UiT The Arctic University of Norway

Faculty of Science and Technology
Department of Computer Science

An Intent-Based Reasoning System for Automatic Generation of Drone Missions for Public Protection and Disaster Relief

Marcel André Grønvold

INF-3971 Master's Thesis in Health Technology - June 2023

“A wizard is never late, nor is he early. He arrives precisely when he means to.”
–Gandalf the Grey

“Life is a sum of all your choices. So, what are you doing today?”
–Albert Camus

Abstract

The utilization of drones for search and rescue operations has become more prevalent over the years. Drones can provide an aerial perspective which can aid first responders in gaining an overview of a situation. Autonomous drones can automate search and rescue operations by removing the human pilot, which can increase efficiency and lower costs. The increased development of machine learning models and techniques has paved the way for intent-based reasoning systems that can understand users' intent. This can allow users to control autonomous drones by expressing their intent. Which can be utilized for search and rescue operations.

However, machine learning models require vast computational power and data storage. In addition, autonomous drones have high-performance requirements. The development of 5G can provide the infrastructure required to meet the stringent performance requirements of machine learning models and autonomous drones. By leveraging the advanced features of 5G, such as network slicing, high-speed communication, and low latency, it provides the infrastructure that supports the use of machine learning models in coordination with drones.

This thesis proposes a system prototype that can generate drone missions based on user intent which can be used for rescue operations. The system utilizes a large language model and automatic speech recognition model to capture the intent of the user and generate drone missions that integrate with a 4G-enabled drone. The evaluation of the system reveals that the system can reliably capture the user's intent with simple commands, but struggles with more complex commands. The prototype demonstrates that intent-based reasoning systems for controlling autonomous drones using 5G technology can aid first responders during PPDR missions.

Acknowledgements

I would like to thank my supervisor Anders Andersen for providing me with guidance and help. I would also like to thank my co-supervisor Arne Munch-Ellingsen and for his insight into cellular technology and for inspiring me to write this thesis.

I want to thank my family for supporting me over the course of this study.

A special thanks to Serine for your love and support.

Contents

List of Figures

List of Tables

1

Introduction

The global increase in temperature caused by global warming has been proven to be correlated with an increase in natural disasters [54]. This poses a serious risk to public safety and the health of society. Drones have been used in search and rescue operations and for providing real-time information during disasters to aid health care services. However, drones are usually remotely piloted by a human due to the stringent performance requirements of autonomous drones such as high latency, capacity, and computational power. Autonomous drones provide major benefits over human-piloted drones such as being able to function automatically and continuously without interruption, they can be programmed to perform specific tasks, and they can coordinate with other drones. However, autonomous drones need to be instructed to carry out specific tasks. The inclusion of an intent-based reasoning (IBR) system can provide first responders with an interface in which to generate missions for drones. IBR systems analyze the user's inputs to understand the user's input to perform certain actions. They can include large language models (LLM) and automatic speech recognition (ASR) system. The challenge with these models is that they require large amounts of computing power and data storage. The advancement of mobile cellular network technology in 5G has increased the performance of cellular networks meeting the performance requirements related to autonomous drones. This allows drones to be used for public protection and disaster relief (PPDR). In addition, 5G meets the performance requirements held by machine learning models.

1.1 Methodology

The problem that this thesis attempts to solve is to automate mission creation for drones. From this, a research questsion was formulated; *Can an IBR system for Autonomous drones using 5G technology be used for public protection and disaster relief?* The background research was carried out regarding subjects relevant to the hypothesis. This included research into 4G and 5G cellular technology, container technology, drone technology, and machine learning. The system's goal is to generate drone mission plans automatically based on the user's intent. A system prototype was designed and implemented using a DevOps approach. Experiments were then conducted to evaluate the performance of the system.

1.2 Research problem

Automatic mission creation for drones can aid first responders during PPDR missions. A system utilizing intent-based reasoning can automatically generate missions for drones based on the users intent. Advancement in cellular technologies such as 5G meets the performance requirements for IBR systems and drones. The research problem is defined as:

> Resarch Problem: IBR systems for controlling autonomous drones using 5G technology can aid first responders during PPDR missions.

The research problem is divided into smaller sub-problems which will be used in order to answer the main research question. Sub-problem 1 discusses whether or not it is possible to implement a prototype of the system using the available technology. This includes whether the users intent can be understood by the system in order to automatically generate drone missions.

> Sub-problem 1: Is it technically feasible to create a prototype of the IBR system?

A critical aspect of a prototype is the system evaluation. If a system prototype can be implemented, how does it perform in terms of correctness and latency?

Sub-problem 2: What is the system performance, and can the system reliably generate missions?

After answering whether it is possible to implement a prototype, and what its performance is, it is important to discuss whether the system is a solution to the problem presented. Is it a viable solution for automatic mission creation for first responders during PPDR missions?

Sub-problem 3: Is an IBR system a viable solution for first responders during PPDR missions?

1.3 Scope and Limitations

Although this project discusses the possibilities of using 5G cellular technology as a solution to the performance requirements of drones. Neither a cellular 5G network nor a 5G compatible drone is commercially available. In order to simulate a 5G edge, one of the components of the system was hosted in a docker container. This is an edge simulation because the computation is virtualized in a container like it is imagined in a 5G edge node. A 4G connection to the drone was attempted, but was only managed through WiFi due to firmware problems.

Chat GPT 4 and an API to GPT 4 was released by OpenAI during the writing of this thesis. Since it occurred before planning, and it was unavailable, it is considered outside of the scope of this project. This thesis will, therefore only discuss its predecessor GPT 3.5.

The system was implemented as a prototype in order to test if it is possible to create an IBR system for controlling drones and is not meant to be used in production. The system was therefore not implemented with regard to security and privacy. In order to use the system, the user is required to have a drone certificate and use extreme caution when evaluating the correctness of the missions.

The focus of the system is automatic mission generation. Other helpful features for drone missions, such as a piloting interface, real-time drone map, and manual mission creation, are effectively addressed by other applications such as

FreeFlight [53] and QGroundControl [45] and therefore outside of the scope of this thesis.

1.4 Significance and Contribution

The contributions of this thesis are:

- **Prototype**: A system prototype that can interpret the user's intent to automatically generate drone missions. This includes a client for capturing the user's intent through voice, a web API for managing audio transcription and drone mission generation, a server that handles a drone command based on client input, and an integration to an audio transcription model and a chat completion model.

- **Evaluation**: An evaluation of the system that measures the performance in terms of correctness and latency of the system and its components.

1.5 Organization

The paper is organized as follows:

Chapter 2: Theoretical Framework describes the 4G and 5G cellular networks and their differences. An introduction to cloud native 5G core is given before an overview of drone technology is given. A brief overview of machine learning, transformers, a description of how the speech recognition system whisper works, and an overview of OpenAI Chat API is given.

Chapter 3: Design & Implementation describes the design and implementation of the system prototype

Chapter 5: Experiments presents the experiment setup, describes how the experiments were conducted, and presents and evaluates the results.

Chapter 6: Discussion discusses the technical feasibility of the system, the system evaluation, and the viability of an IBR system for PPDR operations

Chapter 7: Conclusion presents the thesis conclusion

2

Theoretical Framework

This chapter will present the theoretical framework used in this paper. A basic presentation of 4G, its core functionality, and its performance will be presented. Then a comparison between 4G and 5G will be presented as well as a basic explanation of 5G and its functionalities including cloud native 5G core. Then a brief introduction to drones and their regulations will be presented, and then the drone communication protocol MAVLink. An overview of machine learning is given, in addition to an introduction to Open AIs Whisper and Chat models.

