University of South-Eastern Norway
Faculty of Technology and Maritime Sciences
-
Master Thesis in System Engineering With Embedded Systems
Kongsberg Department of Engineering
November 26, 2020

Bilgehan GUNAYDIN

M2M technologies for UAV-to-UAV communication

Abstract

Drones have gained huge popularity in the recent years, and with the huge increase in usage and demand has brought open a new Ad-Hoc network specifically made for drones. The existing Ad-Hoc networks, MANET and VANET, was shown to be unsuitable for flying drones as the current Ad-Hoc network were not fitted for 3D spaces and whereas only focused on the 2D plane.

This thesis will focus on the new Ad-Hoc network that specialises for drones, FANET. FANET Ad-Hoc network focuses on the 3D plane and thus brought open other new Routing Protocols. A design implementation with two drones will be presented in this thesis as the proposed solution. The design will have wireless M2M communication between the drones. The solution is able to achieve wireless communication through WiFi network and able to transmit video to the client through the intermediate drone.

Acknowledgements

First and foremost, I would like to convey my sincere and deepest appreciation to my supervisor Professor Antonio L. L. Ramos, for his guidance and inspiration during my undergraduate and graduate education. I am thankful for suggesting working on this topic and for the international experience I was able to acquire through his collaboration projects with prestigious higher education institutions abroad. This was a unique opportunity that enriched my learning experience at USN.

I would also like to express my deepest appreciation to Associate Professor Wemerson Delcio Parreira, for his comments and feedback that helped improving this thesis.

The collaboration with Associate Professor Parreira was made possible owing to the Branortech project (https://app.cristin.no/projects/show.jsf?id=2488728), a collaboration between USN and Norwegian University of Science and Technology (NTNU), with prestigious higher education institutions in Brazil. This project is co-funded by the Norwegian Agency for International Cooperation and Quality Enhancement in Higher Education (Diku) and the Coordination for the Improvement of Higher Education Personnel (CAPES), Brazil.

Finally, I would like to thank my family very much. I am very indebted to my family for being with me in the whole process of my master's thesis. Thank you for giving me the support and motivations.

Bilgehan Gunaydin
Kongsberg, Norway, June 11, 2021

Contents

List of Figures

List of Abbreviations

AMPS	**A**dvanced **M**obile **P**hone **S**ystem.
Bluetooth BR	**B**luetooth **B**asic **R**ate.
Bluetooth LE	**B**luetooth **L**ow **E**nergy.
CMDA2000	**C**ode **D**ivision **M**ultiple **A**ccess**2000**.
EDGE	**E**nhanced **D**ata rate for GSM Evolution.
FANET	**F**lying **A**d-Hoc **Net**work.
G	**G**eneration.
GLONASS	**Gl**obal **O**rbiting **N**avigation **S**atellite **S**ystem.
GPS	**G**lobal **P**ositioning **S**ystem.
GSM	**G**lobal **S**ystem for **M**obile communication.
HSPA	**H**igh-**S**peed **P**acket **A**ccess.
IEEE	**I**nstitute of **E**lectrical and **E**lectronics **E**ngineers.
LAN	**L**ocal **A**rea **N**etwork.
LTE	**L**ong-**T**erm **E**volution.
MANET	**M**obile **A**d-Hoc **Net**work.
MAVLink	**M**icro **A**ir **V**ehicle **Link**.
MIMO	**M**ultiple-**I**nput and **M**ultiple-**O**utput.
MMS	**M**ultimedia **M**essaging **S**ervice.
NMT	**N**ordic **M**obile **T**elephone.
QAM	**Q**uadrature **A**mplitude **M**odulation.
QZSS	**Q**uasi-**Z**enith **S**atellite **S**ystem.
TACS	**T**otal **A**ccess **C**ommunications **S**ystem.
SNS	**S**atellite **N**avigation **S**ystem.
SMS	**S**hort **M**essaging **S**ervice.
SOC	**S**ystem **O**n a **C**hip.
SIG	**S**pecial **I**nterest **G**roup.
SSID	**S**ervice **S**et **I**dentifier.
RREQ	**R**oute **R**equest.
UART	**U**niversal **A**synchronous **R**eceiver/**T**ransmitter.
UAV	**U**nmanned **A**erial **V**ehicle.
VANET	**V**ehicle **A**d-Hoc **Net**work.
W-CDMA	**W**ideband **C**ode **D**ivision **M**ultiple **A**ccess.
WiFi	**Wi**reless **Fi**delity.
WiMax	**W**orld-Wide **I**nteroperability for **M**icrowave **A**ccess.

Chapter 1

Introduction

Over the years the popularity of UAV have increased dramatically, and UAVs are becoming more common especially in households and in military. The reason for its drastic rise of popularity comes from its usability. A lot of different types of sensors can be attached to UAVs, thus increasing its capabilities of performing various tasks. Cameras to monitor environments, or sensors to detect movements as used in military are only a few of the examples of UAV' capabilities [39]. Different types of machine-to-machine (M2M) technologies have been applied on UAV based systems to be able enhance its strengths and minimize its weaknesses. In this thesis it will be focused on facilitation communication for data sharing and UAV cooperation in a mission. The proposed network is called FANET, a network that can be applied on multi-UAV systems. By adopting the multi-UAV system, more than one UAV can work together to achieve a more efficient performance throughout the system, thus leading to decreased latency, increased performance and a more robust system overall when compared to the single-UAV systems [34][11]. I would like to mention that creating a new solution is not a part of this project, rather to use an existing solution and implement the FANET network on two UAVs with the Mamba F405 micro-controller board installed on each UAV with a possibility to monitor each individual UAV with a smartphone or a computer.

There are different sizes of UAVs all over the world. And each different size has their own goal. According to CASA (Civil Aviation Safety Authority) drones less than 2 KG of weigh is considered as "very small". Two sets of very small drones will be the goal of the size for this project.

1.1 Problem statement

While flying a UAV there will be some key factors that cannot be overlooked. Those are latency, bandwidth and communication range. To reduce the overall latency of the system and to increase its bandwidth and communication range will result in a favorable UAV system. The most popular existing M2M technologies applied on UAVs are 4G, Bluetooth and WiFi (802.11n and 802.11ac) and according to [34], these M2M technologies are implemented in single UAV systems with a base station, thus the communication range is highly depended on the station. In places as rural areas where the existence of a base station is a concern will there be a cause for huge problem for single UAV systems [34]. Mavlink is one of many communication protocols used to control single UAV systems while needing a base station, mavlink is only able to control a single UAV systems with a base station and is unable to control multiple UAVs at once

[9]. The key factors when dealing with communication methods in UAVs are bandwidth, communication range, latency, but there are also few other factors that need to be taken into account in UAVs. Topology of the UAV network may change over time because we are in a three-dimensional environment, and that will cause variations in the x, y and z coordinates of the UAVs in a network and so forth, a variation in latency between the UAVs may be occurred [21]. It is also important for the UAVs to cooperate with each other, should be not forgotten that the gathered data from each individual UAV plays a big role in decision making [23].

1.2 Contributions

The purpose of this project is to create multi UAV system that is able to achieve better performance metrics and overcome the limitations when compared to single UAV system. In this project, the limitation of a single UAV system will be overcome, this will be done by creating two smaller drones instead of a big single drone. The two smaller drones will cooperate with WiFi technology by utilizing the FANET Ad-Hoc Network. The main objective of this project is:

- Facilitation the communication for data sharing and cooperation among drones in a mission.

The project is also divided in other tasks:

- Cooperation and decision making during missions is based on the combined information (or data) collected by the individual drones.
- The type of data that can be used are: a. Position and orientation in a x-y-z coordinate system (3D) b. Additional sensor data, as in audio and images from the surroundings.

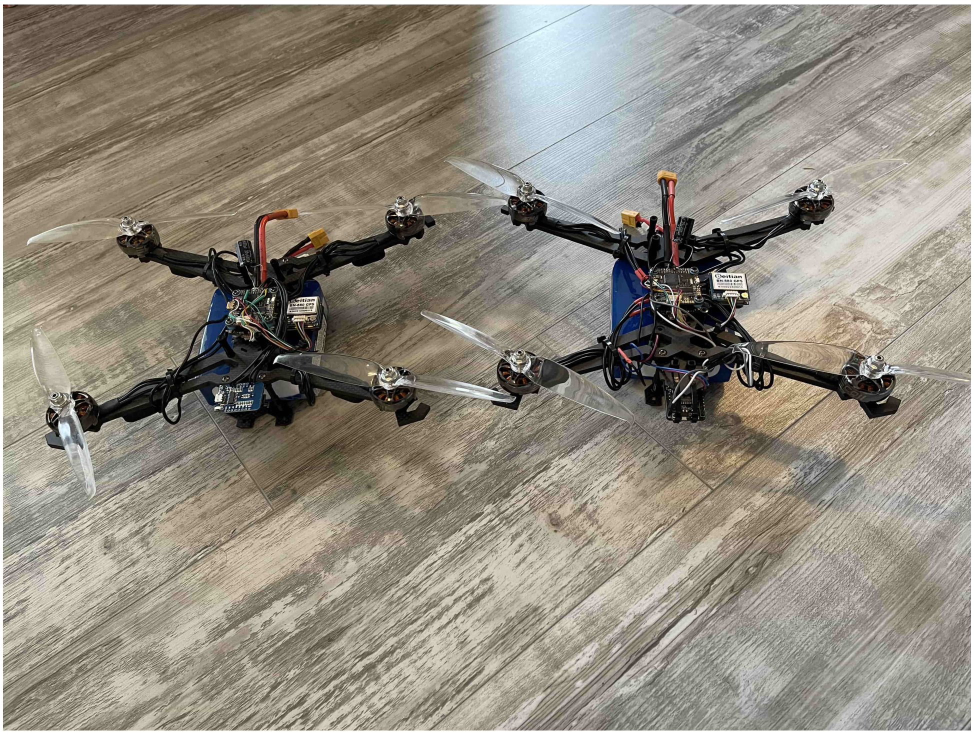


FIGURE 1.1: Design of a multi-drone system

1.3 Outline

The rest of the this thesis is constructed as follows. Chapter 2 discusses various wireless network protocols, including 1G to 4G, Bluetooth, satellite navigation systems, plus three of the most popular Ad-Hoc networks. Lastly, Several important Routing Techniques and Protocols are also been presented. Chapter 3 addresses the project design. Firstly, software and implementation details addressed in details along with the selected hardware parts. Finally, Chapter 4 presents a summary with conclusions and related future work for improvement of this project.

Chapter 2

Background

2.1 Wireless Network Protocols

In this section discusses wireless network and routing protocols. These protocols are the most essential technologies in UAVs. The Wireless network protocols are used to establish a network connection between the UAV to be able to exchange data wirelessly with each other using radio frequency signals.

2.1.1 Mobile networks

We have seen constant technological improvements from the mobile wireless networks over the past 40 years. In this section the mobile networks will be explained in a less detailed manner, only to have a glance of possible mobile networks that can be a probable solution to the project as to create a wireless communication between the drones.

1G Networks

1G is the first generation of the mobile wireless network and is also the least technologically advanced. The first generation of the wireless network was released in 1984 but its design started in 1970. 1G support radio signal frequencies between 824 to 894 MHz. 1G did support only analog voice transmission with maximum data bandwidth of 2.4 Kbps. The three most popular standards for 1G were AMPS, NMT and TACS, which were released in North America, Nordic countries and United Kingdom respectively as noted in [24]. 1G's bandwidth capability and its coverage put its selection for this project even beyond consideration.

2G Networks

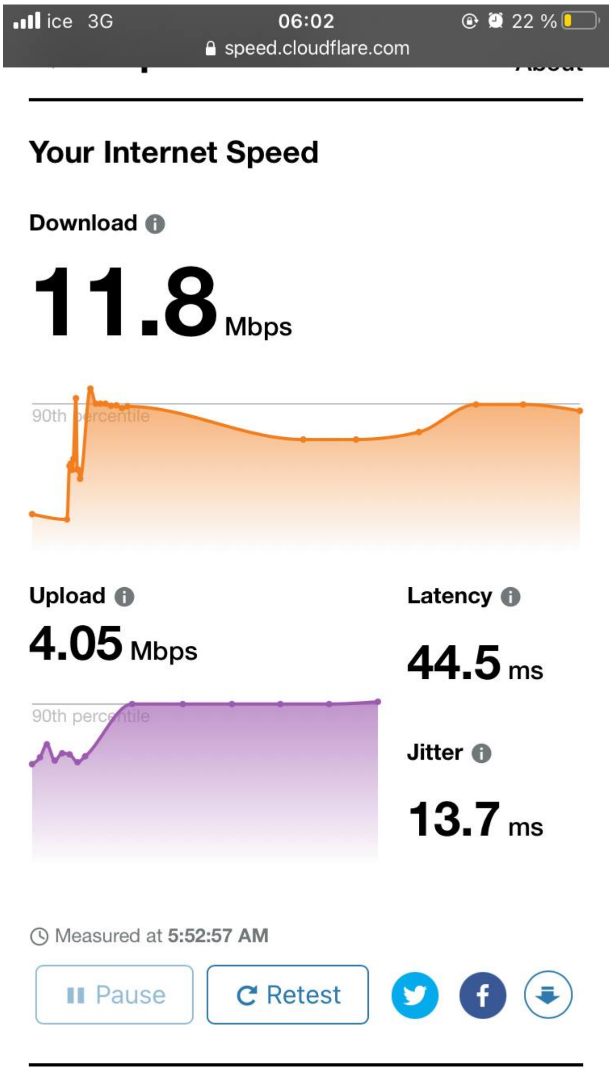
The second generation of the mobile wireless network is the first step towards the digitization of the wireless networks. As described in [43], with the arrival of the 2G mobile network various services were born, some of those were SMS, MMS, voice and data.

Several upgradements have occurred during the 2G network's lifetime; GSM is the first installment of the 2G network launched in 1991. GSM operated in 900 MHz and 1800 MHz frequency signals, as GSM handles digital voice in a adequate manner the data and internet services were a limiting factor. GSM had a download and upload link at 64 Kbps as stated in [40].

3G Networks

The third generation mobile wireless network was released on 2001 by the Japan. 3G supports three network interfaces that are also compatible to the previous network standards as explained in the 2G wireless network standards. Those 3G interfaces are WCDMA, CDMA2 and WiMAX. The 3G network supports higher speed and capacity compared to the previous network technologies [17]. By the release of the HSPA and HSPA+ standards which was an upgrade from the WCDMA technology, the capacity and the performance of 3G network were increased. HSPA has a theoretical maximum bandwidth of up to 14.4 Mbps and 5.76 Mbps download- and upload data-rates respectively, with a latency between 70-100 ms on the download link and even less than 50 ms on the upload link. The highest network standard for 3G was released later called HSPA+ that had an immense increase for both the download- and the upload links. According to [10], HSPA+ could provide bandwidth of 336 Mbps for the download link and 69 Mbps for the upload link. The announcement of HSPA+ made possible for better video streaming experience and real-time applications, like video calls, got an improved latency. 3G supports radio frequencies at 900 MHz and 2100 MHz. According to [32], 900 MHz is in use in rural places and 2100 MHz signal in urban places.

Figure 2.1, displays performance of the 3G network I tested from my smartphone. To test these performance metrics, the 3G network was provided by Ice Norway/Telia Norway. Latency achieved from this test is 44.5 ms. With download and upload link speed at 11.8 Mbps and 4.05 Mbps respectively, these speeds are upholding the download and upload links provided by the HSPA standard.



ice 3G 06:02 22 %

speed.cloudflare.com

Your Internet Speed

Download 11.8 Mbps

Upload 4.05 Mbps

Latency 44.5 ms

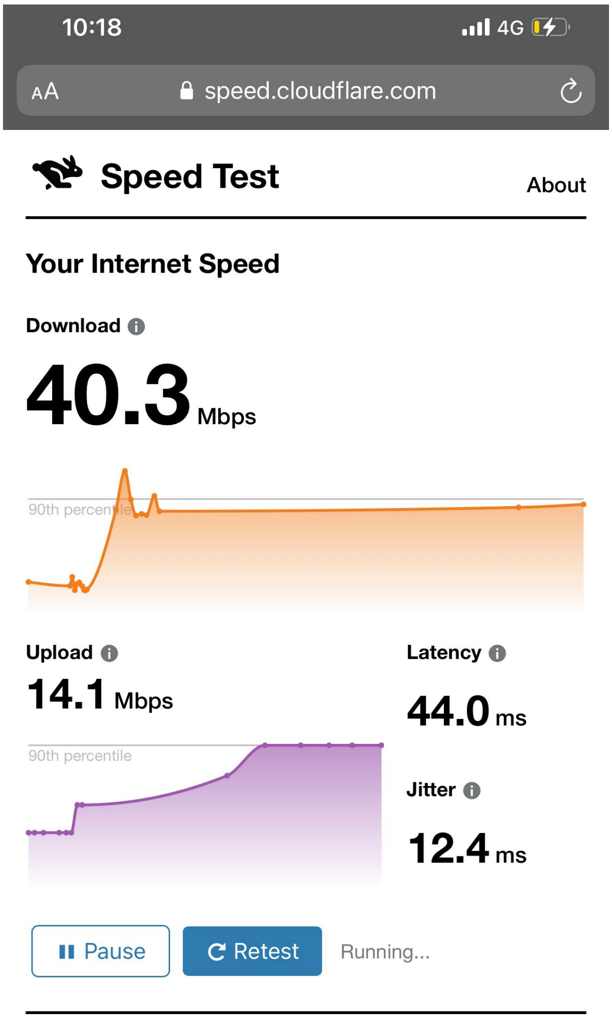
Jitter 13.7 ms

Measured at 5:52:57 AM

FIGURE 2.1: Performance metrics of 3G mobile network

4G Networks

The 4G network is the most used mobile wireless network currently. The design of the 4G work began in the year of 2000. The 4G network is an upgradement from the previous wireless network, namely 3G. With the 4G wireless network, faster speeds and broader network coverage have been increased. According to [25], the 4G network has at least two times more efficient at spectrum usage. Latency is improved compared to the 3G network. Two most popular technologies used in 4G wireless networking ares, LTE (Long-Term Evolution) and WiMAX (Worldwide Interoperability for Microwave Access). The first generation of LTE technology is able to achieve a download speed of 300 Mbps of data rates. With a latency of around 5ms for single path transport. The second generation of LTE network gained an increased spectrum efficiency and data rate speeds. The second generation LTE network is called LTE-Advanced. LTE-A was able to achieve 1000 Mbps and 500 Mbps download and upload data links respectively. The average performance of the LTE-Advanced was multiplied by three compared to its first generation technology. LTE-Advances was able to achieve less than 5ms for single path transport, hence less than the first generation LTE technology, according to Abed, Ismail, and Jumari, [3]. The figure 2.2, is showing the performance metrics of a 4G wireless network. As seen from the figure, 4G will get a faster speeds and a lower latency compared to the previous wireless network technology.



Speed Test

Your Internet Speed

Download 40.3 Mbps

Upload 14.1 Mbps

Latency 44.0 ms

Jitter 12.4 ms

FIGURE 2.2: Performance metrics of 4G mobile network

2.1.2 WiFi

According to Al-Alawi, WiFi is the name of group of wireless networking products appointed by IEEE and given the code-name 802.11. WiFi is used as internet access and has been the superior choice for wireless LAN. WiFi operates on unlicensed frequency signals. WiFi network is a very popular wireless internet primarily used at homes and throughout the years several WiFi standards have been integrated to the WiFi technology.

