on how the assembling will be done, if you build a frame by yourself. The thrust of the motor after mounting on

the frame is really important. It will tell you whether your drone will float in the air or fall down or become imbalanced. To calculate the thrust of the motor, you can follow the equation that we will speak about next.

If *P* is the payload capacity of your drone (how much your drone can lift. I'll explain how you can find it), *M* is the number of motors, *W* is the weight of the drone itself, and *H* is the hover throttle % (will be explained later). Then, our thrust of the motors *T* will be as follows:

$$T = \frac{\frac{1}{H} \times (P + W)}{M}$$

The drone's payload capacity can be found with the following equation:

$$P = T \times M \times H - W$$

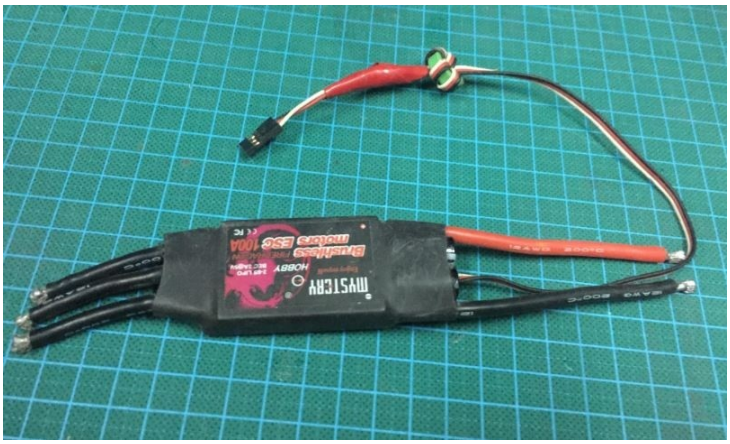
4. Types of motors used for drones

There are a few types of motors that are use to build drones. But as the drone needs to be thrust in the air to float, we should use some powerful motors. The cheap, lightweight, small, and powerful motors used in drones are **Brushless DC motors** (**BLDC**). For small drones, we do not use BLDC motors, but instead use small DC gear motors.

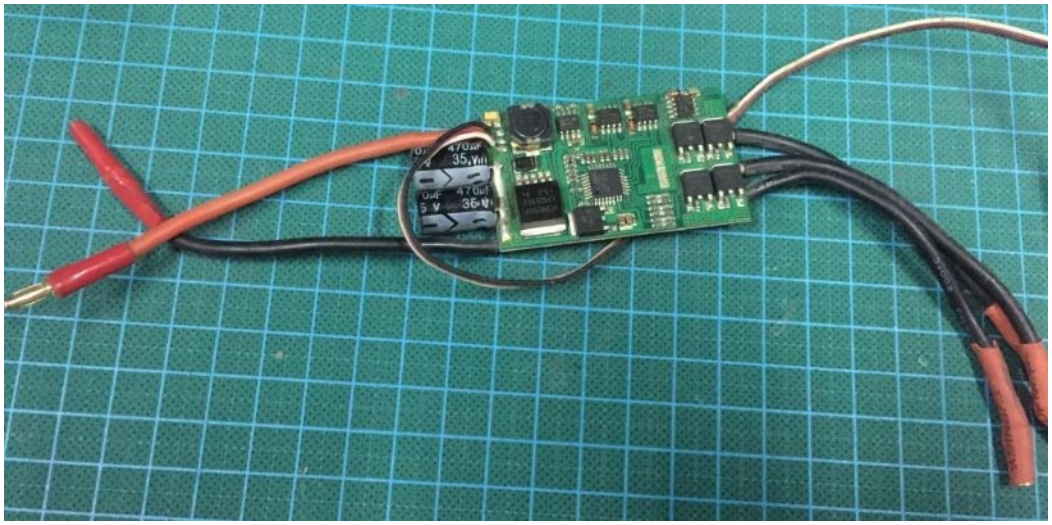
5. Several types of speed controllers

You cannot control the speed of motors of your drone unless you use speed controllers. They enable you to control the voltage and current of the motors and hence control the speed, which is the first priority to move the drone one place to another, after floating in the air. You need to increase and decrease the speed of motor(s) to move the drone forward, backward, left, or right.

The connection between the controller board of the drone and ESC and the battery/power distribution board will be shown in Chapter 2, *Assembling Your Drone:*



Refer to the following circuit diagram:



Flight control board

This is one of the most important things to control the drone from the ground. There are a number of flight control boards on the market. Some of them are open source and some of them are not. The following list has the most famous and top-rated flight controllers:

KK 2.0

- CC3D
- Naze32
- KISS
- ArduPilot
- Vector
- 3DR Pixhawk
- DJI Nava M

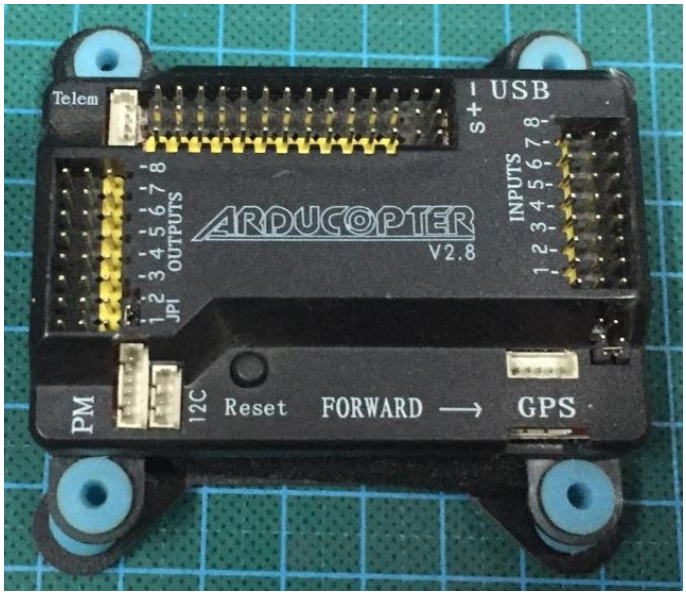
- LUX flight controller
- DJI A3

In this book, we will use ArduPilot, as it is cheap and it is best for copters. It also covers our book title. The following picture of some flight controllers.

The ArduPilot is one of the best flight controllers for drones because of the following reasons:

- It has a free, open source autopilot framework supporting different types of drones
- It supports hundreds of 3D waypoints
- It allows you to do the autonomous take-off, landing, and camera control
- It has 4 MB onboard data-logging memory
- It has a built-in hardware fail-safe processor
- It has full mission scripting
- It is really simple to set up

The following picture is an ArduPilot:



6. Radio transmitter and receiver

What the transmitter does is it sends a signal to the receiver. The receiver receives this signal and does according to the command from the transmitter. Since the drone floats in the sky, it needs to send signals to command the drone to move or do something. So we need the transmitter and receiver. There are lots of transmitters and receivers out there. The transmitter looks like a remote, which is controlled by the drone pilot and the receiver is connected to the flight controller. So, if the pilot gives commands from the transmitter to the drone, the drone receives it via the receiver and the flight controller processes the signal and does, as per the command of the pilot.

7. Mention about the battery

A drone is useless without a battery. All motors, flight controllers, radio, and processing require power. But it is not a wise decision to use the heavy battery to fly your drone because most of the energy will be spent on the thrust of a drone to fly. So, we need to choose light but powerful batteries. In a drone, we usually use lithium polymer batteries.

Choosing the right battery for the drone is one of the most critical things. Before choosing batteries for your drone, keep the following things in mind:

- Battery size and weight

Battery discharge rate
Battery capacity
Battery voltage

- Battery connectors

You can easily calculate the continuous current output of the battery with the help of the following formula. If the current is *I*, battery capacity is *C*, and discharge rate is *D*, then the instantaneous current draw is $I = C X D$. So always choose the highest capacity batteries, depending on the size and weight of the batteries. The LiPo, or Lithium Polymer battery has cells. Say you have three cells in your battery and each of them is 5,000 mAh and the discharge rating is 10 C.

So, the current draw is:

$$I = 5.0 \times 10 = 50\ A$$

Here, we converted mAh to Ah by dividing by 1,000. The following figure shows LiPo batteries for a drone:

8. Give the types of propellers used.

The following figure shows different types of propellers:

9. Mention about the Battery adapters/chargers

Battery chargers are required to recharge your LiPo batteries. There are lots of LiPo chargers on the market. Always buy according to the manual of your battery. My suggestion is to buy a balance charger, which allows your battery to be charged with balance for all the cells.

10. Write about the connectors used.

Connectors are the most important things for the power and other parts of the drone. If the connectors lose your drone, it might meet with an accident. So, buy connectors with special care according to the drone's power distribution system. You need to solder connectors properly with the batteries, ESCs, and other parts of the drone.

You need to buy bullet connectors, XT60, or T-plug connectors and use them where they suit:

11. Explain in detail about assembling Your Drone

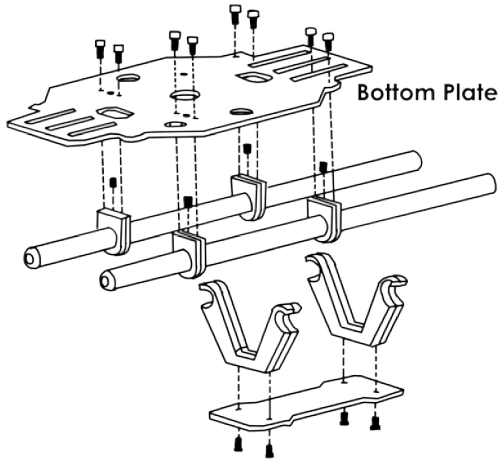
a. Assembling the frame

The assembly of the drone frame requires a lot of patience and you are once again advised to follow the instruction manual when doing it for the first time. For reference, in this book, we will assemble a drone frame from HobbyKing (S500). Inside the box, you will get the items displayed in the following picture:

These items are listed here:

- Four frame arms (two blue and two black, if you buy the blue color)
- Four leg plates
- Two rods for the base mount
- One top plate
- One bottom plate for mounting the power lines
- And a lot of screws

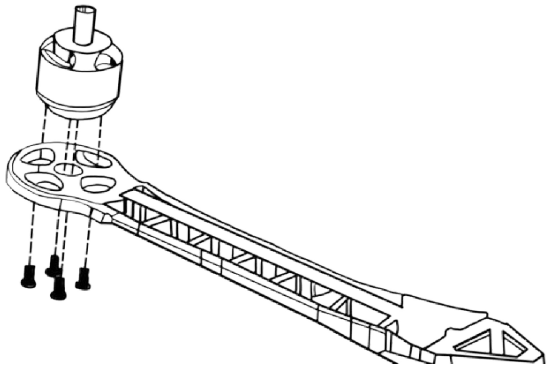
Firstly, see the following figure and connect the parts as shown here:



Use a proper screwdriver to tighten the screws. Do not keep any screws untightened or tilted.

b. Connecting the motors

To connect the motors, you need to place the motor on the frame arm and attach the screws, as shown in the following figure, making sure you tighten the screws as much as you can, without breaking the frame arm:



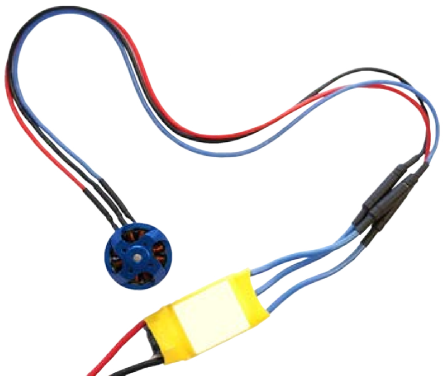
The BLDC has three wires coming out of the motor. We need to solder the bullet

connector to them to be connected to the ESC. Now, connect the other three motors to the frame arms.

c. Connecting the ESC

Connecting the ESCs is one of the most important tasks in building a quadcopter or any other drone. You can buy four pieces of ESCs or a four-in-one ESC. I suggest you use a four-in-one ESC, which is lighter and easy to use.

If you use single a ESC, connect the wires of the motor to the ESC, as shown in the following figure. The connection of the motor and ESC do not matter because the wires are for changing the phase only:



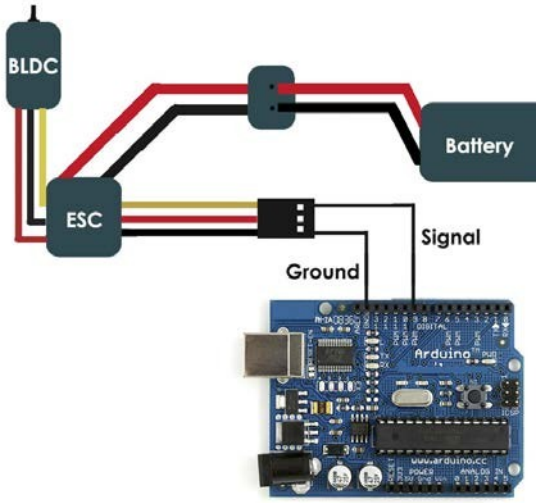
If you use a four-in-one ESC, the connections are also almost the same. I personally use Multirotor four-in-one ESCs from EMAX. Refer to the following figure to know how to connect the wires.

Inside the box

Back View

After connecting the wires

The *3 x 4 = 12* wires will be connected to the four BLDC motors. Let's look at the wire configuration. The single **ESC** has eight wires. Three wires go to the **BLDC** motor, two wires go to the power unit, the remaining three wires are for signal, ground, and power. If you configure the ESC with an Arduino, the diagram will be as follows:



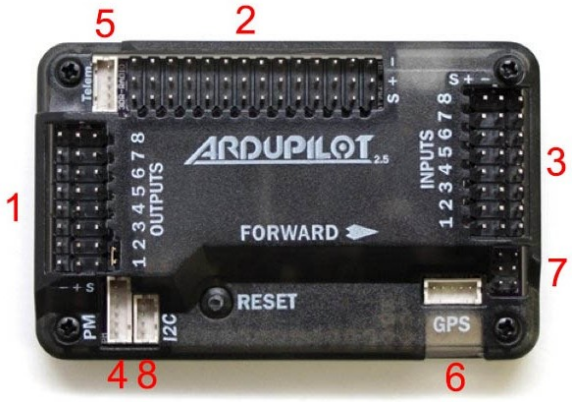
For connecting to the Arduino, we do not use the power of the ESC cable. The main wire is the signal cable. We use this to simulate the core of the motor to turn. If you want to use an Arduino as the main control board of your drone, then you need to connect all of your ESCs, as shown in the previous figure, choosing different signal pins.

d. Connecting the ArduPilot

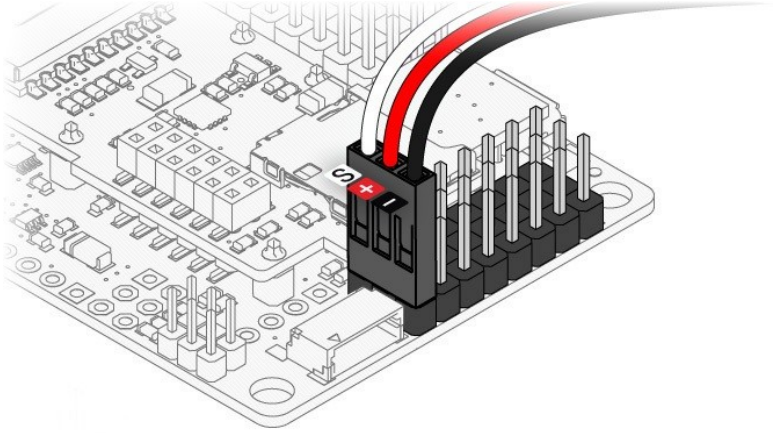
Connecting the ArduPilot is one of the most important tasks for flying and controlling the drone. The ArduPilot is basically the brain of our drone. It enables the drone to control the movement, the camera, and the other sensors connected to it. We will connect a radio to our ArduPilot later, so that we can control the drone remotely.

Let's look at the ArduPilot first.

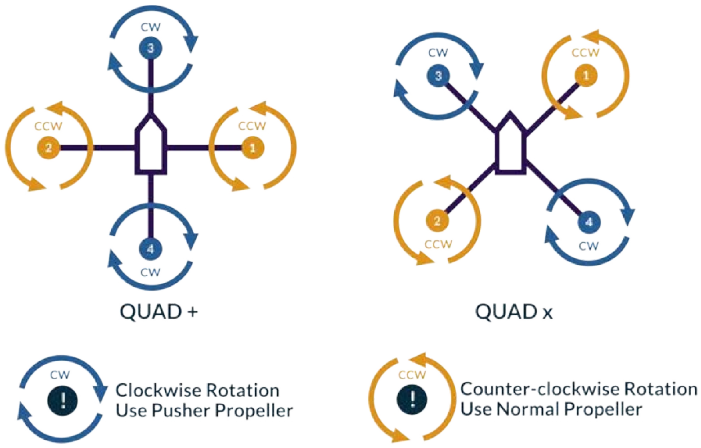
The ArduPilot has basically eight types of pins, as marked in the following diagram. Lets look at the what the uses of them are:



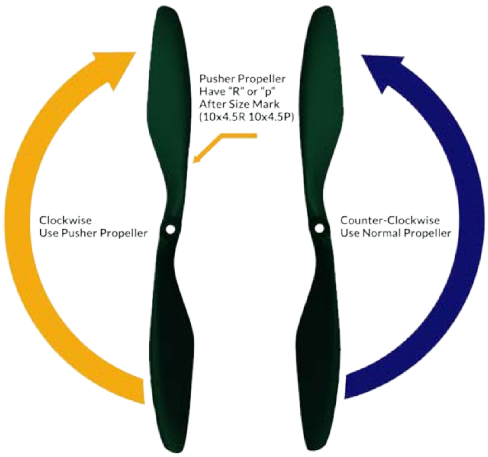
These pins are used to add the ESCs signal or output cables. Remember the three wires of the ESCs? They will be connected here. You can add up to eight ESCs in the ArduPilot. The rightmost pin of a row is used for the signal cable, the middle pin is for the power cable, and the leftmost pin is the ground pin. See the following diagram for a better idea:



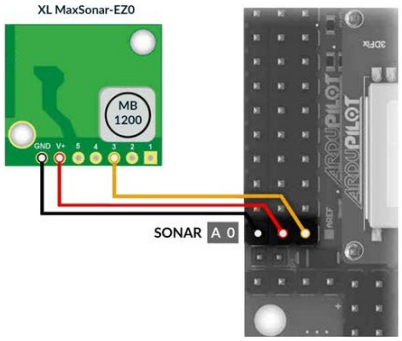
There are some specific rules in which orientation you should connect your ESCs output wired to the ArduPilot. As we are building a quadcopter, the opposite motor of a motor should rotate in the same direction (clockwise or anticlockwise). There are two kinds of quadcopters depending on the shape of the quad's hands. One is plus shaped and the other is cross shaped. The motors must be connected, as shown in the following figure:



If you are unsure about which is the pusher propeller and which is the normal propeller, you may remember, pusher propellers rotate clockwise and normal propellers rotate anticlockwise. In the propellers, you may find **P** or **R** after size marks. P or R denotes that it is a pusher propeller, otherwise it is a normal propeller. See the following diagram for more clarification:



These pins are used for the analogue sensor pins of the drone (for example, sonar sensor, airspeed sensor, voltage or current sensor, and so on). We will learn more about the pins' uses in the coming chapters. Refer to the following diagram to know how the sonar sensor is connected to the ArduPilot:



These pins are used for connecting the remote or radio or RC receiver to the ArduPilot. We will learn how to connect a radio to the ArduPilot in a while.

These pins are used for adding a common power module to the ArduPilot board. The common power module provides 5.37V and 2.25 Amp power supply. It allows the ArduPilot to work more accurately with a lot of sensors connected to it, such as the compass sensor. A small variation of the power will cause the direction to go wrong. If there is a small variation of power, the direction will be changed because of that, as it will trigger the compass inaccurately. So, using a common power module to these pins is needed if you want good accuracy of the measurements from the sensors connected to and built into the ArduPilot.

In these pins, a Bluetooth device or an RF transreceiver (having 900 MHz or 2.4 GHz frequencies) is connected, so that it can communicate to the ground control station. It can cover up to 50 m or the radius, depending upon the model of the Bluetooth device. Using these pins, we can easily connect our drone to the computer without any wire and download flight data without any wire. We will discuss more about these devices in the coming chapters.

The ArduPilot does not come with a **Global Positioning System** (**GPS**) sensor or GPS Receiver. So, we have to connect an external GPS sensor or GPS Receiver, so that your drone can do some complex things related to GPS (for example, position holding and waypoints navigations). If you want to make your drone fully autonomous, this port is needed. We will discuss more about this port in the coming chapters.

If you look closely, you can see that a jumper is there. If the jumper is connected, the board will use the internal compass; if not, then the board will use the external compass. It is not necessary to remove the jumper unless you buy an external compass, or the on-board compass is not good. The compass is used to

define the direction of the drone while flying. So, this is important for a smart drone.

This is a multifunction MUX port (in network communication systems, MUX is the short form of multiplexing, it can send multiple signals at the same time to another device), which is used to connect to the computer via UART0, UART2, I2C, or OSD. If you do not alter anything, the default setting is OSD.

e. Connecting the radio

Connecting the radio to the ArduPilot is one of the easiest things. You can choose a short-range or long range radio. I would suggest you use the 3DR radio. It is easy to set up via the ArduPilot software and is easy to use. The radio will be directly connected to the ArduPilot and will send signals to the receiver connected to the PC. This is required when your drone is fully assembled and you need to calibrate the drone without any external wire connected to it. The radio will be connected via the Telem port.

f. Connecting the RC receiver and transmitter

The RC transmitter allows the drone pilot to control the drone wirelessly. It would be silly to fly a drone with a wire in the sky, so connecting the **Radio Controlled** (**RC**) transmitter is a must for a drone.