2.1 4G

4G is the fourth generation of broadband cellular network technology and is an improvement upon the 3G and 2G networks. LTE and 4G LTE (Long Term Evolution) is often used interchangeably with 4G. The 4G standards were developed by 3GPP [1], an organization comprising multiple telecommunication companies that manage standards in the cellular industry. The most significant change from previous generations' networks (1G-3G) was moving from circuit switching to packet switching in the network. In a circuit switch, there is a direct connection between the senders and receiver, often a physical connection as seen in older cellphones. Packet switching refers to a connection-less

network, where messages from the sender are grouped together with other messages in a payload and routed dynamically in a network to the sender. The advantage of switching from circuit switching to packet switching is that there is no delay when sending packets, multiple users can operate on the same channel and uses bandwidth more efficiently. Other important changes in 4G were overall increased performance of the network with better end-to-end latency, download/upload speed, and increased capacity.

2.1.1 System Overview

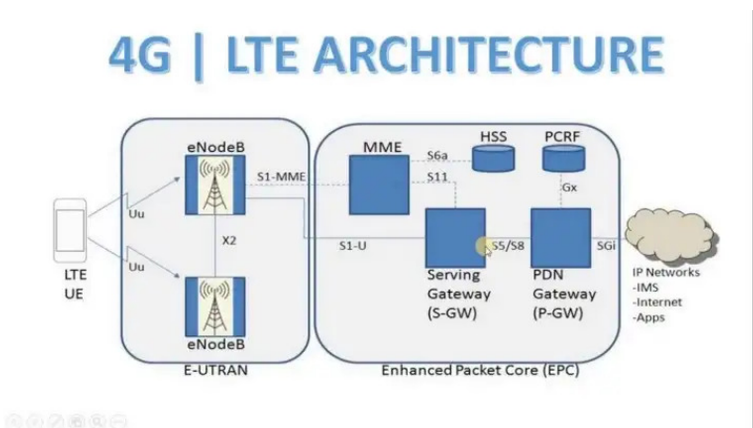


Figure 2.1: The 4G Network Architecture [18].

As seen in figure 2.1, the 4G network can be divided into 3 main parts:

- User Equipment (UE)
- Evolved UMTS Terrestrial Radio Access Network (E-UTRAN)
- Enhanced Packet Core (EPC)

Communication in the network is split into two categories; the user plane and the control plane. The user plane carries the user data traffic across the network, and the control plane carries the signaling traffic between the internal network functions. 4G interfaces allow the components in the 4G network to interact with each other either within the core or from either basestations and user equipment outside the core.

2.1.2 User Equipment

User Equipment (UE) refers to the equipment used by the user when interacting with the network and can be any device from a phone to a computer as long as it is connected to the 4G network. The UE will register itself to the network by using Non-access Stratum signaling (NAS) to contact the Access and Mobility Management Function (AMF), which will register the UE to the network.

2.1.3 E-UTRAN

Evolved Universal Terrestrial Radio Access Network (E-UTRAN) is the air interface part of the 4G network, allowing UEs to connect to the network core. It consists of the base station called Evolved Node B (eNodeB) and uses radio signaling to connect the UE to the network.

2.1.4 Evolved Packet Core

The Evolved Packet Core (EPC) is a part of the SAE (System Architecture Evolution), and is the brain of the network in charge of most of its logic. The EPC mainly consists of these different sub components:

- **Mobility Management Entity (MME):** Is the control node in the Core, in charge of the session and mobility management.
- **Serving Gateway (SGW):** Routes user plane packer data, and serves as a mobility anchor during UE eNodeB handovers.
- **Packet Data Network Gateway (PGW):** Connects the UE to any external network, eg. the internet, and filters packets.
- **Home Subscriber Server (HSS):** Contains user data including subscription information.
- **Access Network Discovery and Selection Function (ANDSF):** Assists the UE in discovering non-3GPP networks such as WiFi, and provides rules on how to manage these networks.

If a user wanted to access the internet through the cellular network from their phone, the UE would connect itself to the EPC through the eNodeB base stations. The base station would then connect itself to the MME, which would contact

the HSS to get the user subscription. After validating the UE, the MME would contact the SGW in order to route the user plane data, and then the SWG would route the data into the PDN which would in turn route the data to the third-party network. This would allow the UE to send and receive packets on the internet.

A drone can be controlled over the 4G network by having a 4G modem which allows it to connect to a nearby eNodeB station. The drone's 4G connections enable it to send and receive packets over the network, with an average end-to-end latency of 200 ms.

2.2 5G

5G is the 5th generation mobile cellular network provided in the 3GPP specifications [4]. In comparison to the older generation of mobile cellular networks (1G-4G), 5G restructures the broadband network architecture. The previous generations can be seen as having a static network topology due to hardware and system architecture requirements. 5G restructures the architecture, its interfaces, and the network functions in order to allow the network to be programmable and dynamic.

2.2.1 System Overview

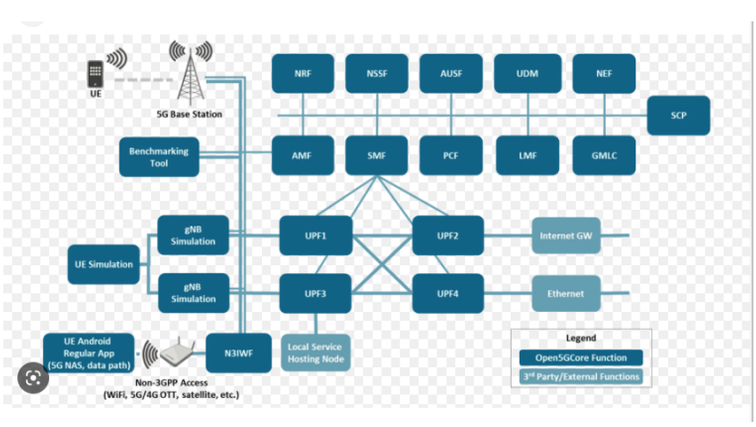


Figure 2.2: The Network Functions of the 5G Core [29].

The 5G network is split into three main parts:

- New Radio User Equipment (NR UE)
- Next Generation Radio Access Network (NG-RAN)
- System Architecture Evolution (5G Core NW)

In 5G the user plane is defined as the data transmissions between the user's UE and the core network. The control plane is defined as the data transmissions in the core network.

2.2.2 New Radio User Equipment (NR UE)

5G New Radio (NR) is a new radio technology that is developed by 3GPP which uses the frequency ranges of 410 MHz – 7125 MHz and 24250 MHz – 71000 MHz. New Radio User Equipment is any user equipment that has NR connectivity. NR UE will be needed to connect to the new 5G network. The higher frequency band allows for a higher information transmission rate (Gbit/sec), in addition, the new technology has been optimized for low latency.

2.2.3 Next-Generation Radio Access Network (NGRAN)

Next-Generation Radio Access Network (NGRAN) is the air interface that utilizes NR. NGRAN consists of the next-generation base station gNodeB.

2.2.4 Core Network

The 5G core network has the same responsibilities of the 4G core and is in charge of connecting the user to its services. However, its architecture is vastly different as it has a service-based architecture (SBA). In a SBA the software components are used in order to create business applications. The components are built for reusability, allowing services to be quickly deployed.