IEEE 802.11 was the first standard in the WiFi wireless LAN Technology released in 1997 as this was the first release of its technology the download and upload links were much slower than the up comings. The IEEE 802.11 standard was able to support up to 2 Mbps data speed rate. This standard was running at 2.4 GHz frequency band and had a transmission range of up to 100 meters.

IEEE 802.11a was the second installment of the WiFi technology released on 1999. IEEE 802.11a standard was the first time the WiFi technology went for the 5 GHz radio frequency, this arose both advantages and a disadvantages. The first advantage was the maximum download and upload link rates got an immense increase compared to the first standard. The download and upload link rates got a jump from 2 Mbps to 54 Mbps, thus a 27-fold increase. More than double of a radio frequency increase reduced to the transmission range down to 80 meters, and in addition the 5 GHz radio frequency has worse characteristics than 2.4 GHz radio frequency when it comes to penetrating obstacles, especially walls and therefore the transmission range will be reduced even more when there are hindrances in place.

IEEE 802.11b, the third installment is an upgrade of the first WiFi standard. The operated radio frequency is the same 2.4 GHz, and by adopting the same radio frequency the transmission range has maintained the same distance.IEEE 802.11b had an increase in download and upload link rates from 2 Mbps to 11 Mbps, compared to the first standard. IEEE 802.11b was released on at the same year as the IEEE 802.11a standard. Compared the the second installment IEEE 802.11a, the IEEE 802.11b had better maximum operational range and better at penetrating obstacles in hindrance, at the expanse of having close to 5-fold decrease in link speeds.

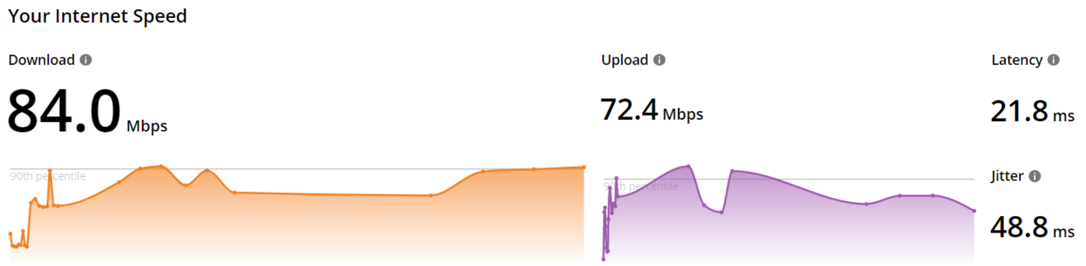
In 2003 a new standard called IEEE 802.11g was released for the WiFi technology. IEEE 802.11g was operational in the same 2.4 GHz radio frequency as IEEE 802.11b and the first IEEE 802.11 WiFi standards. The IEEE 802.11g WiFi standard uphold the same transmission range as the other 2.4 GHz standards, yet got data rate increase. Data rates at 54 Mbps was achieved, the same as the IEEE 802.11a standard without including the disadvantages it possessed.

According to [8] is the next installment of the WiFi technology IEEE 802.11n is applying MIMO technology. MIMO is the technology that uses multiple antennas to transmit and receive data, that allows for better throughput and data rates will increase. With the new MIMO technology technology applied, data link speeds may increase up to 900 Mbps. The IEEE 802.11n standard is able to operate at both 2.4 GHz radio frequency and 5 GHz radio frequency. IEEE 802.11n standard is able to take advantage of 40 MHz channel frequency as opposed to only 20 MHz channel on the previous standards, IEEE 802.11a,b, and g. The use of 40 MHz channel frequency will occupy twice as bandwidth capacity from the radio frequency band, but operating at 40 MHz channel will increase the network's download and upload link rates by 2-fold. The radio radio frequency at 2.4GHz has lower channels compared to 5 GHz frequency band, consequently the the 40 MHz technology is generally only applied to the 5GHz radio frequency band.

As described by Aruba-Networks,[7], IEEE 802.11ac is the next upgrade of the previous IEEE 802.11n WiFi standard released in 2013. IEEE 802.11ac includes all of the existing technologies from the previous standard IEEE 802.11n, but with an step-up upgrade for each technology. IEEE 802.11ac standard is able to operate at 80 MHz channel as the default and obligatory channel width, but 160 MHz channel as an optional choice for the user, this will reduce the total bandwidth capacity, thus reduce the total channels by 2-fold or 4-fold and at the same time increase the throughput by 2-fold or 4-fold based on which channel width in use. IEEE 802.11ac standard is only able to transmit radio frequency signals at 5GHz unlike the previous standard. 802.11ac supports up to 8 antennas to transmit and receive signals, hence the throughput will increase even more. The signal will be strengthened with the new 256-QAM technology for more efficient encoding. With three antennas and 80 MHz channel width use, the IEEE 802.11ac does have a download and upload data rates at 1300 Mbps.

IEEE 802.11ax is the latest available installment. IEEE 802.11ax is also called WiFi 6. WiFi 6 does have backward compatibility with all other previous WiFi standards and WiFi 6 supports both 2.4GHz and 5GHz radio frequency signals. The 256-QAM support from IEEE 802.11ac has been improved to 1024-QAM, and according to Afaqui, Garcia-Villegas, and Lopez-Aguilera, [5] the upgrade will increase the average performance of WiFi 6 network by 4 times.

The image below is showing the different performance attributes of the WiFi network I tested at home. By looking at the figure we can easily tell the latency performance of the network. Latency with 21.8 ms is showing a lower latency compared to other network as mobile wireless network. The test was performed on a WiFi 802.11ac protocol with a signal frequency at 5GHz.



Your Internet Speed

Download 84.0 Mbps

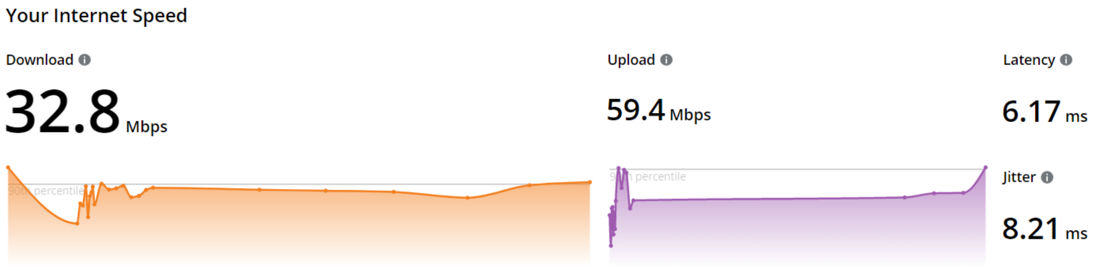
Upload 72.4 Mbps

Latency 21.8 ms

Jitter 48.8 ms

FIGURE 2.3: Performance metrics of WiFi 802.11ac network

The image below is a similar test compared to the test above, except the test below is a result of WiFi 802.11n network instead of WiFi 802.11ac network. The test was performed on 2.4 GHz signal frequency. This test got lower latency than 802.11ac network and the cause may be from performing the test indoor, and 5GHz network does have worse performance at penetrating walls producing lower range than 2.4GHz signal frequency based WiFi network. This is the effect of having better performing metrics at download link and upload link speeds.



Your Internet Speed

Download 32.8 Mbps

Upload 59.4 Mbps

Latency 6.17 ms

Jitter 8.21 ms

FIGURE 2.4: Performance metrics of WiFi 802.11n network

2.1.3 Bluetooth

Bluetooth operates at the same unlicensed radio frequency as WiFi at 2.4 GHz. The Bluetooth technology is released in 1994 by the Bluetooth sig. Bluetooth is invented to operate at short range and at low cost systems. Bluetooth has usually range of up to

10 meters, according to [37]. As opposed to 20 MHz- 160 MHz channel width usage from the WiFi technology, the channel width Bluetooth uses alternate between 1 MHz and 2 MHz. This technology comes in two packs, BLE and the BBR. BBR is meant for higher throughput with higher power usage and BLE for low power devices with lower performance compared to BBR. BLE supports 1 MHz channel width and BBR supports 2 MHz channel width, thus BLE will have 79 available channels and BBR only 40 available channels. BBR can transmit up to 3 Mbps of data and lower power version 2 Mbps. Both of the Bluetooth version can support from 1 to 100 mW of transmission power as stated in [41].

2.1.4 Satellite navigation systems

SNS' goals are to determine a system's current location by utilizing satellites deployed in the space. SNS' are able to determine the user's three dimensional locations, being altitude, longitude and latitude from that, the location and time. SNS' manage to determine all the mentioned abilities by receiving radio signals from the orbiting satellites and receiving back to the electronic device [4, 44]. There are different variations of SNS' throughout the world.

GPS

GPS is one of the available SNS. GPS is U.S. based SNS and is capable of establishing positioning, navigation and timing services.

According to [18] the accuracy of the depends several varying factors, being satellite geometry, signal blockage, receiver design features and quality and lastly atmospheric conditions. Signal blockage may be caused by buildings and other obstacles that are in interference with the radio signal. Indoor usage of GPS may also cause worse accuracy.

Frequency Each of SNSs' supports varying signal frequencies, but there is one type of navigation signal which, GPS, GLONASS, Galileo, Beidou and QZSS are adapting at the same frequency, the L1C signal. The L1C signal is operating at 1575.42 MHz frequency and is currently operating on the all the mentioned SNSs. The GPS do support three other navigation frequency signal, which are L2C and L5. L2C and L5 operates at 1227.60 MHz and a lower 1176.45 MHz frequencies respectively.

2.2 MANET, VANET and FANET

This project is only focused on FANET, but other Ad-Hoc networks will be also be explained as FANET is deprived from the other Ad-Hoc networks. FANET is based on VANET, and VANET is based on MANET. In this section each of the Ad-Hoc networks will be briefly explained. Their advantages, disadvantages and key features will be pointed out. MANET, VANET, FANET are all Ad-Hoc networks that tend to include from as low as three nodes to as high hundreds of nodes. The point of each of the three Ad-Hoc network are to wirelessly transport data packages from a source node to one ore more destination nodes while going through all the necessary intermediate nodes, and it is not necessary to through all nodes. While data packages are being transported from a source to destination, a routing technique and a routing protocol must be utilized.

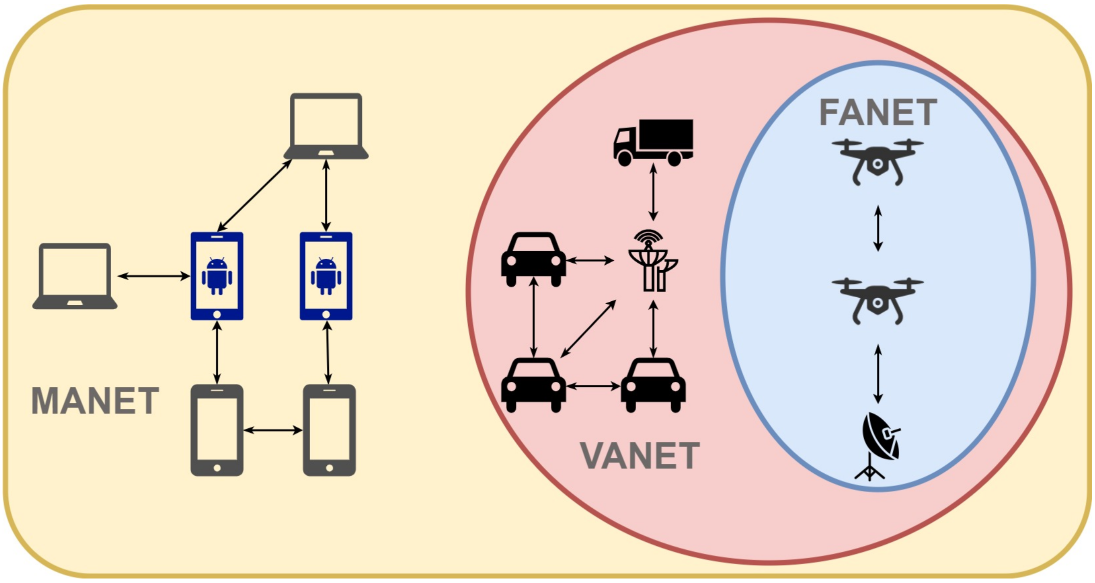


FIGURE 2.5: Illustrational difference between MANET, VANET and FANET

2.2.1 MANET

MANET, as its name suggests, is a Ad-Hoc network used mobile devices. The mobile devices that may benefit from MANET are including, but not limited to, smartphones, laptops, tablet, and other mobile devices that have network connection. As described by [30], MANET is a infrastructure-less network that works by connecting mobile devices with each other wirelessly. MANET is anticipated to be used for a complete specific tasks or missions. As MANET is mainly used to wirelessly connect mobile devices, the network may also be used to connect other devices such as, satellite networks. MANET is specifically used in military and disaster management applications [12]. In MANET all the mobiles devices (nodes) are able to leave or join the network as they want. MANET is a multi-hop network, also called hop-by-by communication. A multi-hop network works by hopping through more than one node while delivering a package from source to destination. The intermediate nodes in MANET helps transporting the packages by jumping from node to node. The opposite of a multi-hop or hop-by-hop network would be called single-hop network, and in this case the network would not have any intermediate nodes from source to destination, thus creating a direct network connection. In a MANET, power consumption plays a big role. Mobile devices have their size limited, thus having limited battery capacity and soc performance. As described by [31] These limitation sets the MANET in a role to prioritize efficiency, and therefore power saving techniques have been implemented to make mobile devices in MANET more efficient. MANET is based on dynamic network topology, meaning the topology in MANET can be changed to meet a specific goal. Dynamic network topology is mainly used to increase the power efficiency in a network, as its one of the power saving techniques in MANET. In a dynamic topology, the nodes in a network is not stationary and may be moved, and no central node being in present, thus creating a heterogeneous network, as explained by [42].

2.2.2 VANET

VANET, is Ad-Hoc network similar to MANET, but instead composing of mobile devices, it is composed of vehicles. Some of the vehicles that are used in VANET are, police cars, ambulances, trucks, autonomous cars and fire trucks. As in MANET, vehicles used in VANET must have wireless communication to be implemented in a VANET. As described in [35], the main differences between VANET and MANET are its performance limitations, battery limitations, and lower storage. Mobile devices, especially smartphones, are hugely limited by its performance and battery capacity, these two limitations main factor is their limited physical size. As in VANET, even the smallest vehicle has enough space to include a SOC that is more powerful than the most powerful SOC in a smartphone. The battery performance of a vehicle will outlast a smartphone in all use cases, the battery performance do not tend to be a problem for a vehicle, the reason being vehicles do recharge their batteries with dynamo or alternator continuously while driving. As MANET, VANET is also based on dynamic network topology, but as opposed to MANET, VANET tend to follow a geographical topology pattern. Because VANET is based on vehicles, the vehicles as it may be stationary they are mostly non-stationary and at high movement speed. Vehicles at high speed will lower connection time between the nodes, thus having possibility to lose a connection mid-way. The vehicular traffic will change throughout the day and will highly depend on the hour of the day, and there will be likely more vehicles in the morning and noon than in evening and at night, that will cause a unforeseeable situation in the node density.

2.2.3 FANET

FANET is the latest Ad-Hoc network that will be explained in this chapter, and also the Ad-Hoc network that will in use in this project. The devices that consists in a FANET can be UAVs, helicopters, aircrafts, etc. With the use of FANET Ad-Hoc network, smaller UAVs is able to be created rather than a bigger UAV. With FANET, the smaller UAVs will be able to communicate with each other and also with the base station without an actual infrastructure, this will achieve a faster data transfer between the UAVs and the base station. Each of the UAVs will also be able to transmit data to each others. The main goal with a FANET, is to create an Ad-Hoc networking between the drones to achieve, better network coverage when compared to a single UAV system. One of the disadvantages of a single UAV system will also cease to exist with the implementation of the FANET Ad-Hoc network, which is, failure tolerance. With a single UAV system, failure is not tolerable, the reason being, with only a single UAV in the whole system, will cease to work if the only one UAV stops working because of bad weather, software error, hardware problems. With the FANET technology, even if one or several UAVs stops working because of any occurred problems, the system will still be able to continue transferring data from the A to B. The FANET network does also come with a disadvantage, and that is, topology change. FANET, as compared to the other Ad-Hoc networks, MANET and VANET, will the topology changes into account. This will create an importance towards the 3D space when the other two Ad-Hoc networks purely focuses on the 2D space. As stated by Khan et al. FANET will be dealing with UAVs, constant changes in 3D coordinates will occur.

2.3 Routing Techniques

Routing techniques are wireless techniques that are being applied on top of the chosen Routing Protocols. Routing techniques adds different behaviour on the applied Ad-Hoc network. Each routing technique will handle the data from source to the destination differently, some of the routing techniques may not handle the data as intended and vice versa. To find the routing technique that is most appropriate to the chosen network is a crucial part in Ad-Hoc networking platform.

2.3.1 Single Path

In this routing technique the source node forwards the data package to the next node by following a single route pattern. The nodes will follow a fixed route pattern no matter the design of the network. The problem with this Single Path technique, as suggested by [33], which is that a critical problem may occur if one or more of the nodes has gotten into a trouble, as wireless communication problem, low battery on a drone or a failure on the node itself. These problems may cause the node to lose communication with the other node that was going to get the data from and thus, fail to forward the data to the next node. Because the Single Path route was taken, the node that was going to forward the data cannot choose another route to forward it, and that will cause a package loss in the network.

2.3.2 Path Discovery

The path discovery routing technique is used when the path is unknown by the network and thus, the source node will not be able to deliver the package forward. To overcome this, a technique called route request package (RREQ) will be transmitted to discover all possible routes the source node can take, to deliver the package to the destination node. After that, a path will be chosen based based on network criteria. The chosen path will be used to deliver the data package from the source node to the destination node by forwarding the package through the intermediate nodes. According to [33], path discovery routing technique is one of the most popular routing technique in FANET.

2.3.3 Greedy Forward

In the greedy forward technique the closest intermediate node will be used to transfer the data package through the network. The source node will try to transmit the data package to the destination node by making least amount of jumps through the intermediate nodes. That will be achieved by jumping to the next node that is the closest node to the destination node. This node comes with a huge disadvantage, and that is the local optimum problem, the this problem is caused when the closest node to the destination node is unable to deliver the package to the next intermediate node.

2.3.4 Multi Path

Multi path routing technique is similar to the single path routing technique, except this routing technique does include with several routing path the network can take to deliver the package. This routing technique tend to be more difficult to maintain because an intermediate node can be used for more than one routes. The advantage of this routing technique is that the network can find problems if a problem appears, then the network can take another route to follow.

2.3.5 Prediction

The prediction based routing technique will take, speed, geographical location and direction into account to predict the next node's location. The predication routing technique will will minimize the package loss of the network and will occasionally reduce delay between the intermediate nodes.

2.3.6 Stone Carry and Forward

The network's will sometimes data transmission range problems, the node that holds the package will not always find the next intermediate node in range of transmission, therefore will not be able to forward the data package. The stone carry and forward technique, as opposed to the other routing technique, will fly physically closer to the next intermediate node, so that the node is in range for data transmission. Therefore, this routing technique is mostly used in networks that are not densely populated, thus the physical movement can hinder the densely populated network.