The RC transmitter sends signals and the receiver receives the signal. The RC transmitter is also known as the TX and the RC receiver is known as the RX. Before discussing about the RC receiver, let's know the transmitter first.

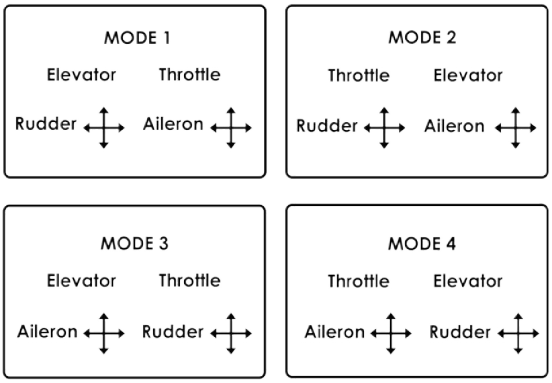
Basically, the transmitter is the remote controller of the drone. Before choosing a perfect transmitter, you need to consider a few things, such as the number of channels, the modes, and the frequency technology, and so on.

I would suggest you buy a good transmitter because this is the least thing that is destroyed while flying drones.

The number of channels gives you the ability to control how many individual

actions you can control of the drone. Let's make it clear. Say a drone needs a few actions from the transmitter to be received by the receiver and enables the controller to execute the commands, such as the throttle, pitch, or yaw. These three actions will require three individual channels of the transmitter. To fly a drone properly, you need to choose at least a four-channel transmitter. Then you can easily control the throttle, roll, pitch, and yaw. We will know about them later. There are transmitters with a higher number of channels. There, you can also control the auxiliary channels.

The mode of the transmitter is also needed to be taken care of. There are usually four kinds of modes of a transmitter. They are known as numerically mode 1, mode 2, mode 3, and mode 4:



The previous diagram shows all four modes of the transmitter.

We see more details of the transmitter in Chapter 3, *Preparing Your Drone for Flying*, as it is a must to fly a drone.

Connecting the RC receiver to the ArduPilot is one of the hardest parts. Remember the input pins of the ArduPilot? We will connect our RC receiver there. Usually, the pin configuration is as follows:

- **Pin 1**: Roll/aileron
- **Pin 2**: Pitch/elevator
- **Pin 3**: Throttle
- **Pin 4**: Yaw/rudder
- **Pin 5**: Auxiliary channel 1 (for example, mode switch)
- **Pin 6**: Auxiliary channel 2

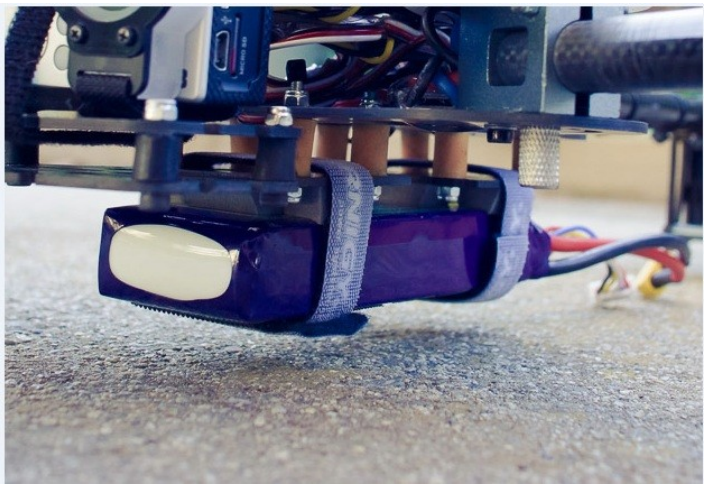
On the RC receiver, you will see the pin number/channel number. Just connect to the ArduPilot as you desire. Alternatively, you can follow the previous pin

configuration to avoid complexity. See the following figure, if you need more clarification:

We will configure the transmitter as the setup of the RC receiver. I will be using a six-channel displayless transmitter from Fly Sky (model: FS-CT6B) throughout the book. You can use another transmitter, if you want. Both the transmitter and the receiver come with the single box, so you don't have to worry about the configuration now.

g. Connecting the battery

We are almost finished assembling our quadcopter. Now the last part is to connect the battery. As we are using, S500 frame, it comes with a board, where you can connect the battery by soldering the connectors to the board and later just plug the battery while flying the drone. You can use zipties to lock the battery under the body of the copter, as shown in the following figure:



To summarize the connections, let's recap. The BLDC motors will be connected to the four-in-one ESC (or four individual ESCs). The ESCs will be connected to the ArduPilot's output pins (only signal cables, 5V, and ground cables). The power cables of the ESC (or ESCs) will be directly connected to the battery (in our case, we have connected on the body plate first and later used connector pins to be connected to the battery.). The radio and the RC receiver will be connected to the Telem pins and input pins of the ArduPilot, respectively. That is it for a simple quadcopter.

h. Binding transmitter to the receiver

Binding is really important. Without proper binding, you cannot configure the drone's actions properly. You can bind your transmitter with at least the following things. Note that the binding process differs from the RC Receiver's model to model. The steps of the most common method is as follows:

A binding cable

- An ESC
- A servo motor (or you can use a BLDC motor too, but it is too risky; I don't recommend it while binding the transmitter to the receiver)
- A battery

Firstly, connect the binding cable to the RC receiver on the BAT pin, as follows. In some models, the BAT pins might be known as B/VCC:

Now, take an ESC and connect the signal, 5V, and ground pin to the CH1 pins.

Connect the battery to the ESC. The ESC will make a beeping sound and the RC receiver will show an LED blinking. Now, turn on the RC transmitter after installing the battery to it. Press and hold the Bind Range Test button until the LED of the RC receiver stops blinking. Do this process again if the LED does not stop blinking. You can try switching off and on the transmitter, and then finally disconnect the binding cable from the RC receiver.

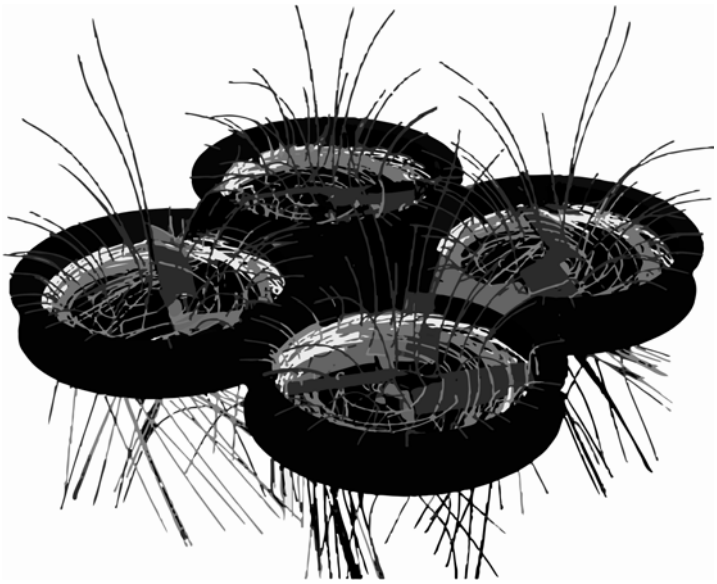
If you have successfully bound the transmitter to the receiver, you can now test your transmitter by connecting a few servo motors to the RC receiver and moving the throttle, roll pitch, or yam of the transmitter. If successfully bound, the servo will rotate, as per the rotation of the knobs or gimbal of the transmitter.

12. Know the aerodynamics needed for flying a drone

Just imagine your drone produces infuriating noises. Do you know why? It is because of the BLDC motors and the propellers, right? But theoretically, drones should not make that much loud a noise. Can you imagine why? Because of the proper uses of the aerodynamics inside a drone. The physics for flying a drone is really necessary to be known by all the drone pilots because, if you cannot master the air, your drone will not fly properly. Refer to the following figure to

get a rough idea on how air is effected by the propellers of the drone.

The figure is taken from the NASA website. They simulated the aerodynamics via computers:



So, basically a drone (specially quadcopters) has two pairs of propellers (two in a clockwise direction and another two in a anticlockwise direction). The speed of each motor is individually controlled to control the movement of the drone. We need to think about two things for flying a drone, the torque, and the thrust. So what are these?

Well, a torque is nothing but a twisting force that tends to cause rotation.

Alternatively, we can say, in physics, the capability of rotating an object around a fixed axis is known as torque. It is symbolized as τ (Tau). Mathematically, torque is the vector product of force (*F*) and the distance (*r*) of the axis. So, we can write:

$$\vec{\tau} = \vec{F} \times \vec{r}$$

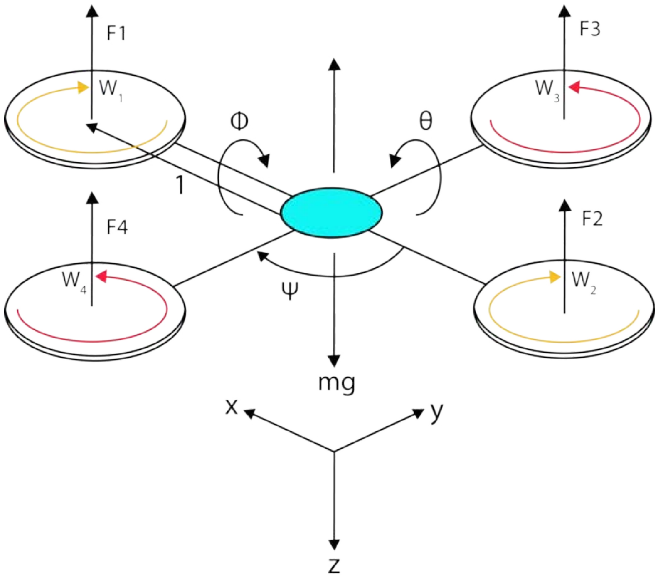
or

$$\tau = Fr\sin\theta$$

Where θ is the angle between the force and the distance from the center of the axis. We will know more about torque a little bit later. Let's speak about thrust now. Thrust is simply pushing something suddenly or with propulsive force. In physics, thrust is defined as the forward force that impels it to go faster or keeps it going in the intended direction. Mathematically, thrust is the product of pressure (*P*) and area (*A*).

So, we can say, *Thrust = P x A*.

We use a small control board to control the drone. The control board has a few sensors that provide the necessary signals to move the propellers at the proper speed, and in the right direction. Inside the control board, there is a gyroscope and accelerometer that provide the orientation information of the drone. The RC receiver gets a signal from the RC transmitter and sends it to the microcontroller of the control board, and the ESCs connected to the microcontroller are then controlled to provide the necessary speed. The following figure shows the forces and movements of the quadcopter:



Mathematically, Thrust (*T*) will be proportional to the square of the angular velocity (*w*) of the propellers. The thrust is perpendicular to the *Z* direction of the drone. So, we can write:

$$T \propto \omega^2$$

or

$$T = K_a \times \omega^2$$

Here, K_a is a constant. As the propellers rotate and create a thrust in the *Z* direction, there must be an opposite force in the drone. Let the movement be M_i, which will also be equal to the right side of the previous equation.

Therefore, $M_i = K_b \times \omega^2$, where K_b is a constant. We used two different constants because the force might be slightly decreased or increased, due to the friction of the air particle or the dust.

The opposite pair of propellers are M_x and M_y. According to the definition of movements, if the distance of the center of the drone and a propeller is *l*, we can write the following equations:

$$M_x = |F_1 - F_2| \times l$$

$$M_y = |F_3 - F_4| \times l$$

The weight of the drone $W = mg$. The weight always acts in the direction of the quadcopter. From Newton's second law of motion, we know:

$$force = mass \times acceleration\ (linear)$$

The torque can be defined with the help of inertia as follows:

$$Torque = Inertia \times acceleration\ (angular)$$

Hovering

To hover the quadcopter, the weight of the drone must be equal to all the upward forces of the propellers, where the movement must be equal to zero:

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

For flying, the upward force must be greater than the weight. So, if we subtract the mass of the drone from the upward forces of the propellers, we will get the equation of motion of the drone flying. Let's say it is D_*. So, we can write:

$$D_* = F_1 + F_2 + F_3 + F_4 - mg$$

Rising or climbing or taking off

To fly the drone over the ground, the equations will be changed as follows:

$$mg < F_1 + F_2 + F_3 + F_4$$

$$D_* = F_1 + F_2 + F_3 + F_4 - mg > 0$$

Dropping or descent or falling

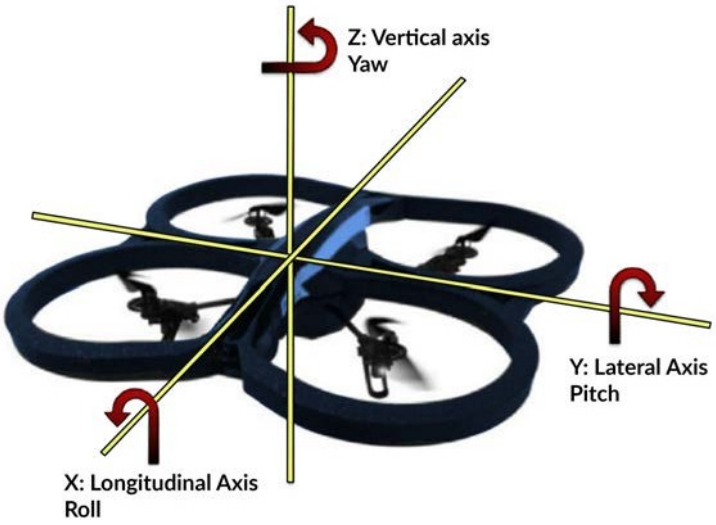
The drone will not fly if the weight is more than the upward forces or $D_* < 0$:

$$mg > F_1 + F_2 + F_3 + F_4$$

$$D_* = F_1 + F_2 + F_3 + F_4 - mg < 0$$

Yaw

Yaw is the motion of the drone in the *xy* plane, or the horizontal plane as shown in the previous figure. The opposite kind of pair of propellers will create reaction movements. If the sum of all the movements of each propeller is equal to each other, then there is no yaw motion. But if there is difference movements between any pair of propellers, there will be yaw motion, and the drone will move, as shown in the following figure:



If the drone simply rotates along the *z* direction, it is known as a yaw motion. This will occur, if there is a stable upward force and the propeller forces are as follows:

$$I_{zz} = M_1 + M_2 + M_3 + M_4$$

Here, M_1, M_2, M_3 *and* M_4 are the movements of the propellers.

Pitch and roll

The pitch is the rotation of the drone in the *Y* direction, while the rotation along the *X* direction is known as roll or vice versa, depending on the front of the drone. This will happen if one pair of opposite propellers provide, thrust more than the other two propellers.

This is the simple aerodynamics of a quadcopter.

Saving your drone from crashing

You hardly can prevent your drone from crashing and breaking some of the propellers, or even the body. The damage will depend on the crash and the height it falls from. It will also depend on the surface your drone lands on or crashes.

The best place to fly a drone is open fields where there are no trees or any electrical wires hanging. Do not uplift the drone with high throttle at first, increase the throttle gradually. We will know about safe throttling later. For now, you can use the Stabilize mode first. Just remember, throttle is nothing but speeding up the velocity of the propellers of the drone. If you are a beginner, my suggestion is to learn how you can levitate your drone above the ground. Once you master hovering the drone, increase the throttle a little bit. Practice more with that high. Increase the height gradually. Then, start doing the yawing to the drone. It will rotate the drone in parallel to the surface, up above the ground. Once you can yaw your drone, you can then pitch it to a forward direction or a backward direction. Then, learn how to roll the drone. From my personal experience, when I flew my first DIY drone, I became nervous about which knob/gimbal does the throttle or which one does roll/pitch and crashed my drone in the ground after 2 seconds of flying, and ended up breaking it. I can remember that the drone suddenly gave a thrust to the ground and flew about 10/12 feet up. Then I did something more horrible. I rotated the roll and pitch gimbal of the transmitter 360 degrees and the drone fell from above and landed on a rock.

So, never get nervous, do not press any button of the transmitter without knowing its task; also be careful about the weather conditions, obstacles, and air flow. That is how you can avoid most of the crashes.

13. What are the things to Check before flying

There are a few things you must check before flying a drone. Here is a checklist for flying a drone safely:

- Check all the connections
- Check transmitter and receiver bindings
- Check the battery charge and voltage
- Check whether all the propellers are attached tightly
- Check all the motor mountings
- Check all the screws
- Check the balance of the drone to see if any side is heavier than the other
- Always unplug the battery after flying; only attach the battery few a seconds before flying
- Check any obstacles outside
- Keep children away from the flying area
- Keep a distance from the drone while you first throttle it
- Turn on autopilot and return to the home/launch feature if they are available
- Do not fly a drone with unbalanced propellers or broken propellers
- Always wear safety glasses
- Maintain security protocols

14. Things to check the security protocols for flying a drone outside

There are some rules by the government of the country where you fly things in the sky, specially the drone. Always check the security protocols. A few common rules are as follows:

- You cannot fly a drone within 5 miles of an airport
- You must keep the drone within your eyesight
- You are not allowed to go higher than 400 feet (around 0.12km)
- You cannot fly a drone in busy traffic areas

- You must register your drone if you use it for business purposes or professionally; you must have a license
- Always know the local rules before flying a drone

15. Preparing Your Drone for Flying

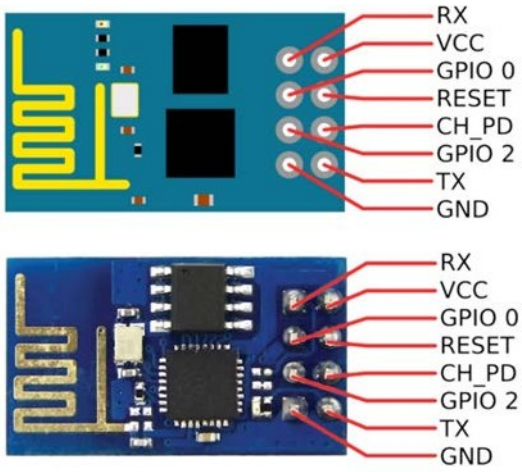
As you are reading this chapter, I assume you have assembled your DIY quadcopter and now you want to program it for flying. In this chapter, we will make our drone ready for flying. But as I said before, you can use an awesome Wi-Fi module, ESP8266, as the Wi-Fi telemetry device to receive data from your drone to your computer. In this chapter, we will learn about using the ESP8266 module with our ArduPilot and Arduino, so that you can use an ESP8266 wherever you need to. We will also learn about calibrating our quadcopter and configuring our ArduPilot software. Some of the things in the following list will be discussed in this chapter too:

Types of software to control and program the Arduino
- How we can connect the ESP8266 to the Arduino board
- Details about the ESP8266 module
- Coding for the ESP8266
- Controlling the ESP8266 from a smartphone
- Calibrating the drone after connecting the ArduPilot

What is ESP8266?

Basically, an ESP8266 is a Wi-Fi module. It has the capability for 2.4 GHz Wi-Fi, which is 802.11 b/g/n. It supports WPA and WPA2. It is a system-on chip integrated with a 32-bit processor which runs 80 MHz (it can also be overclocked to 160 MHz). It has 64 KB of RAM and a 64 KB boot ROM. The data RAM of ESP8266 is 96 KB. It is cheap, small, and powerful. That's why everyone uses it for different kinds of projects.