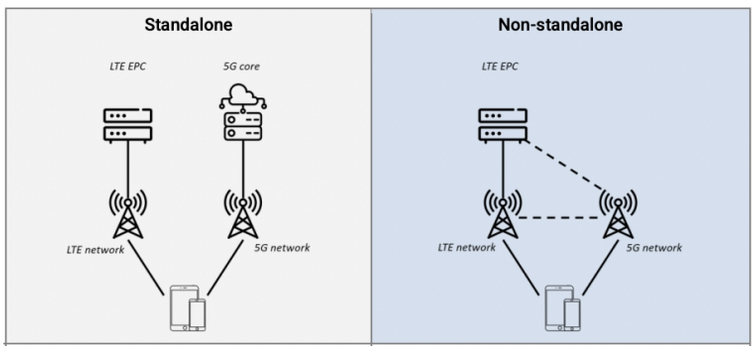


Figure 2.3: Standalone and Non-Standalone Deployment Models, retrieved from STL Partners [40]

There are two different deployment models for the 5G Core; standalone and non-standalone. A standalone core is a 5G core that utilizes NGRAN and NR UE. While a non-standalone core uses NGRAN and NR but uses a 4G core. We might expect this in the early stages as operators roll out 5G upgrades such as 5G NR. Just like 4G LTE was marketed as 4G even though it was not true 4G, telecommunication companies are allowed to market their cellular networks with a 4G core, and New Radio as 5G. Even though it does not provide the full features of 5G in terms of reliability, performance, security, and programmable network.

2.2.5 Network Functions

The 5G network is divided into network functions that have specific responsibilities; the main ones are the following:

- **Access and Mobility Management Function (AMF):** manages connection, registration, and the mobility of the UE. Contains some of the functionalities from the MME in the EPC.
- **Session Management Function (SMF)** manages the UE session, in charge of session authorization.
- **User plane Function (UPF)** handles the user plane routing and forwarding in the 5G network.
- **Policy Control Function (PCF)** provides network policies eg: access and mobility policies, for example, maximum bit-rate, and priority for service.

- **Unified Data Management (UDM)** is a database containing subscriber data. The data includes subscriber profiles, preferences, and bill information.

- **Authentication Server Function (AUSF)** provides authentication for 3GPP and non-3GPP users.

- **Application Function (AF)** provides application services to operators or 3rd parties.

- **Network Exposure Function (NEF)** is a gateway between the 5G core network and the outside world and can expose services and capabilities to 3rd parties.

- **Network Repository Function (NRF)** maintains an updated list of which functions are present in the network. Allows network functions to register themselves, and look up other network functions.

- **Network Slice Selection Function (NSSF)** manages the network slice instances to provide the UE.

2.2.6 Overview

The user connects to the 5G network by establishing a wireless connection from their phone to the nearest gNodeB base station. The gNodeB relays the wireless connection to the AMF, which registers the UE with the identity and authenticates their subscription before establishing a secure connection. Once the UE is authenticated, the SMF will establish a connection between the UE and external networks by providing it with an IP address. The UPF handles the data forwarding between the UE and external networks, while the PCF enforces the quality of service and policy compliance.

2.2.7 Network Slicing

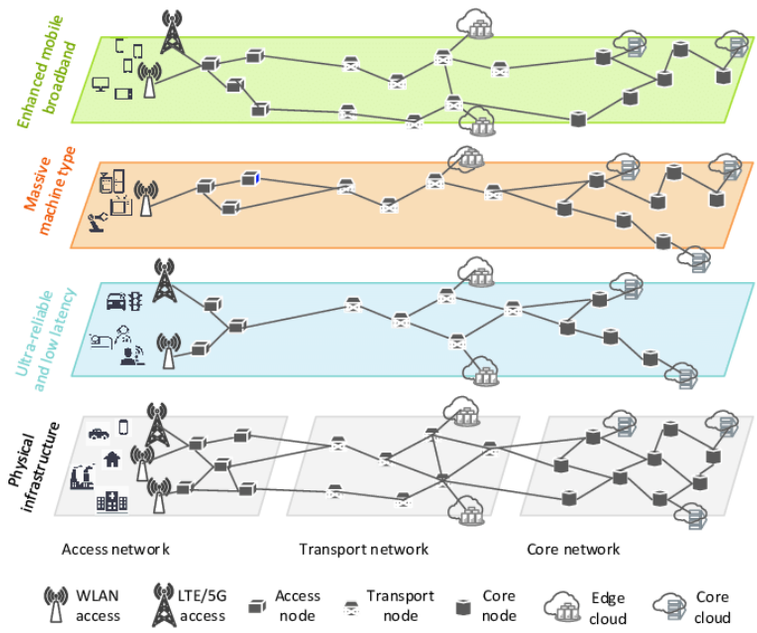


Figure 2.4: 5G Network Slicing [13].

Network slicing refers to multiple virtual network overlays on top of a physical network. The 4G network follows a one size fits all principle which does not efficiently cover different network use cases. As seen in figure 2.4 the 5G network can be divided into network slices with different configurations. The slices can be configured to fit customer needs like low latency, high capacity, or high reliability. An example use case for this is if a customer wanted to deploy a service that required high reliability with low latency such as autonomous drones. The 5G network could create a network slice configured with these properties. 3GPP has defined three pre-configured 5G network slices for different use cases.

- **Enhanced Mobile Broadband (eMBB):** Provides high speed, high connectivity mobile broadband. Can be used by users who wish to stream 4K videos.

- **Ultra-Reliable and Low Latency Communications (URLLC):** Provides high reliability and low latency for applications that have high reliability and latency requirements such as drone operators.

- **Massive Machine-Type Communications (mMTC):** Provides low power

and low data for devices for equipment used in factories or IoT devices.

2.2.8 Summary

The most significant difference between 4G and 5G is the architectural change. The architectural change and its implementation provide better overall performance and network slicing. Network slicing is particularly useful because it customizes the network to fit users' needs. 5G also offers better performance in terms of lower latency, higher bandwidth, and making it ideal for drones that require reliable connections and low latency. The advancement of 5G technology can meet the performance requirements for drones and allow them to be piloted more accurately with increased performance.

2.3 Cloud Native 5G Core

Container Technology refers to configuring and managing virtual containers, which are lightweight, self-contained software consisting of code and dependencies. This allows them to be deployed across different computing environments while keeping the same behavior. Docker [11] is an open-source software platform that allows software to be containerized. Kubernetes [44] is an open-source container orchestration tool that can automate container deployments and scale them dynamically. Microservice architecture is often implemented using container orchestration tools like Kubernetes that manage and deploy Docker containers. This type of architectural style structures systems as collections of small, loosely coupled, and independent services. The system is broken down into specialized services that are often responsible for business functionality. The advantage of microservice architecture is that systems can be deployed and scaled more efficiently. The disadvantage of the microservice architecture is added complexity provided by managing inter-service communication and data consistency. Cloud Native 5G Core is a microservice deployment of the 5G Core. Each network function is containerized and managed by a container orchestration tool like Kubernetes to provide 5G functionalities. This can provide more flexibility in services, faster time to market, and flexibility to update the services. Edge computing is a distributed network paradigm that brings computation and processing closer to the data source. Cloud Native 5G Core can provide the network architecture which enables edge computing which can provide coordination between machine learning models and drone technology, by bringing the models closer to the drone which can decrease the latency.