2.4 Summary

In this chapter Wireless Networks, Ad-Hoc Networks, Routing Protocols and Routing Techniques have been discussed. All the way down to the first generation of wireless network, 1G, and all the way up to 4G, including Bluetooth, have been explained. Their various limitations and advantages are shown in this chapter. The most popular Satellite Navigation Systems are also discussed, including GPS.
MANET, VANET and FANET Ad-Hoc networks' advantages and disadvantages are been clearly shown in this chapter, and why FANET is more suited for this project. Several important and most used Routing Techniques are also been gone through.

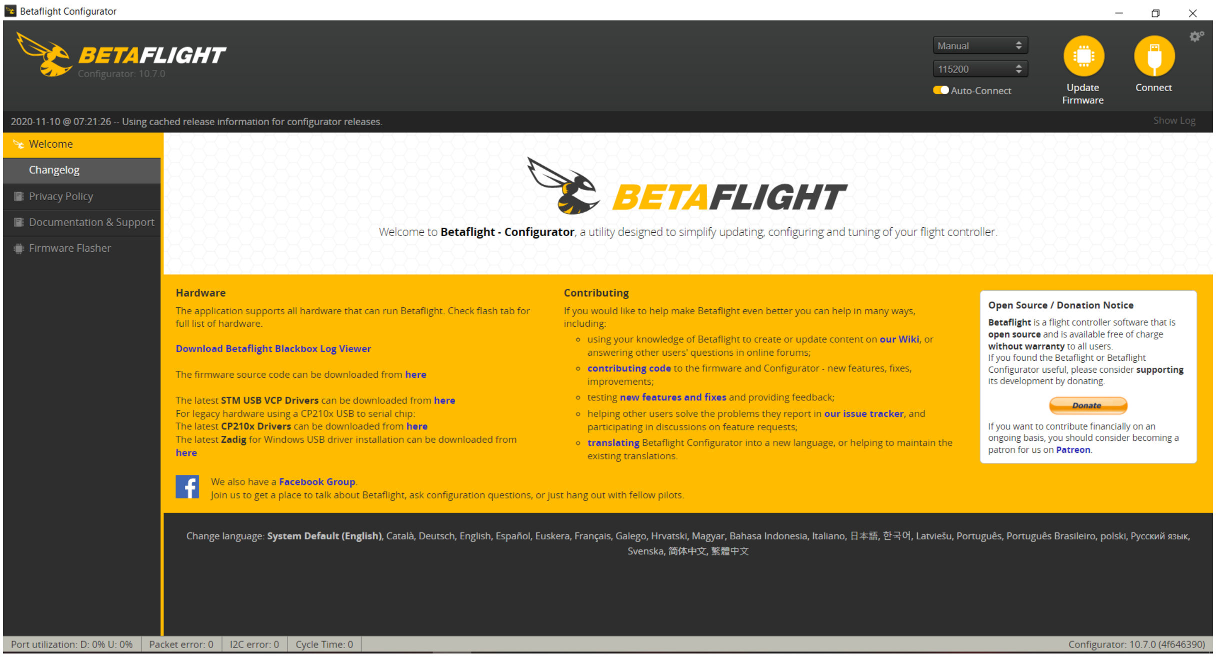
Chapter 3

Implementation

3.1 Software

In this project an open source program has been used. Betaflight is a flight controller software that is used to program UAVs, as quad-copters in this case. With the Betaflight software each of the connected sensors on the flight controller can be initialized and programmed individually. Betaflight version 10.7.0 was used in this project.

The figure below 3.1 is showing the homepage of Betaflight software before connecting the flight controller of the drone to the computer the software is installed on.



2020-11-10 @ 07:21:26 -- Using cached release information for configurator releases.

Welcome to **Betaflight - Configurator**, a utility designed to simplify updating, configuring and tuning of your flight controller.

Hardware

The application supports all hardware that can run Betaflight. Check flash tab for full list of hardware.

Download Betaflight Blackbox Log Viewer

The firmware source code can be downloaded from **here**

The latest **STM USB VCP Drivers** can be downloaded from **here**
For legacy hardware using a CP210x USB to serial chip:
The latest **CP210x Drivers** can be downloaded from **here**
The latest **Zadig** for Windows USB driver installation can be downloaded from **here**

Contributing

If you would like to help make Betaflight even better you can help in many ways, including:

- using your knowledge of Betaflight to create or update content on **our Wiki**, or answering other users' questions in online forums;
- **contributing code** to the firmware and Configurator - new features, fixes, improvements;
- testing **new features and fixes** and providing feedback;
- helping other users solve the problems they report in **our issue tracker**, and participating in discussions on feature requests;
- **translating** Betaflight Configurator into a new language, or helping to maintain the existing translations.

Open Source / Donation Notice

Betaflight is a flight controller software that is **open source** and is available free of charge **without warranty** to all users.
If you found the Betaflight or Betaflight Configurator useful, please consider **supporting** its development by donating.

If you want to contribute financially on an ongoing basis, you should consider becoming a patron for us on **Patreon**.

We also have a **Facebook Group**.
Join us to get a place to talk about Betaflight, ask configuration questions, or just hang out with fellow pilots.

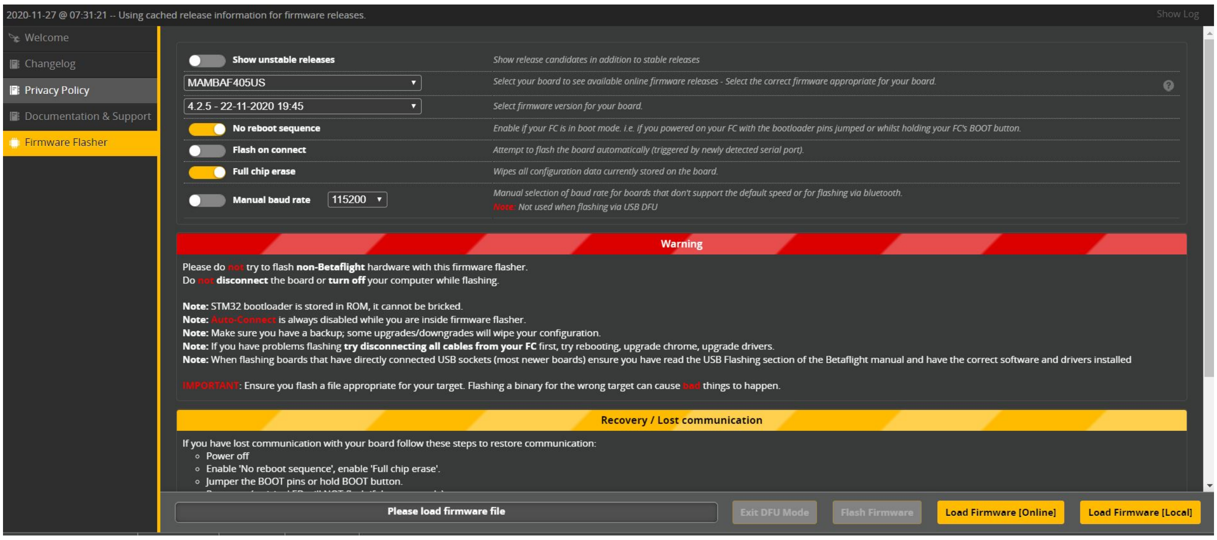
Change language: **System Default (English)**, Català, Deutsch, English, Español, Euskera, Français, Galego, Hrvatski, Magyar, Bahasa Indonesia, Italiano, 日本語, 한국어, Latviešu, Português, Português Brasileiro, polski, Русский язык, Svenska, 简体中文, 繁體中文

Port utilization: D: 0% U: 0% | Packet error: 0 | I2C error: 0 | Cycle Time: 0 | Configurator: 10.7.0 (4f646390)

FIGURE 3.1: Homepage of Betaflight software

3.1.1 Firmware Flash of Flight Controller

First thing first to do after having installed Betaflight, is to upgrade the firmware of the flight controller. When I got the Mambfa F405 MK2 flight controller, it came with an outdated firmware. I downloaded the latest firmware from `https://bit.ly/3l6YUzX` and set the flight controller into the boot mode, that can be done by holding the boot butting right before inserting the USB power cable, then I installed the firmware, by clicking the "Firmware Flasher" as seen in the figure below 3.2. After we are in the "Firmware Flasher", we can click on "Load Firmware [LOCAL]" button to select the



2020-11-27 @ 07:31:21 -- Using cached release information for firmware releases.

○ **Show unstable releases** — *Show release candidates in addition to stable releases*

MAMBAF405US — *Select your board to see available online firmware releases - Select the correct firmware appropriate for your board.*

4.2.5 - 22-11-2020 19:45 — *Select firmware version for your board.*

● **No reboot sequence** — *Enable if your FC is in boot mode. i.e. if you powered on your FC with the bootloader pins jumped or whilst holding your FC's BOOT button.*

○ **Flash on connect** — *Attempt to flash the board automatically (triggered by newly detected serial port).*

● **Full chip erase** — *Wipes all configuration data currently stored on the board.*

○ **Manual baud rate** 115200 — *Manual selection of baud rate for boards that don't support the default speed or for flashing via bluetooth.*
Note: Not used when flashing via USB DFU

Please do not try to flash **non-Betaflight** hardware with this firmware flasher.
Do not **disconnect** the board or **turn off** your computer while flashing.

Note: STM32 bootloader is stored in ROM, it cannot be bricked.
Note: Auto-Connect is always disabled while you are inside firmware flasher.
Note: Make sure you have a backup; some upgrades/downgrades will wipe your configuration.
Note: If you have problems flashing **try disconnecting all cables from your FC** first, try rebooting, upgrade chrome, upgrade drivers.
Note: When flashing boards that have directly connected USB sockets (most newer boards) ensure you have read the USB Flashing section of the Betaflight manual and have the correct software and drivers installed

IMPORTANT: Ensure you flash a file appropriate for your target. Flashing a binary for the wrong target can cause bad things to happen.

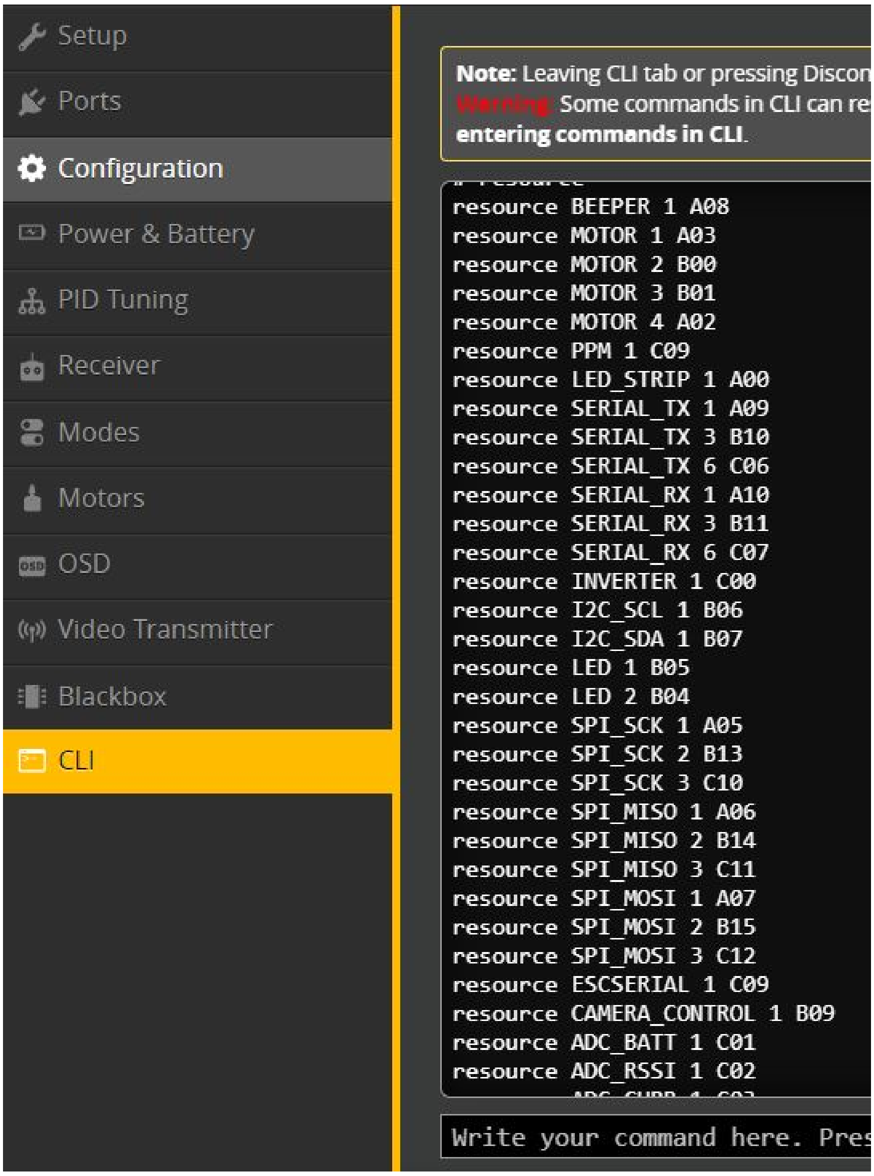
If you have lost communication with your board follow these steps to restore communication:
- Power off
- Enable 'No reboot sequence', enable 'Full chip erase'.
- Jumper the BOOT pins or hold BOOT button.

FIGURE 3.2: Firmware flasher

downloaded firmware. After the firmware is selected it is important to toggle on "No reboot sequence" and "full chip erase". That will make sure that no problems acquire while flashing. Lastly I flashed the firmware by pressing the "Flash Firmware". USB cable can be re-connected to successfully connect to flight controller and Betaflight.

3.1.2 Resource Mapping

The flight controller had adequate UARTs so that I could be able to connect WiFi and GPS modules seamlessly, thus there was only space for one of the module to be connected at the same time, unless resource mapping and soft serial was performed. Resource mapping makes it possible to swap pins throughout the flight controller, thus as an example RX3 of UART3 can be swapped with TX1 of UART1 or even with the LED pin. As seen in the figure 3.28, it can be seen that GPS' RX and TX pins are connected to TX6 and LED pins respectively. Normally TX6 is a part of the UART6 works a a transmitter alongside RX6, which works as a receiver. By doing two resource mapping for TX6 and LED pin on the Betaflight I can create an UART communication with the GPS. As the Betaflight had no other UARTs left to resource map to, I decided to use soft serial. Soft serial stands for Software Serial, Soft-serial is able to create serial communication through software but at the cost of CPU performance, instead of the through usual hardware. So to make the GPS work, the RX and TX from the GPS will directly connect to TX6 and led pin, and then with software add soft-serials to create an extra UART from software [27, 28].



Note: Leaving CLI tab or pressing Discon
Warning: Some commands in CLI can re
entering commands in CLI.

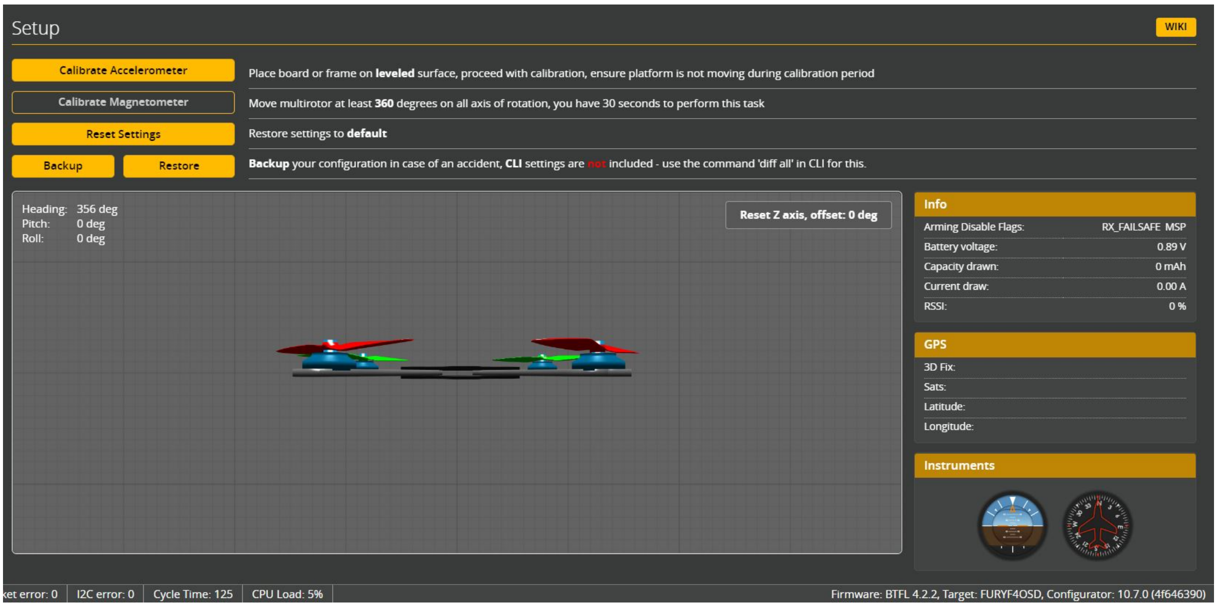
```
resource BEEPER 1 A08
resource MOTOR 1 A03
resource MOTOR 2 B00
resource MOTOR 3 B01
resource MOTOR 4 A02
resource PPM 1 C09
resource LED_STRIP 1 A00
resource SERIAL_TX 1 A09
resource SERIAL_TX 3 B10
resource SERIAL_TX 6 C06
resource SERIAL_RX 1 A10
resource SERIAL_RX 3 B11
resource SERIAL_RX 6 C07
resource INVERTER 1 C00
resource I2C_SCL 1 B06
resource I2C_SDA 1 B07
resource LED 1 B05
resource LED 2 B04
resource SPI_SCK 1 A05
resource SPI_SCK 2 B13
resource SPI_SCK 3 C10
resource SPI_MISO 1 A06
resource SPI_MISO 2 B14
resource SPI_MISO 3 C11
resource SPI_MOSI 1 A07
resource SPI_MOSI 2 B15
resource SPI_MOSI 3 C12
resource ESCSERIAL 1 C09
resource CAMERA_CONTROL 1 B09
resource ADC_BATT 1 C01
resource ADC_RSSI 1 C02
```

Write your command here. Pres

FIGURE 3.3: Resource mapping

By enabling "CLI it is possible to map resources. By first typing "resource" we can see all resources available on the flight controller. Each Resources is mapped to an address, and that address is the physical location on the flight controller. The LED, which is called LED_STRIP in Betaflight has an address of "A00" and the TX6 an address "C06". These two addresses will be important when we are going to map the soft-serials to those addresses.

First it is needed to free the unused resources from their original address, those are namely TX6 and LED on the flight controller. Typing "resource SERIAL_TX 6 none" and "resource LED_STRIP 1 none" will remove the resources mapping and "SERIAL_RX 11 A00" will assign the soft-serial RX1 to the address of the LED pin and "SERIAL_TX 11 C06" will assign the soft-serial TX1 to the address of the TX6 pin, and lastly type "save" to save all the changes made, otherwise it will not be saved. Now GPS will be connected to the flight controller through an UART communication. The only disadvantage of soft-serial is that is work quite good at 9600 baud rate but, at higher baud rate, there might be communication problems.