You can use an ESP8266 almost everywhere you need to make IoT wireless and smart. The following image is an ESP8266 with its pin out:



The standard ESP8266 has eight pins, as shown in the previous diagram. Let's look at their details:

Pin: Function

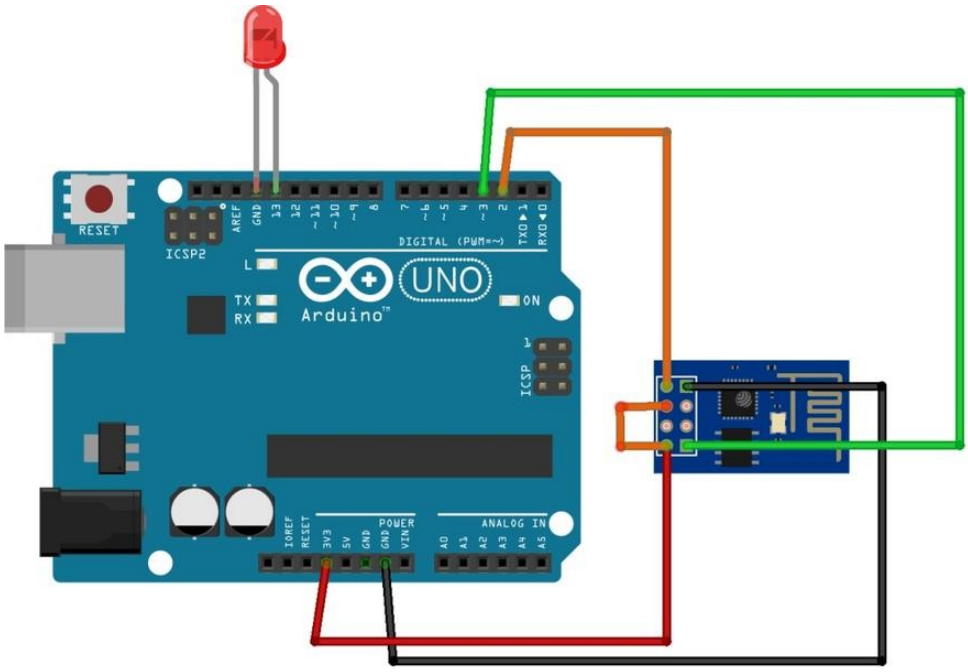
- **RX**: It receives data
- **VCC**: Power pin (usually 3.3V maximum)
- **GPIO 0**: General purpose input output pin 0
- **RESET**: It is a reset pin
- **CH_PD**: Chip power down pin
- **GPIO 2**: General purpose input output pin 2
- **TX**: It transmits data
- **GND**: Ground pin

Connecting the ESP8266 to Arduino

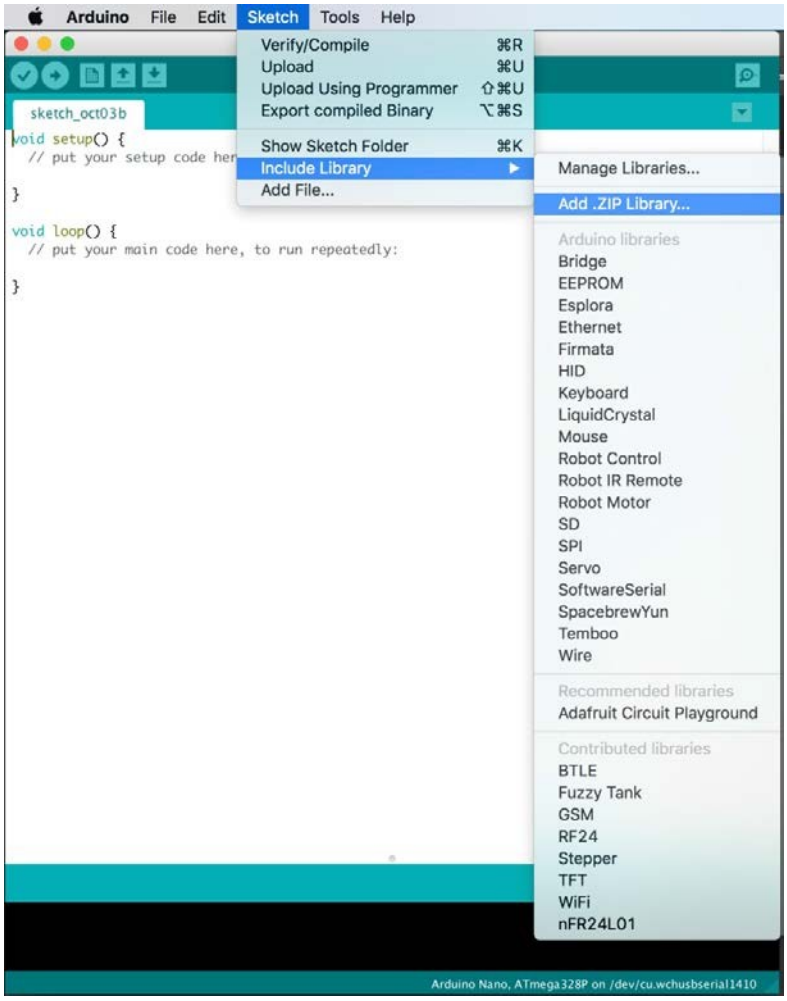
Let's connect our ESP8266 to an Arduino board. You can use any Arduino. Just follow the following pin configuration. We are going to control an LED with the ESP8266. So the pin settings will be as follows:

ESP8266	Arduino
RX	3
TX	2
VCC	3V
CH_PD	3V

GPIO 0	No connection
GPIO 2	No connection

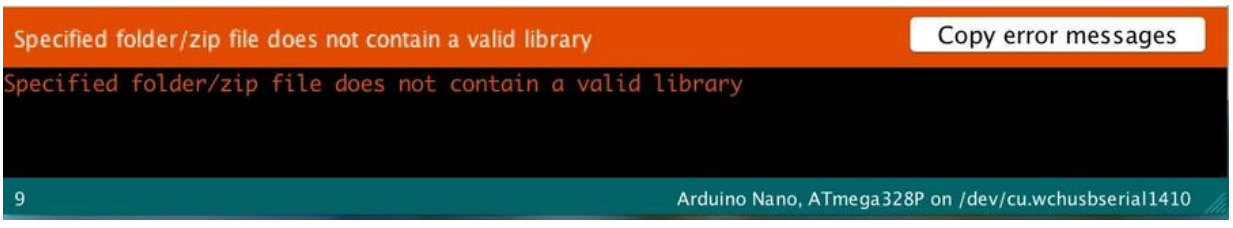


We will use an app on our smartphone to control the LED turn on and turn off functions. Add an LED on pin 13 of the Arduino for avoiding complexities for the first time. First of all, download and install the Arduino IDE from https://www.arduino.cc/en/Main/Software. Now you need to install a library on your Arduino IDE. Go to https://github.com/blynkkk/blynk-library/releases/latest and download the latest release of the Blynk library in ZIP format. Install the library in the Arduino IDE from Sketch | Include Library | Add .Zip Library:



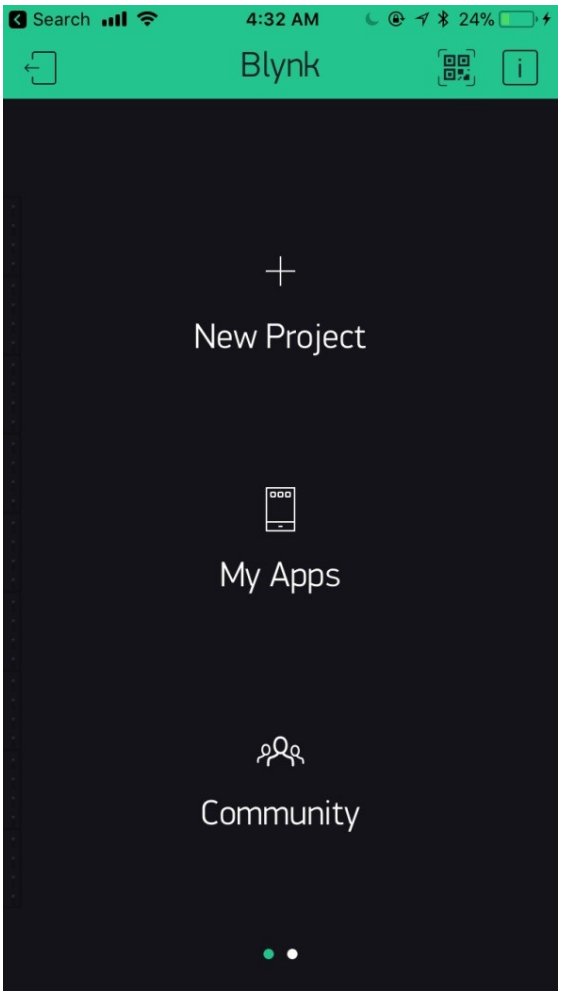
If you get an error like the following image, don't worry; just unzip the zipped file and copy all the contents of the libraries folder to the Arduino IDE library folder. The library location is as follows:

- **On Mac**: Right-click on the Arduino app icon and click on the Show Package Contents | Contents | Java | libraries
- **On Windows**: Go to Program Files or Program Files (x86) |Arduino| Libraries
- **On Linux**: Go to User | Share | Arduino| Libraries:



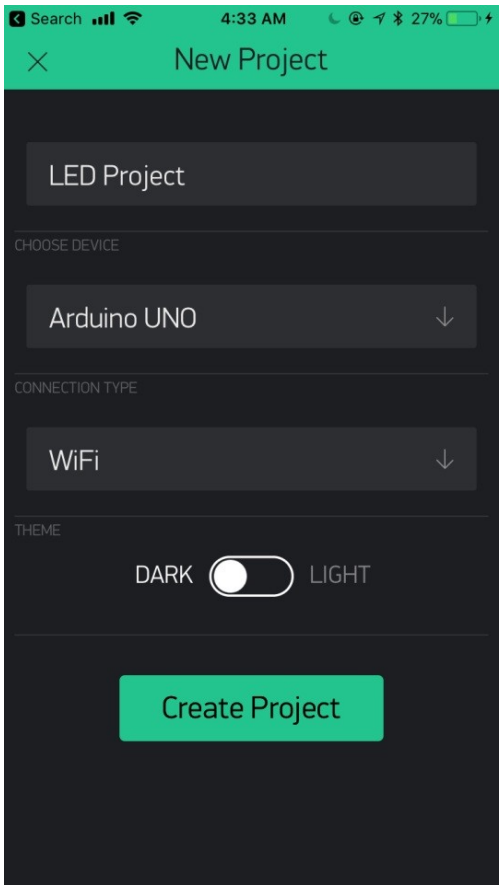
Once you have installed the library and connected the ESP8266 to the Arduino, now is the time to download the Blynk mobile application. The application is available on both the App Store and Play Store. Install the application and finish the registration process, if required.

You will see the following page after the installation:

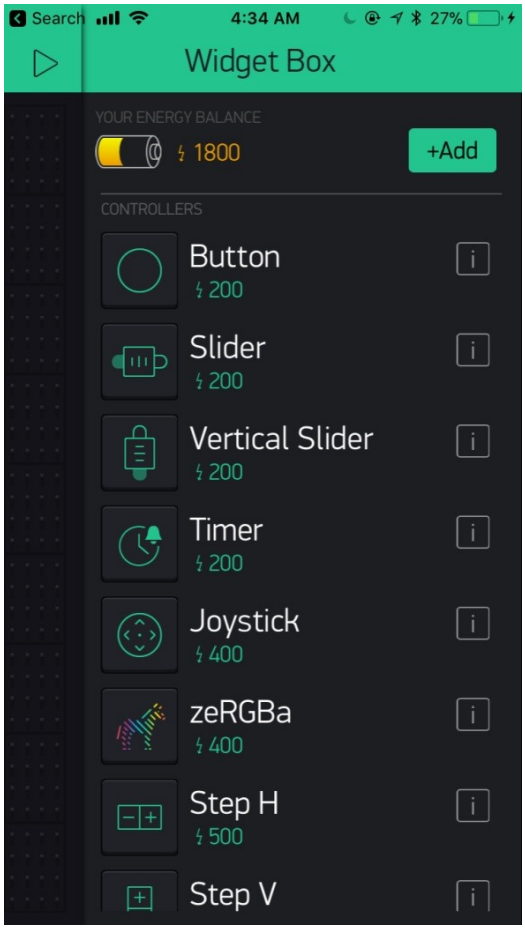


Refer to the following steps:

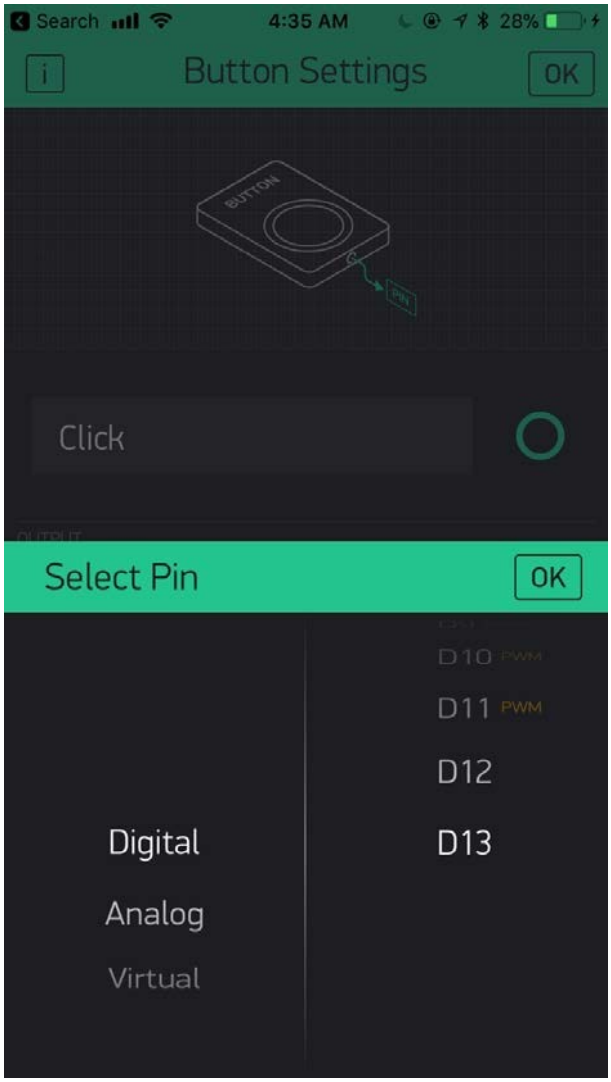
1. Click on New Project and name your project anything you want.
2. Choose the device, Arduino Uno (or your own device) and connection type, WiFi:



3. You will get a prompt that an authentication code was sent to your email, which can also be found on the project setting of the Blynk application. Ignore it for now. Swipe right ride of your screen to access the Widget menu and select a Button from there:



4. You will get a Button on the project page. Click on the Button and set the button properties. Select a pin type, Digital**,** and pin number 13, as we have connected our LED on the Arduino pin D13:



5. Now, fire up your Arduino IDE and write the following code there:

```
#define ESP8266_BAUD 9600 #include
<ESP8266_Lib.h>
#include <BlynkSimpleShieldEsp8266.h> #include
<SoftwareSerial.h>
char auth[] = "YourAuthToken"; char ssid[] =
"YourNetworkName"; char pass[] =
"YourPassword"; SoftwareSerial EspSerial(2,
3);

ESP8266 wifi(&EspSerial); void

setup()
{
    Serial.begin(9600);
    EspSerial.begin(ESP8266_BAUD); delay(10);
    Blynk.begin(auth, wifi, ssid, pass);
}
void loop()
```

```
{
  Blynk.run();
}
```

6. Remember the authentication token you got in the email? Assign the value of the auth[]string with your assign token. Assign the ssid[] and pass[] strings with your Wi-Fi name and Wi-Fi password, respectively.
7. Verify and upload the code to the Arduino and open the Blynk app.
8. From the Blynk app, press the Button and if everything is OK, the LED connected on your Arduino should turn on.

Since you can now control an LED with your smartphone, you can now control a motor with your phone, right? For more about the Blynk code and documentation, go to www.blynk.cc. You can use tons of things with your ESP8266 using Blynk. Let's go back to our drone, which needs to be configured.

16. How the Downloading and installing APM Planner or Mission Planner is done?

As we have been using ArduPilot as the control board of our drone, we need to use the APM Planner software for configuring our drone. Let's download APM Planner from here: http://firmware.ardupilot.org/, and download AMP Planner 2.0 or Mission Planner:

After successful installation, open the APM Planner or Mission Planner. The interface will look like this:

At the top left, you can see the following six menus:

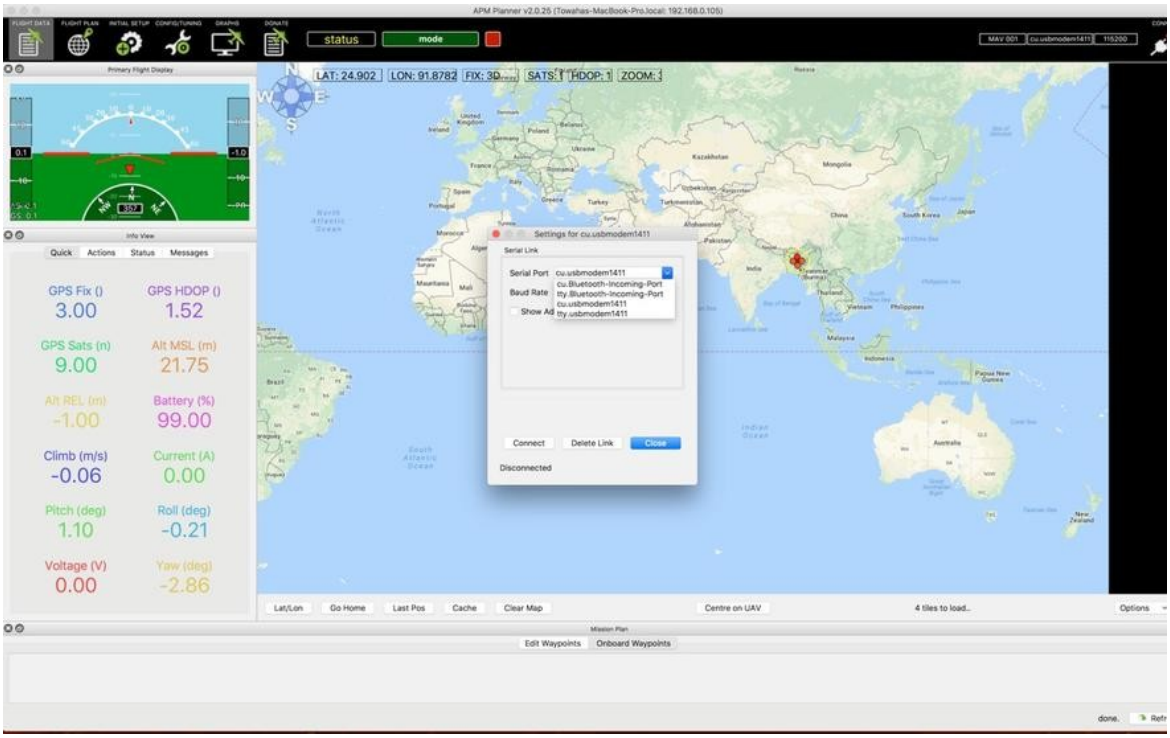
- Flight data
- Flight plan
- Initial setup
- Config/tuning

- Graphs
- Donate

The option names are self-explanatory. At the top right, you can see a Connect button. Before the Connect button, you can find the MAV ID option, which allows you to select the right port for your ArduPilot connected to your computer by USB or by telemetry. In Windows, you can select Auto mode to find out the perfect port number, if you are not sure which is the COM port number. But on OSX, you need to figure out the perfect port.

For the first time, we need to connect our ArduPilot via USB to your computer. Remember to unplug the drone battery, if you are connecting via USB. You will see that the LEDs are blinking on the ArduPilot. This means your ArduPilot is getting powered. Now on your computer, open APM Planner or Mission Planner, select the MAV ID**,** and click onConnect. If you are not sure which is the correct MAV ID, then follow my lead:

1. On OSX, you will see the Serial Port name, as shown in the following screenshot:



2. Select cu.usbmodem1411. The last number may be changed for your computer.

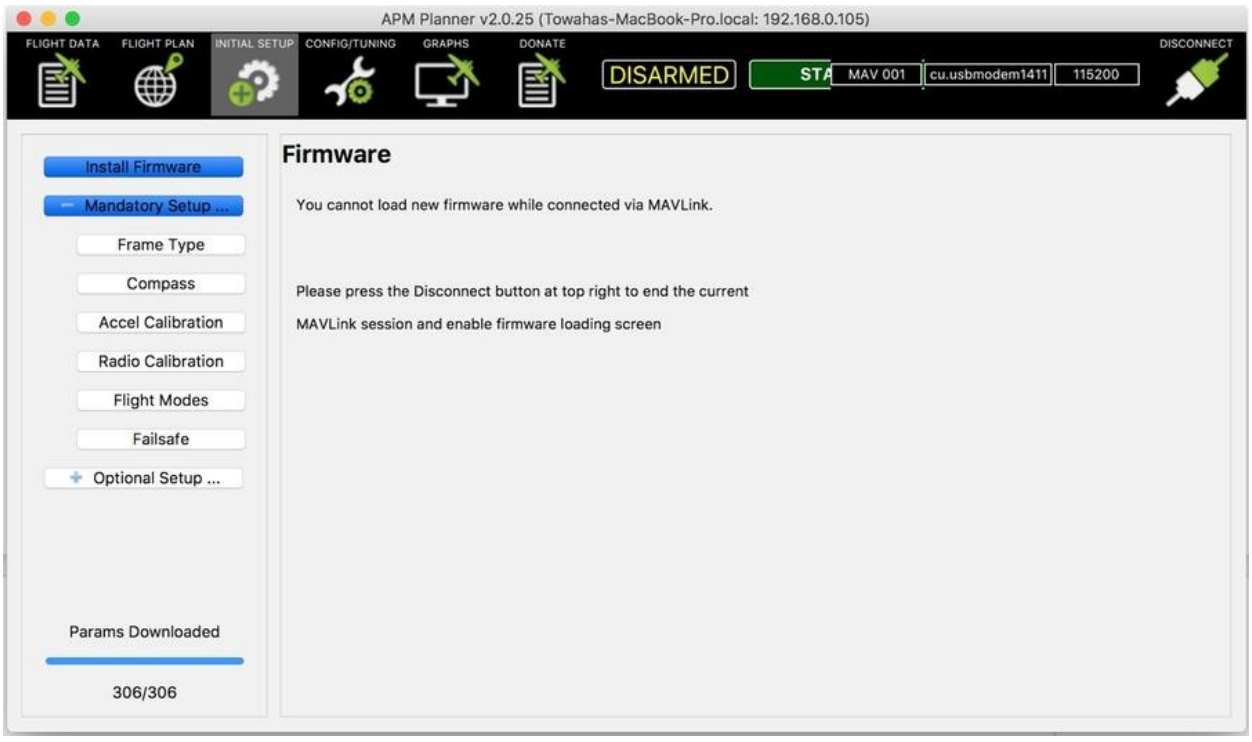
Select a baud rate of 115,200 for a higher speed and click on Connect**.**

3. Once connected, you will see some messages under the Info View panel of the APM Planner.

On Linux, you will see almost the same but in Windows you will see the Port number on the top-right side. Carefully choose the port number before clicking the Connect button of the software. Or you can choose Auto on the Mission Planner version for Windows.