2.4 Drone Technology

Drones or UAVs (Unmanned Aerial Vehicles) are remotely or autonomously controlled without the need of input from a pilot. Controllers (ground stations) are the devices used to control the drone, often with radio signals. A drone is usually made up of a power source, a controller, a sensor, and a camera. Drones come in various sizes, from the micro drone Black Hornet [28] weighing 32 grams to large drones like the American MQ-1 Predator drone weighing 513 kg [17]. They can be used for crop monitoring in agriculture [20], 3D mapping of local areas [7], and search and rescue operations [42]. Drone piloting in Norway is regulated by the Civil Aviation Authority Norway (CAAN) [9], which enforces EU regulations 2019/947 on civil drones [14]. Citizens operating drones of more than 250g (open category) must register themselves for the drone certificate. Drone operators must apply for permission to CAAN if the drone operation cannot be conducted in the open category. This applies to operating drones over 25kg or flying beyond visual line of sight (BVLOS). Some of the rules that must be adhered to when operating in the open category are:

- **Height**: Drones can fly up to 120 meters above ground.
- **Visual Contact:** Drone operator must be able to get visuals on the drone.
- **Satefy Distance:** A safety distance of 500 meters between buildings and other objects is recommended.
- **Night Flying:** It is illegal to fly during the night.
- **Airport:** It is illegal to operate a drone within 8 kilometers of an airport, unless given the authorization.
- **Military:** It is illegal to fly over military bases.

2.4.1 Parrot Drone

Parrot SA is a wireless electronics manufacturer based in Paris, France [35]. They focus primarily on the production and sale of drones and software development kits (SDKs) surrounding it. The Parrot ANAFI Ai is the world's first commercial 4G drone [38]. The drone is controlled by the Parrot Sky Controller 4, when attached to a tablet allows the drone to be piloted with the FreeFlight [53] IOS app developed by Parrot.

Figure 2.5: The Parrot ANAFI Ai with the Skycontroller 4 [27].

Some of the features of the drone are the following [37]:

- **Camera** 4k video, and 14 MP camera quality.
- **Connectivity** WiFi connection, and 4G connectivity.
- **Obstecal Avoidance** Avoids obstacles automatically with sensor data.
- **Photogrammetery** Drone is designed for photogrammetry (turning video to 3d models), compatible with PIX4D.
- **Server** Exposed web API be used to access and manage flight plans, media files, and flight reports.

Other relevant information about the drone is that it has a top speed is 17 ms forwards, a battery time of 32 minutes, and satellite GPS. Parrot also provides drone users with the open-source software development kit (SDK) platform Parrot SDK [36] which can be used by developers to use code to interact with the drone. The Parrot SDK includes:

- **Open Flight [39]** The open source core of FreeFlight.
- **Air SDK [21]** A C++ framework for running code on the drone, has access to sensors and creates flight plans.
- **Ground SDK [34]** A Ground Control Station framework for Android mobile devices.

- **Sphinx [60]** A drone simulation tool built using the game engine Unreal Engine 4 [56].

- **Olympe [2]** An SDK that provides a drone controller interface in Python.

The Parrot ANAFI AI possesses unique attributes such as 4G Connectivity and Parrot SDK compatibility, allowing it to be used for rescue operations. 4G connectivity enables the drone to be piloted in remote areas, and Parrot SDK compatibility opens up many possibilities for augmenting it with programming. The drone also allows for drone mission plans to be uploaded and executed, allowing the drone operate autonomously.

2.4.2 MAVLink

MAVLink is short for Micro Air Vehicle Link and is an open-source communication protocol for communicating between UAVs and ground stations [22]. The protocol was released by Lorenz Meier in 2009 and is commonly used by other drone manufacturing companies. The ANAFI AI drone supports its own version of MAVLink and allows MAVLink files to be uploaded to the drone as a GroundSDK flight plan. A flight plan is a description of a planned flight, that the drone will follow. This can include flying towards way points which are coordinates defining a geographical point in space, or other actions such as taking a photo, recording a video, streaming, or returning to home. MAVLink can be uploaded and started on the ANAFI Ai as a flight plan allowing the drone to fly autonomously. Missions can be created and uploaded using the FreeFlight app, or by using another ground control station such as QGroundControl [45], which supports the MAVLink protocol. A MAVLink mission is formatted shown in figure 2.6.

```
QGC WPL <VERSION>
<INDEX> <CURRENT WP> <COORD FRAME> <COMMAND> <PARAM1> <PARAM2> <PARAM3> <PARAM4> <PARAM5/X/LATITUDE> <PARAM6/Y/LONGITUDE> <PARAM7/Z/ALTITUDE> <AUTOCONTINUE>
```

Figure 2.6: The MAVLink file format.

The first line of the file specifies the file format and version information, and subsequent lines show mission items. Index refers to the index of the mission item. Current WP refers to the active waypoint the drone is flying towards. Coord Frame provides information on the position and orientation of the drone. Command refers to drone commands such as takeoff (22), go to waypoint (16), or start image capture (2000). Parameters 1-7 are reserved for command

arguments. Command parameters 5-7 are reserved explicitly to latitude, longitude, and altitude. Autocontinue informs the drone to continue executing the command sequence automatically.

```
QGC WPL 120
0       1       3       22      0.000000        0.000000        0.000000        0.000000        69.657800       19.61418        20.000000       1
1       0       3       16      0.000000        0.000000        0.000000        0.000000        69.662758       19.85094        20.000000       1
2       0       3       21      0.000000        0.000000        0.000000        0.000000        0.0000000       0.000000        0.0000000       1
```

Figure 2.7: A MAVLink file.

Figure 2.7 is MAVLink file instructing the drone to launch (command 22) from the GPS coordinates 69.65, 19.61 with an altitude of 20 meters. The drone is then instructed to navigate to the waypoint (command 16) at GPS coordinates 69.66, 19.85. Finally, it is told to return to launch (command 20). MAVLink files can be sent over 4G, which allows drones to execute the drone mission autonomously.

2.5 Machine Learning

Machine learning (ML) is a sub-field of Artificial Intelligence (AI) dedicated to developing algorithms and models that perform specific tasks without programming. The models are developed using techniques from mathematics and statistics to analyze patterns in the data. A model is usually trained by using an algorithm to analyze a data set. The patterns, relationships, and structures in the data that have been learned can be used to predict outcomes based on unseen data. There are three main sub-fields in machine learning:

1. **Supervised Learning** refers to training models with a labeled data set. The provided data is manually labeled, which the model uses to train. An algorithm generates a function mapped to the label's input and output, and the goal is to generalize the training data so that the model can accurately predict outcomes of unseen data. The model uses an evaluation function to provide determine the performance of the model.

2. **Unsupervised Learning:** refers to machine learning attempts to recognize patterns in unlabeled data sets. The goal is to discover patterns and relationships hidden from humans. A common technique in unsupervised learning is clustering. It involves grouping data from the data sets based on similarities or dissimilarities.

3. **Reinforcement Learning:** is a machine learning method that trains a model by rewarding or punishing desired behavior. It does not use labeled data sets but focuses on finding a balance between letting the model explore and letting it learn.

A neural network is a machine learning model that is inspired by human neurons and is designed to learn from data to make predictions. It is a system of connected nodes (neurons) where the information moves from the input through layers of neurons to the output. Each neuron performs computation based on the input from a previous neuron and the output is given to the next neuron in the layer. The neurons apply an activation function to the received input which determines the output. Neural networks can be trained to recognize and categorize images.