Setup

Calibrate Accelerometer	Place board or frame on **leveled** surface, proceed with calibration, ensure platform is not moving during calibration period
Calibrate Magnetometer	Move multirotor at least **360** degrees on all axis of rotation, you have 30 seconds to perform this task
Reset Settings	Restore settings to **default**
Backup / Restore	**Backup** your configuration in case of an accident, **CLI** settings are not included - use the command 'diff all' in CLI for this.

Heading: 356 deg
Pitch: 0 deg
Roll: 0 deg

Info

Arming Disable Flags:	RX_FAILSAFE MSP
Battery voltage:	0.89 V
Capacity drawn:	0 mAh
Current draw:	0.00 A
RSSI:	0 %

GPS

3D Fix:
Sats:
Latitude:
Longitude:

ket error: 0 | I2C error: 0 | Cycle Time: 125 | CPU Load: 5%

Firmware: BTFL 4.2.2, Target: FURYF4OSD, Configurator: 10.7.0 (4f646390)

FIGURE 3.4: Setup page of the Betaflight software

After making soft-serial protocol, the soft-serial protocol must be enabled from the "Configuration" tab of the Betaflight, located in the "Other Features" and then "Save and reboot".

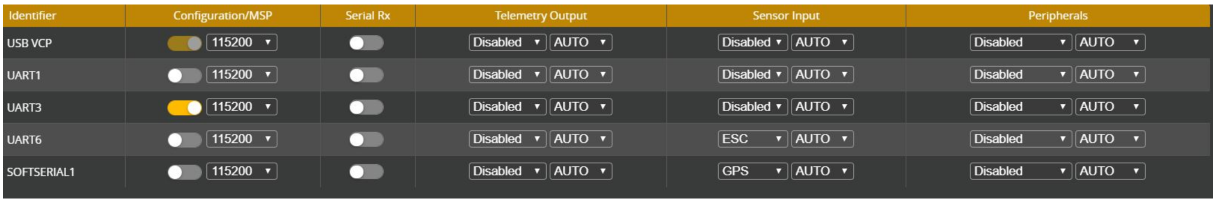
The enabling of soft-serial, has made another Identifier appear on the "Ports tab". "SOFTSERIAL1"'s all toggles will be off except the "Sensor Input", which the GPS will be selected, as shown in the next section's figure 3.5.

3.1.3 Settings and Setups

In this section several settings will be gone through, and which settings were toggled and which turned off will be shown.

Firstly the Setup tab. In this tab it is possible to calibrate the accelerometer and magnetometer. Accelerometer can be calibrated on the go, but to calibrate the magnetometer a GPS needs to be installed. To calibrate the Accelerometer it is very important to lay down the drone on flat surface without having any movements on the drone.

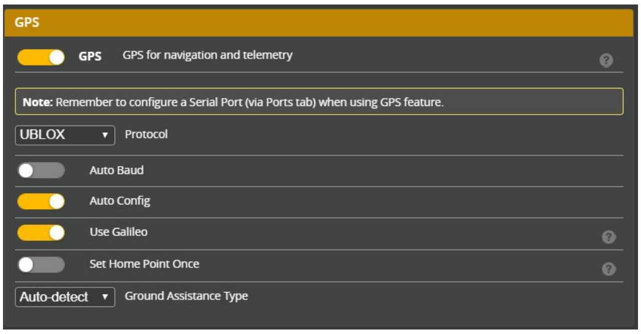
In the Ports tab I have configured different UARTS and a serial-port to the hardware on the drone. UART3 is connected to the ESP32-CAM on one of the drones and ESP8266 on the other one. The "Configuration/MSP" on UART3 most be on to be able to remotely controll the drone via WiFi technology. UART6 is connected to the ESC, on default. Lastly soft-serial 1 is connected to the GPS.



Identifier	Configuration/MSP	Serial Rx	Telemetry Output	Sensor Input	Peripherals
USB VCP	● 115200	○	Disabled AUTO	Disabled AUTO	Disabled AUTO
UART1	○ 115200	○	Disabled AUTO	Disabled AUTO	Disabled AUTO
UART3	● 115200	○	Disabled AUTO	Disabled AUTO	Disabled AUTO
UART6	○ 115200	○	Disabled AUTO	ESC AUTO	Disabled AUTO
SOFTSERIAL1	○ 115200	○	Disabled AUTO	GPS AUTO	Disabled AUTO

FIGURE 3.5: Settings page of the Betaflight Software

After enabling soft-serial for GPS, there will be one more setting left to be tuned to enable GPS. Toggling settings according to the figure 3.6 and rebooting after saving will enable GPS on the system.



GPS

☑ GPS GPS for navigation and telemetry

Note: Remember to configure a Serial Port (via Ports tab) when using GPS feature.

UBLOX ▾ Protocol

☐ Auto Baud

☑ Auto Config

☑ Use Galileo

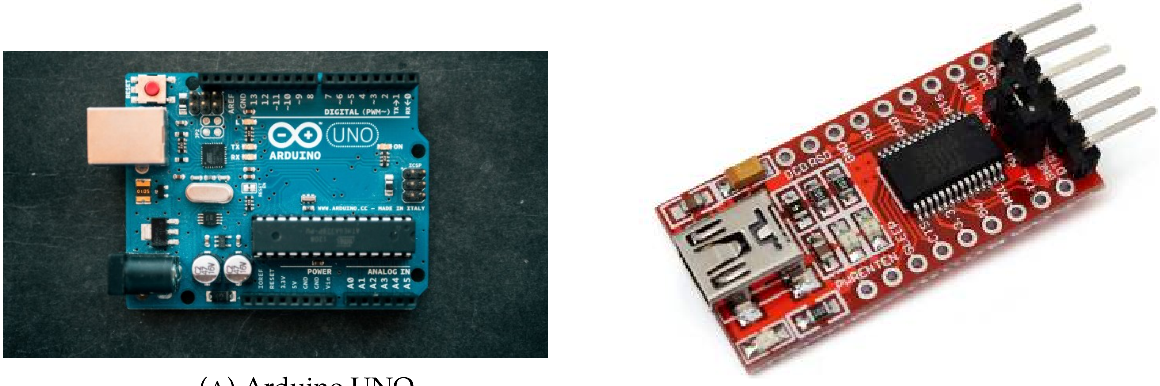
☐ Set Home Point Once

Auto-detect ▾ Ground Assistance Type

FIGURE 3.6: GPS settings page of the Betaflight Software

3.2 Flashing the WiFi Modules

In this section the concept flashing will be discussed and how to perform flashing will be shown. Right of the boxes, the WiFi modules, ESP32 and ESP8266 used in this project will not have access to WiFi, Bluetooth any other technologies it features. To make the module work as intended, the modules need to know what to do and how to do it, therefore it is necessary to upload codes onto the module. That is what is called flashing. By uploading codes, are we able to flash the modules. There are many different methods to flash a WiFi module, through flight controller, via the embedded USB, by using Arduino or to use an USB-to-serial-adapter. The method applied in this project was the two middle explained, embedded USB and through Arduino.



(A) Arduino UNO

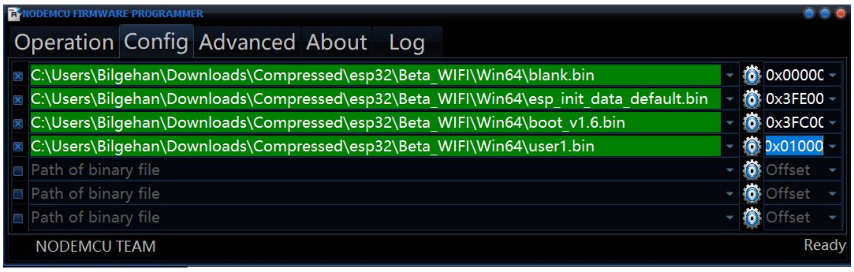
(B) USB-to-serial adapter, ref: https://bit.ly/3nRT7QK

FIGURE 3.7: Two of the four different methods of flashing

To flash ESP8266 WiFi module a software specifically made for flashing ESP8266 was used. To access the files, go to [29]. The ESP8266 has an embedded USB connector, thus is is possible to directly connected ESP8266 to a computer. The computer that is going to flash needs to be Windows based computer, and have USB port. After connecting to a computer, we can start downloading the files and get ready to flash. Downloaded file will include 2 folders and an executable (EXE) file. The first folder is named Win32 and the other is Win64, the compatible folder will be used, based on the computer's Windows version installed, 32- or 64-bit. After opening one of the folders, you will be presented with 4 bin files and an exe file, run the exe file continue the steps below on the opened software.

First thing first, select Com Port, and go to the second tab, Config.

Config tab: Tick the first four boxes located at the most left side of the window. As a "Path of binary file" click the gear symbol and select "boot_v1.6.bin", "blank.bin", "esp_init_data_default.bin" and "user1.bin" for the 1st, 2nd, 3rd and 4th gear symbol from top. For the offset choose, "0x00000"for the 1st, "0x3FE000" for the 2nd, "0x3FC000" for the 3rd and lastly "0x01000" for the 4th.



☒ C:\Users\Bilgehan\Downloads\Compressed\esp32\Beta_WIFI\Win64\blank.bin 0x0000C
☒ C:\Users\Bilgehan\Downloads\Compressed\esp32\Beta_WIFI\Win64\esp_init_data_default.bin 0x3FE00
☒ C:\Users\Bilgehan\Downloads\Compressed\esp32\Beta_WIFI\Win64\boot_v1.6.bin 0x3FC0C
☒ C:\Users\Bilgehan\Downloads\Compressed\esp32\Beta_WIFI\Win64\user1.bin 0x01000
□ Path of binary file Offset
□ Path of binary file Offset
□ Path of binary file Offset

FIGURE 3.8: Config tab of ESP8266 flash program

Advanced tab: Some ESP8266 may differ, and not all may support the same settings, but nevertheless, recommended settings are presented.

- Baudrate: 115200,
- Flash size: 4MByte,
- Flash speed: 40MHz,
- SPI Mode: DIO

After all the configuration, the ESP8266 can successfully be flashed. This method only applied for the ESP8266 module, and will not work for ESP32. After successfully been flashed, the ESP8266 will be automatically configured as a soft-AP (software enabled Access Point). What it soft-AP do, is to create a network SSID (Service Set Identifier) of its own without any password lock. It is important find the SSID to set a password as soon as it is flashed. The default SSID it got by flashing starts with "ESP". After connecting to the network, connect to 192.168.4.1 and it will be forwarded to the configured page, where several settings can be changed, as changing password.

To flash ESP32 another method will be implemented, as explained in the 3.3.5 section, ESP32 as opposed to ESP8266 does not have embedded USB. Therefore one of the other three methods needed to be applied. The ESP32 was connected to the computer through an Arduino Mega board.

As seen from the figure below, few wires were connected to establish a connection between the ESP32 and Arduino.

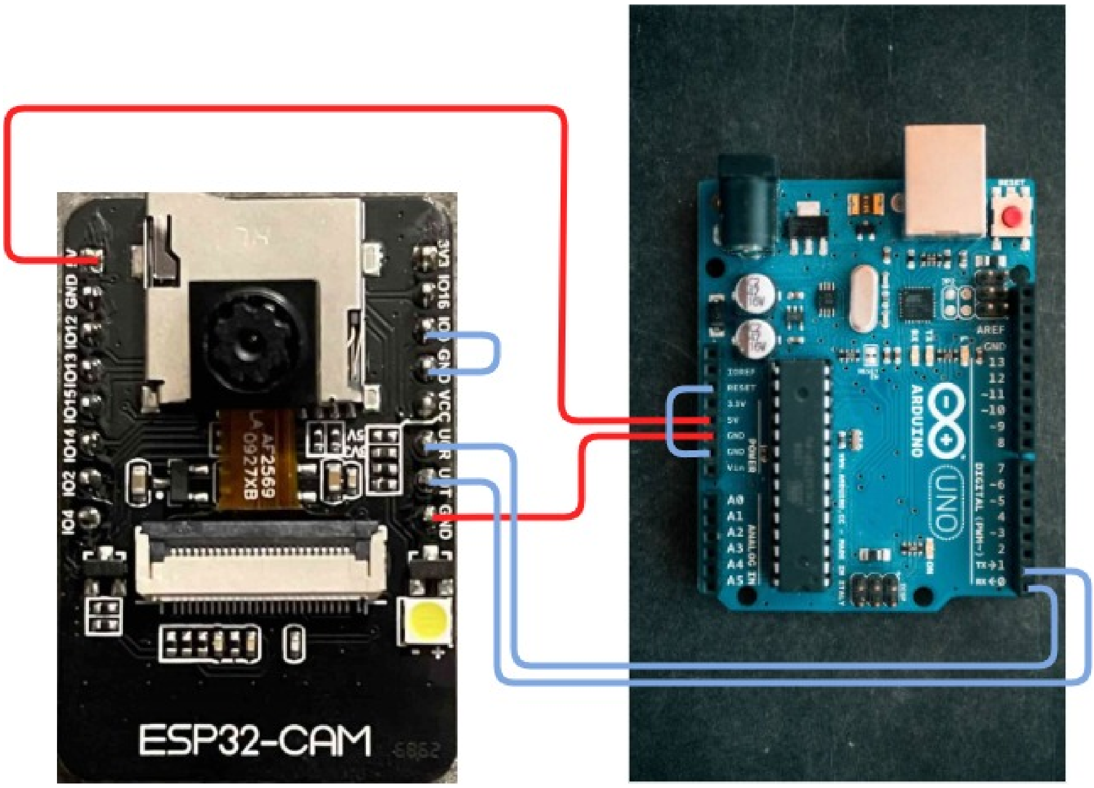

ESP32-CAM
3V3 IO16 IO0 GND VCC U0R U0T GND
5V GND IO12 IO13 IO15 IO14 IO2 IO4
ARDUINO
UNO

FIGURE 3.9: Arduino and ESP32 connections while flashing

To successfully flash ESP32, connecting the wires as shown in the figure is important, or follow the steps below.

- 5 volt from Arduino ↔ 5 volt from ESP32
- Ground from Arduino↔ Ground from ESP32
- Reset from Arduino ↔ Ground from Arduino
- RX from Arduino ↔ U0R from ESP32
- TX from Arduino ↔ U0T from ESP32
- Ground from ESP32 ↔ IO0 from ESP32

The Arduino will not be able to deliver sufficient amount of power to the ESP32 module, therefore ESP32 and Arduino connection will only be used to flash the ESP32, thus after flashing the flight controller will be used to deliver more power to the module.

After the wiring is done, the required software needs to be installed. First of all, Arduino software is needed. The latest version 1.8.13 seems to be having uploading issues, and therefor version 1.8.12 version is used.

To be able to work on ESP32 through Arduino, the library for the boards must be installed. That can be done by following the steps as stated below:

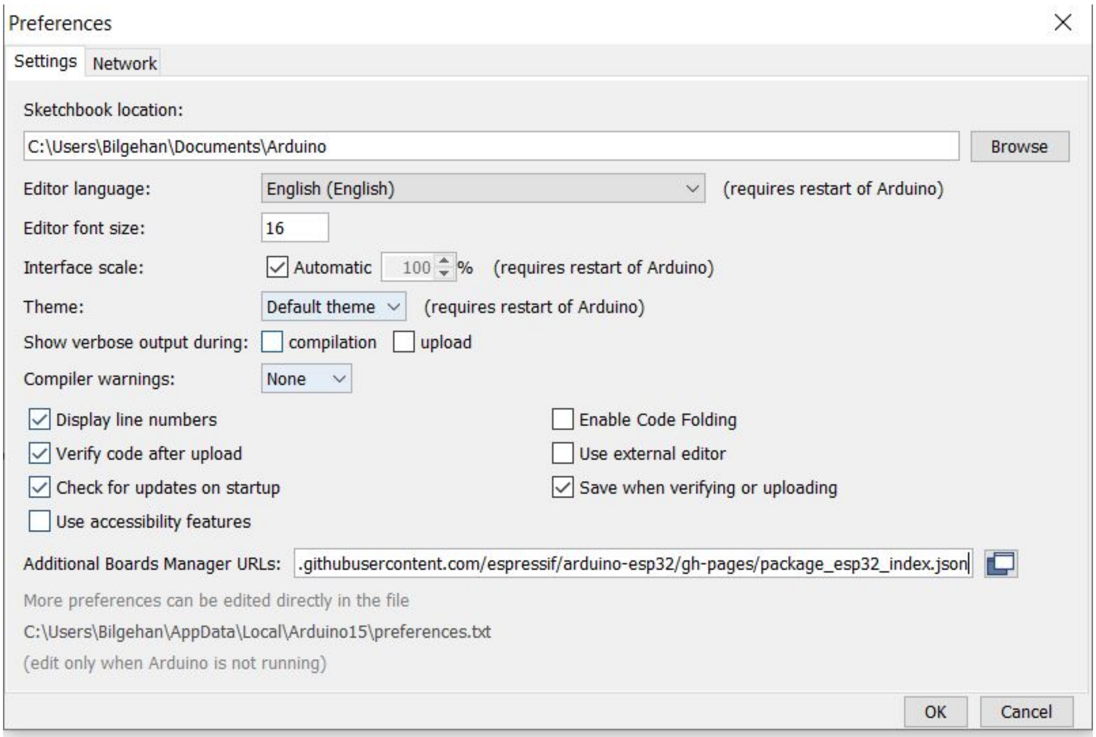
- Choosing File → Preferences.
- Go to "Additional Boards Manager URLs:"

- Paste the following:

```
"https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package\_esp32\_index.json"

```

- Thereafter click "OK".



Preferences

Sketchbook location:
C:\Users\Bilgehan\Documents\Arduino

Editor language: English (English) (requires restart of Arduino)

Editor font size: 16

Interface scale: ☑ Automatic 100 % (requires restart of Arduino)

Theme: Default theme (requires restart of Arduino)

Show verbose output during: □ compilation □ upload

Compiler warnings: None

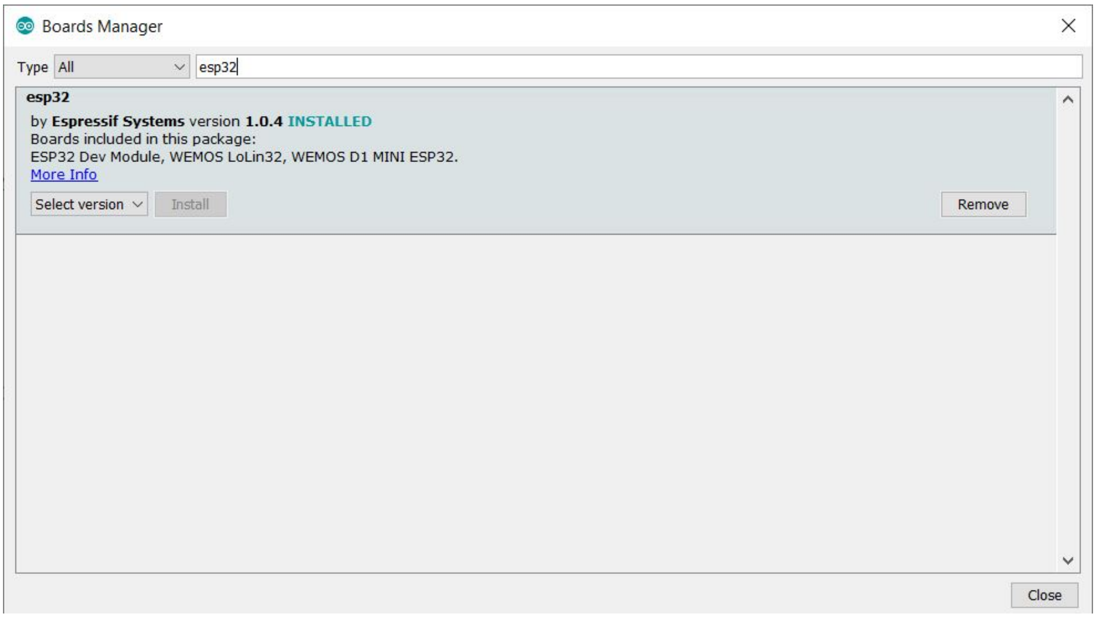
☑ Display line numbers
☑ Verify code after upload
☑ Check for updates on startup
□ Use accessibility features
□ Enable Code Folding
□ Use external editor
☑ Save when verifying or uploading

Additional Boards Manager URLs: .githubusercontent.com/espressif/arduino-esp32/gh-pages/package_esp32_index.json

More preferences can be edited directly in the file
C:\Users\Bilgehan\AppData\Local\Arduino15\preferences.txt
(edit only when Arduino is not running)

FIGURE 3.10: Preferences tab on Arduino

Go to "Tools" → "Board" → "Boards Manager..." and download esp32 library. By doing that the libraries for ESP32 boards will be installed.