Configuring the quadcopter

Let's configure our quadcopter. Before going any further with the configuration, remove all the propellers of your drone. It can be dangerous if the propellers are connected. Connect the USB cable to the ArduPilot and connect it to your computer. Open APM Planner and go to the Initial Setup tab. Remember, you need to disconnect the connection before going to the Initial Setup tab or you might see some errors (especially when installing new firmware). You will see the options, as shown the following screenshot:



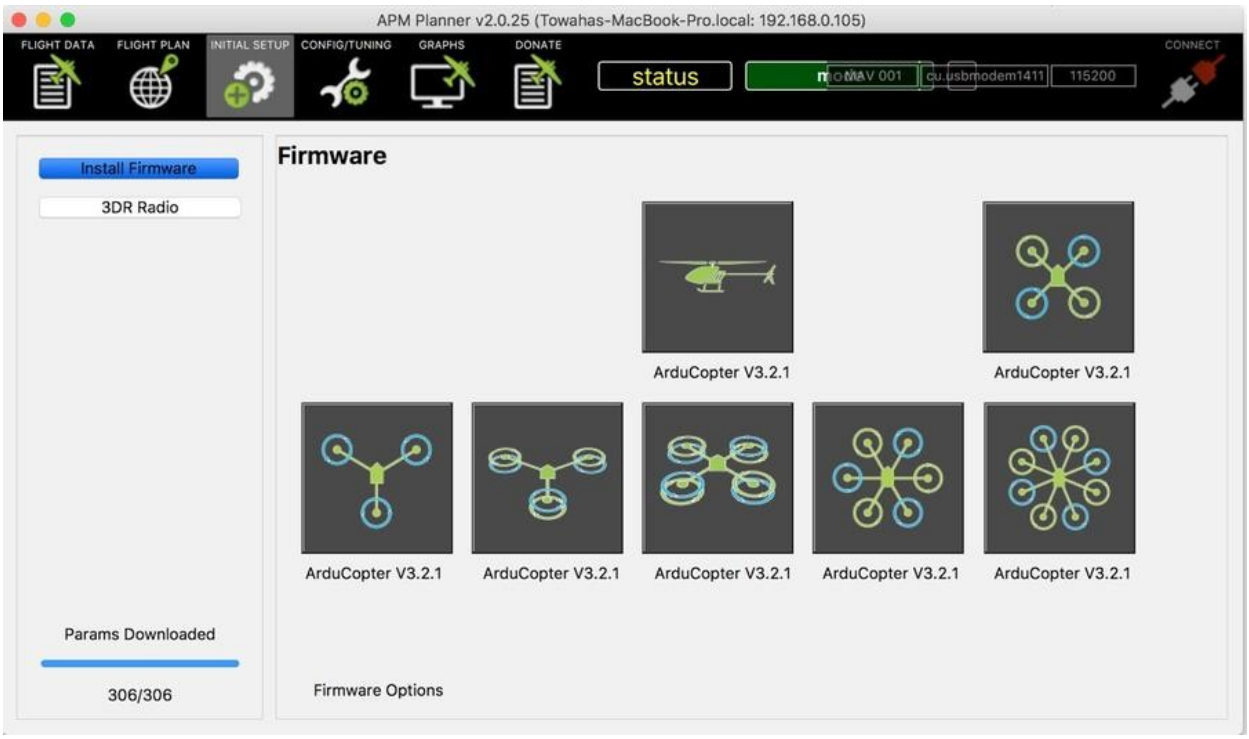
Firstly, we are going to install the firmware, which is the most important work for configuring the ArduPilot:

1. Disconnect the connection between the ArduPilot and your computer by

clicking the Disconnect button but keep the USB cable connected.

2. Click on Install Firmware from the left-hand side. You will be shown several types of drones.
3. Since we are configuring a quadcopter, select the quadcopter. In my case, it

is ArduCopter 3.2.1. The version may vary depending on the version of the ArduPilot control board:



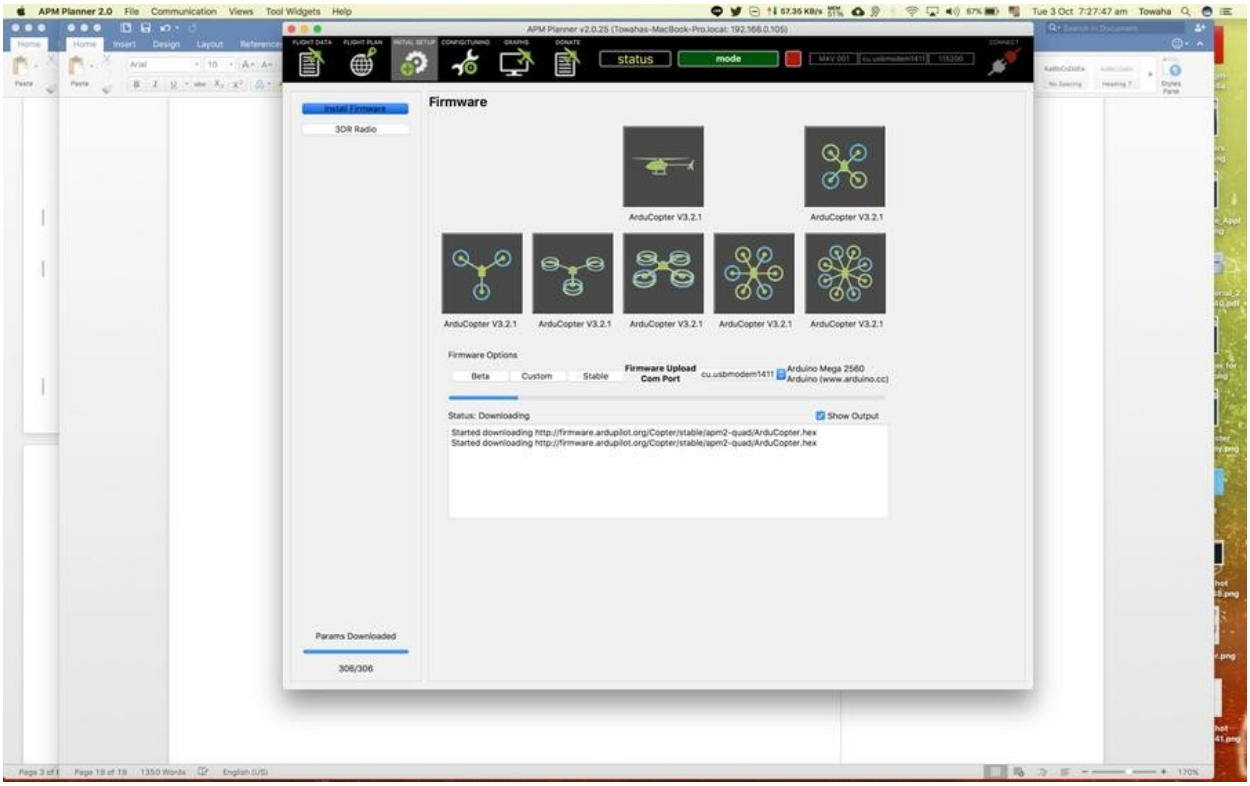
4. You will be prompted with the following message:

| **"You are about to install ArduCopter.hex for apm2-quad"**

5. Click the OK button to start the process.

Make sure the USB cable is connected firmly and properly.

6. After clicking OK, you will see that the download has begun. To see the output, you may check the Show Output option.
7. After downloading, the firmware will be flashed into the ArduPilot automatically. Do not unplug or loosen the connections. After successful flashing, you will see the following messages:


APM Planner 2.0
File
Communication
Views
Tool Widgets
Help
APM Planner v2.0.25 (Towahas-MacBook-Pro.local: 192.168.0.105)
status
mode
Install Firmware
3DR Radio
Firmware
ArduCopter V3.2.1
ArduCopter V3.2.1
ArduCopter V3.2.1
ArduCopter V3.2.1
ArduCopter V3.2.1
ArduCopter V3.2.1
ArduCopter V3.2.1
Firmware Options
Beta
Custom
Stable
Firmware Upload
Com Port
cu.usbmodem1411
Arduino Mega 2560
Arduino (www.arduino.cc)
Status: Downloading
Show Output
Started downloading http://firmware.ardupilot.org/Copter/stable/apm2-quad/ArduCopter.hex
Started downloading http://firmware.ardupilot.org/Copter/stable/apm2-quad/ArduCopter.hex
Params Downloaded
306/306

Firmware

Install Firmware

3DR Radio

ArduCooter y'3.2.1

ArduCopter V3.2.1

Ard CagTer V3.2.*1

Afd+JCagter V3.2.1

Afd+iCoptef VG.2.1

AyduCQgtey Vs:2*.T

A¥duCoptey vs.2.1

Firmwa e oe‹ "

Beta Custom Stable Firmware Upload Com Port cu.usbmodem1411 Arduino Mega 2560 Arduino (www.arduino.cc)

Show Output

8tatña: Downlaâding

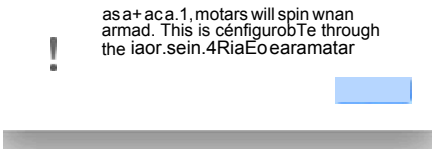
siarrad dowrtioaoing http://rifmware.ardupifot.org/copTerIa\aaio/aomz•ouad/arducopim.nex

S\aned downioading http://r'rmware.ardueito‹.org/Coeter/s\ ab+e/apm2-qpad/>rduCop\ er.hex

H6f306

Frame type selection

Now, click the Connect button and go to the Initial Setup tab. From the Fame Type option, select the x shape (be aware that the four propellers are on each corner of the shape of x) quadcopter, as our ArduPilot is connected like x's center but pointed forward in between the middle of the two hands of the quadcopter:

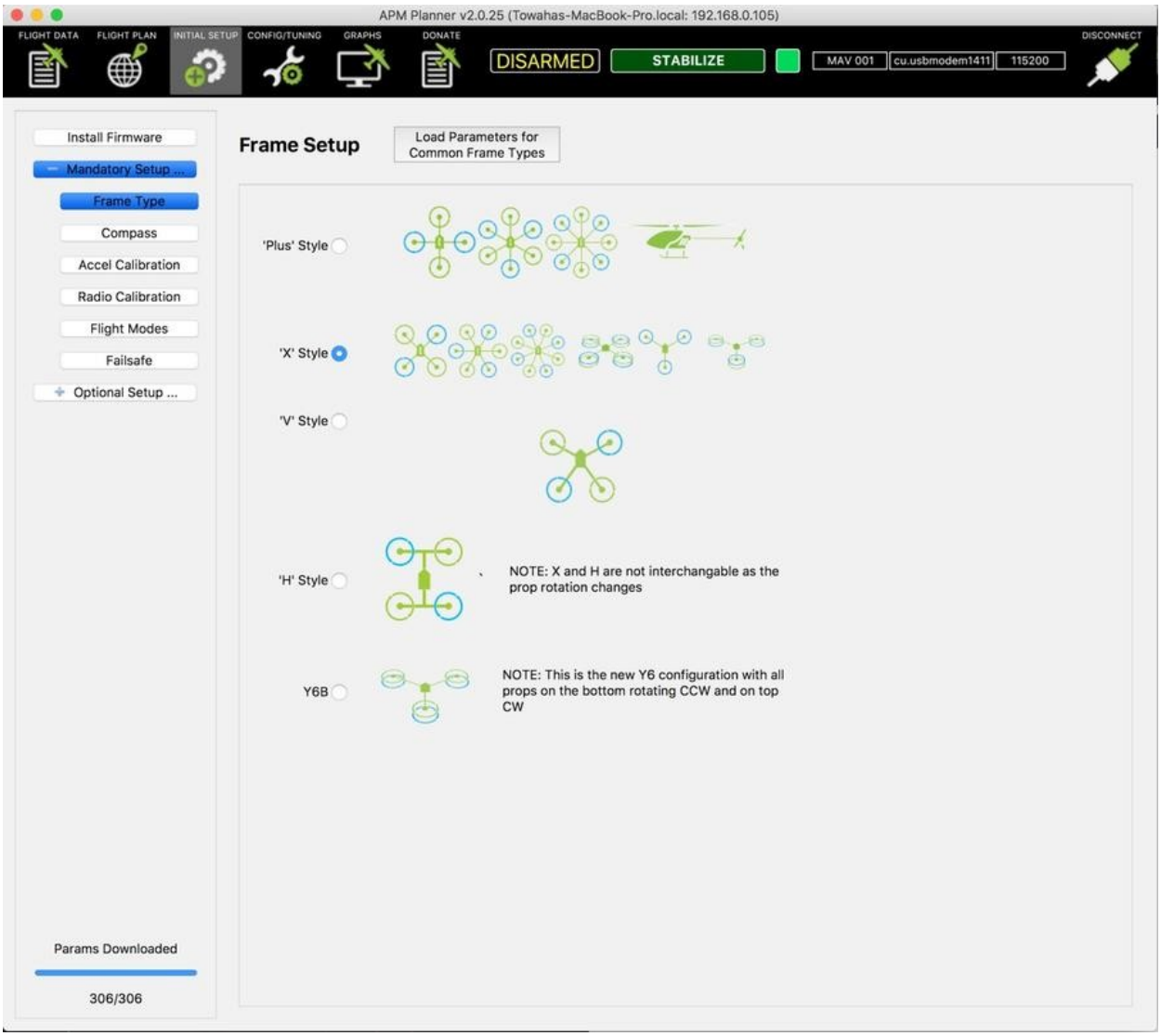


Compass calibration

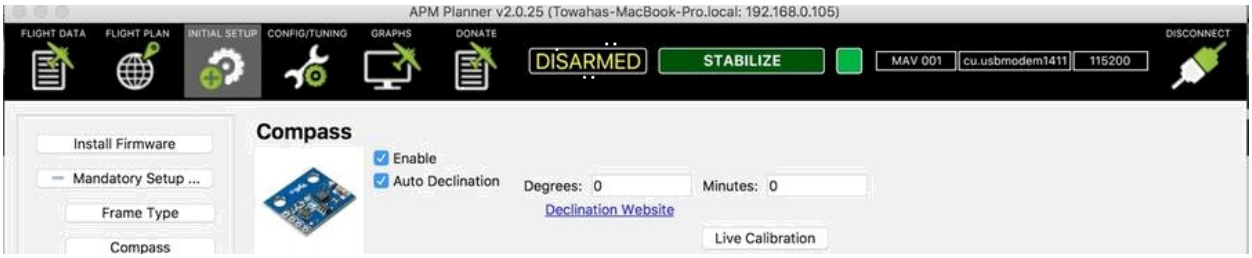
Then, go to the compass and check the following options:

- Enable
- Auto detection

Click on Live Calibration. And rotate your quadcopter all 360 degrees for about 60 seconds. The software will collect a few calibration points for calibrating the compass to the ArduPilot. You may rotate only the ArduPilot by removing other parts from it, if you want to get a good calibration:



After successful calibration, you will see the following message:



Compass Not calibrates tne compass for magnetic intarfarence created by sha Q0w4r

Fai Isa Ie

nata wifi i» •oiieciea i•• we s»-•a ,

PTeaee click oX and move the epmaroun• •!!

ine ie. 0°

\T 180°

306y306

Access calibration

Now, select Access Calibration. Hit the Calibrate button and you will see instructions as to which orientation you should place your drone. Hit any key or a button. Just do what the software says and complete the calibration. Make sure you choose the orientation (above-left-right-down-up-back) correctly because it is really important when flying a drone. And make sure the USB cable is firmly connected and does not disconnect while calibrating. After successful calibration, you will see the following message:

[redacted]

309 Rad io

9ew elf sets t9omPsss 4i are

s:-11a.1oo y:a.v z z:ze.soa aav la:

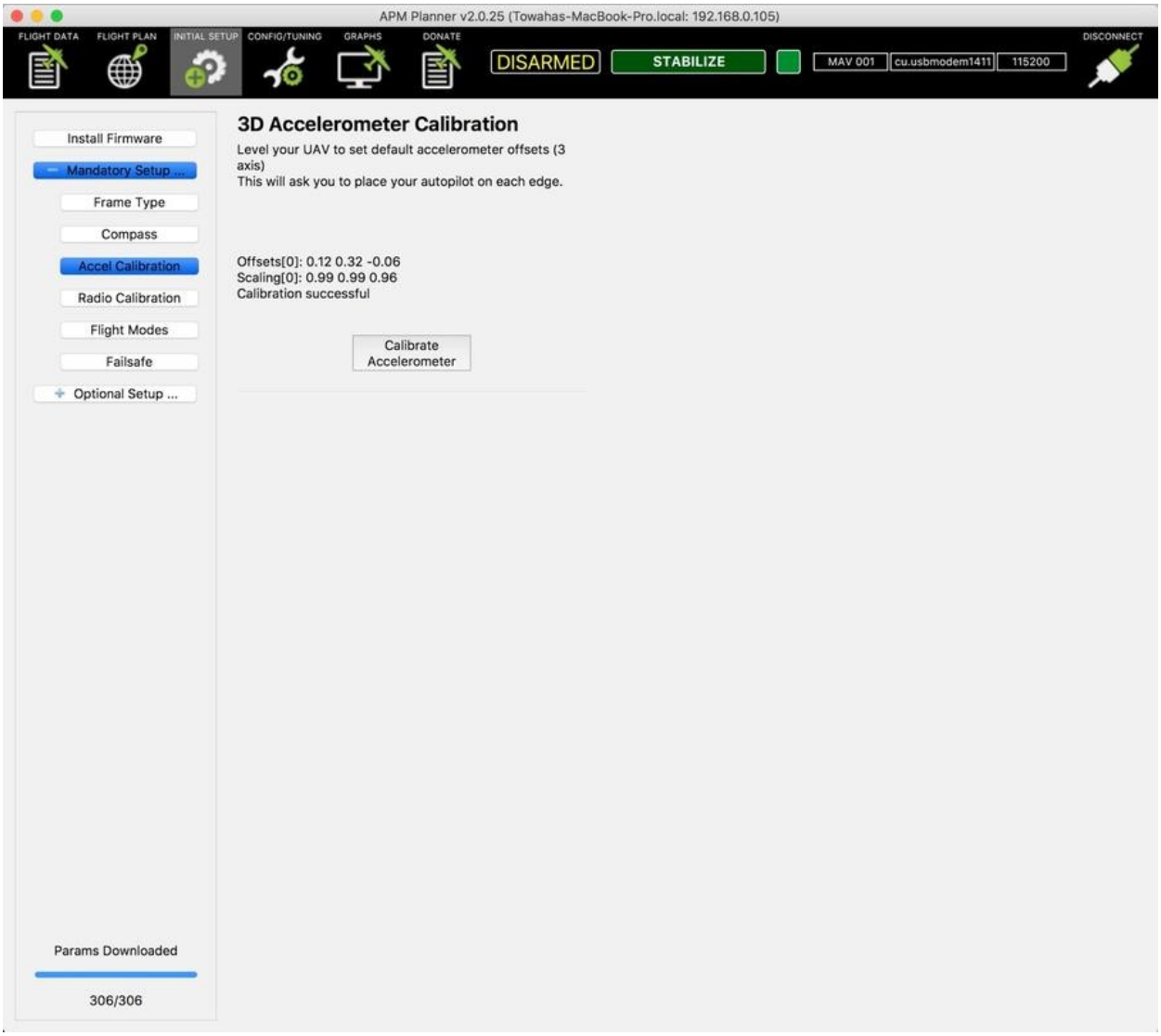
OK

Params Downloaded

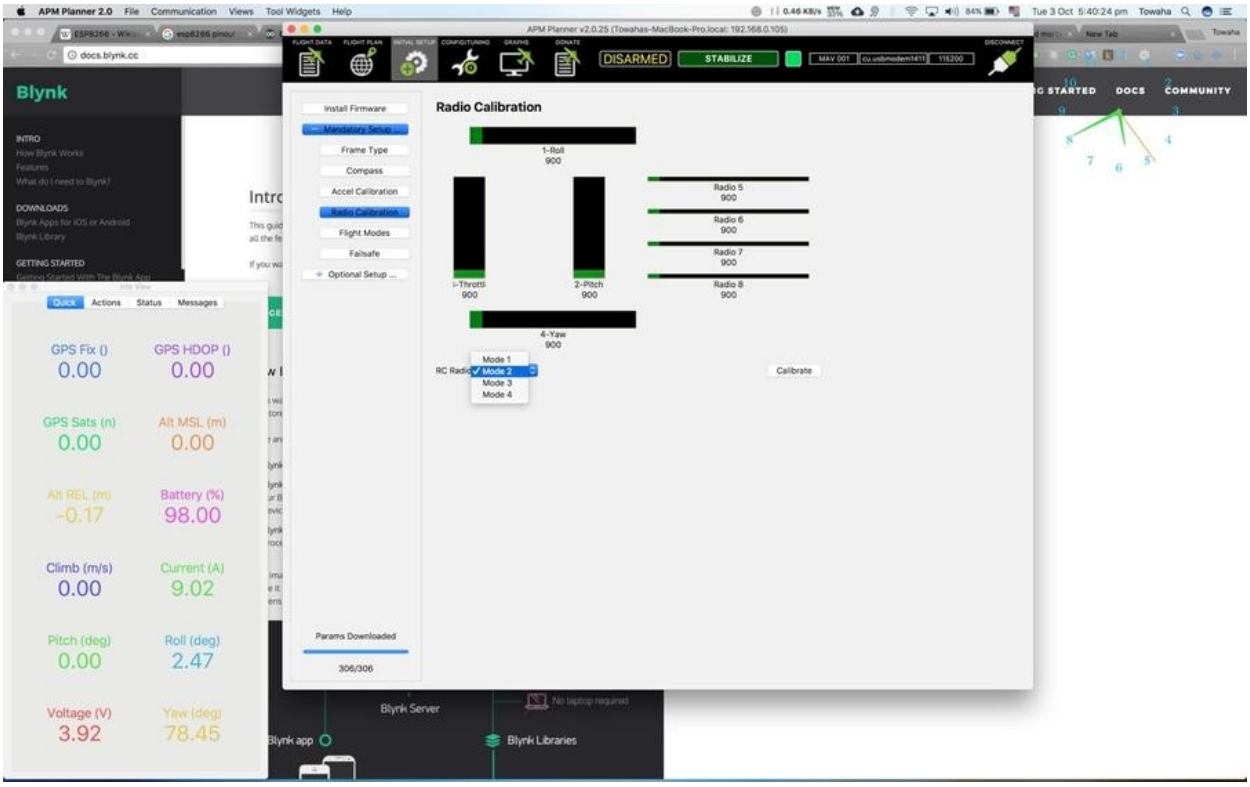
306/306

Radio calibration

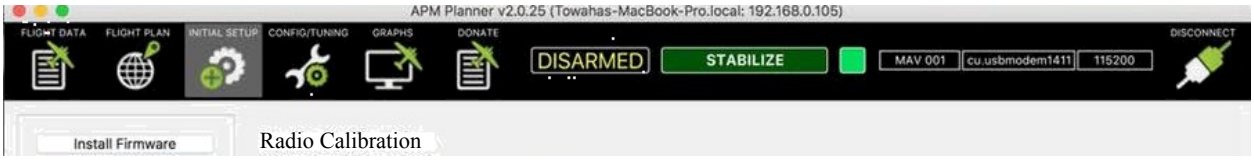
As we bound our radio with our transmitter in the Chapter 2, *Assembling Your Drone*, we need to connect it to the ArduPilot and click on Radio Calibration. You will see the following screen:



Now, choose a flying mode which is suitable to you. Make sure the transmitter is turned on. I prefer Mode 2. Now click on **Calibrate** and rotate all the gimbals and knobs to every position. You will see that the progress bars will show the increments and decrements while you move the gimbals and knobs. After doing this, click the End Calibration button. Remember, you must disconnect all the motors and propellers for this calibration or you might get a serious injury. After clicking the End Calibration button, you will see the following message:



After clicking the OK button, you will see your calibrated settings for your transmitter and radio, as shown in the following screenshot:



Install Firmware

Radio Calibration

Mandatory Setup ...