The applications of machine learning spread into many industries like healthcare, finance, and manufacturing. Machine learning can synergize with drones by providing object detection, autonomous navigation, and video analysis. The deployment of 5G can significantly enhance the capabilities of drones by providing an infrastructure to support the processing power required by machine learning. 5G supports edge computing, allowing machine learning models to process and store data closer to the drone without relying on cloud servers. This can decrease the latency and improve decision-making, which is critical when operating drones.

2.5.1 Transformer

A transformer is a deep learning architectural model introduced in 2017 by Google Research [57]. It's commonly applied to training natural language and is used for training the GPT language model family and the automatic speech recognition system Whisper and language model BERT [10]. The key components of the transformer are the self-attention mechanism and the feed-forward neural network. The self-attention mechanism mimics human cognitive attention and is used to apply focus on specific parts of the input data during processing, which allows the model to capture relationships and contexts in the data.

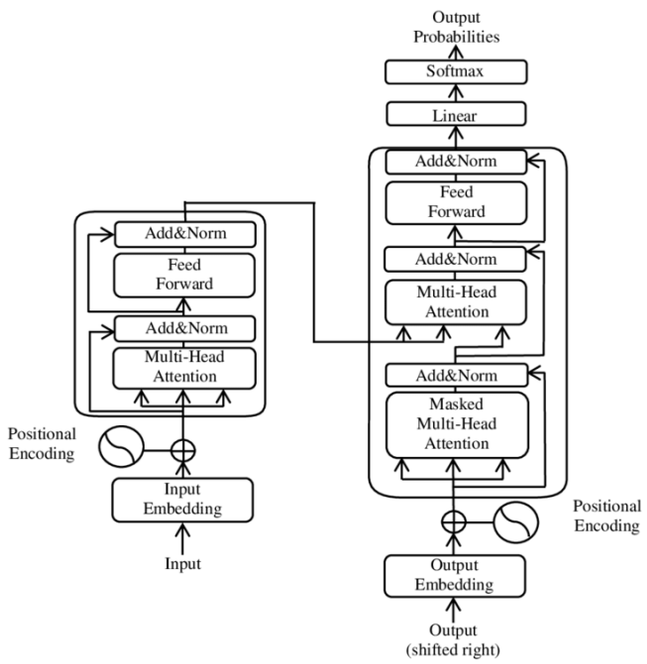


Figure 2.8: The Transformer Architecture [57].

Overview

The transformer takes an input sequence $X = [x_1, x_2, ..., x_n]$ which can be a set of characters. The input is split into tokens and embedded with encodings. The encoder processes the input sequence and generates the weights that capture the contextual information of each token. The decoder generates an output sequence based on the encoded sequence. The output is passed through a linear transformation and a Softmax function to calculate the probability distribution of the output sequence.

The input given to the encoder is first embedded with an input, output, and positional embedding. Which can be denoted as $E = [e_1, e_2, ..., e_n]$, where each e_n is the embedding vector for the corresponding token x_n. The input embedding captures the token's semantic meaning, and the output embedding captures the token's contextual data based on the attention mechanism. The positional embedding capture the token's position in the input sequence.

The encoder then processes the embedded input data (not the input sequence) by passing it through the self-attention sub-layer. An important feature of the encoder is that each token traverses its independent path, allowing the model

training process to be parallelized. The self-attention layer computes new self-attention weights and applies them to the value matrix. The self-attention output contains information on how each token relates to other tokens in a sequence. The feed-forward sub-layer passes the data through a neural network using non-linear transformations. The output is calculated for each token in the input sequence. In the transformer, the encoder can be stacked N times to encode more information. The Decoder generates sequences based on the input given by the encoder. The decoder contains both a self-attention and a feed-forward sublayer, but a masked Multi-Head attention layer is between them. This layer is a mask that removes any tokens ahead of the current token.

The output of the final encoder is passed through a linear neural network which acts as a classifier. The Softmax activation takes the weights in order to generate a probability distribution of each position of the output sequence. The output of the transformer can be used to words or sentences, which can be used for complex use cases such as chat bots, language completion models, and speech recognition models.

2.5.2 Whisper

Whisper is a general-purpose speech recognition model developed by Open Ai [46], which can transcribe audio to text. The open-source model is a paid service at $0.006 / minute [43] through the Open AI API [31]. The API supports audio files size up to 25MB, meaning longer audio will need to be separated. It is trained on 680 000 hours of multilingual data, allowing audio to be transcribed in multiple languages.

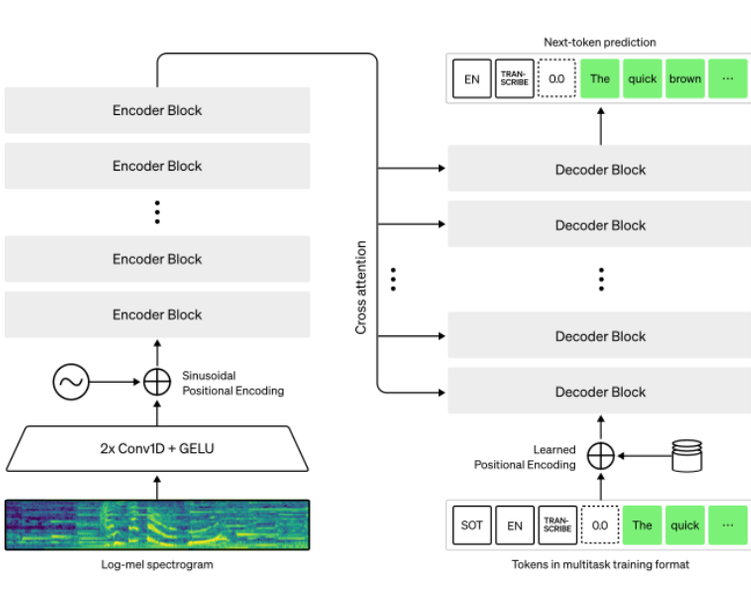


Figure 2.9: The Whisper Architecture [46].

Whisper uses an encoder-decoder transformer which takes an audio input and outputs text. The audio input is split into 30 seconds chunks and converted into a log-Mel spectrogram. A Mel scale [26] is a pitch scale in which listeners perceive the pitch to be of equal distance. Humans perceive lower frequencies such as 500 MHz and 1000 MHz to be very different, but higher frequencies, such as 6000 MHz and 8000 Mhz, to be barely noticeable. A log-Mel is a logarithmic spectrogram that provides a visual representation of noise. The log-Mel spectrogram is fed into the encoder, which parses it for tokens. The decoder receives the encoder output and predicts the text using tokens that it has received.

2.5.3 OpenAI Chat

OpenAI Chat is a chat completion model family developed by OpenAI that provides chat completion from a prompt. A prompt is an input sent to the chat completion model containing instructions or requests to complete the provided text. The request can be to draft an email, translate a sentence, or generate a MAVLink file. The model is accessed using an API and should not be confused with ChatGPT, a general-purpose chat bot AI developed by OpenAI. They are both built on top of the large language model Generative Pre-trained Transformer 3.5 which is closed-source and uses a transformer architecture.

The API service charges per token, meaning per word. OpenAI Chat offers the following GPT 3.5 model family, which understands natural language and code:

Model	Description	Max Tokens	Training Data	Pricing per 1k tokens
gpt-3.5-turbo	Chatbot using GPT-3.5, updated iteratively	4,096 tokens	Up to Sep 2021	$0,002
text-davinci-003	Chat completion model using GPT-3.5, can translate text, trained using reinforcement learning	4,096 tokens	Up to Sep 2021	$0,02
text-davinci-002	Similar capabilities as text-davinci-003, but trained using supervised fine-tuning	4,097 tokens	Up to Jun 2021	$0,02
code-davinci-002	Optimized for code completion tasks	8,001 tokens	Up to Jun 2021	$0,02

Table 2.1: Model description, with maximum tokens, training data, and pricing [43].