Boards Manager

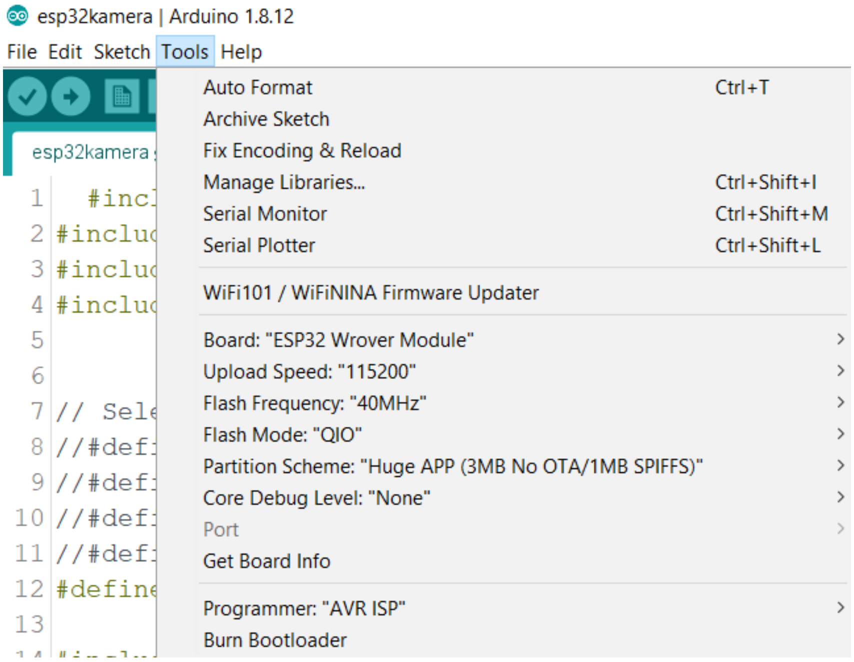
Type All esp32

esp32
by Espressif Systems version 1.0.4 INSTALLED
Boards included in this package:
ESP32 Dev Module, WEMOS LoLin32, WEMOS D1 MINI ESP32.
More Info

FIGURE 3.11: Boards manager tab on Arduino

After having the necessary libraries, it is time to select the board we want to program on. Starting from "Tools" tab there are some settings that need to be changed.

- Board: ESP32 Wrover Moduleæ
- Upload Speed: 115200
- Flash Frequency: 40MHz
- Flash Mode: QIO
- Partitioning Scheme: Huge APP (3MB No OTA/1MB SPIFFS)
- Core Debug Level: None
- Port: Select the port Arduino is connected to
- Programmer: AVR ISP


esp32kamera | Arduino 1.8.12
File Edit Sketch Tools Help
esp32kamera
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter Ctrl+Shift+L
WiFi101 / WiFiNINA Firmware Updater
Port
Get Board Info
Burn Bootloader

Board: "ESP32 Wrover Module"
Upload Speed: "115200"
Flash Frequency: "40MHz"
Flash Mode: "QIO"
Partition Scheme: "Huge APP (3MB No OTA/1MB SPIFFS)"
Core Debug Level: "None"
Programmer: "AVR ISP"

FIGURE 3.12: Selecting board on Arduino

After this point, the code from Appendix A.1 will be used. The ESP32-CAM module needs to be connected as station mode, meaning it will not have its own SSID as an AP connection would have, the ESP8266 previously flashed is an example of AP based connection. The line 16 and line 17 are showing the SSID and the password respectively that the ESP32-CAM is going to connect on. Thus the variable "ssid" needs to be changed to the same SSID created by the ESP8266 module, and the password most also be the same so that the ESP32 is able to connect directly. At this point a mesh network has been established between the ESP32 and ESP8266 WiFi modules and that has created local networking between these two nodes. The mesh networking will extend the total WiFi range.

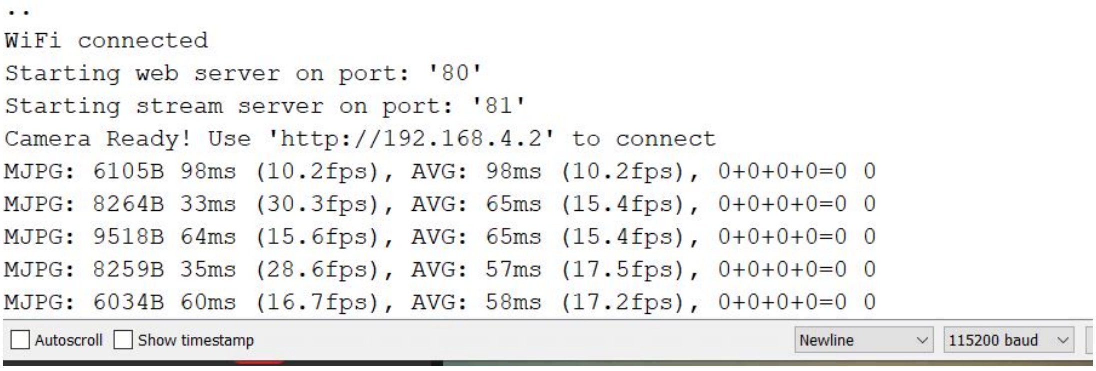
Now the code is ready to be uploaded to ESP32 through Arduino. It may sometimes help to press the reset button right before uploading to increase the success rate of flashing.

3.2.1 Video Streaming via WiFi Ad-Hoc

For this solution, I have decided to go for a routing technique that is suitable for the system. The system has three devices in use, two drones and a device that can communicate with the Drone Beta. Therefore, I decided to go for Single Path Routing Technique. In this project to achieve a FANET Ad-Hoc network design, the ESP32-CAMs and ESP8266 will be connected in mesh network to extend the wireless network coverage. The Drone with ESP32-CAM (also called the Drone Alpha) is the drone with that takes video feed and sends it the next drone (also called the Drone Beta). Therefore when the base station, or in this project's case, the smartphone, wants to the view the video feed, will the smartphone not directly be able to access the Drone Alpha, but rather have to connect to the ESP8266's WiFi. After the smartphone is successfully connected to the ESP8266, the ESP8266 will get the video feed from ESP32-CAM, thus be able to view the data to the smartphone, like so FANET Ad-Hoc network was created.

After successfully uploading, the connection between the Ground and IO0 from ESP32 can be removed and the physical reset button can be pressed to start streaming video from ESP32-CAM.

Local IP-address is necessary to get before streaming. The local IP-address can be gotten by accessing "Serial Monitor" from Arduino. Right after successfully connecting to the assigned SSID, the local IP-address will be print out, the local IP-address is usually "192.168.4.2" and occasionally "192.168.4.3". Before pasting the IP-address to the web-browser, it is needed to establish a WiFi communication between the computer and ESP8266.



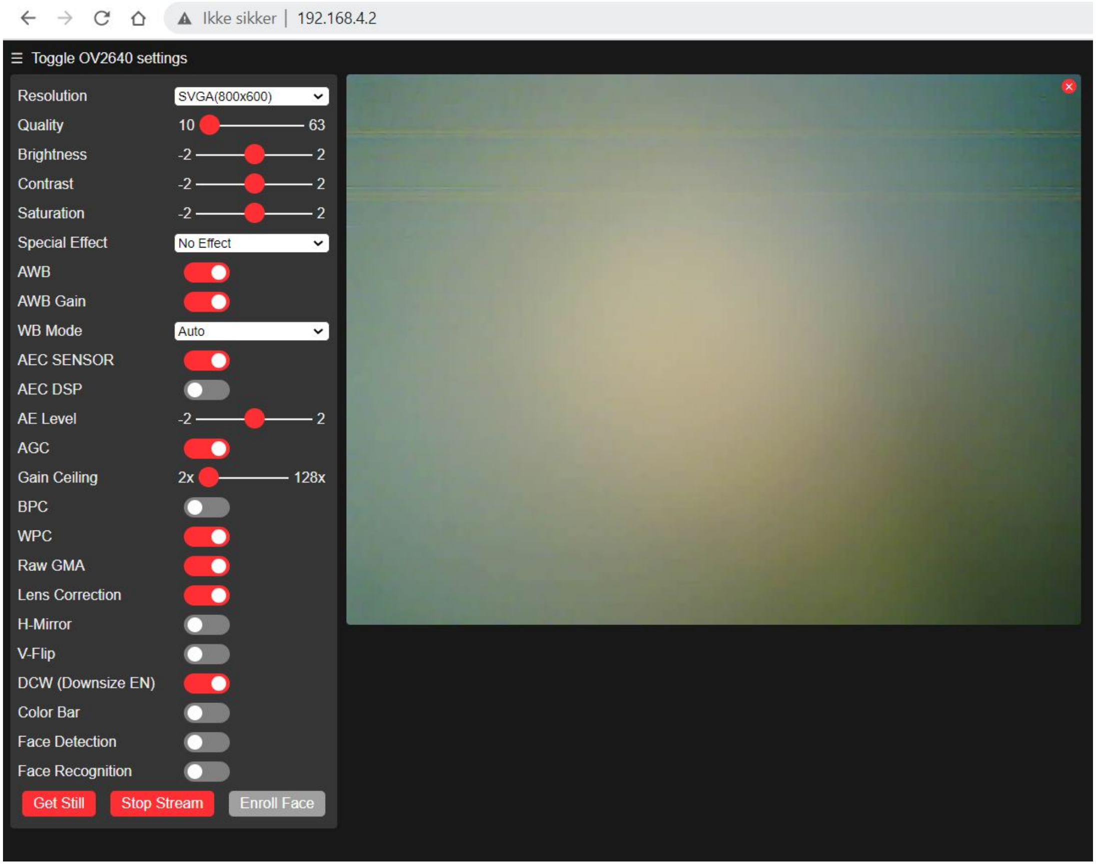
```
..
WiFi connected
Starting web server on port: '80'
Starting stream server on port: '81'
Camera Ready! Use 'http://192.168.4.2' to connect
MJPG: 6105B 98ms (10.2fps), AVG: 98ms (10.2fps), 0+0+0+0=0 0
MJPG: 8264B 33ms (30.3fps), AVG: 65ms (15.4fps), 0+0+0+0=0 0
MJPG: 9518B 64ms (15.6fps), AVG: 65ms (15.4fps), 0+0+0+0=0 0
MJPG: 8259B 35ms (28.6fps), AVG: 57ms (17.5fps), 0+0+0+0=0 0
MJPG: 6034B 60ms (16.7fps), AVG: 58ms (17.2fps), 0+0+0+0=0 0
```

FIGURE 3.13: ESP32-CAM connected to ESP8266

The figure above is showing the WiFi establishment between the ESP32-CAM module and ESP8266 module, and local IP-address is also printed out.

After connecting to the ESP8266's SSID, it is time to paste the "192.168.4.2" IP-address on a web-browser.

Figure 3.14 is showing the first presented page after pasting IP-address. In this page it is possible to tweak the settings of the camera connected to the ESP32-CAM module. I have left the settings as it is, because I found out the default settings to be best working. The only settings I made a change for is the resolution. By adjusting the first setting, it is possible to change the video resolution. With this module I found out the SVGA setting with 800 x 600 pixels resolution to be best balanced between FPS and quality.



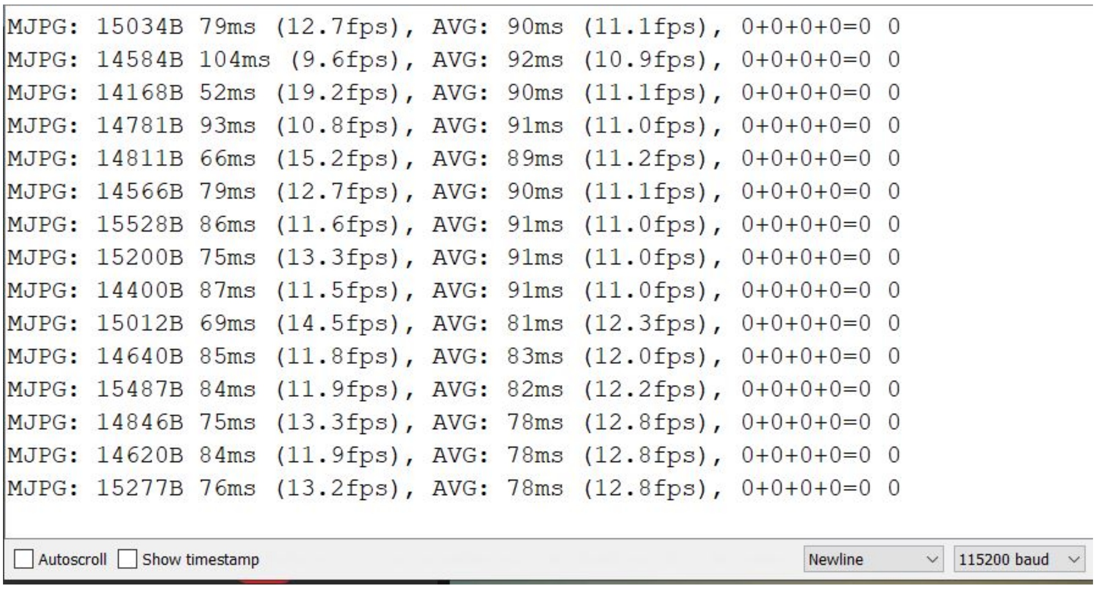
Ikke sikker | 192.168.4.2

☰ Toggle OV2640 settings

Resolution SVGA(800x600)
Quality 10 63
Brightness -2 2
Contrast -2 2
Saturation -2 2
Special Effect No Effect
AWB
AWB Gain
WB Mode Auto
AEC SENSOR
AEC DSP
AE Level -2 2
AGC
Gain Ceiling 2x 128x
BPC
WPC
Raw GMA
Lens Correction
H-Mirror
V-Flip
DCW (Downsize EN)
Color Bar
Face Detection
Face Recognition

FIGURE 3.14: Homepage of ESP32-CAM WiFi Module

By going back to the serial monitor is it possible to monitor the delay and FPS that we are getting from the module. In this particular example, the camera was streaming at SVGA quality and the camera was able to achieve an average FPS close to 13 FPS according to the figure shown below 3.15. A lower quality can be instead configured, to get a lower delay, thus avoid possible failures while flying the drone.

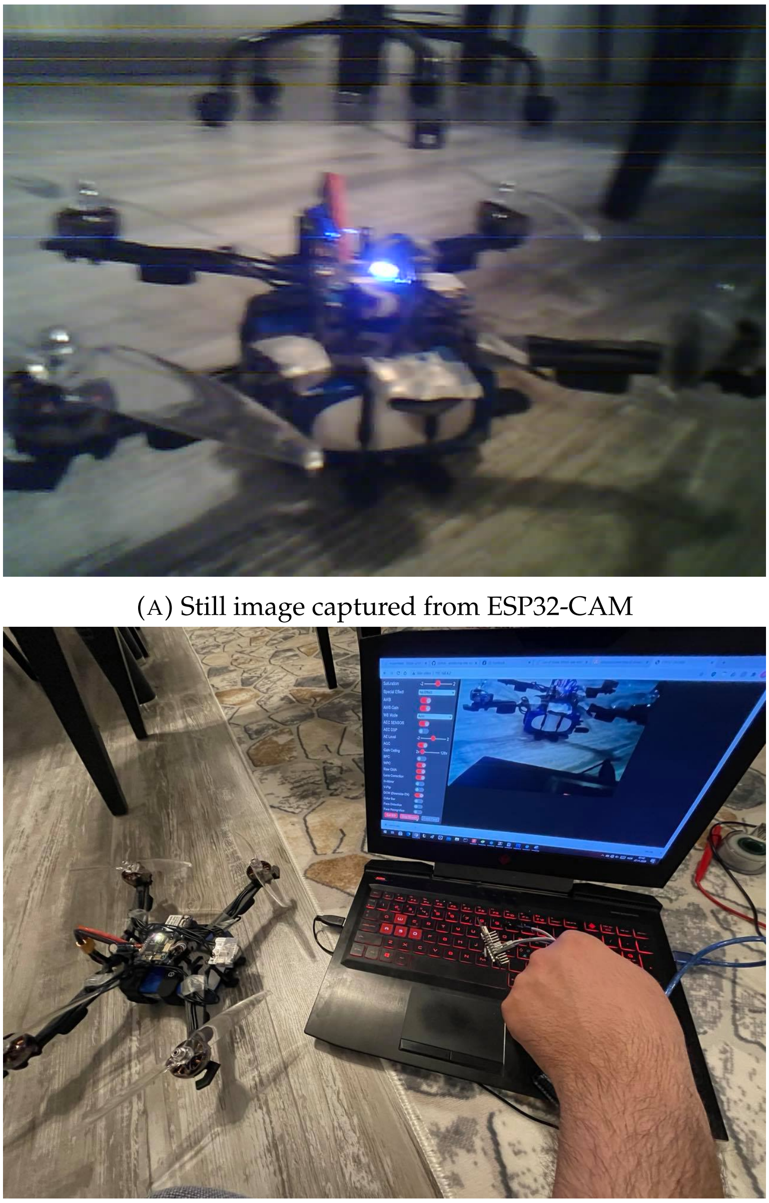


```
MJPG: 15034B 79ms (12.7fps), AVG: 90ms (11.1fps), 0+0+0+0=0 0
MJPG: 14584B 104ms (9.6fps), AVG: 92ms (10.9fps), 0+0+0+0=0 0
MJPG: 14168B 52ms (19.2fps), AVG: 90ms (11.1fps), 0+0+0+0=0 0
MJPG: 14781B 93ms (10.8fps), AVG: 91ms (11.0fps), 0+0+0+0=0 0
MJPG: 14811B 66ms (15.2fps), AVG: 89ms (11.2fps), 0+0+0+0=0 0
MJPG: 14566B 79ms (12.7fps), AVG: 90ms (11.1fps), 0+0+0+0=0 0
MJPG: 15528B 86ms (11.6fps), AVG: 91ms (11.0fps), 0+0+0+0=0 0
MJPG: 15200B 75ms (13.3fps), AVG: 91ms (11.0fps), 0+0+0+0=0 0
MJPG: 14400B 87ms (11.5fps), AVG: 91ms (11.0fps), 0+0+0+0=0 0
MJPG: 15012B 69ms (14.5fps), AVG: 81ms (12.3fps), 0+0+0+0=0 0
MJPG: 14640B 85ms (11.8fps), AVG: 83ms (12.0fps), 0+0+0+0=0 0
MJPG: 15487B 84ms (11.9fps), AVG: 82ms (12.2fps), 0+0+0+0=0 0
MJPG: 14846B 75ms (13.3fps), AVG: 78ms (12.8fps), 0+0+0+0=0 0
MJPG: 14620B 84ms (11.9fps), AVG: 78ms (12.8fps), 0+0+0+0=0 0
MJPG: 15277B 76ms (13.2fps), AVG: 78ms (12.8fps), 0+0+0+0=0 0
```

□ Autoscroll □ Show timestamp Newline 115200 baud

FIGURE 3.15: Average FPS and delay generated by ESP32-CAM video streaming

The figure 3.16a is an example of capturing a still image from ESP32-CAM module. The image is showing the second drone that is equipped with the ESP8266 WiFi module. The figure 3.16b is a still image captured from a smartphone. In the figure it is shown that a USB cable is connected to computer from the ESP32-CAM module, but that is only to power the module and not to send and receive data.