Frame Type

*ccel Gali bra: ion

Radio Calibration

Flight Modes

Options i Setup

1- Roll
900

900

900

9Q0

Rooo8
900

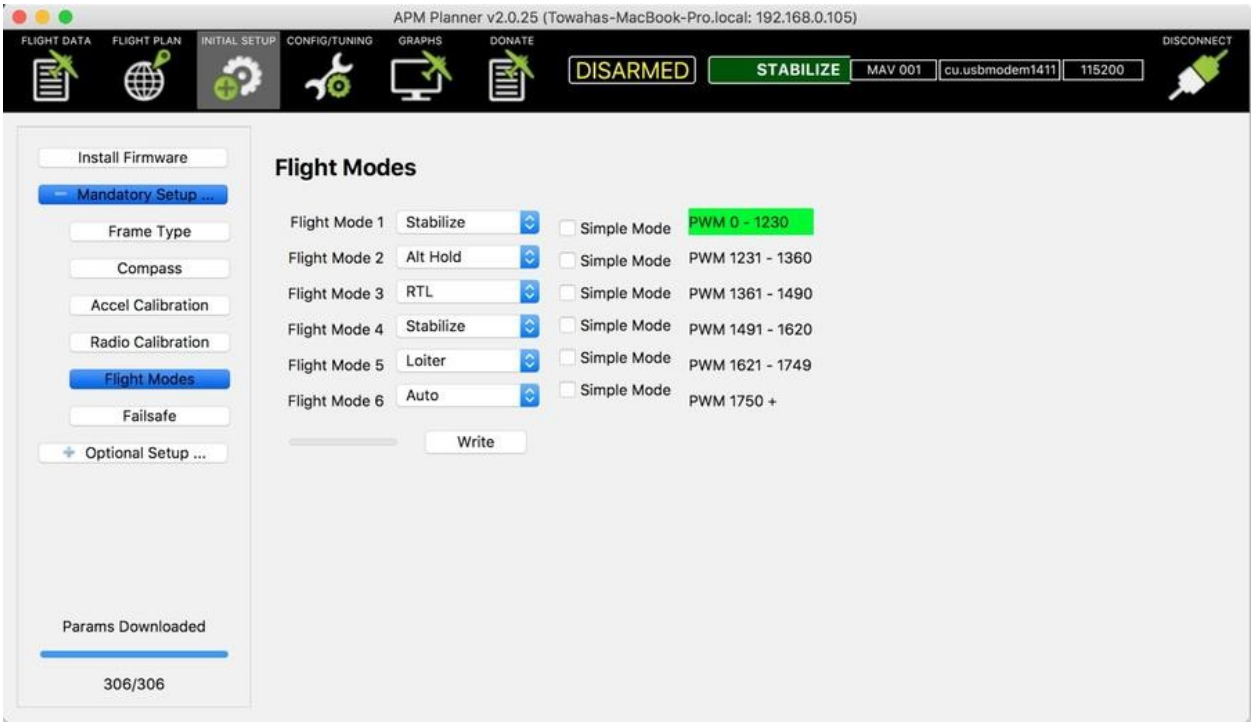
4-
Ya\v
000

Porams Ooyrnloaoeo

306/306

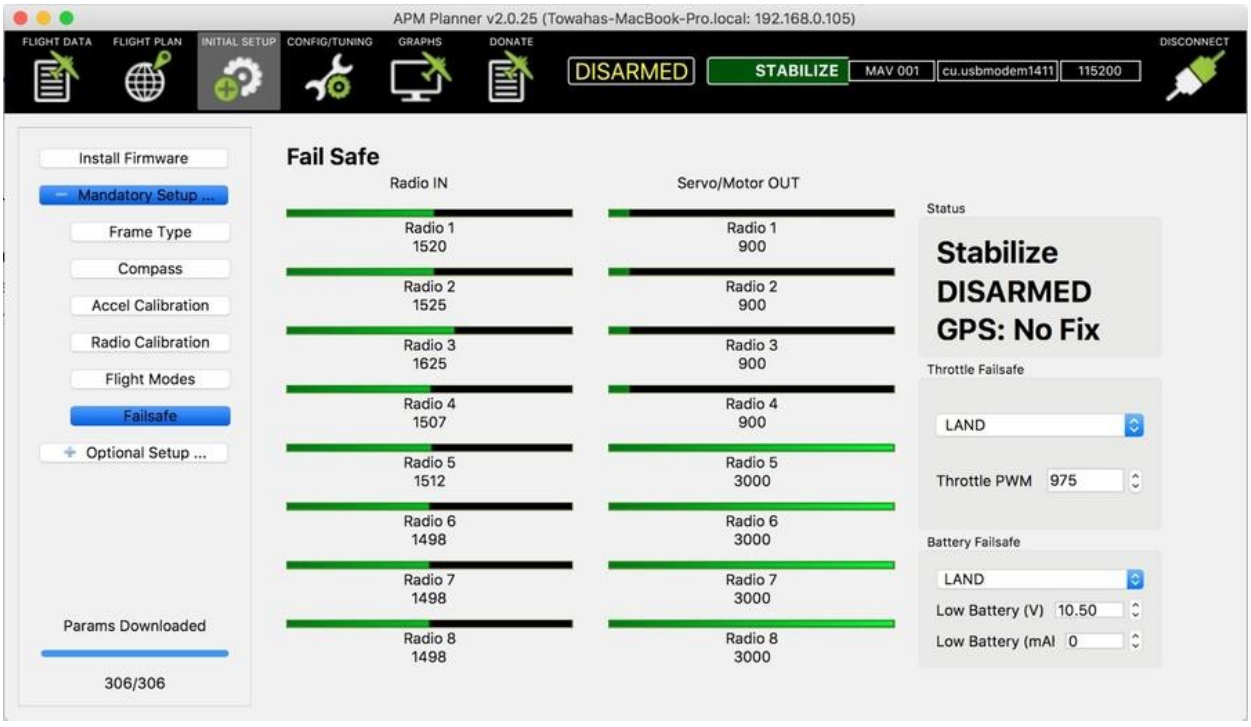
Flight mode calibration

For the flight mode, you can choose any type you want. But to avoid complexities, you can choose and write the settings, as shown in the following screenshot:



Failsafe calibration

My personal preference is for the Failsafe configuration for the throttle fail when returning to the land. Failsafe helps to protect the drone so as to avoid crashes when it loses the connection from the transmitter. You may choose the **Return to Launch** (**RTL**) option too:



There are other optional calibrations you may need in the future. But for now, your drone is ready to fly.

Let's reconnect all the propellers and motors carefully and find a suitable place where you can test your first flight:

1. Turn on your transmitter first. Connect the battery on your drone and you will hear beeps from the ESCs. Now if everything is OK, you move the left gimbal (throttle) to the bottom right and hold it until the propellers start rotating. Keep your finger away from the other gimbal for now.

2. Make sure all the propellers are moving, now you can throttle the motors a little bit more, but don't be quick; do it gradually. Whenever the thrust is greater than the drone weight, your drone will start hovering, as we learned in the Chapter 2, *Assembling Your Drone*. Do not give your drone rolling or pitching until it is at least more than two meters away (just as an extra precaution).
3. Now you can move the pitch and roll of your transmitter and control your drone. To stop the drone from flying, remove your finger from the roll/pitch gimbal and slowly decrease the throttle. Your drone will come to ground gradually. Do not hurry, as your drone might get a bounce on the ground and get misdirected. It can also break the propellers. When the drone is almost on the ground, just push the throttle gimbal to the bottom left and hold it until all the propellers stop. As an extra precaution, switch off the transmitter first and then remove the battery of the drone.

17. How Building a Follow Me Drone

What is a Follow Me drone?

A Follow Me drone follows a device or an object; the device can be your phone or a device with some sensors that continuously communicate with the drone to get the right position. If it follows an object, then there is some machine learning involved with the system. Some drones can do image processing and follow an object. For example, if you train your drone to follow dogs, you need to teach it how to detect an object and compare it with a dog and follow it, which is a little complicated. But in this chapter, you will be given some ideas that can be implemented for building your own Follow Me drone. As we were talking about following a device with the Follow Me drone, you may have guessed that the position might be determined by the GPS sensor. And, of course, you need other sensors for the drone to locate your position correctly; thus, the receiver and sender create a communication between them and follows the device. The speed of communication also needs to be faster in some cases, because you know what happens when you are walking and your pet dog is not following you. Follow Me drones are like air dogs to me. Simply, they can follow something, and they can do some simple things such as taking photos or recording videos.

If your drone is capable of doing complex things like face detecting, object targeting, and so on. then the processing speed of the drone's brain should be higher than the communication speed between the RC receiver and the drone.

Making a Follow Me drone using ArduPilot

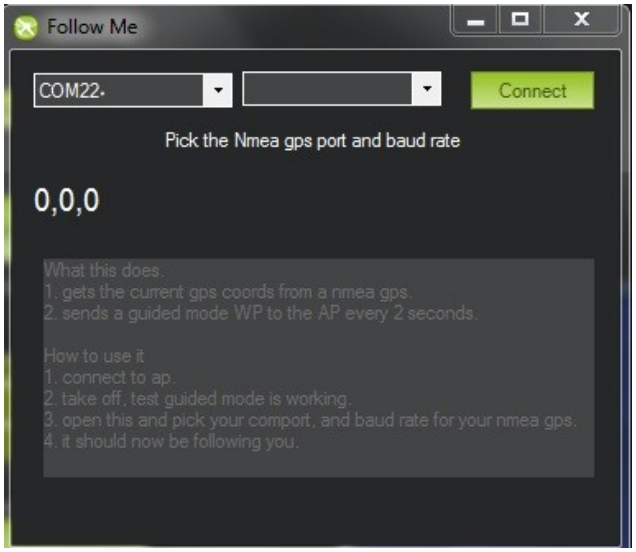
It is super easy to build a Follow Me drone using ArduPilot. You just need to change some settings and buy a USB dongle, which is a GPS receiver. There are a few types of USB dongles that go with ArduPilot, but the official one to use is the GlobalSat ND-100S USB GPS dongle, or you can use a USB dongle or a GPS-embedded Bluetooth module such as the GlobalSat BT-368i Bluetooth GPS receiver. The following picture shows both types of modules:

GlobalSat ND-100S USB GPS Dongle

GlobalSat BT-368i Bluetooth GPS Receiver

You will also need telemetry for setting up the modules with the ArduPilot software. Setting up the modules is easy:

1. Firstly, take your drone to a suitable place for flying and connect it via MAVLink using telemetry.
2. Now, connect your USB dongle or Bluetooth receiver to your laptop.
3. You can check if your module is working by using the software of the module you use, or the LED built on it.
4. Set the GPS position locked before the take off or increasing the throttle speed. Now, gradually take off the drone and keep it at a sufficient altitude (I suggest at least 7-8 feet), and switch the flying mode to Loiter. For those who do not know what the Loiter mode is, you may check the upcumming note. This will lock the drone in to position.
5. Now, on your Mission Planner software, go to the Flight Data Screen and right-click any position you want and select Fly to Here. If your drone flies to the place you just selected, then your Follow Me drone is almost ready.
6. Now, if everything works, on the Mission Planner click *Ctl+F* or *Command+F* to open a setting and click the Follow Me button.
7. When you click the Follow Me button, your computer will try to use the dongle you connected to it by showing a window, as follows:



8. Select the proper COM port and the name of the device and click on Connect.

Loiter mode is an automatic mode in which the copter tries to maintain the current location, the altitude, and the heading. To do this, the copter must have a good GPS receiver. In this mode, the pilot feels safe as the hand is kind of released from the sticks of the controller. You can find the settings in the Mission Planner (Configuration | APM Copter | PIDs? | Loiter PID). There, you can set up the PIDs and speed as you want. Once you connect the USB dongle to the Mission Planner software, your drone will consider the dongle the Fly to Here location all the time.

Now, if you move with your laptop, your drone will follow your laptop. Your Follow Me drone is now ready. Why don't you walk a little bit and see if your air pet follows you? But, you might be wondering, why do I have to carry my laptop? Yes, this is irritating. You can walk with only the dongle, but the setting will be slightly different. Let's try another way of making your drone work like a Follow Me drone.

Using a smartphone to enable the Follow Me feature of ArduPilot

You can make your drone follow you if you use a smartphone. Your smartphone already has a built-in GPS, and the drone will follow your phone if properly configured. You will need an OTG connector to connect the telemetry to your phone. Wherever you take your phone, your drone will follow it, as long as the data transmission is good. You can use any of the following two applications (there might be other applications that I have not tested) for this:

- DroidPlanner2
- Tower
-

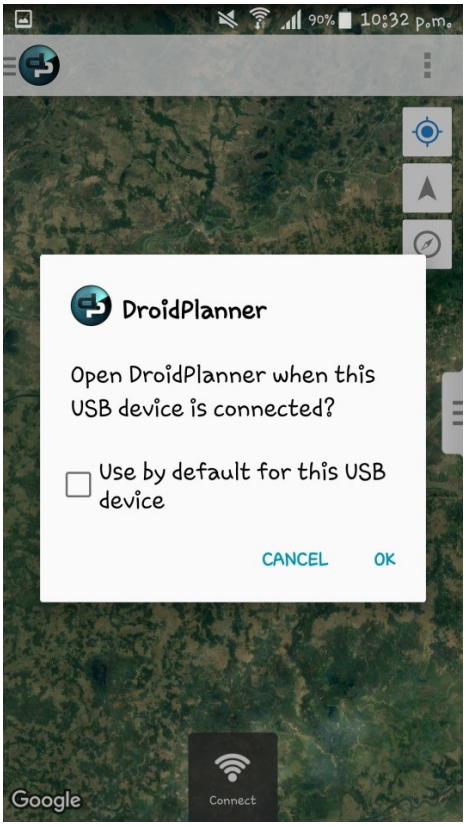
Unfortunately, neither of the applications is available for iPhone (there is the 3DR Solo application for iPhone that can be used to control the on-drone camera by using Wi-Fi and Sidepilot, which is able to configure the drone to Follow Me mode if you use the ArduCopter or 3DR Solo platforms). So, we need to use Android phones. Let's learn how you can configure your drone with your smartphone. Firstly, we will learn how you can configure by using the DroidPlanner 2 application.

Using DroidPlanner 2

Let's see how to use DroidPlanner 2:

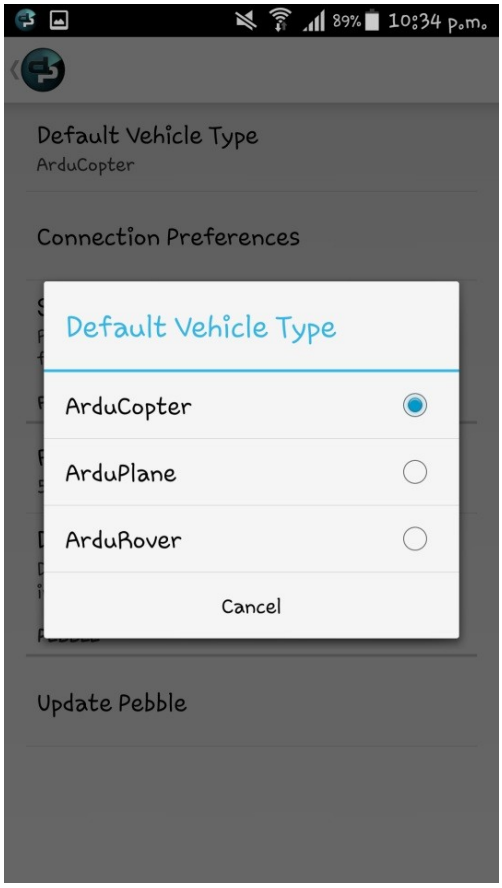
1. First of all, go to the Play Store and download and install DroidPlanner 2 (https://play.google.com/store/apps/details?id=org.droidplanner&hl=en). After installation, open the app and you will see the following page:

2. Now, connect your OTG cable/converter to your phone and connect the USB telemetry, such as SiK Telemetry Radio, to your phone via the OTG. You may see the following message:



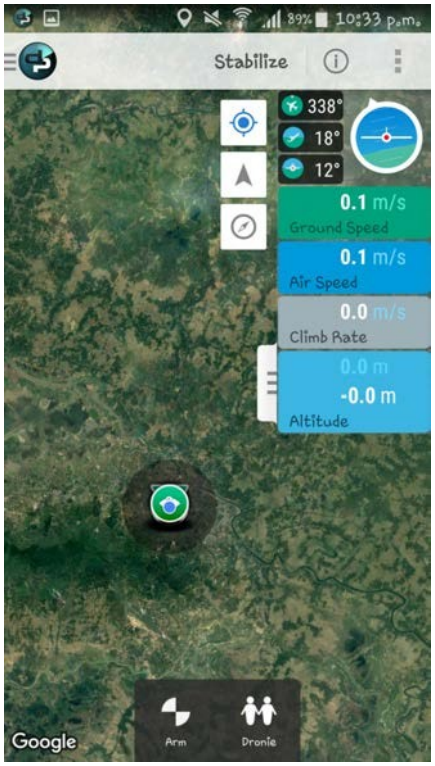
3. Click OK there or your telemetry won't be powered from your phone.

4. You may need to change some settings of the application before connecting the drone. Go to the settings menu of the application (top-left corner) and select the vehicle type as ArduCopter:



5. And from the home menu, select Loiter as the flight mode. You can change it while flying too:

6. Now, take your drone to a suitable place for flying. Arm your drone and set the mode to Loiter.
7. While the drone is hovering, click the Connect button at the bottom of the application on your phone.
8. Once the connection is established between your drone and the phone, you will be able to see the following screen:



9. If you can see the preceding screen without any error, then you are ready to use your drone as a Follow Me drone. Just click the Dronie icon at the bottom of the screen, and your drone is now a Follow Me drone.
10. Take your phone anywhere; the drone will go with you now. You may control your drone from the application too. You can see all the flight data on the screen and know your drone's condition.

Remember, you must enable the location of your phone before using this app or this will not work.

Now we will see how we can use the Tower app for making the drone act like a Follow Me drone.

Using the Tower application

The Tower application is similar to the DroidPlanner 2 application, but has a few extra features such as camera control and smart user interface:

1. Go to the Play Store and download and install Tower (https://play.google.com/store/apps/details?id=org.droidplanner.android&hl=en). You might get a warning about the OTG connection on your screen. Just do what you need to do. After installation and connection, you will see the following page on your phone:

2. Now, change the flight mode to Loiter and click the Dronie button.
3. Your drone is now a Follow Me-type drone. If there is a camera connected to your drone, you will be able to see the camera signal on your phone on the no video available section. We will learn how to connect a camera to your drone in Chapter 5, *Building a Mission Control Drone*.

If you are a hardcode programmer and hardware enthusiast, you can build an Arduino drone, like the following one, and make it a Follow Me drone by enabling a few extra features. The following guidelines are not for new readers, so if you need help, you can reach me anytime. Let's get started.