2.6 Summary

This chapter presented the 4G and 5G cellular network. This included presenting network slicing and cloud-native 5G core and its possibilities to be used with drones. Drone technology the Parrot Drone and SDK, and MAVLink were then presented. An overview of machine learning and the transformer was given, and Whisper and OpenAI Chat API was presented.

3

Design & Implementation

This chapter presents the design and implementation of the system. The system aims to enable automatic mission creation for drones based on expressed intent to aid first responders during PPDR missions. The user interacts with the system using a mobile device, the communication is managed using a REST API, a drone mission plan is automatically generated based on transcribed voice commands using machine learning models, and the drone can be programmed to follow mission plans.

3.1 System Overview

To automatically generate drone missions based on the users intent, the system is designed to provide the following functionalities:

1. User Input
2. Communication Management
3. Mission Generation
4. Programmable Drone

The user input functionality allows users to interact with the system using their expressed intent. This is implemented using the audio recording captured by a microphone on a mobile device. The communication management functionalities handle the communication between the system components and is implemented as a REST API. The mission generation functionality automatically generates drone plans based on the user intent. This is implemented using an automatic speech recognition model to transcribe the intent in the form of an audio recording, and the mission plan is generated by a large language model from the transcribed intent. The programmable drone can be programmed to execute drone mission plans. This is implemented by using the ANAFI AI drone, which can execute MAVLink files. The system implementation is illustrated in figure 3.1.

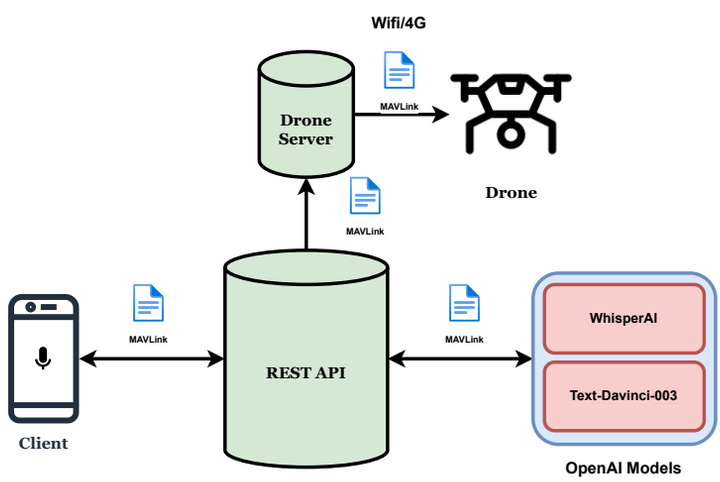


Figure 3.1: The System Implementation.

3.2 User Input

The user input functionality aims to provide the user a way to express their intent to the system. This is implemented using a mobile device because they have small weight which is practical during PPDR missions, and allows the user to stay connected to the cellular network, giving them access to the internet at all times. This enables them to communicate with the system. Mobile devices are also usually equipped with a microphone allowing users to express their intent through audio recording which is more practical using a keyboard for input. The chosen mobile device was an iPhone 11 Pro [41] running IOS [**ios**] because it is less susceptible to malware and security breaches than a device

running the Android operating system [19].

The mobile device should provide the user with a user interface that allows it to interact with the system. This is implemented as a mobile application using the open-source mobile development JavaScript framework React Native [47] with the programming language typescript [23]. React Native allows the code base to be developed once for Android and IOS platforms. The programming language Typescript was used because it improves the development experience of Javascript by providing a type system that allows better type safety. Expo [15] is an open-source platform for app development and was used to bundle and run the application software in the IOS application Expo Go [52]. Expo was used because it allows the mobile application to be set up quickly and provides an ecosystem with many libraries. The functionalities that are provided by the mobile application are:

1. Voice Recording
2. Mission Plan Verification
3. Intent Execution

Each functionality is implemented as a separate screen in the mobile application and is accessed with a bottom tab navigator as depicted figure in 3.2.

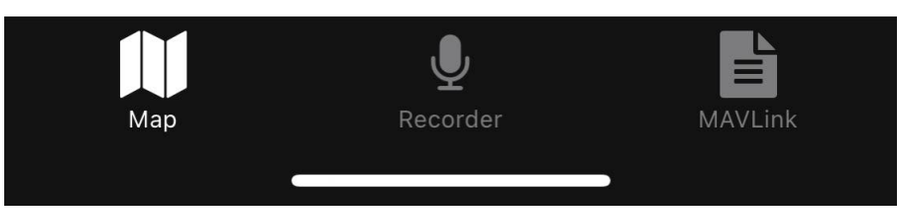


Figure 3.2: The Mobile Application Tab Navigator.

3.2.1 Intent Input

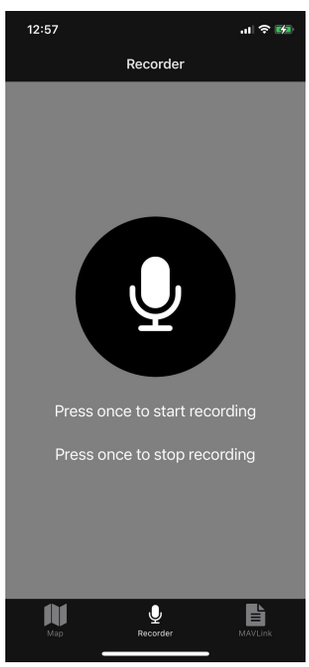
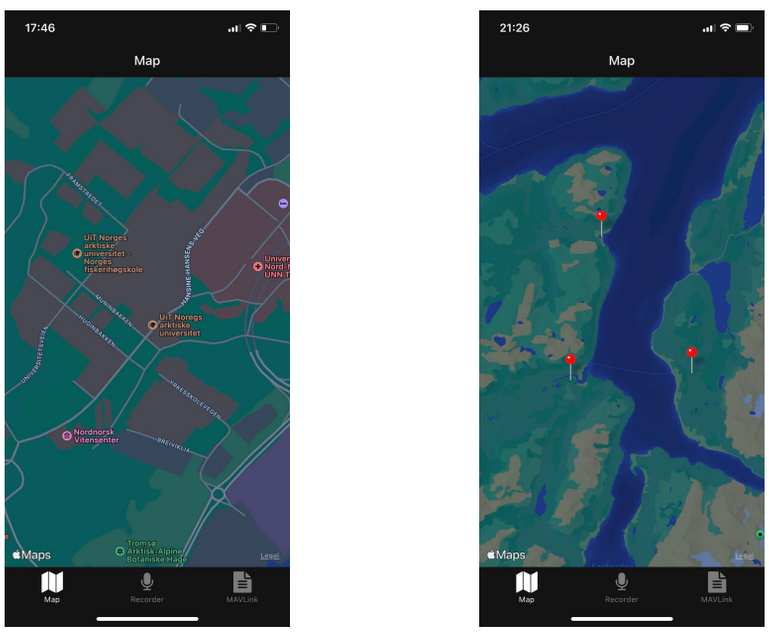


Figure 3.3: The Mobile Application Recording Screen.