(A) Still image captured from ESP32-CAM

(B) Overall view of ESP32-CAM and the Drone

FIGURE 3.16: Still photos captured from smartphone and ES32-CAM

3.2.2 Wireless Flight

In this project I have also shown the possibility to fly the drone wirelessly. To be able to do a wireless flight, the ESP8266 was used to create a MESH network to extend the WiFi Range. ESP8266 WiFi module will also be used to send wireless data commands to the flight controller.

The application used in this scenario is from the author cs8425 https://git.io/JkQus. The application is downloaded from https://bit.ly/3o0UPz8 and works for Android based Smartphones only. It made possible to make a wireless flight possible to via a computer (through an Android Emulator) or smartphone.

One the "Configurator" tab it is necessary to choose "MSP RX Input" as the receiver to tell Betaflight that the receiver is the one that had MSP enabled on the "Ports" tab, which is UART3, the ESP8266 WiFi Module. After that "Modes" tab needs to be taken a look as seen in the figure 3.17.

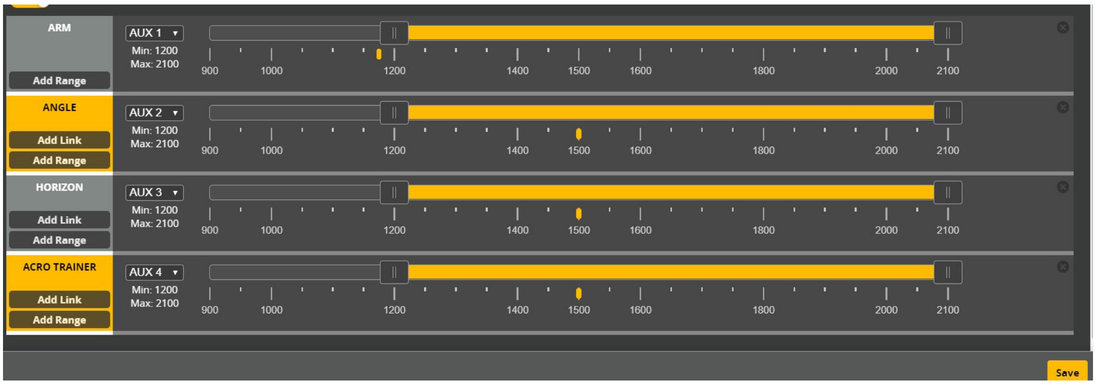


FIGURE 3.17: Modes tab in BetaFlight Configurator

Connecting to the ESP8266's SSID is, then launching the installed android app are few of the first steps. To steer the drone, it is necessary to connect to "tcp://192.168.4.1:2323", by clicking on the top left button.

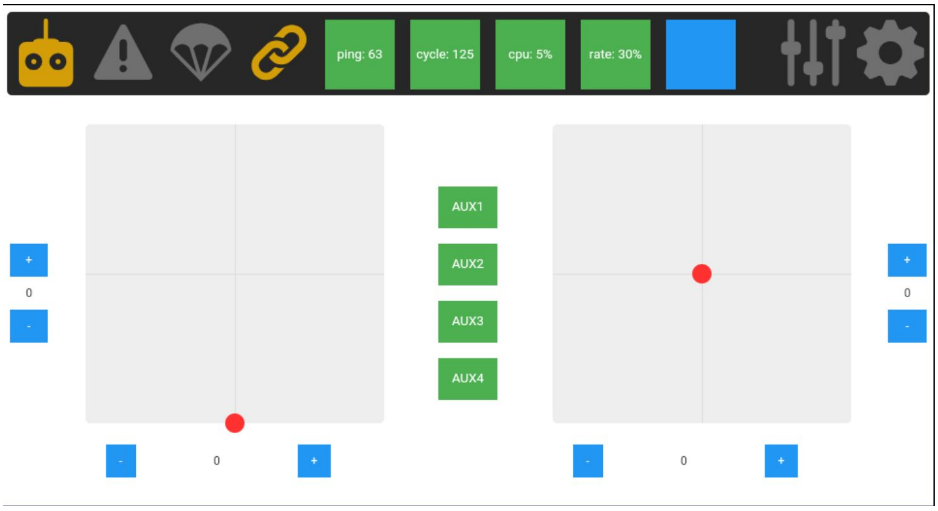


FIGURE 3.18: The interface of the App to control the drones wirelessly

3.3 Hardware

In this section all of the hardware used in this project will be shown. The reason why each specific hardware was chosen for that specific part will also be explained, as there are many hardware included in this project with their own role to fulfill the project's goal.

3.3.1 Flight Controller and ESC

The first hardware I will go trough will be the one of the most important part in a drone, the flight controller. The flight controller works as a brain in a drone and that is the hardware that sends data signals to other hardware connected to the flight controller (itself), the GPS and WiFi components can be examples in this regard. For this project Mamba f405 MK2 is used as the flight controller. The flight controller is compromised of a micro-controller with two embedded sensors, gyroscope and accelerometer. The flight controller is also able to power low power required hardware through its BEC components. BEC stands for Battery Eliminator Circuit and that is usually an electronic circuit on itself, but it is embedded in this flighth controller. BEC is able to alter the voltage that is coming from the battery itself to power other necessary components connected to the flight controller, thus BEC can be regarded as a voltage regulator. BEC in this flight controller is able to regulate the voltage from the battery's voltage, which is up to (4.2v x 6 cell) 25.2 volt to lower 9, 5 and 3.3 volt. Three different voltages with the help of the BEC will be very helpful to power all types of hardware adequate for those voltages. According to Diatone, the flight controller's designer, BEC at 9 volt is able to deliver 2 A at peak and 2.5 A at 5 volt and lastly 500 mA at 3.3 volt. It should be noted 2.5 A at 5 volt can only be achieved of there is no hardware connecting to the 3.3 volt, as 3.3 volt is achieved by regulating the voltage from 5 volt directly. Therefor if there is one or more hardware connected to the 3.3 volt rail using all the available 500 mA, will it then be possible to achieve 2A at peak from the 5 volt rail, the 9 volt rail is unaffected from these scenarios.

Mamba F405 MK2 FC comes with four electronic speed control (ESC) for each motor. ESCs' role are quite similar to the BEC. As the BEC delivers power to other hardware with regulated voltage, the ESCs deliver power to the motor connected to the ESC. Many ESC comes with different power rating, this particular ESC comes with 50 A peak power or 40 A continuous while using a 6 cell battery at 25.2 voltage peak. Meaning the each ESC connected to the flight controller is able to deliver quite a power, at 1008 W continuous power and 1260 W peak power.

$$W = V * A => W = 50 * 25.2 = 1008W \tag{3.1}$$

With all of the ESC combined, it is possible to deliver 5040 W and 4032 W, peak and continuous power respectively. As ESC is able to transfer all that power from the battery to the motors, that will be not possible because of the limitation factors presented by the battery itself, the problems approached with the battery will be explained later on the battery section, 3.3.7.

As seen in the figure 3.19, is an image of both the flight controller Diatone Mamba F405 MK2 and F40 ESC stacked together. The components in the lower part is the ESC and the one above is the flight controller. It is possible to connect the flight controller via USB connection to the PC to update/upgrade firmware and connect to the open source Betaflight software to configure settings of the drone. Section 3.1 will explain the software part in-depth.

Diatone Mamba F405 MK2 flight controller's CPU operates at 168 MHz clock frequency and includes 16 MB memory. Flight controller weights 20.5g according to Diatone.us.

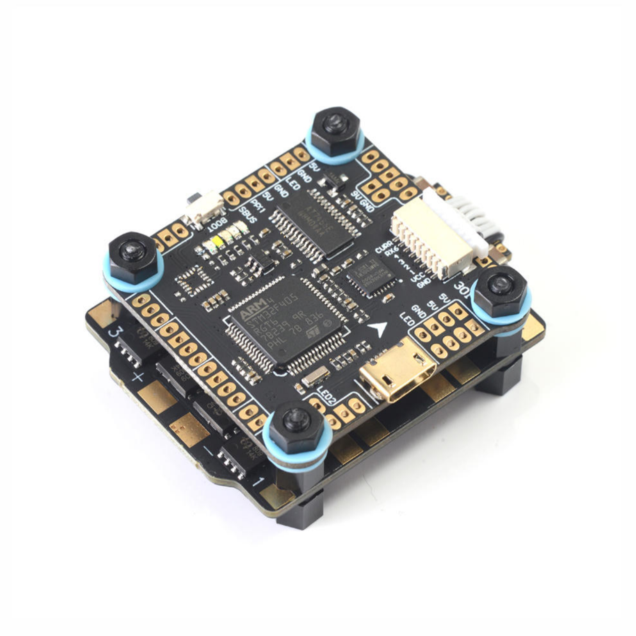


FIGURE 3.19: Diatone Mamba F405 MK2 Flight controller & F40 ESC, ref: https://www.unmannedtechshop.co.uk/product/mamba-f4-f40-power-tower-stack-30x30-f4-fc-40a-esc/

The figure below, 3.20, is the diagram showing different pin layout of the flight controller used.

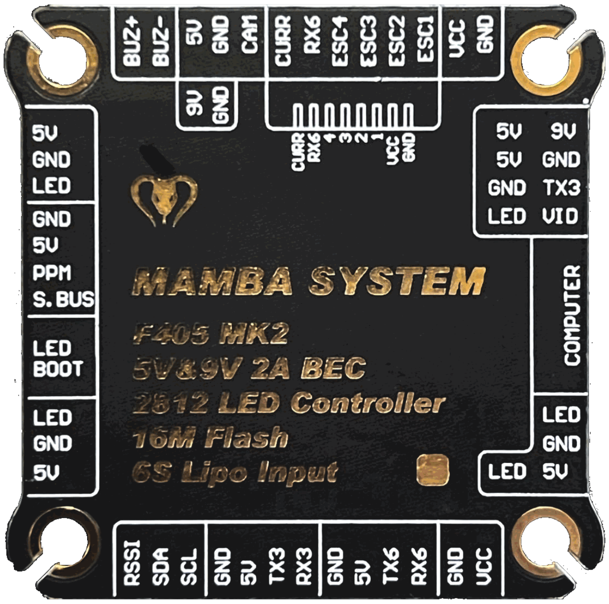


FIGURE 3.20: Diatone Mamba F405 MK2 Flight controller pin layout

As shown in the figure 3.20, the flight controller includes 3 UART (Universal asynchronous receiver-transmitter) ports, named UART1 , UART3 and lastly UART 6. UART is a serial communication protocol used to data signals bit by bit trough cable. UART protocol is a necessary communicating protocol used to connect WiFi, Bluetooth, GPS and other types of transmitters and receivers. Each UART protocol consists of a transmitter and a receiver. UART1 do have RX1 and TX1 as the receiver and transmitter respectively, and UART3 do have RX3 and TX3 for the same reason and so on. The flight controller does also support different types of hardware as, buzzer, camera and led stripes do have their own connection able to be directly connected without going through UART or other types of communication methods. SBUS port as seen in the figure is just renamed RX1 of UART1 but inverted, therefore only specific types of hardware supported by this protocol will work. RX6 is been used by ESC by default and RX1 being inverted puts two receivers of two different types of UARTS unusable on the go, some flight controllers are able to uninvert the SBUS to make it work as a normal RX1, unfortunately this flight controller does not support that feature.

3.3.2 Satellite navigation system

A navigation module that support multiple satellite navigation system has been chosen for this project. The BN-880 is a navigation module, with a duty to incorporate positioning to each of the drones to be able to identify their locations. That will be quite helpful to create a better communication bond between the drones. The satellite positioning can also be used as distance calculations, which can be used to avoid collisions with the help of the WiFi 802.11 technology to send the satellite positioning data between each drone. BN-880 module supports all of the satellite navigation systems described in the section 2.1.4. The reasoning behind picking the BN-880 module is to be non-reliant on only one type of navigation system, but several. Delay and precision will be the two main components that will be looked into By that, choosing the best possible navigation system automatically based on these components will be the aim. That approach will aid in finding the best possible distance between the drones with the lowest delay as possible. Detailed explanations of the different variations of satellite navigation systems are explained in 2.1.4

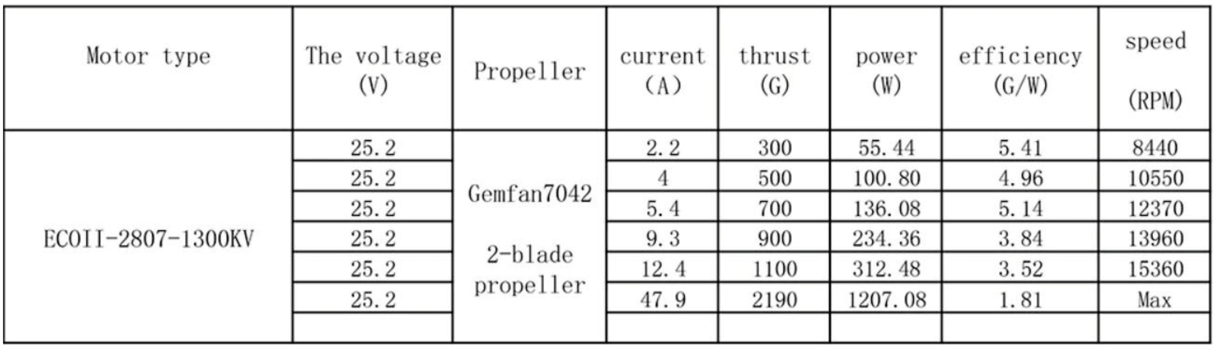
3.3.3 Motor

Motors are essential components while making a drone. Without the motors, the drone will just not fly. But as the motors are an essential part in a drone, so is choosing the correct motors an important factor. In this section chosen motor will be talked through and the reason why the selected motors were chosen will be explained. The chosen motors for this project are called Emax ECO II Series 2807. Emax 2807 motors are brushless DC motors specifically made for flying drones. The motors are made for 6 or 7 inch propellers. The bigger the motors the more torque they have, and by having the higher torque they can be equipped with bigger propellers, thus generating a better efficiency curve. Brushless DC motors consists of several different factors [13].

- Thrust: Thrust is the airflow of the motor and thrust is showing how much air it can push down with the help of the propellers. Thrust can also be regarded as the motor′s performance Thrust is measured in G (grams).
- Power: As in any other electronic components, power tells us how much Watts the motors is consuming.
- Speed: Speed is the showing the rotational speed of the propellers in RPM (rounds per minute).

- Current: How much current flows through the motors, higher current will generate more heat through the motors.
- Voltage: Not all motors run at the same voltage, therefore Voltage is quite an important factor in motor. The voltage the motors supports much equal to the battery's voltage.
- Efficiency: Efficiency can be regarded as one of the most important factors, as higher efficiency will increase the battery life of the drone. Efficiency can be calculated by dividing Power with Thrust.
- KV: KV is telling how fast the motor is able to rotate at given voltage. The rpm of a given motor can be calculated by multiplying KV with applied voltage from battery. Higher KV tends to have higher current usage and lower efficiency.

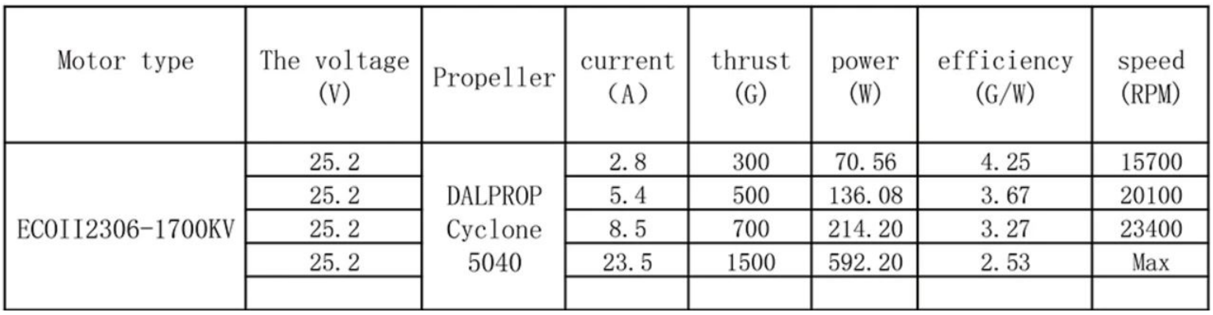
The figure 3.21 is showing the specifications of the motors that are going to be equipped for the drones. The motor is able to deliver up to 2.19 KGs of thrust with no load at 1207 W power, but by doing so the motor will consume 47.9 A of current which will exceed the maximum continuous current capability of the currently equipped ESC, which is at 40 A.



Motor type	The voltage (V)	Propeller	current (A)	thrust (G)	power (W)	efficiency (G/W)	speed (RPM)
ECOII-2807-1300KV	25. 2	Gemfan7042 2-blade propeller	2. 2	300	55. 44	5. 41	8440
	25. 2		4	500	100. 80	4. 96	10550
	25. 2		5. 4	700	136. 08	5. 14	12370
	25. 2		9. 3	900	234. 36	3. 84	13960
	25. 2		12. 4	1100	312. 48	3. 52	15360
	25. 2		47. 9	2190	1207. 08	1. 81	Max

FIGURE 3.21: Emax ECO II Series 2807 specifications, ref: https://bit.ly/3fFpoaI

The figure 3.22 is smaller motor compared to the 2807 version from the same company. The reason to show the smaller variant of the motor is to present different specifications each motor bestow while running at the same voltage. Both of the motors support a 6 cell battery. Right from the image, it is possible to differentiate the two different types of propellers used. As the 2306 motor variant have lower torque, it is equipped with smaller propellers than the 2807 motor. Smaller propellers will generate lower thrust at the same blade amount, thus as to compensate for that, as explained in the 3.3.4, the 2306 went with 3-blade propeller types.