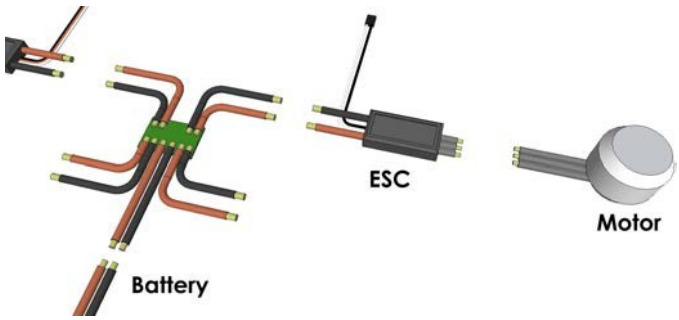
Building an Arduino-based Follow Me drone

For this section, you will need the following things:

- Motors
- ESCs
- Battery
- Propellers
- Radio-controller
- Arduino Nano
- HC-05 Bluetooth module
- GPS
- MPU6050 or GY-86 gyroscope.
- Some wires

Connections are simple:

1. You need to connect the motors to the ESCs, and ESCs to the battery. You can use a four-way connector (power distribution board) for this, like in the following diagram:



2. Now, connect the radio to the Arduino Nano with the following pin configuration:

Arduino pin	Radio pin

D3	CH1
D5	CH2
D2	CH3
D4	CH4
D12	CH5
D6	CH6

3. Now, connect the Gyroscope to the Arduino Nano with the following configuration:

Arduino pin	**Gyroscope pin**
5V	5V
GND	GND
A4	SDA
A5	SCL

4. You are left with the four wires of the ESC signals; let's connect them to the

Arduino Nano now, as shown in the following configuration:

Arduino pin	**Motor signal pin**
D7	Motor 1
D8	Motor 2
D9	Motor 3
D10	Motor 4

Our connection is almost complete.

Now we need to power the Arduino Nano and the ESCs. Before doing that, making common the ground means connecting both the wired to the ground.

Before going any further, we need to upload the code to the brain of our drone, which is the Arduino Nano. The code is little bit big. I am going to explain the code after installing the necessary library. You will need a library installed to the Arduino library folder before going to the programming part. The library's name is PinChangeInt.

Install the library as you did in Chapter 3, *Preparing Your Drone for Flying*. Now, we need to write the code for the drone. The full code can be found at https://github.com/SOFTowaha/FollowMeDrone.

Let's explain the code a little bit.

In the code, you will find lots of functions with calculations. For our gyroscope, we needed to define all the axes, sensor data, pin configuration, temperature

synchronization data, I2C data, and so on. In the following function, we have declared two structures for the accel and gyroscope data with all the directions:

```
typedef union accel_t_gyro_union
{
  struct
   {
     uint8_t x_accel_h;
     uint8_t x_accel_l; uint8_t
     y_accel_h; uint8_t
     y_accel_l; uint8_t
     z_accel_h; uint8_t
     z_accel_l; uint8_t t_h;
     uint8_t t_l; uint8_t
     x_gyro_h; uint8_t
     x_gyro_l; uint8_t
     y_gyro_h; uint8_t
     y_gyro_l; uint8_t
     z_gyro_h; uint8_t
     z_gyro_l;
   } reg;
  struct
   {
     int  x_accel; int
     y_accel; int  z_accel;
     int temperature; int
     x_gyro;
     int y_gyro; int
     z_gyro;
   } value;
};
```

In the `void setup()` function of our code, we have declared the pins we have connected to the motors:

```
myservoT.attach(7); //7-TOP
myservoR.attach(8); //8-Right
myservoB.attach(9); //9 - BACK
myservoL.attach(10); //10 LEFT
```

We also called our `test_gyr_acc()` and `test_radio_reciev()` functions, for testing the gyroscope and receiving data from the remote respectively. In our `test_gyr_acc()` function. In our `test_gyr_acc()`
function, we have checked if it can detect our gyroscope sensor or not and set a condition if there is an error to get gyroscope data then to set our pin `13` high to get a signal:

```
void test_gyr_acc()
{
  error = MPU6050_read (MPU6050_WHO_AM_I, &c, 1); if (error !=
  0)
   {
     while (true)
```

```
        {
            digitalWrite(13, HIGH); delay(300);
            digitalWrite(13, LOW); delay(300);
        }
    }
}
```

We need to calibrate our gyroscope after testing if it connected. To do that, we need the help of mathematics. We will multiply both the rad_tilt_TB and rad_tilt_LR by 2.4 and add it to our x_a and y_a respectively. then we need to do some more calculations to get correct x_adder and the y_adder:

```
void stabilize()
 {
    P_x = (x_a + rad_tilt_LR) * 2.4; P_y = (y_a
    + rad_tilt_TB) * 2.4;
    I_x = I_x + (x_a + rad_tilt_LR) * dt_ * 3.7; I_y = I_y + (y_a +
    rad_tilt_TB) * dt_ * 3.7; D_x = x_vel * 0.7;
    D_y = y_vel * 0.7;
    P_z = (z_ang + wanted_z_ang) * 2.0;
    I_z = I_z + (z_ang + wanted_z_ang) * dt_ * 0.8; D_z = z_vel *
    0.3;
    if (P_z > 160)
       {
          P_z = 160;
       }
    if (P_z < -160)
       {
          P_z = -160;
       }
    if (I_x > 30)
       {
          I_x = 30;
       }
    if (I_x < -30)
       {
          I_x = -30;
       }
    if (I_y > 30)
       {
          I_y = 30;
       }
    if (I_y < -30)
       {
          I_y = -30;
       }
    if (I_z > 30)
       {
          I_z = 30;
       }
    if (I_z < -30)
       {
          I_z = -30;
       }
    x_adder = P_x + I_x + D_x; y_adder =
    P_y + I_y + D_y;
 }
```

We then checked that our ESCs are connected properly with the `escRead()` function. We also called `elevatorRead()` and `aileronRead()` to configure our drone's elevator and the aileron.

We called `test_radio_reciev()` to test if the radio we have connected is working, then we called `check_radio_signal()` to check if the signal is working. We called all the stated functions from the `void loop()` function of our Arduino code. In the `void loop()` function, we also needed to configure the power distribution of the system. We added a condition, like the following:

```
if(main_power > 750)
  {
    stabilize();
  } else
  {
    zero_on_zero_throttle();
  }
```

We also set a boundary; if `main_power` is greater than 750 (which is a stabling value for our case), then we stabilize the system or we call `zero_on_zero_throttle()`, which initializes all the values of all the directions.

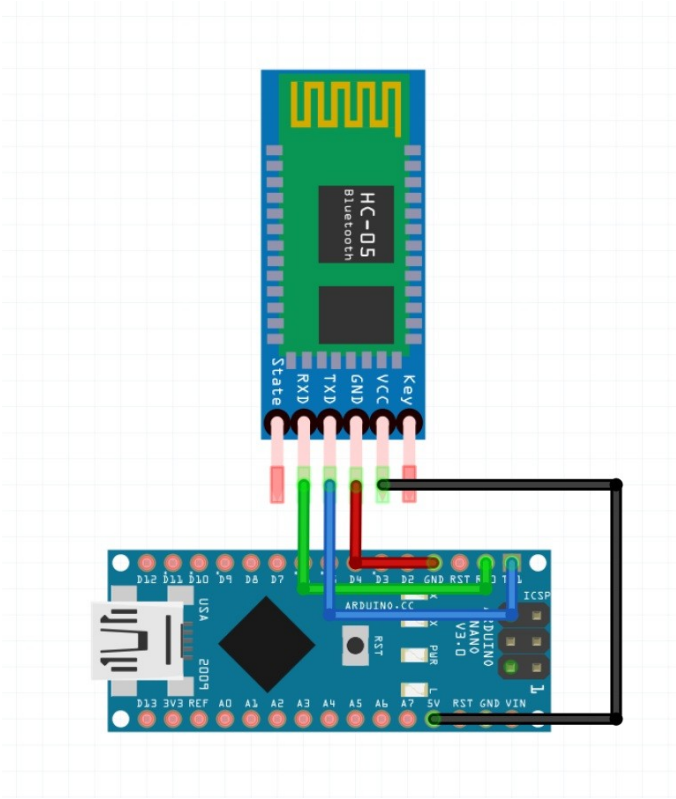
After uploading this, you can control your drone by sending signals from your remote control. Now, to make it a Follow Me drone, you need to connect a Bluetooth module or a GPS. You can connect your smartphone to the drone by using a Bluetooth module (HC-05 preferred) or another Bluetooth module as master-slave usage. And, of course, to make the drone follow you, you need the GPS. So, let's connect them to our drone.

To connect the Bluetooth module, follow the following configuration:

Arduino pin	**Bluetooth module pin**
TX	RX
RX	TX
5V	5V

GND	GND

See the following diagram for clarification:

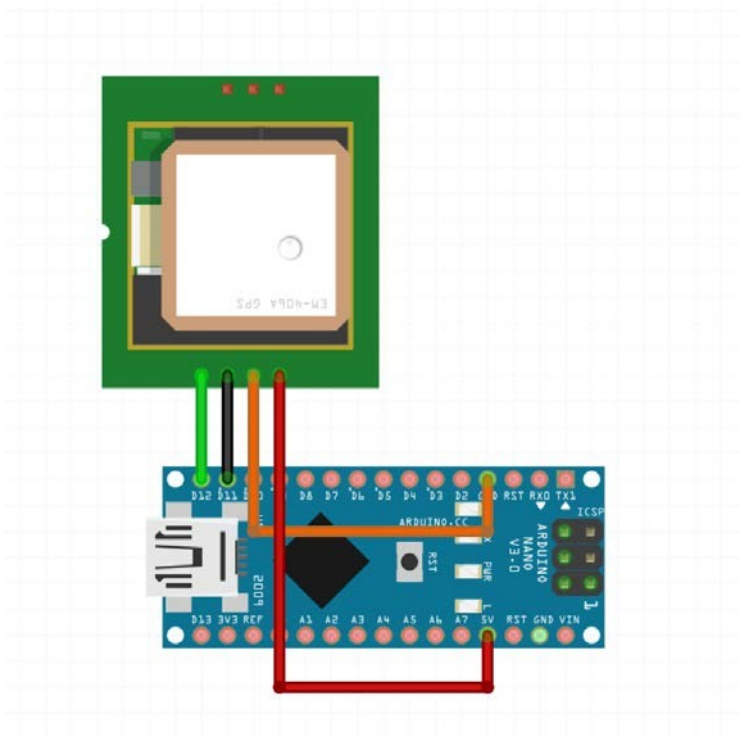


For the GPS, connect it as shown in the following configuration:

Arduino pin	**GPS pin**
D11	TX
D12	RX
GND	GND

5V	5V

See the following diagram for clarification:



Since all the sensors usages 5V power, I would recommend using an external 5V power supply for better communication, especially for the GPS.

If we use the Bluetooth module, we need to make the drone's module the slave module and the other module the master module. To do that, you can set a pin mode for the master and then set the baud rate to at least 38,400, which is the minimum operating baud rate for the Bluetooth module. Then, we need to check if one module can hear the other module. For that, we can write our `void loop()` function as follows:

```
if(Serial.available() > 0)
   {
     state = Serial.read();
   }
if (state == '0')
   {
      digitalWrite(Pin, LOW);
```

```
        state = 0;
    }
else if (state == '1')
    {
        digitalWrite(Pin, HIGH); state = 0;
    }
```

And do the opposite for the other module, connecting it to another Arduino. Remember, you only need to send and receive signals, so refrain from using other utilities of the Bluetooth module, for power consumption and swiftness.

If we use the GPS, we need to calibrate the compass and make it able to communicate with another GPS module.

We need to read the long value from the I2C, as follows:

```
float readLongFromI2C()
    {
        unsigned long tmp = 0;
        for (int i = 0; i < 4; i++)
          {
            unsigned long tmp2 = Wire.read(); tmp |= tmp2
            << (i*8);
          }
        return tmp;
     }
float readFloatFromI2C()
    {
        float f = 0;
        byte* p = (byte*)&f;
        for (int i = 0; i < 4; i++) p[i] =
        Wire.read();
        return f;
    }
```

Then, we have to get the geo distance, as follows, where `DEGTORAD` is a variable that changes degree to radian:

```
float geoDistance(struct geoloc &a, struct geoloc &b)
    {
          const float R = 6371000; // Earth radius float p1 = a.lat *
          DEGTORAD;
          float p2 = b.lat *  DEGTORAD;
          float dp = (b.lat-a.lat) * DEGTORAD; float dl = (b.lon-
          a.lon) * DEGTORAD;
          float x = sin(dp/2) * sin(dp/2) + cos(p1) * cos(p2)
          * sin(dl/2) * sin(dl/2);
          float y = 2 * atan2(sqrt(x), sqrt(1-x)); return R * y;
       }
```

We also need to write a function for the Geo bearing, where `lat` and `lon` are latitude and longitude respectively, gained from the raw data of the GPS sensor:

```
float geoBearing(struct geoloc &a, struct geoloc &b)
{
    float y = sin(b.lon-a.lon) * cos(b.lat);
    float x = cos(a.lat)*sin(b.lat) - sin(a.lat)*cos(b.lat)*cos(b.lon-a.lon); return atan2(y, x) * RADTODEG;
}
```

You can also use a mobile app to communicate with the GPS and make the drone move with you. Then the process is simple. Connect the GPS to your drone and get the TX and RX data from the Arduino and spread it through the radio and receive it through the telemetry, and then use the GPS from the phone with DroidPlanner or Tower. You also need to add a few lines in the main code to calibrate the compass. You can see the previous calibration code. The calibration of the compass varies from location to location. So, I would suggest you use the try-error method. In the following section, I will discuss how you can use an ESP8266 to make a GPS tracker that can be used with your drone.

GPS Tracker using ESP8266

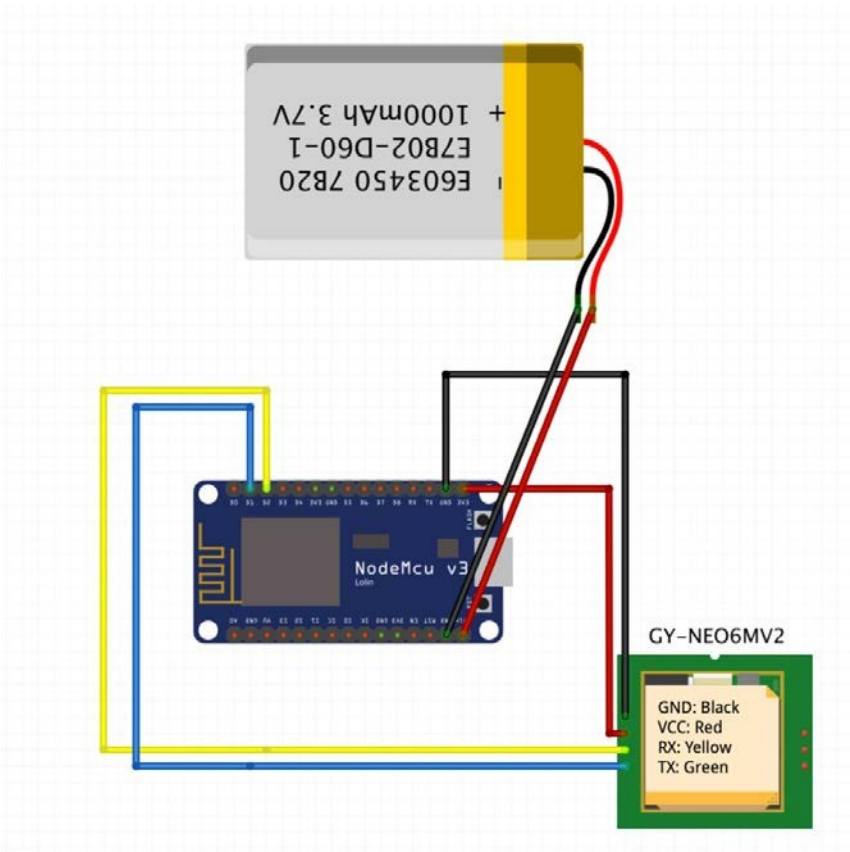
We will use the following components:

- NodeMCU (it has a built-in ESP8266)
- GPS receiver (you can use the Ublox NEO-6M GPS Module)
- A few cables
- Power source
- A smartphone (for the Blynk)

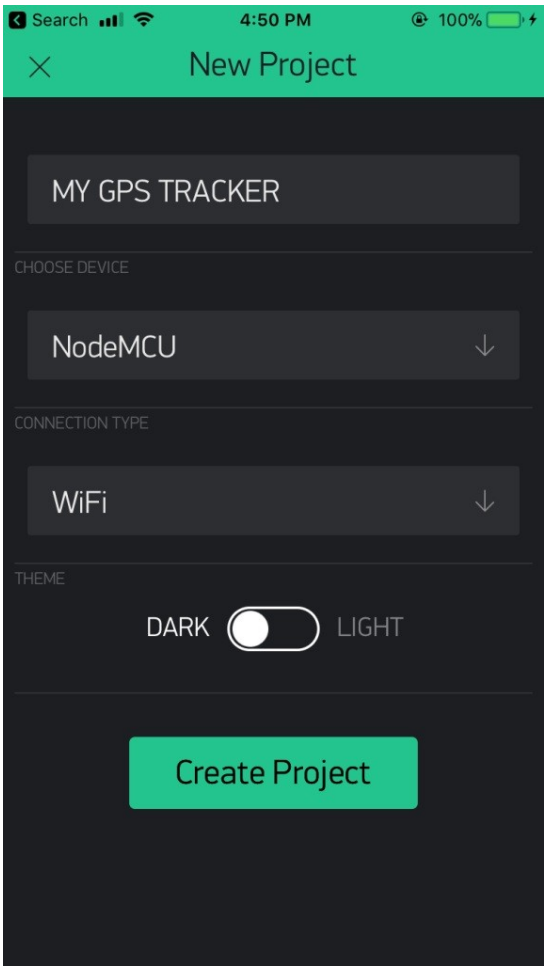
To make the connection, follow the pin configurations with NodeMCU and the GPS receiver:

NodeMCU	**GPS receiver**
D1	TX
D2	RX
GND	GND
VCC	3V

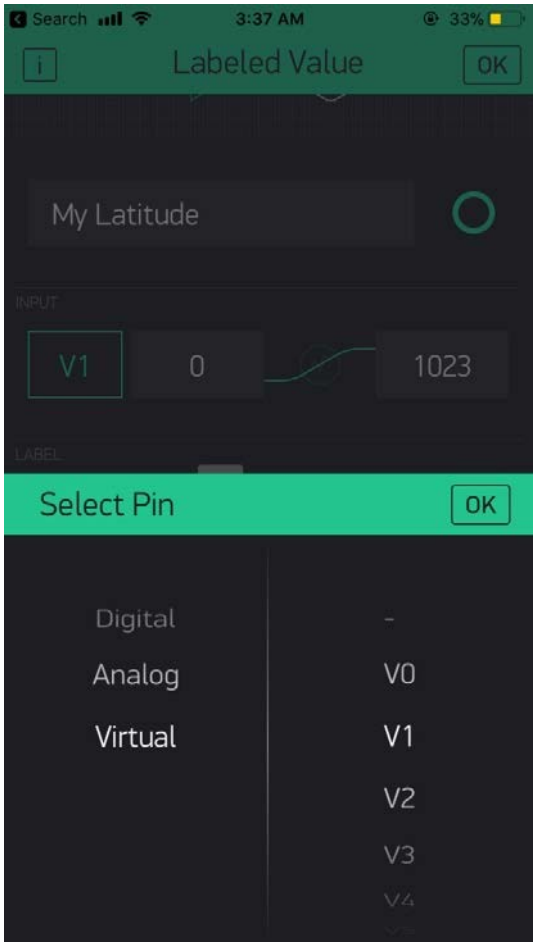
See the following circuit for clarification:



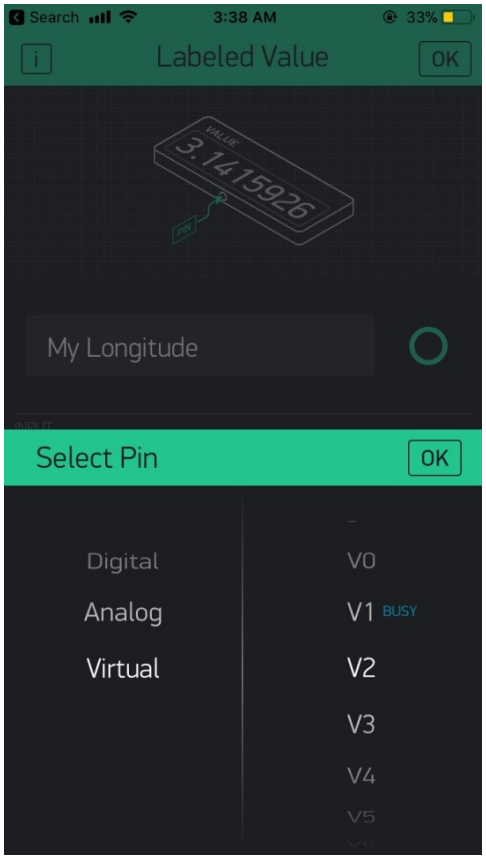
1. Now, connect the NodeMCU to the computer and open the Arduino application to program the NodeMCU. Before uploading the code, let's open our Blynk app.
2. Create a new project, as we discussed in Chapter 3, *Preparing Your Drone for Flying*.
3. Remember the authentication code sent to the email address you registered with the app; this will be needed later. In my case, I have set my project name as MY GPS TRACKER, device type as NodeMCU, and connection type WiFi, as follows:



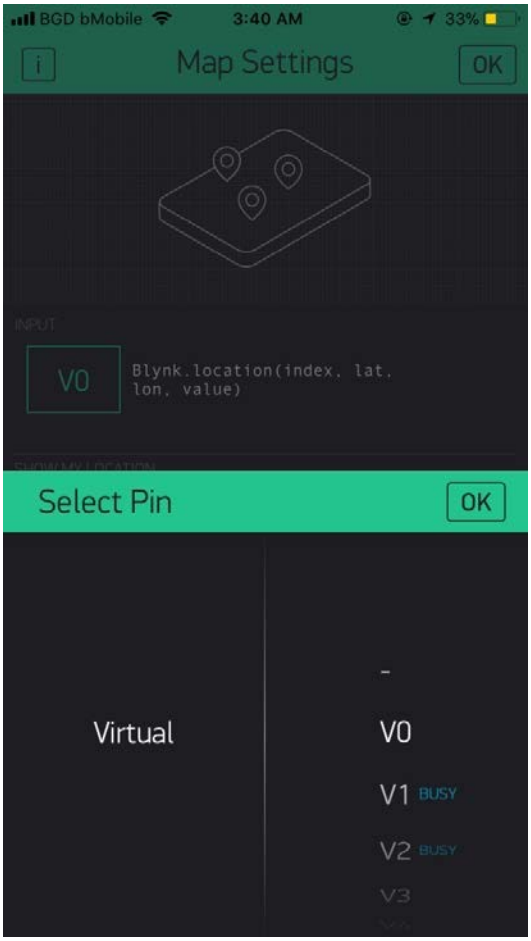
4. To see the real-time location in text value, add two text fields with Labeled Value.
5. For latitude, set the pin type Virtual and pin V1:



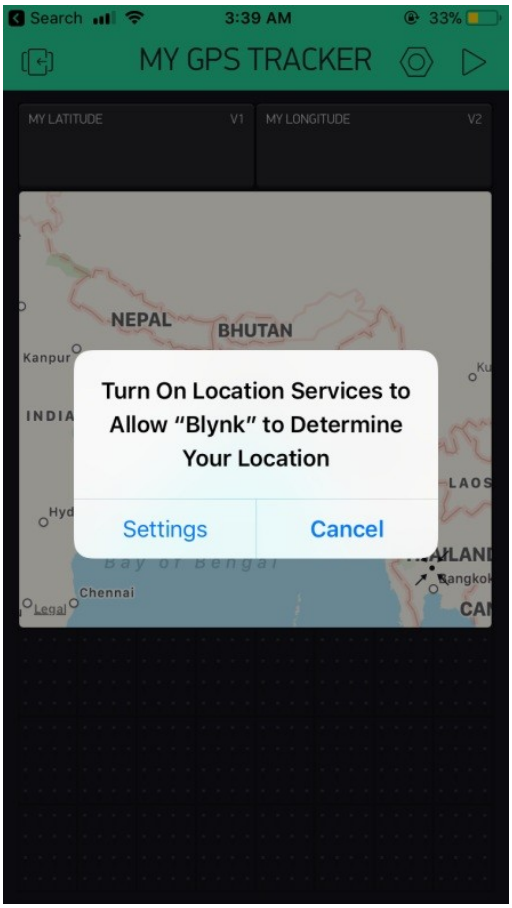
6. For longitude, select pin type Virtual and pin V2, as follows. You may change the color of the text by tapping the circular color icon:



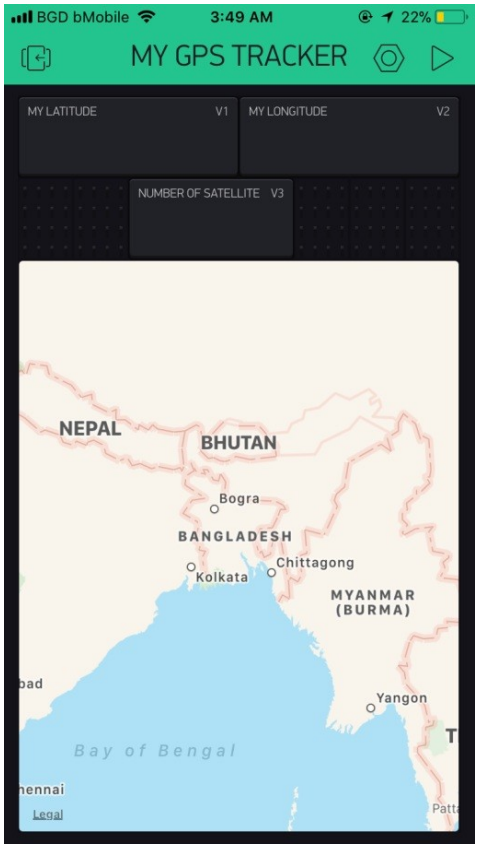
7. Now, add the map from the widget menu and select pin type to be Virtual and number to be V0:



8. The Blynk application will need your permission to use your phone's GPS. You may see the following if the location service is turned off:



9. We may need to see how many satellites there are, so select a Display Value or Labeled Value to the screen with Virtual pin and V3 configuration. Our visual should look like the following:



10. Now we need to code for the NodeMCU. We need to include the following libraries in our code:

```
#include  <TinyGPS++.h> #include
<SoftwareSerial.h> #include
<ESP8266WiFi.h> #include
<BlynkSimpleEsp8266.h>
```

11. To add these libraries, you need to download the libraries and install them to your Arduino software. To install the libraries, go to Sketch | Include Library | Add .zip Library and select the folder you can download from http s://github.com/blynkkk/blynk-library/releases/download/v0.5.0/Blynk_Release_v0.5.0.z ip and https://github.com/SOFTowaha/FollowMeDrone/blob/master/TinyGPS-master.zip.
12. After installing the libraries, let's declare some variables, as follows:

```
#define BLYNK_PRINT Serial //for defining the Blynk Serial static const int RX = 4,
TX = 5;
static const uint32_t GPSBaud = 9600;
```

13. Form the pinout of the NodeMCU, you can see that pin $_4$ is actually D2 and pin $_5$ is D1. We declared the GPS baud rate to be $_{9600}$. If this speed does not work with your GPS, then choose a lower baud rate such as half of $_{9600}$.

14. Now, create an object for the TinyGPS:

```
TinyGPSPlus mygps;
```

15. Since we have added a virtual pin V0 for our map, we need to define it:

```
WidgetMap myMap(V0);
```

16. To start communicating, we need to pass the TX and RX value to the SoftwareSerial:

```
SoftwareSerial test_GPS(RX, TX);
```

17. The Blynk timer and the number of satellites should be declared too:

```
BlynkTimer timer; float
noofsats;
```

18. For the authentication and the Wi-Fi connection part, add the following lines:

```
char auth[] = "********"; //This key can be found on your email char ssid[] = "********";
//Your WiFi name
char pass[] = "********"; //Your WiFi Password
```

19. Now, we will write our void setup() function. Inside the function, we will start our Blynk connection and check the GPS. In the void loop() function, we will display the latitude, longitude, and number of satellites in the Blynk. The full code can be found at https://github.com/SOFTowaha/FollowMeDrone/ blob/master/GPS_Blynk.ino.

20. Now, upload the code to the NodeMCU after connecting it to the computer.

You can see some information on the Serial Monitor of the Arduino software. To see the real-time data only, start the project from the Blynk application and you will see your location in the Blynk:

You can add the same technique to your Follow Me drone and control it from your Blynk application.

Summary

In this chapter, we have learned how we can build a Follow Me-type drone. We have learned how we can use DroidPlanner 2 and Tower to configure our drone to be a Follow Me drone. We have also learned how we can build a custom drone from scratch. In Chapter 5, *Building a Mission Control Drone*, we will do something even more exciting. We will build a mission control drone that will do a task given by you, and then return to you. I hope you will love to build one for you. So, why wait? Brace yourself for the next adventure.

Building a Mission Control Drone

In Chapter 4, *Building a Follow Me Drone*, we learned how we can build a Follow Me-type drone by using a smartphone or an external USB dongle. In this chapter, we will build or modify an existing drone to do something on its own. Actually, not on its own, but by our commands. Before going any further, let's discuss mission control drones.

Mission control drones are one of the common and most useful drones these days. We have seen how Amazon ships its products by such drones. They ship the packages and return to the base. The other types of mission control drones do surveys. They fly in an area and collect data. In this chapter, we will build such a kind of drone. We can do this with ArduPilot. So, let's get started.

Surveying with a drone

In Chapter 3, *Preparing Your Drone for Flying*, we have built a DIY drone using ArduPilot. We will first set up our existing drone for doing surveys. We do not need to change any kind of hardware in that system:

1. Open the Mission Planner software and go to the Config/Tuning page of Mission Planner and select **Planner**. You will see the following page:

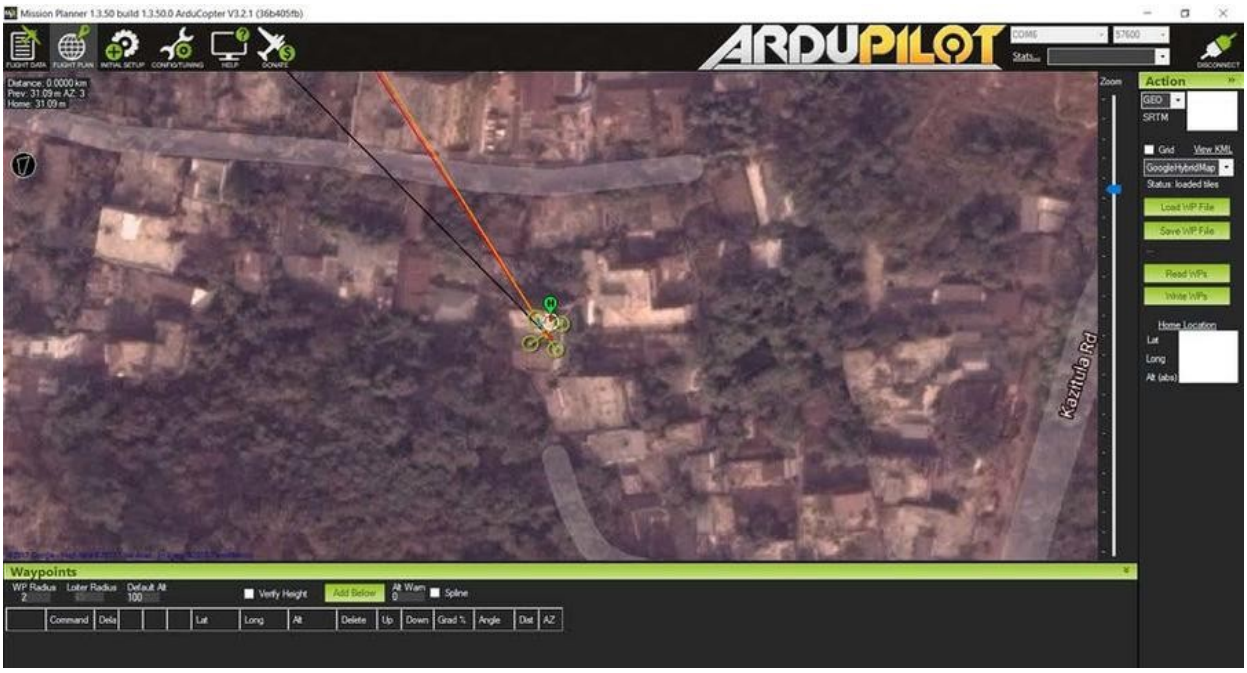
2. Now, choose a suitable unit for the distance and the speed of the copter, because these are vitally important to send a drone somewhere precisely.

3. From the top-left corner of the Mission Planner, select Flight Plan, and you will see the following screen:

This is just a map, and you can see a green pointer which says Home/H when the mouse is hovered over it. We can click and move the pointer anywhere we want.

4. We first place the pointer from where we will depart the drone to our destination:

Remember, this is very important for a mission control drone because this is used for the returning point of the drone. So, zoom in for placing the pointer to a certain place. I chose the rooftop of my house from where I will send the drone.

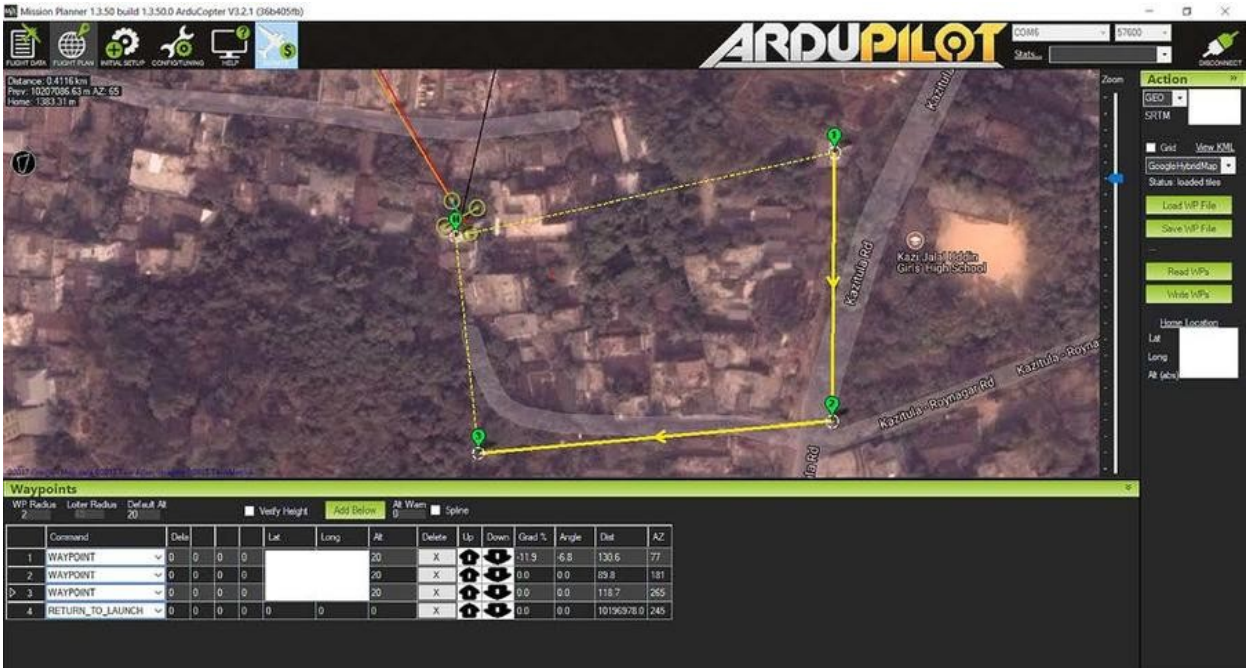


5. Now, under the map screen, you can see a panel called Waypoints. We need to change some settings there too:

6. Set the WP Radius between 2 and 5 meters, and the suitable Loiter Radius if you use as a Follow Me drone. Select a perfect altitude in the Default Alt box.

   Make sure that, where the drone will fly, there are no obstacles at that height, and set the altitude slightly higher than that height. If the region is hilly, or there are some houses, then select the Verify Height Box for avoiding obstacles.

7. Now, we need to select a place to which we want to send our drone. I have selected three waypoints by clicking the left button of the mouse, as you can see on the following screenshot:



I want to hover my drone to these three selected points and see what is happening there and come to the place from where I launched the drone. So, I need to set the **Return to Launch** (**RTL**) option on the waypoints.

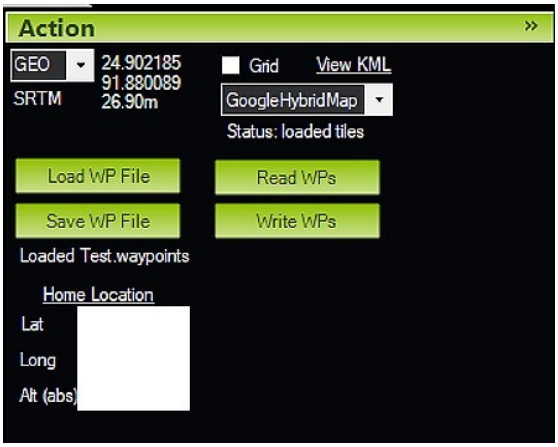
8. Click Add Below on the Waypoints panel and you will see a new waypoint.

*Before clicking the **Add Below** button, make sure you select the last waypoint so that the next waypoint of the last destination is the RTL waypoint.*

9. Then click the command menu and select the RETURN_TO_LAUNCH option:

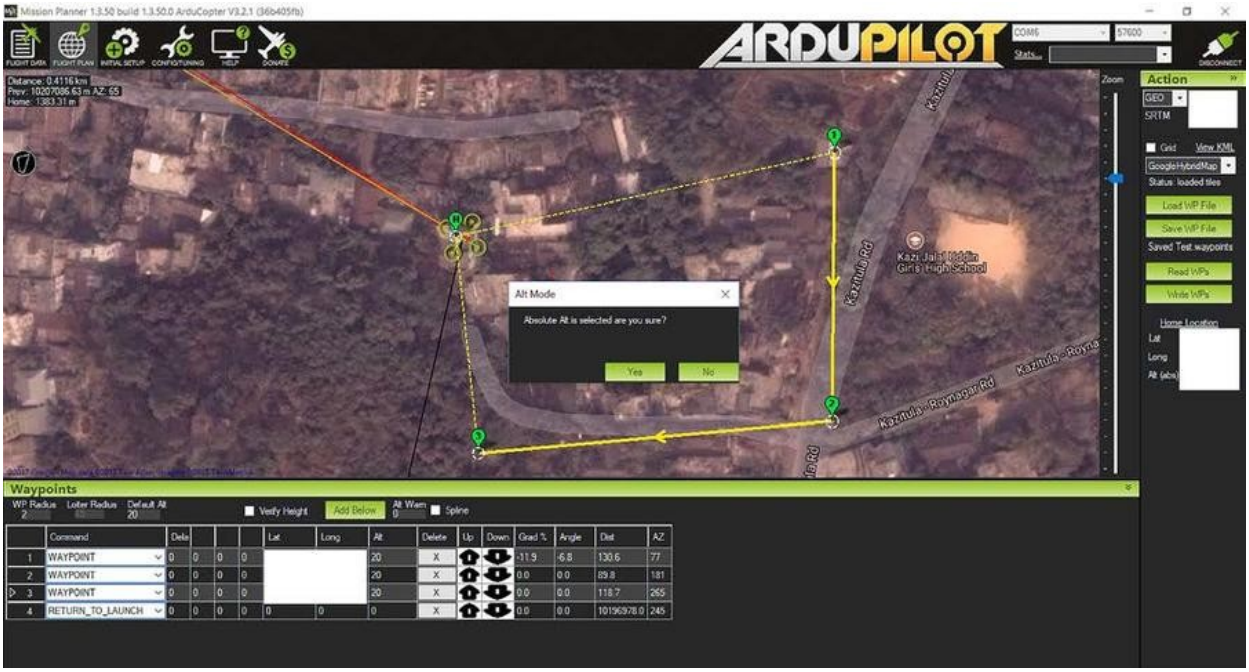
10. That's it. Your survey mission control drone is ready to do the survey in the air. You need to save the waypoint file to write it to ArduPilot. To do that, right-click on the map and go to File Load | Save | Save WP File:

11. Give a name to your waypoint file and click on Save. You can reuse it by clicking the Load WP File on the right-side of the screen. You can change the type of the map and add some extra features to the map from the right-side of the map:



12. Now all we need to do is load the mission to ArduPilot. You can do this by connecting the drone to the computer via telemetry or USB cable. Check for the correct COM port or select the Auto mode. Now, click on the Connect button and select the Load WP File from the right-side of the map (or right-click on the map). Select the file we just saved and hit the Write WPs button. You will see a pop-up message, and after that your copter will

memorize the places you just uploaded to it. You can erase the memory by uploading another waypoint file or resetting the waypoint:



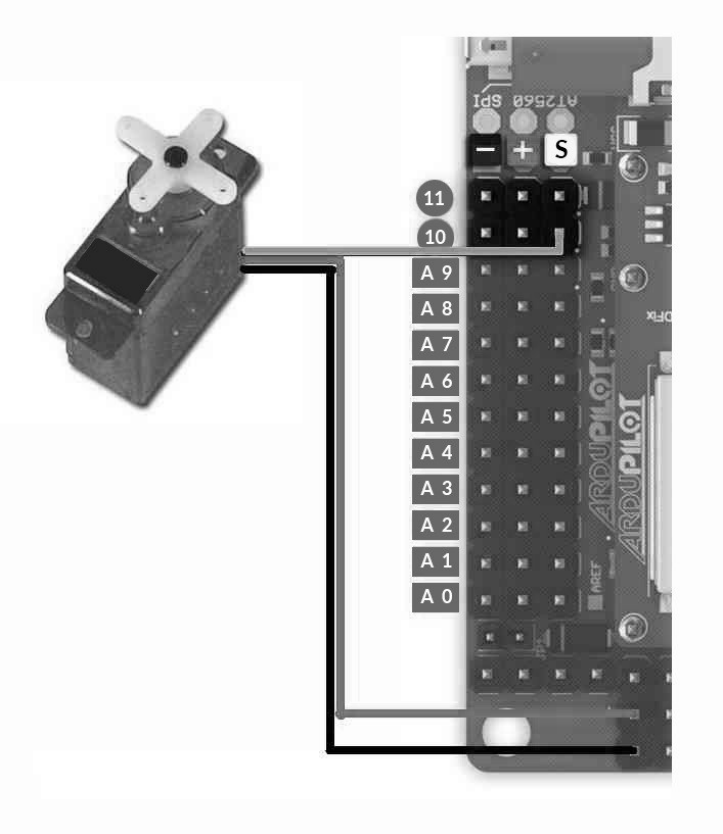
13. Now, take the drone to the place where we selected the Home/H point, and arm it (in auto mode). Our drone will fly to the three points we have selected and then finally it will return to the Home/H point. The timing of hovering depends on the speed you selected, on the **Config/Tuning** page. To select the auto mode, go to **Config/Tuning** and select Flight Mode. You can set any mode you want to be auto. Play with other settings too. Don't forget to hit the Save Modes button after changing any mode:


Mission Planner 1.3.50 build 1.3.50.0 ArduCopter V3.2.1 (36b405fb)
FLIGHT DATA
FLIGHT PLAN
INITIAL SETUP
CONFIG/TUNING
HELP
DONATE
ARDUPILOT
COM6
57600
Stats
DISCONNECT
Flight Modes
GeoFence
Basic Tuning
Extended Tuning
Standard Params
Planner
Current Mode: Stabilize
Current PWM: 5: 900
Flight Mode 1
Flight Mode 2
Flight Mode 3
Flight Mode 4
Flight Mode 5
Flight Mode 6
Stabilize
Acro
AltHold
Auto
Guided
Loiter
RTL
Circle
Land
Drift
Sport
Flip
AutoTune
PosHold
Brake
Throw
Avoid_ADSB
Guided_NoGPS
Simple Mode
Super Simple
PWM 0 - 1230
PWM 1231 - 1360
PWM 1361 - 1490
PWM 1491 - 1620
PWM 1621 - 1749
PWM 1750 +
Simple and Super Simple description

Using drones and delivery man

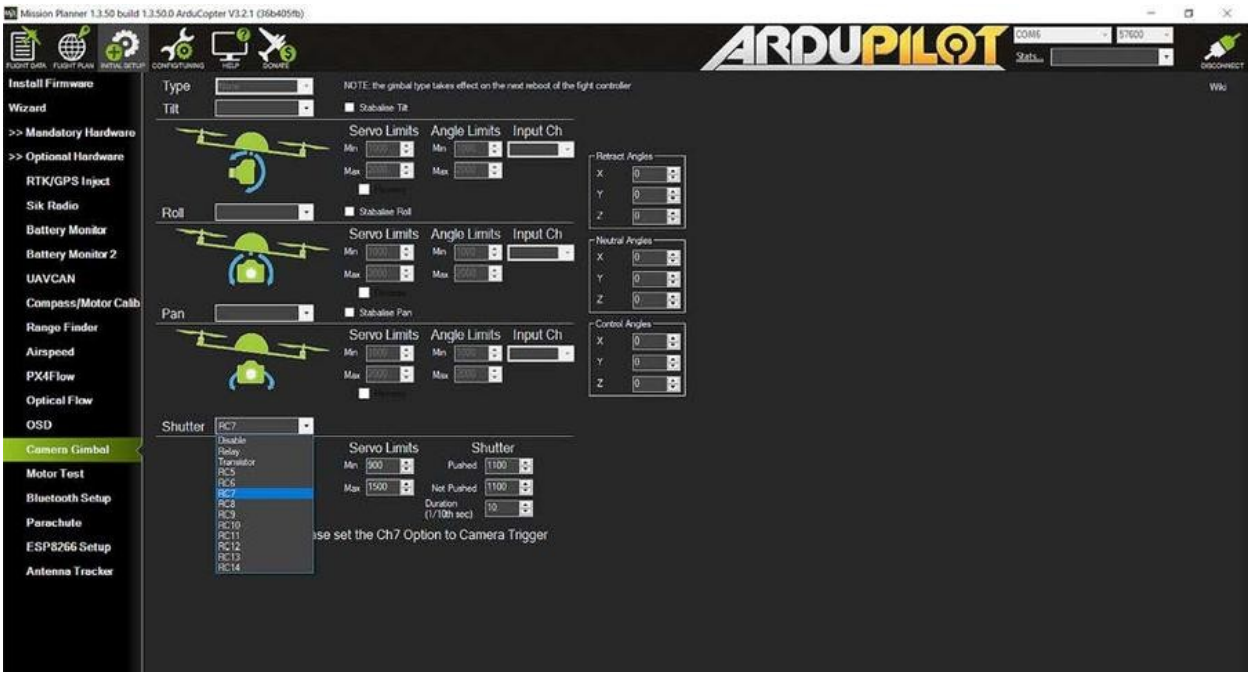
In this section we will ship or drop a package to a destination by our drone. This is a tricky part in the Mission Planner software. There is no direct way to do it. We can modify the camera shutter button to drop a package from our drone. We need to connect a servo motor with ArduPilot, and we will trigger a signal to the channel we connect and move the servo. If we have a placeholder connected to our servo, we can drop the package from the placeholder by triggering the servo from our remote or the Mission Planner software.

Let's connect the servo motor first. For our quadcopter, we have used one-four output channels of ArduPilot. On any other channel, we will connect a servo, as shown in the following picture:



1. Now, go to the initial setup of the Mission Planner software, and from Optional Hardware, select Camera Gimbal. At the bottom you will see the

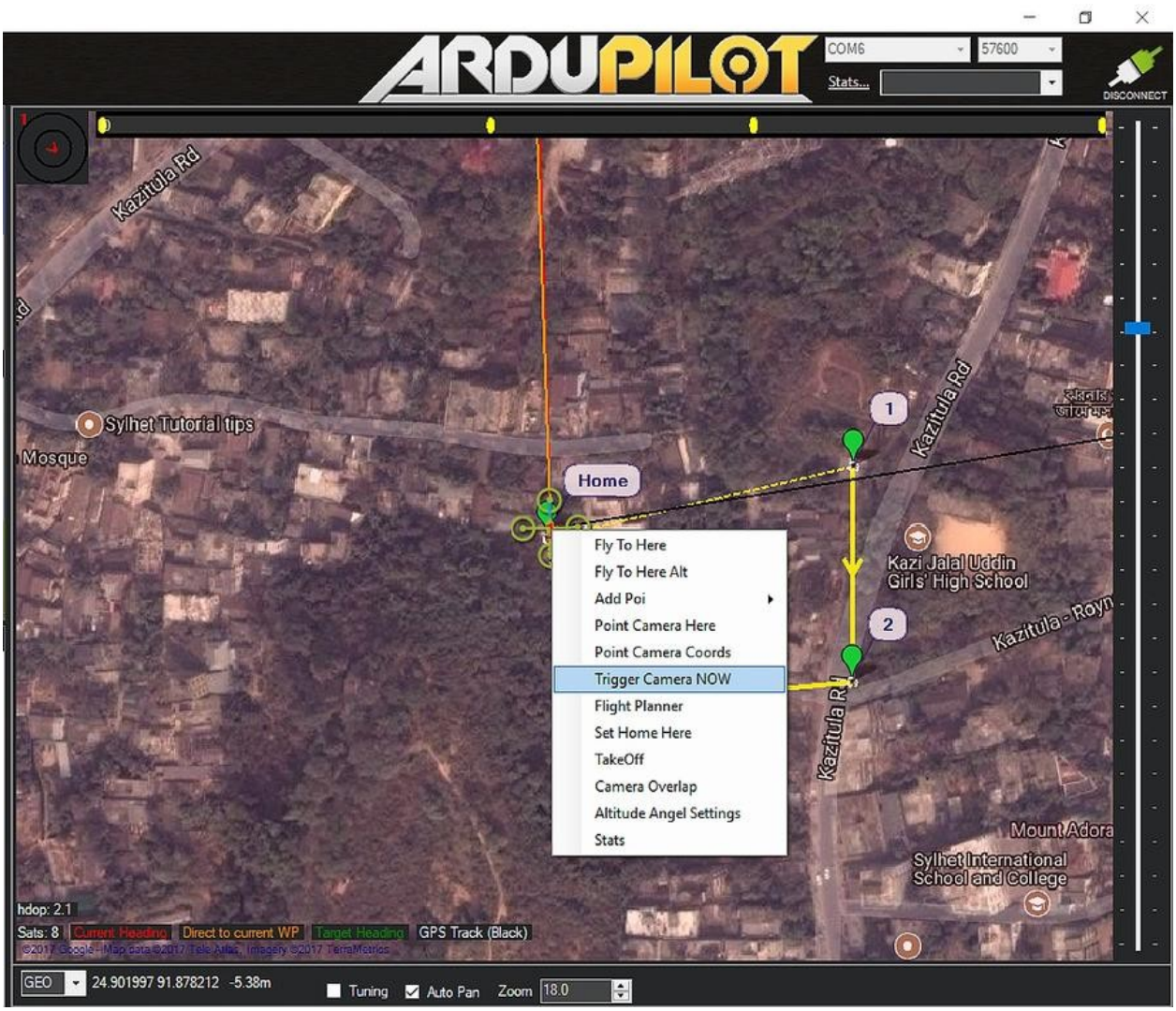
camera Shutter option. Select an unused channel (I chose $_{CH7}$) and set the Shutter Pushed to $_{1100}$ for a better push of the servo motor. You can also change the minimum and maximum of the servo rotation according to your need:



Here is a suggestion of what type of gimbal you can use to your drone:

2. Now, let's get started with dropping the package to a destination. To do that, you need to select a waypoint from the **Flight Plan** menu. Then click the Add Below button and set it to RETURN_TO_LAUNCH. Save the waypoint file and connect your drone to the computer. Load the saved waypoint file and write it to ArduPilot. Then attach your package to your gimbal in such a way that when the gimbal is triggered by the servo, the package falls:

3. Now, arm your drone and wait for it to reach the destination. Then, from the Flight Data screen, right-click on the map and click the Trigger Camera NOW option:



The package will be dropped at the destination and your drone will return to the launch point, as described in the waypoint.

Some other tweaks with the Flight Plan screen

You can select the waypoints from the WayPoint panel, as shown before. When you select a waypoint, the latitude and longitude are added to the waypoint. It means the drone will follow the position and go there by following the shortest path algorithm. We can do some actions on the location points by changing the command from the WayPoint panel. Please try them yourself and let me know what happens and which command you liked most.

Communicating with the drone via the ESP8266

In this chapter, we have successfully delivered a package to a place by using our mission control drone. But mission control needs to be aware of the weather and the environment for a successful flight and mission. So, if we can monitor the weather surrounding the drone, we can fly it safely and the package can be delivered without any problem.

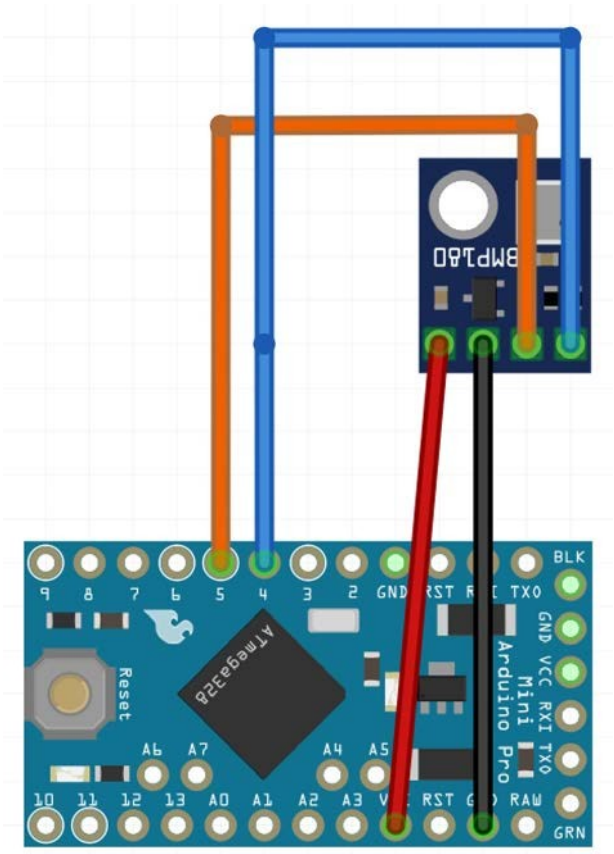
To monitor the weather, we need the following things (for demonstration purposes, we will only look at the air pressure in this chapter):

- A barometric pressure sensor (BMP 180 preferred)
- An Arduino (pro mini preferred)
- An ESP8266 transreceiver module

Firstly, connect the Arduino to the barometer sensor. The pin configuration is as follows:

Arduino	Barometer sensor
GND	GND
VCC	VCC
A4	SCL
A5	SDA

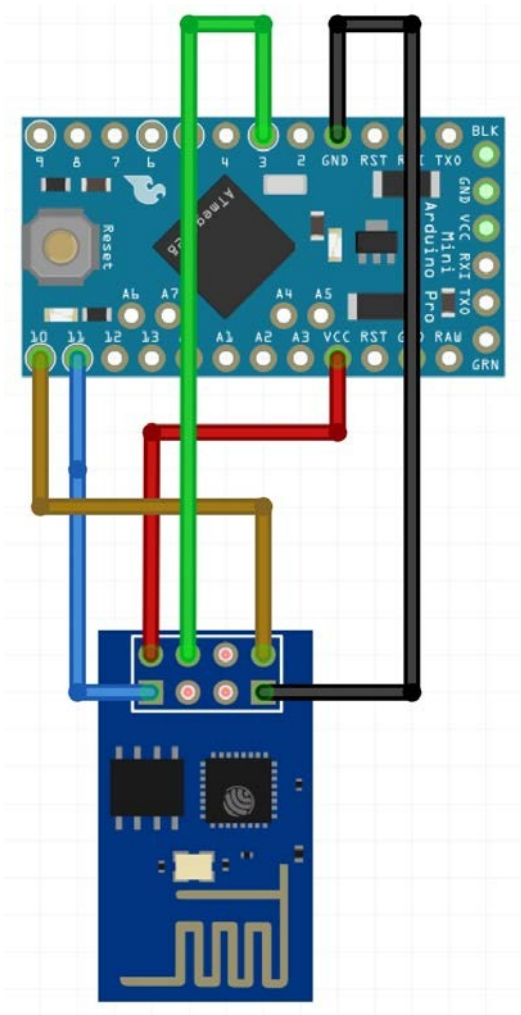
See the following circuit diagram for further clarification:



Now, connect the ESP8266 to the Arduino. The pin configuration is as follows:

Arduino	**ESP8266**
VCC	VCC
GND	GND
10	UTXD
11	TRXD
3	RST

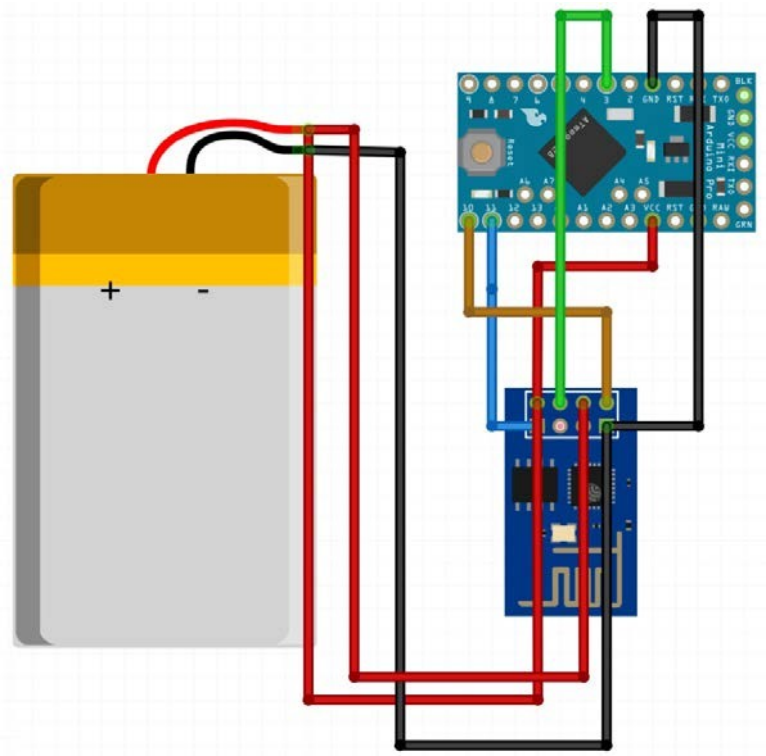
See the following circuit diagram:



Also, we need to connect a 3.3V power supply (for example, a battery) to the system. The configuration will be as follows:

ESP8266	**Power supply**
VCC	3.3V
GND	GND
CH_PHD	3.3V

See the following diagram for more clarification:



Now, we need to configure a server from which we will monitor the weather. We can use any kind of server, but I liked the EasyIoT (https://easyiot-cloud.com/) server:

1. Create an account there and let's get started:

    - We will upload code to our Arduino. The full source code can be found at https://github.com/SOFTowaha/MissionControl-Drone/blob/master/Baromet erCode.ino
    - We can discuss the main functions of the code
    - We used the following libraries in our code:

```
#include <Esp8266EasyIoT.h>
#include <SFE_BMP180.h> #include
<ESP8266WiFi.h>
```

    - To install the first library to Arduino, download all the files from https://github.com/iot-playground/Arduino/tree/master/Esp8266EasyIoT and install

them from Arduino software or add them to the Arduino software's library

- To install the barometer library, download and extract the ZIP file (https://github.com/SOFTowaha/MissionControl-Drone/blob/master/SFE_BMP180.zip) and install it in the Arduino/libraries folder
- To install the ESP8266WiFi library, all you need to do that is to go to the Board Manager install ESP8266

2. From the code, define the drone's altitude:

```
#define ALTITUDE
```

Which is important to get correct pressure data.

3. We declared our hardware as follows:

```
SFE_BMP180 bmp180;
Esp8266EasyIoT esp;
```

4. The kind of data we want to see from the server needs to be defined in the code too. We want to see temperature, pressure, and forecast in the server. So, define them as follows:

```
Esp8266EasyIoTMsg msgTemp(CHILD_ID_TEMP, V_TEMP);
Esp8266EasyIoTMsg msgPress(CHILD_ID_BARO, V_PRESSURE);
Esp8266EasyIoTMsg msgForec(CHILD_ID_BARO, V_FORECAST);
```

5. Our void setup() function is simple. We need to set the baud rates for the ESP8266. For debugging on the Serial Monitor, we can also add another baud rate:

```
Serial1.begin(9600);
Serial.begin(115200);
```

6. We need to initialize our sensor, which is a pressure sensor. So, in the void setup()function, we need to check the barometer by adding a condition:

```
if (bmp180.begin()) Serial.println("BMP180 initialization
    successful");
else
    {
        Serial.println("BMP180 initialization failed\n\n");
        while(1);
    }
```

7. Here, `while(1)` will pause the system forever. Then we need to prepare the ESP8266 for other purposes by writing the following code, passing the parameters:

```
esp.begin(NULL, ESP_RESET_PIN, &Serial1, &Serial);
```

8. If the ESP8266 is connected successfully, we can pass a few other parameters to `esp.present()` to send our data to the ESP8266, and hence send it back to the server:

```
esp.present(CHILD_ID_TEMP, S_TEMP);
esp.present(CHILD_ID_BARO, S_BARO);
```

9. For the forecast, we have created a function in the code file:

```
calculateForecast(double pressure){
  }
```

   Here we have set some variables and conditions, depending on the time, and get the average data and show it for the next few minutes.

   The `void loop()` function is loaded with some checking with the time and status of the sensors.

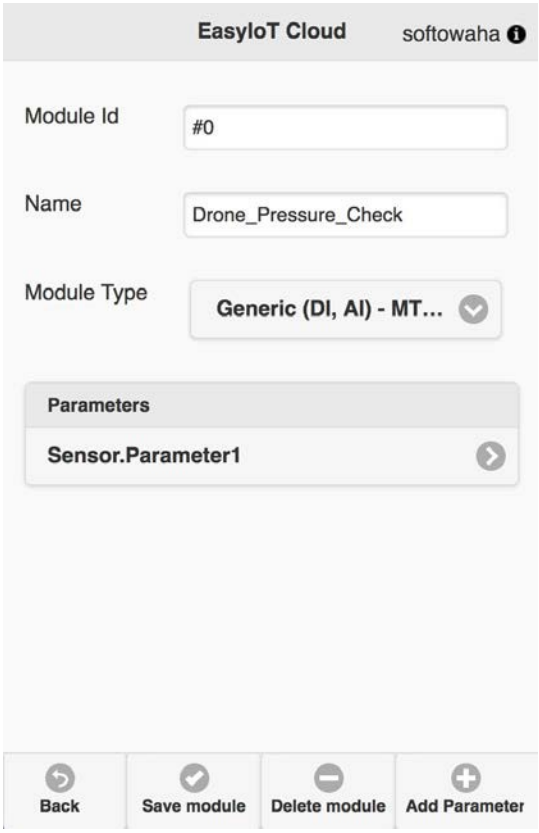
10. To connect to the EasyIoT server, we need to add the following credentials at the top of the code:

```
#define    AP_USERNAME    "******"
#define    AP_PASSWORD    "******"
#define INSTANCE_ID "******"
```

11. Make sure you connect your ESP8266 to the internet. To do that, define your router's ID and password in the code and connect it. For the drone, I would recommend using a portable router:

```
const char* ssid = "The WiFi Name"; const char*
password = "WiFI Password";
in the void setup() file add the following line. WiFi.begin(ssid, password);
```

12. Now, from the EasyIot dashboard, go to Configuration | Modules | Add Module. Give your module an ID (you cannot edit this), a Name**,** and select Module Type to be generic:



13. Save the module after adding the sensor parameters from our temperature and pressure.
14. Then go to Config | User Info and set the Instance Id for the module we have just made.
15. Now, you can upload the code to the Arduino and check the data shown in the server from the Module list.