The intent input functionality allows users to express their intent to the system as an audio recording. This is implemented using Expo Audio Recorder component[6] that allows audio recording and playback in the mobile application. The user can record their intent on the audio recording screen as depicted in 3.3. It allows the user to start and stop a voice recording by pressing a microphone button on the screen. When the user has pressed and recorded the intent, the recording is stored as an M4A audio file before it is packaged into a multipart JSON object and sent to the REST API.

3.2.2 Mission Verification

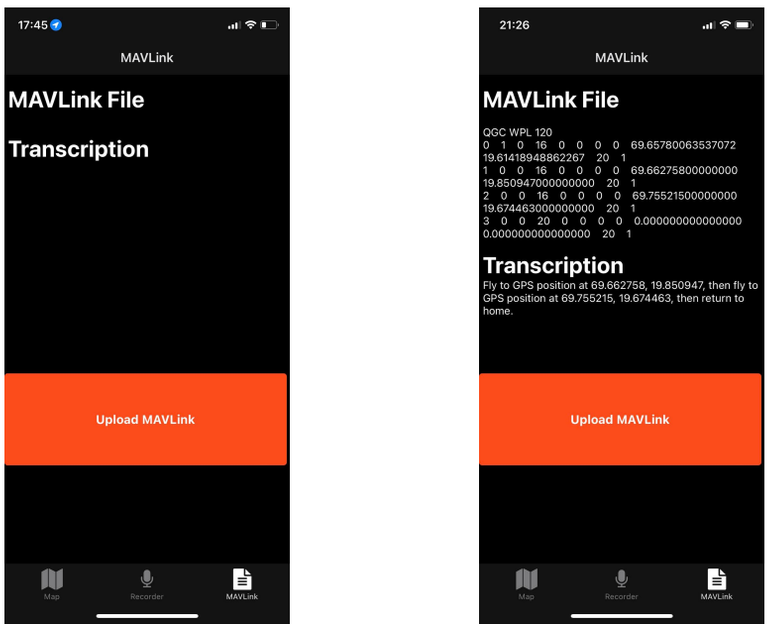


(a) Map screen initial render. **(b)** Map screen with coordinates.

Figure 3.4: Before and after the mobile application has received coordinates.

The mission plan verification functionality allows the user to verify that the system has understood their intent. This is achieved by displaying the drone mission plan coordinates as markers on a map. The map is implemented using the Expo MapView component [25], which provides the mobile application access to Apple Maps. Once the system has processed the user's intent, the system will send the markers to be rendered on the map. As depicted in 3.5, the coordinates are rendered as red markers, allowing the user to visually verify that their intent has been translated correctly to a drone mission. Before the user has expressed their intent, the map will ask for the user's location to render the map in their area, in order to not render an empty map.

3.2.3 Intent Execution



(a) MAVLink screen upon start. **(b)** MAVLink screen after intent processing.

Figure 3.5: Before and after the system has processed the users intent.

The intent execution functionality allows the user to execute the translated intent on the programmable drone. This is implemented as a screen (MAVLink screen) that contains an upload button that, when pressed, informs the system to upload and execute the drone plan on the programmable drone. The screen also displays the intent translated from a voice recording to a text transcription, allowing the user to verify that the system has correctly translated their intent. In addition, the raw unparsed drone plan is displayed, allowing the user to inspect the content that the drone receives and executes.

3.3 Communication Management

The communication management functionality handles communication between the components of the system. This is implemented using two REST APIs; One for managing the communication required for mission generation and the other to control the programmable drone. Both REST APIs are implemented as web servers to allow the other system components to interact with them. The web servers are implemented using the Ubuntu version 22.04.2

LTS x86_64 [8] because it provides a good user experience for developers. In addition, the drone requires the Python [59] library Parrot Olympe to be programmed, which is only compatible with Ubuntu. Both web servers were implemented using the following hardware:

CPU 11th Gen Intel i7-11700 (16) @ 4.800GHz

GPU Intel Rocket Lake-S GT1 [UHD Graphics 750]

RAM 14GB (16GB)

3.3.1 Mission Generation Communication

Intent generation communication manages the communication between the system components to generate the drone mission based on the user's intent. It is implemented using a web server using the Rust programming language [49] and the Rust web framework Actix Web [3]. Rust was chosen because of its memory safety without a garbage collector. Instead, Rust uses a borrow-checker and a lifetime system, allowing for both memory safety and high performance. Actix Web was chosen because it is a production-ready, stable, and fast web framework for Rust. The server was containerized in a Docker application in order to simulate the deployment of a 5G edge node.

The web server receives the intent from the mobile application to generate the drone plan based on the user's intent. It sends it to the mission generation models that translate the user's intent into a drone mission. The drone mission is sent to the mobile application for the user to verify that the system has correctly translated the intent it can choose to upload and execute the mission on the drone. The web server exposes two API endpoints that allows the mobile application to interact with the system:

POST \api\prompt receives intent as an input and outputs a drone mission.

GET \api\upload\id uploads the drone mission plan and executes it on the drone.

Translating Intent

The \api\prompt endpoint allows the user to translate intent into a drone mission by sending a POST request from the mobile client to the web server

containing an audio file. The server parses the request and writes the M4A audio file to the disk. To translate the user's intent, the audio file is transcribed to text using OpenAI Whisper API. This is implemented by establishing a session client using the Rust crate async-openai [5]. The client is provided an Open AI API Token, which allows a transcription request of the audio recording to the "whisper-1" model.

Once the user's intent has been transcribed from the audio recording to text, the text is inserted into a prompt that is sent to a chat completion model to generate a drone mission. The user's intent is expressed by inserting the transcribed audio into a pre-designed prompt containing instructions on generating a drone plan. The drone plan is implemented using the MAVLink file format because it is compatible with the ANAFI AI drone, allowing the plan to be uploaded and executed on the drone; in addition, the chat completion model can generate MAVLink files based on instructions. Therefore, the prompt contains instructions on generating a MAVLink file using the audio transcription of the user.

Mission Generation

The mission is generated by prompting the chat completion model. The prompt requests the model to generate a MAVLink file based on a hard-coded GPS position and provide it with the official MAVLink file format [22] to use. A MAVLink file example is also provided before the user's transcribed text is inserted, containing instructions on what the MAVLink file should contain. This provides the chat completion model with an understanding of the MAVLink format before being instructed on how the file should be generated based on the user's transcribed text. Finally, the prompt contains restrictions such as altitude restrictions, no identical GPS positions, GPS positions of the same length, and expressing the desired MAVLink version of 120. It also requests the model not to explain the output to parse it easier and save money on tokens. The instruction "*think step by step*" is added in the end because studies have shown that large language models perform better by adding the instruction to their prompt [24].

> Generate a MAVLink mission file from the following starting position: 69.65780063537072, 19.61418948862267. Using the MAVLink Mission Plain-Text File Format. Using the following format:
>
> QGC WPL <VERSION> <INDEX> <CURRENT WP> <COORD FRAME> <COMMAND> <PARAM1> <PARAM2> <PARAM3> <PARAM4>

<PARAM5/X/LATITUDE> <PARAM6/Y/LONGITUDE <PARAM7/Z/ALTITUDE> <AUTOCONTINUE>

For example:

QGC WPL 120
0 1 0 16 0 0 0 0 8.54800000000000004 47.3759999999999977 20 1
1 0 0 16 0 0 0 0 8.54800000000000004 47.3759999999999977 20 1
2 0 0 16 0 0 0 0 8.54800000000000004 47.3759999999999977 20 1

Based on the provided starting position, create a MAVLink mission file that does the following [TRANSCRIPTION HERE]. The altitude parameter should always be 20. GPS position should be precise. No waypoint should have identical GPS positions. GPS positions in the file should be of the same length, fill the remaining length with 0. The file format should be in MAVLink Mission Plain-Text File format QGC WPL 120. Provide no explanation. Think step by step.