Motor type	The voltage (V)	Propeller	current (A)	thrust (G)	power (W)	efficiency (G/W)	speed (RPM)
ECOII2306-1700KV	25. 2	DALPROP Cyclone 5040	2. 8	300	70. 56	4. 25	15700
	25. 2		5. 4	500	136. 08	3. 67	20100
	25. 2		8. 5	700	214. 20	3. 27	23400
	25. 2		23. 5	1500	592. 20	2. 53	Max

FIGURE 3.22: Emax ECO II Series 2306 specifications, ref: https://bit.ly/36amSWZ

By plotting the efficiency numbers from figures 3.22 and 3.21, it is possible to have a better look at the efficiency at different power usages. As seen from the figure 3.23, at all levels of power usage, the bigger 2807 motor is able to achieve a higher efficiency ranking than the smaller 2306 motor. That was the reason I went for the 2807 motor, efficiency to get more battery life of the drones.

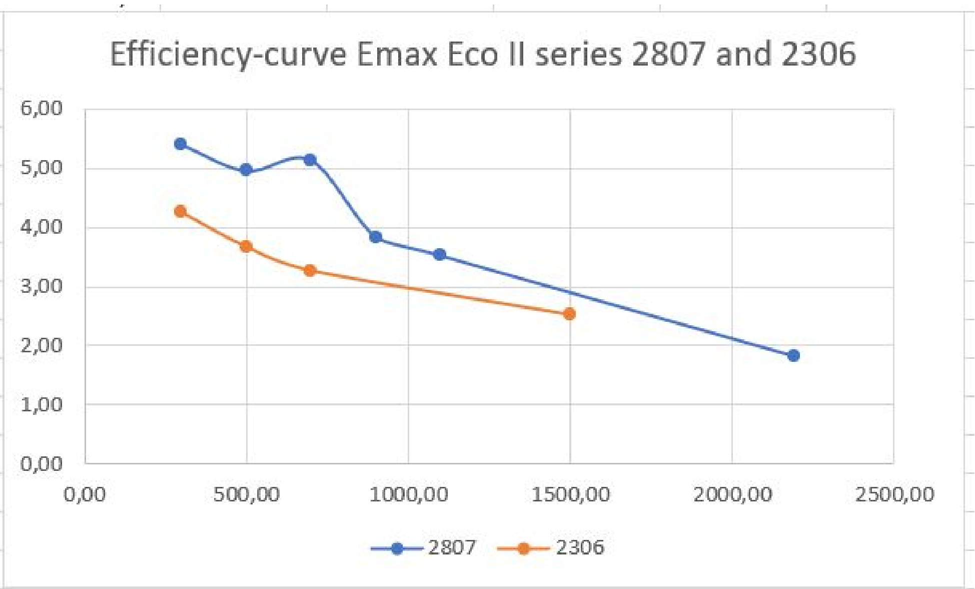


FIGURE 3.23: Efficiency curve of 2807 and 2306 motors

3.3.4 Propeller

For this project it was decided to go for a 7 inch (17.78 cm) propeller with 2 blades, Gemfan 7042 2-blade propeller. propellers can be equipped with various amount of blades. Even then, propellers with 2 and 3 blades are the most common for UAV systems, and there are reasons for that. The higher amount of blades attached to the propellers will increase the air resistance and stability created by the propellers, thus an increase of blades will decrease the efficiency. Therefore 2-blade propellers will be more efficient than 3-blade propellers. The intention for the usage of 3-blade propellers are mainly the thrust the blade can generate. The 3-blade propellers are able to utilize the motors full potential by generating higher thrust (g) at a given rotational speed (rpm). Usually it bigger UAVs that priority battery life are seen with 2-blade propellers and smaller UAVs that priority speed over efficiency with 3-blade propellers. The size of the propellers also play a big role in efficiency. As 2-blade propellers are more efficient than 3-blade are also bigger propellers more efficient. The bigger the propellers are the more efficient and the more thrust (g) it will generate but at the cost of requiring adequate torque from the motors. That is the reason why small UAVs with smaller motors are only seen with small sized propellers. The reason I went for a 7 inch propeller with 2 blades was because the chosen motor explained in 3.3.3 was able to provide adequate torque to go for a 7 inch propeller and as I aimed for efficiency and not top-speed for the UAVs, thus the only important factor being the efficiency unlike the thrust did I go for 2 blade propeller.

3.3.5 ESP32, ESP8266 and camera

As for the WiFi components used for this project, I went for two different WiFi modules, ESP32 and ESP8266. ESP32 is a module that supports two features, Bluetooth 4.2v and WiFi 802.11b/g/n at 2.4GHz radio frequency. ESP32 module will help to establish a WiFi connection. There are a lot of different variants bearing the same module name, ESP32. The variant I went for this project is called ESP32-CAM from the company AI-Thinker, according to `esp.net`. This is also the one of the variant that supports camera module attached directly with a flex cable. The ESP-32S currently supports two camera modules, OV2640 and OV7640. OV2640 is a 2 MegaPixels (MP) camera, that supports of 60 FPS (frames per second) of video recording at 352 x 288 resolution, 30 FPS at 800 x 600 resolution or 15 FPS at 1600 x 1200. When compared to other camera sensors inside smart phones, OV2640 has quite camera sensor. The size of the sensor is 1/4 inch [1]. Compared to Huawei mate 40 pro, which has a main camera sensor size of 1/1.28 inch, can we easily see that OV2640 is 3.125 times smaller than the compared smartphone's [36]. OV7640 is camera module lower performance, with the same sensor size, but lower, maximum 640 x 480 resolution support at 30 fps and 15 FPS at 320 x 240 resolution and at a lower power consumption than OV2640 at 40 mW, whereas OV2640 have a maximum power consumption at 140 mW, that is a whopping 3.5 times more power [2]. For this project, I decided to go for the OV2640 camera, the reason being inadequate video performance of OV7640, even though having higher power consumption.

ESP8266 is the other WiFi module used in this project. ESP8266 is the predecessor of ESP32 WiFi module, and has lower clock speed, 1 core vs. 2 cores, lower ram and less features, but at a lower price [38]. ESP8266 WiFi module does not support camera, and Bluetooth, but only the WiFi Technology. Hence, this module supported only WiFi, this was still adequate choice of a WiFi module, the reason being it is the only wireless network technology I needed for this project.

Even though the WiFi module supports Bluetooth communication protocol, it was not used in this project, the WiFi technology was used both to control the drones and transmit video from the camera module.

ESP8266 with only WiFi capability is connected to the flight controller of one of the drone and the ESP32 connected to the other drone.

Why did I choose one WiFi module capable of Bluetooth, WiFi and camera features and one with only WiFi?

One of the drone needed to transmit video stream to the other drone, thus the drone that needed to transmit had to have a camera included. As for ESP32-CAM, I went for ESP32-CAM, because it had embedded camera controller with an ability transmit camera feed through the use of libraries. The ESP8266 did not need a camera, therefore I went for a simple WiFi which could only do WiFi communication.

3.3.6 Frame

All of the hardware will be mounted on the frame, and therefore choosing a good frame for the drones is nonetheless an important factor. The frame will need to have adequate space for adding all the hardware used in this project and the battery included. The frame must be sturdy and not be easily damaged when the drone is crashed. The frame also need to be lightweight.There are mainly two types of frames, X and H and there are also hybrids, which are a mix of these two types. The X type frames are as the letter is, X-shaped, therefore there will be very little places to mount hardware in the middle of the frame, H type frames will have more headroom of hardware but will be more difficult to fly. The frame that I went for this project is iFlight HL7 V2.1. The frame is an hybrid type between a X and H type frames, called x-hybrid [26]. That has made possible to mount all of the included hardware in this frame. The frame weighs 143 g. Frame can also limit the propeller size that are been used, and in this case, the chosen frame supports up to 7 inch propellers, which is exactly adequate for this project.

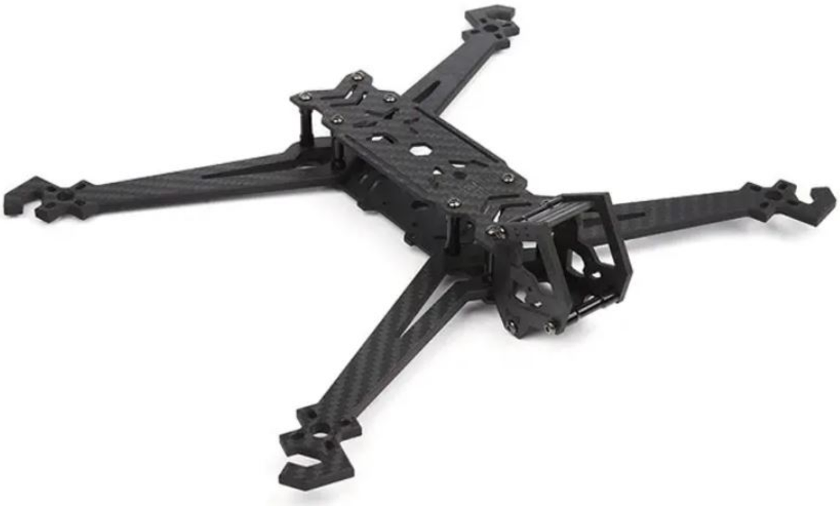


FIGURE 3.24: iFlight HL7 V2.1 frame, ref: https://bit.ly/36bIjH4

3.3.7 Battery

The battery used for the drones are a Li-polymer type battery. The battery has 6 cells connected in serial, achieving a total of six times of a cell voltage.

$$V_t = V_b * N \tag{3.2}$$

The total nominal voltage generated from the battery pack will be $V_t = 3.7V * 6 -> V_t = 22.2V$. The reason for a 6 cell battery pack is to have adequate voltage to power the chosen motors for the drones, as explained in chapter Appendices and section 3.3.3. The battery pack has a total of 5.2 Ampere-hours (Ah), with a total battery capacity of $5.2Ah * 22.2V = 115.44W$ at nominal voltage. The pick for li-polymer battery instead of the popular li-ion battery is that li-polymer batteries have a better discharge characteristics than the li-ion batteries, but in change of costing more and having less energy density.

The discharge capability of the batteries are important factors in drones especially as the batteries must be capable of delivering enough power to the motors to achieve satisfactory flight speeds of the drones. The specification of battery's charge/discharge are shown in C. The C-rating tells us at what current the battery can be discharged or charged at maximum rate. The equation below 3.3 is showing the relation between the charge/discharge rate, C-rating and Ah. For example, a battery with a charge rating of 1C will be able to charge fully in 1 hour, and 2C in half an hour. Li-polymer batteries have two C-ratings, one for its charging capabilities and the other for its discharge. The chosen li-polymer battery has a C-rating for charging at 1C and 12C for the discharging (continuous power), meaning the battery will be able to charge and discharge at 1 hour and 5 minutes respectively at maximum, thus the battery is able to discharge upto 5.2A * 12C = 62.4A, meaning a total power of 1385.28 W at nominal voltage or 1572.48W at peak voltage. As the drone's weight is an important factor for the project does the battery's weight also have a huge impact on flight times. The battery is heaviest of all the components, weighing 682g, thus importance of choosing a correct battery is a crucial part. The reasoning behind the battery I have chosen is to have as much as flight time as possible. It is important to note that having a bigger battery will not always increase the flight time, but will only do it for a degree. The reason for that is coming from the inefficiency generated from the motors at high RPMs, hence a logarithmic function can be plotted with the battery capacity or total drone weight at the x-axis and flight time at the y-axis, [20].

$$W_{C/D} = C * Ah \tag{3.3}$$

3.3.8 Battery life

I have not made battery life test of the drones, and the reason why is to avoid any harm causing the drone if the user fails to operate it successfully. Another reason is that the PID is not well regulated on the flight controller. Nevertheless, it is still possible to make a very close estimate of the drones battery life. By using date from the figure 3.21 and the total weight of the drone is it possible to determine battery life. The test has been done on the drone with ESP8266 WiFi module installed. By weighing on the drone, I found out its weight being 1120 g. Assuming only hovering will be taken into account. At 1120 g, means all of the motors in total need to generate a total thrust of 1120 g to hover the drone. Thus, each motor will need to generate 1120 / 4 = 280 g of thrust. By taking the marginal error into account, we can simply assume each motor needing 300 g of thrust. By looking at the figure 3.21, and thrust (G) panel. Can we say it the battery need to deliver 55.44 W of power to each of the motor. That means a total power consumption to hover the drone will be 221.76 W. By taking the nominal battery capacity into consideration, at 115.44W. Therefore it is possible to conclude that the battery life of the tested drone will be $(115.44W) * 60/221.76W \rightarrow Battery_life = 31min$. By going through this test, the battery life of the drone was found out to be maximum 31 min, that will be lower when taking wind, flying at higher speed, powers used by other hardware components into account.

3.3.9 Charger

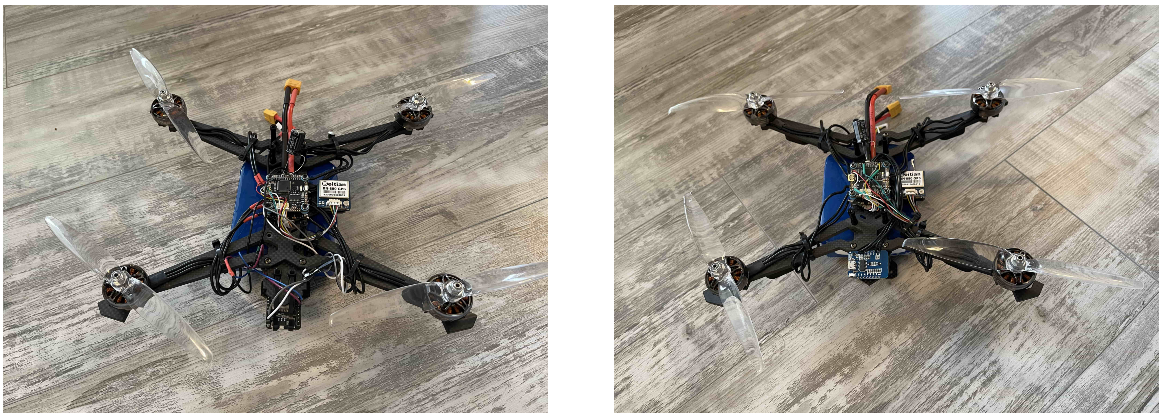
After a while of flying the drones there will be no juice left from the batteries, therefore the batteries will need to be charged back up. Having a charger may be a crucial part in a drone, but performance metrics of a charger does not have that much an impact as other hardware in the drones. Some charger will be able to charger faster than the others, and some chargers support dual charging capabilities. Therefore there is only one important factor, that needs to be fulfilled before a charger can successfully charge a battery, and that is the charger's maximum cell charging amount. There are all types of battery, all the way back from charging up 2 cell batteries and up to 10 cell batteries, or even higher. The battery I had chosen for this project was a 6 cell battery, meaning the charger needed to have a charging capability of charging 6 or higher cell batteries. The other factor I went for when looking for chargers was its maximum battery charging capability, meaning how many batteries the charger could charge at the same time. For this project it was needed to have two batteries, one for each drone, therefore it could be handy to have a charger that could charge two batteries at the same time. As the project involved in using two drones at the same in MESH networking system, and that the two drones would be flight in the air at the same time, would it be plausible to think that their battery life would be quite similar. Similar battery life would mean that both of the batteries could most likely be emptied at the same. Thus we can conclude that dual charging capability of a charger can be, if not a very important, but a little important factor in this regard. Hence the choice of the charger for this project, is iSDT D2 charger. The iSDT D2 charger as shown in the figure below is able to connect to two batteries and do balance charging. It is capable of charging at maximum 100 W for each port. The 100 W power is very close the maximum C-charging rating and thus charger is able to charge at a near maximum charging speed supported by the battery.

FIGURE 3.25: A charger capable of charging two batteries at the same time, ref: https://bit.ly/3fCO2IY.

3.3.10 Design

After mounting all the hardware to the frame and connecting wires correctly, the design of the drones were fulfilled.

Figure 3.26a, shows the complete design of the first drone. The first drone, or as called "Drone Alpha". Drone Alpha has the same hardware as the second drone, or as called "Drone Beta", except one hardware that is different. Drone Alpha uses ESP32-CAM WiFi Module instead of the ESP8266 WiFi Module, the differences between the modules have been explained in the previous section, namely section 3.3.5. Hence the usage of a different WiFi Module, the wiring of these two modules have been also different.



(A) Picture of the Alpha Drone

(B) Picture of the Beta Drone

FIGURE 3.26: The complete design of the drones

3.3.11 Connection layout

The figure below is showing the Circuit connection layout. The drone system consist of GPS and WiFi module, battery, motors, flight controller, ESC and capacitor and lastly frame. Capacitor with a voltage rating of 35 V and 680μF is stabilize the voltage coming form the battery, because voltage spikes may happen and that may fry the hardware components and even the motors. Capacitor will reduce the spikes and also reduce the trough voltages.

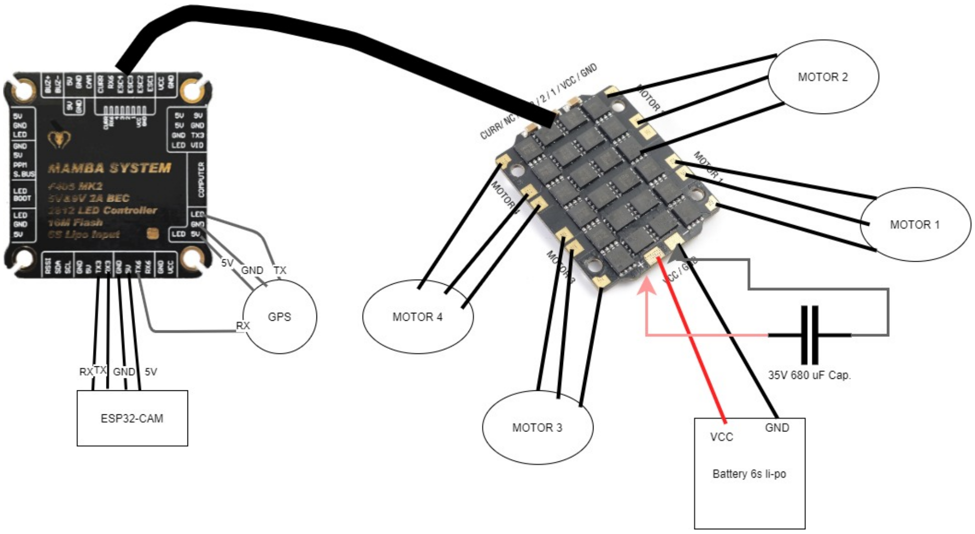


FIGURE 3.27: Connection layout of the Alpha Drone

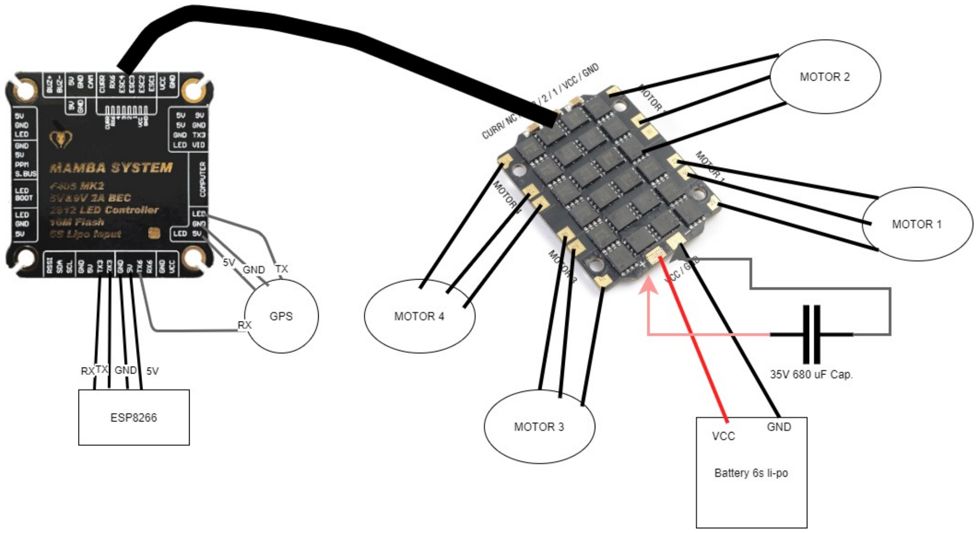


FIGURE 3.28: Connection layout of the Beta Drone

3.4 Summary

In this chapter both the software and hardware methods have been deeply shown. As to configure the drone, the Betaflight software has been chosen. Choosing the correct hardware parts for a specific scenario is a crucial part, and in this chapter different hardware parts have been reviewed and the motive behind the choosing of each hardware part is also explained. Each of the hardware parts need to be gone through software programming or some sort of software techniques to make them work as intended, this chapter will answer these questions.

Chapter 4

Conclusion and Future Work

4.1 Conclusion

As time passes the usage of drones will increase, thus increasing the popularity of the new Ad-Hoc network specifically created for drones, namely FANET. A single Drone will have its limitations and disadvantages compared two two or multiple drones in a system. The main disadvantage with a single drone is that it has to be bigger in size, pricier and louder to achieve the same performance metrics as compared to the state-of-the-art multi type drone systems in an Ad-Hoc network.

In this project it has been created a dual type drone system in an Ad-Hoc network, FANET, to overcome the restrictions and the drawbacks of a single drone. Both of the drones do have WiFi network capabilities thus, the drones are able to communicate with each other to extend the the total communication range. By going for this Ad-Hoc method: A smaller design was achieved, a smaller design will decrease the weight, and decrease total physical size of the drones; A less noisy design was attained, a less noisy design will make the design less noticeable; A more efficient design was achieved, with this solution, the design will be able to have better performance metrics, namely better network coverage, combining the network coverage of two WiFi modules will make it achieve better network coverage per price spent. In this project, two drones were created, with a WiFi capability for each of them and transmitting video feed from Drone Alpha to a smartphone through Drone Beta was achieved.

4.2 Future Work

4.2.1 5G

The 5G technology is the next generation of the current generation mobile network 4G. 5G is currently available in limited cities in Norway. According to Telia Norway, Telia consumers can use 5G in Bergen, which started to be available in 3 November 2020. According to Telenor Norway it is possible to achieve the 5G mobile network in cities as mentioned; Kongsberg, Trondheim, Elverum, Bodø, Fornebu, Kvitfjell, Svalbard/-Longyearbyen, Spikersuppa in Oslo and Askvoll/Flokeneset and more cities coming later. As seen there are still limited amount of cities enabled with 5G connection.

According to Gupta and Jha, 5G will include a technology called massive MIMO, even if bearing the same technological name as MIMO in WiFi network, the massive MIMO is a massive upgradement as the name suggests. As opposed to 4-6 antennas used in the MIMO technology, the massive MIMO is able to use over 100 hundred of antennas at the same time. The massive MIMO will increase the energy efficiency by 100 times and the capacity will be increased by at least 10 times. This upgrade would surely increase the battery life of the drones implemented in this project if 5G mobile network was in use. The delay of transmitting and receiving signals, especially video was a crucial part in this project. Because video streaming takes a huge amount of bps (bit per second). 5G will have up to 10 times less latency compared to 4G network and atop of the a higher bps efficiency of up to 5 times , would definitely help in transmitting video stream to different FANET nodes.

4.2.2 Hardware/Software Upgrades

Several Hardware or Software upgradements can be produced to achieve better performance metrics compared to the current system, and these are:

- Connecting an antenna to the WiFi modules. With an antenna connected, will the WiFi modules achieve a better communication range, thus increasing the total communication range that, according to an article from espressif [14], ESP32 WiFi module is able to achieve a 10KM communication range with the use of an external antenna.

- In this project automatic flying was not implemented and the only way to fly the drones is to use a smartphone with a specific app to control drones.

- In this project, WiFi modules, accelerometer and GPS modules are the only senors used in this project, with the use of other sensors, this project can be extended. An object detection sensor can be added inside the drones to avoid obstacles, as trees while automatically or manually flying the drones.

Appendix A

Appendices

A.1 Codes

Code A.1, is showing the code used to flash the ESP32-CAM module. This code is a slightly modified version of the original code in [15] to minimize disconnecting issues. Code A.1 have several library dependencies, and those codes can be found at [16].

```
#include "esp_camera.h"
#include <WiFi.h>
#include "soc/soc.h"
#include "soc/rtc_cntl_reg.h"


// Select camera model
//#define CAMERA_MODEL_WROVER_KIT
//#define CAMERA_MODEL_ESP_EYE
//#define CAMERA_MODEL_M5STACK_PSRAM
//#define CAMERA_MODEL_M5STACK_WIDE
#define CAMERA_MODEL_AI_THINKER

#include "camera_pins.h"

const char* ssid = "esp8266drone";
const char* password = "PasswordForDrone123";

void startCameraServer();

void setup() {
  Serial.begin(115200);
  Serial.setDebugOutput(true);
  Serial.println();

  camera_config_t config;
  config.ledc_channel = LEDC_CHANNEL_0;
  config.ledc_timer = LEDC_TIMER_0;
  config.pin_d0 = Y2_GPIO_NUM;
  config.pin_d1 = Y3_GPIO_NUM;
  config.pin_d2 = Y4_GPIO_NUM;
  config.pin_d3 = Y5_GPIO_NUM;
  config.pin_d4 = Y6_GPIO_NUM;
  config.pin_d5 = Y7_GPIO_NUM;
  config.pin_d6 = Y8_GPIO_NUM;
  config.pin_d7 = Y9_GPIO_NUM;
  config.pin_xclk = XCLK_GPIO_NUM;
  config.pin_pclk = PCLK_GPIO_NUM;
  config.pin_vsync = VSYNC_GPIO_NUM;
  config.pin_href = HREF_GPIO_NUM;
  config.pin_sscb_sda = SIOD_GPIO_NUM;
  config.pin_sscb_scl = SIOC_GPIO_NUM;
  config.pin_pwdn = PWDN_GPIO_NUM;
  config.pin_reset = RESET_GPIO_NUM;
  config.xclk_freq_hz = 20000000;
  config.pixel_format = PIXFORMAT_JPEG;
  //init with high specs to pre-allocate larger buffers
  if(psramFound()){
    config.frame_size = FRAMESIZE_UXGA;
    config.jpeg_quality = 10;
    config.fb_count = 2;
  } else {
    config.frame_size = FRAMESIZE_SVGA;
    config.jpeg_quality = 12;
    config.fb_count = 1;
  }


  // camera init
  esp_err_t err = esp_camera_init(&config);
  if (err != ESP_OK) {
    Serial.printf("Camera init failed with error 0x%x", err)
  ;
    return;
  }

  sensor_t * s = esp_camera_sensor_get();
  //initial sensors are flipped vertically and colors are a
  bit saturated

  //drop down frame size for higher initial frame rate
  s->set_framesize(s, FRAMESIZE_QVGA);

WiFi.disconnect();

while (WiFi.status() != WL_CONNECTED) {
  WiFi.begin(ssid, password);
  delay(1000);
  Serial.print(".");
}
```

```
 Serial.println("");
 Serial.println("WiFi connected");
  startCameraServer();

  Serial.print("Camera Ready! Use 'http://");
  Serial.print(WiFi.localIP());
  Serial.println("' to connect");
}

void loop() {
  // put your main code here, to run repeatedly:
  delay(10000);
}
```

LISTING A.1: ESP32 flash code, ref: https://git.io/JkQfB

```
Entering CLI Mode, type 'exit' to return, or 'help'

#
# Building AutoComplete Cache ... Done!
#
# resource
resource BEEPER 1 A08
resource MOTOR 1 A03
resource MOTOR 2 B00
resource MOTOR 3 B01
resource MOTOR 4 A02
resource PPM 1 C09
resource SERIAL_TX 1 A09
resource SERIAL_TX 3 B10
resource SERIAL_TX 11 C06
resource SERIAL_RX 1 A10
resource SERIAL_RX 3 B11
resource SERIAL_RX 6 C07
resource SERIAL_RX 11 A00
resource INVERTER 1 C00
resource I2C_SCL 1 B06
resource I2C_SDA 1 B07
resource LED 1 B05
resource LED 2 B04
resource SPI_SCK 1 A05
resource SPI_SCK 2 B13
resource SPI_SCK 3 C10
resource SPI_MISO 1 A06
resource SPI_MISO 2 B14
resource SPI_MISO 3 C11
resource SPI_MOSI 1 A07
resource SPI_MOSI 2 B15
resource SPI_MOSI 3 C12
resource ESCSERIAL 1 C09
resource CAMERA_CONTROL 1 B09
resource ADC_BATT 1 C01
resource ADC_RSSI 1 C02
resource ADC_CURR 1 C03
resource FLASH_CS 1 B03
resource OSD_CS 1 B12
resource GYRO_EXTI 1 C04
resource GYRO_CS 1 A04
resource USB_DETECT 1 C05
```

LISTING A.2: Resource mapping on Betaflight

Bibliography

[1] 2020. URL: https://datasheetspdf.com/datasheet/OV2640.html.

[2] 2020. URL: https://datasheetspdf.com/pdf/499805/OmniVision/OV7640/1.

[3] Dr.Ghassan Abed, Mahamod Ismail, and Kasmiran Jumari. "The Evolution to 4G Cellular Systems: Architecture and Key Features of LTE-Advanced Networks". In: *International Journal of Computer Networks and Wireless Communications* 2 (Jan. 2012), pp. 2250–3501.

[4] Francis Abulude, Akinyinka Akinnusotu, and Adewale Adeyemi. "GLOBAL POSITIONING SYSTEM AND IT'S WIDE APPLICATIONS". In: *Continental J. Information Technology* (Oct. 2015). DOI: 10.5707/cjit.2015.9.1.22.32.

[5] Shahwaiz Afaqui, Eduard Garcia-Villegas, and Elena Lopez-Aguilera. "IEEE 802.11ax: Challenges and Requirements for Future High Efficiency WiFi". In: *IEEE Wireless Communications* PP (Dec. 2016). DOI: 10.1109/MWC.2016.1600089WC.

[6] Adel Al-Alawi. "WiFi Technology: Future Market Challenges and Opportunities". In: *Journal of Computer Science* 2 (Jan. 2006). DOI: 10.3844/jcssp.2006.13.18.

[7] Aruba-Networks. *802.11ac In-Depth*. 2013. URL: https://www.arubanetworks.com/assets/wp/WP_80211acInDepth.pdf.

[8] Aruba-Networks. *Designed for Speed: Network Infrastructure in an 802.11n World*. 2013. URL: https://www.arubanetworks.com/pdf/technology/whitepapers/wp_Designed_Speed_802.11n.pdf.

[9] Maik Basso et al. "A Practical Deployment of a Communication Infrastructure to Support the Employment of Multiple Surveillance Drones Systems". In: *Drones* 2.3 (2018), p. 26. ISSN: 2504-446X. DOI: 10.3390/drones2030026. URL: http://dx.doi.org/10.3390/drones2030026.

[10] KHURSHID LAL BHAWAN. "Study Paper on High Speed Packet Access+ (HSPA+)". In: *Mobile* (2014). URL: http://docshare02.docshare.tips/files/31745/317452043.pdf.

[11] Amira Chriki et al. "FANET: Communication, mobility models and security issues". In: *Computer Networks* 163 (2019), p. 106877. ISSN: 1389-1286. DOI: https://doi.org/10.1016/j.comnet.2019.106877. URL: http://www.sciencedirect.com/science/article/pii/S1389128618309034.

[12] Ali Dorri, Seyed Reza Kamel, and Esmail kheyrkhah. "Security Challenges in Mobile Ad Hoc Networks: A Survey". In: *International Journal of Computer Science & Engineering Survey* 6.1 (2015), 15–29. DOI: 10.5121/ijcses.2015.6102. URL: https://arxiv.org/ftp/arxiv/papers/1503/1503.03233.pdf.

[13] Nodes Drone. *Drone Motor Fundamentals – How Brushless Motor Works*. URL: https://dronenodes.com/drone-motors-brushless-guide/.

[14] *ESP32's Wi-Fi Range Extended to 10 km with a Directional Antenna*. URL: https://www.espressif.com/en/news/esp32%E2%80%99s-wi-fi-range-extended-10-km-directional-antenna.

[15] Espressif. *espressif/arduino-esp32*. URL: https://git.io/JkQfB.

[16] Espressif. *espressif/esp32-camera*. URL: https://git.io/JkQJn.

[17] E. Ezhilarasan and M. Dinakaran. "A Review on Mobile Technologies: 3G, 4G and 5G". In: *2017 Second International Conference on Recent Trends and Challenges in Computational Models (ICRTCCM)*. 2017, pp. 369–373. DOI: 10.1109/ICRTCCM.2017.90.

[18] gps.gov. *The Global Positioning System*. Accessed: 07-nov-2020. 2020. URL: https://www.gps.gov/systems/gps.

[19] A. Gupta and R. K. Jha. "A Survey of 5G Network: Architecture and Emerging Technologies". In: *IEEE Access* 3 (2015), pp. 1206–1232. DOI: 10.1109/ACCESS.2015.2461602.

[20] Hobbyking. *Turnigy High Capacity 5200mAh 6S 12C Lipo Pack w/XT60*. Accessed: 04-nov-2020. 2020. URL: https://hobbyking.com/en_us/turnigy-high-capacity-5200mah-6s-12c-multi-rotor-lipo-pack-w-xt90.html.

[21] Muhammad Khan et al. "Dynamic Routing in Flying Ad-Hoc Networks Using Topology-Based Routing Protocols". In: *Drones* 2 (Aug. 2018), p. 15. DOI: 10.3390/drones2030027. URL: https://www.researchgate.net/publication/327019677_Dynamic_Routing_in_Flying_Ad-Hoc_Networks_Using_Topology-Based_Routing_Protocols.

[22] Muhammad Khan et al. "Flying Ad-Hoc Networks (FANETs): A Review of Communication architectures, and Routing protocols". In: Nov. 2017. DOI: 10.1109/INTELLECT.2017.8277614.

[23] Muhammad Asghar Khan, Ijaz Mansoor Qureshi, and Fahimullah Khanzada. "A Hybrid Communication Scheme for Efficient and Low-Cost Deployment of Future Flying Ad-Hoc Network (FANET)". In: *Drones* 3.1 (2019), p. 16. ISSN: 2504-446X. DOI: 10.3390/drones3010016. URL: http://dx.doi.org/10.3390/drones3010016.

[24] Polynia Kharbuli and Amina Sultana. "A Comparative Study on the Generations of Mobile Wireless Telephony: 1G -5G". In: (Sept. 2018). DOI: 10.1729/Journal.18303.

[25] Ashish Kumar, Ankit Aswal, and Lalit Singh. "4G Wireless Technology: A Brief Review". In: *International Journal of Engineering and management Research* 3 (Apr. 2013), pp. 35–43.

[26] Oscar Liang. *How to choose Mini Quad Frame - The Basics and Evolution*. 2020. URL: https://oscarliang.com/mini-quad-frame-basics/.

[27] Oscar Liang. *Tutorial: Betaflight Resource Remapping*. 2020. URL: https://oscarliang.com/betaflight-resource-remapping/.

[28] Oscar Liang. *Tutorial: How to Setup Betaflight Softserial - Oscar Liang*. URL: https://oscarliang.com/betaflight-soft-serial/.

[29] Mediafire. *Beta WIFI*. URL: http://www.mediafire.com/file/mpngel1k7qg7wja/Beta_WIFI.zip/file.

[30] Ye Miao et al. "Comparison studies of MANET-satellite and MANET-cellular networks integrations". In: *2015 International Conference on Wireless Communications Signal Processing (WCSP)*. 2015, pp. 1–5. DOI: 10.1109/WCSP.2015.7341337.

[31] Anahita Naghshegar, Amir Darehshoorzadeh, and Arash Dana. "Dynamic Topology Control Scheme in MANETs for AODV Routing". In: Dec. 2008. DOI: 10.1109/ATNAC.2008.4783331.

[32] Telia Norway. *3G*. Accessed: 08-nov-2020. 2020. URL: https://www.telia.no/bedrift/oppgradering/.

[33] Omar Oubbati et al. "A survey on position-based routing protocols for Flying Ad hoc Networks (FANETs)". In: *Vehicular Communications* 10 (Nov. 2017). DOI: 10.1016/j.vehcom.2017.10.003.

[34] António L. L. Ramos, Zhili Shao, and José A. Apolinário Jr. "Data Transmission Strategies for UAV: an Overview and Future Perspectives". In: *Society for Design and Process Science* (Nov. 2016). URL: http://sdps2016.conferencespot.org/sdps63262-1.3381173/t007-1.3381354/f007-1.3381355/a023-1.3381358/an023-1.3381480.

[35] Chaudhary Muhammad Asim Rasheed et al. "Vehicular Ad Hoc Network (VANET): A Survey, Challenges, and Applications". In: vol. 548. Mar. 2017, pp. 39–51. ISBN: 978-981-10-3502-9. DOI: 10.1007/978-981-10-3503-6_4.

[36] Lars Rehm. *Huawei Mate 40 Pro Camera review: Dynamic range monster*. 2020. URL: https://www.dxomark.com/huawei-mate-40-pro-camera-review-dynamic-range-monster/.

[37] K. V. S. S. S. S. Sairam, N. Gunasekaran, and S. R. Redd. "Bluetooth in wireless communication". In: *IEEE Communications Magazine* 40.6 (2002), pp. 90–96. DOI: 10.1109/MCOM.2002.1007414.

[38] Sara Santos. *ESP32 vs ESP8266 - Pros and Cons*. 2020. URL: https://makeradvisor.com/esp32-vs-esp8266/.

[39] Mohd. Sayeed and Rajesh Kumar. "An Efficient Mobility Model for Improving Transmissions in Multi-UAVs Enabled WSNs". In: *Drones* 2.3 (2018), p. 31. ISSN: 2504-446X. DOI: 10.3390/drones2030031. URL: http://dx.doi.org/10.3390/drones2030031.

[40] Audrey Selian. "From GSM To IMT-2000 - A comparative analysis". In: *3G Mobile Licensing Policy: GSM Case Study* (2001).

[41] Bluetooth SIG. *Bluetooth Radio Versions*. Accessed: 10-nov-2020. 2020. URL: https://www.bluetooth.com/learn-about-bluetooth/bluetooth-technology/radio-versions/.

[42] Hwee-Xian Tan and Winston Seah. "Dynamic topology control to reduce interference in MANETs". In: Apr. 2005.

[43] Q. K. Ud Din Arshad, A. U. Kashif, and I. M. Quershi. “A Review on the Evolution of Cellular Technologies”. In: *2019 16th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*. 2019, pp. 989–993. DOI: 10.1109/IBCAST.2019.8667173.

[44] A. S. Zaidi and M. R. Suddle. “Global Navigation Satellite Systems: A Survey”. In: *2006 International Conference on Advances in Space Technologies*. 2006, pp. 84–87. DOI: 10.1109/ICAST.2006.313803.