The chat completion model returns the drone mission plan as a MAVLink file based on the prompt containing the user's intent. It is saved on disk as a JSON object and given an id of 0. This is because the user can request to upload the mission plan to the drone after it has inspected it. The MAVLink file is then parsed for coordinates, allowing the client to render it onto a map, allowing the user to visually confirm that their intent has been translated correctly. As depicted in listing 3.1, the MAVLink file, the markers (coordinates), and the transcription are inserted into a JSON object sent to the mobile application. The following prompt is sent to the chat completion model, TRANSCRIPTION HERE is where the audio transcription is inserted:

```
{
  "markers": Array [
    Object {
      "coordinate": Object {
        "latitude": 69.65780063537072,
        "longitude": 19.61418948862267,
      },
      "id": 0,
    },
    Object {
      "coordinate": Object {
        "latitude": 69.662758,
        "longitude": 19.850947,
      },
      "id": 1,
    },
    Object {
      "coordinate": Object {
        "latitude": 69.755215,
        "longitude": 19.674463,
      },
      "id": 2,
    },
    Object {
      "coordinate": Object {
        "latitude": 0,
```

```
        "longitude": 0,
      },
      "id": 3,
    },
  ],
  "mavfile": "QGC WPL 120
    0    1    0    16    0    0    0    0    69.65780063537072    19.61418948862267    20    1
    1    0    0    16    0    0    0    0    69.66275800000000    19.850947000000000    20    1
    2    0    0    16    0    0    0    0    69.75521500000000    19.674463000000000    20    1
    3    0    0    20    0    0    0    0    0.000000000000000    0.000000000000000    20    1",
"transcription": "Fly to GPS position at 69.662758, 19.850947, then fly to GPS position at 69.75521
5, 19.674463, then return to home.",
}
```


Listing 3.1: JSON server response containing raw MAVLink and markers

3.3.2 Drone Programming

Drone programming is implemented by taking the mission plan generated by the chat completion model, and uploading and executing it on the drone. This allows the user to execute their intent. This is implemented by the Rust Web API endpoint \api\upload\id which uploads the mission plan to the drone. The Rust Server does this by retrieving the MAVLink file based on the mission plan id requested by the client and sending it to the drone programmer which is implemented using the Python programming language [59] version 3.11 and the Python Web Framework Flask [58]. The drone programmer is implemented in Python because that is the only programming language that Olympe is compatible with. The drone server uses Flask because it is an industry-standard Python server, which allows it to receive and upload mission plans.

The drone server exposes the endpoint api\upload which allows it to receive a MAVLink file. Then a connection to the drone is established using Olympe the drone's IP Address. The server then uploads the MAVLink file and starts the mission. In order to establish a 4G connection to the drone, the Sky Controller is attached via USB-C to the computer. A paired Sky Controller provides access to the drone cellular modem interface and acts as a multi-path passthrough proxy to the drone, exposing the drone's REST API on the address 192.168.53.1:180. The MAVLink file is uploaded to the drone API endpoint api/v1/upload.

3.4 Mission Generation

The mission generation functionality handles mission generation based on the user's intent. This is implemented by translating the intent as an audio recording to text. The text is then used to generate a drone mission. Audio recording translation is implemented using audio transcription with the voice recogni-

tion model Whisper. It was used because it can transcribe audio text efficiently. While Whisper has been released as open-source OpenAI provides the audio transcription service whisper-1 as a paid service through their API which enables the use of dedicated hardware. This can provide better performance than using a local implementation. The API enables the user to have their intent transcribed from audio to text. The audio is recorded on their phone using Expo Recording, which uses Expo M4A audio which has a bitrate of 128kbps. Whisper API accepts a maximum file size of 25MB. If the file is longer it is split up into different recordings, which can decrease the correctness of the recording especially if the recording is cut off mid-sentence. OpenAI recommends that if the file size limit is exceeded the recording should be split up between sentences. This gives the following equation for calculating the maximum audio recording duration per transcription:

$$\text{Max Recording Duration} = \frac{\text{Max File Size}}{\text{Audio Bit Rate}} \tag{3.1}$$

$$\text{Max Recording Duration} = \frac{25\ \text{MB}}{128\text{Kbps}} = 195s \tag{3.2}$$

Once the user's intent has been transcribed, it is inserted into a prompt which is sent to a large language model to generate a mission plan. The mission plan is implemented as a MAVLink file which can be uploaded to the ANAFI AI drone for execution. The system uses the chat completion model Text-Davinci-003 to translate the user's intent into a drone mission. Text-Davinci-003 is used because it can generate MAVLink files based on instructions. The chat completion model is available through an API provided by OpenAI API. Another language model that was not used is GPT-turbo-3.5 which is a chatbot, which is cheaper but cannot be fine-tuned to your liking like Text-Davinci-003. However, Text-Davinci-003 is more expensive $ 0.0200 / 1K tokens than GPT-turbo-3.5 $0.002 / 1K tokens. A downside of using OpenAis chat completion models is that they are closed-source, and there is limited information on the internals of the models. Ultimately, Text-Davinci-003 can translate the user's intent into mission plans which can be executed by the drone. Whisper and Text-Davinci-003 servers are located in the US.

3.5 Drone Control

The drone control functionality is implemented by using the Parrot ANAFI AI drone. It comes with the SkyController 4 which allows it to be piloted and provides access to the cellular modem interface on the drone. The drone was chosen because it can be programmed and has 4G connectivity which enables it to be piloted in remote areas and BVLOS. Its compatibility with the Parrot SDK allows it to be programmed using MAVLink missions. The drone is equipped with a 4G SIM-card.

3.6 Summary

This chapter presented the design and implementation of the system. The goal of the system was to enable users to automatically generate drone missions plans based on their intent. The system does this by capturing the user's intent using a mobile application that allows the user to record a voice command. The intent is translated into text using a voice recognition model and inserted into a prompt that is sent to a chat completion model to translate the intent into a MAVLink file. This process is managed by a Rust server which stores the MAVLink files and allows the user to upload and execute the drone mission plan on the drone using a Flask Server running Olympe. The ANFAI AI drone exposes an API endpoint that allows it to execute a MAVLink file.

4

Evaluation

This chapter describes how the system was evaluated. This includes the details of the experimental setup. It describes how the system's ability to generate drone missions based on intent was tested using correctness and latency tests for audio transcription and mission generation. The results from the tests are presented and evaluated.

4.1 Experimental Setup

The system specifications for the experiments are presented in the tables below. The goal of the system is to generate drone missions based on the users intent. To evaluate the system's ability to translate intent into drone mission plans, the correctness and the latency of key components are tested. The key components of the systems intent understanding are the voice recognition model which transcribes the user's intent, and the chat completion model which generates drone missions based on the intent. To measure the voice recognition models' ability to transcribe the user's intent and the chat completion models' ability to generate drone plans, a correctness and a latency test was executed on both of the models.

Server Specifications	
OS	Ubuntu Linux 22.04
CPU	11th Gen Intel i7-11700 (16) @ 4.800GHz
Ram	14 GB (16GB)
Programming Languages	Rust and Python

Table 4.1: Server specifications.

Mobile Device Specifications	
Hardware	IPhone 11 Pro
OS	IOS 14
Programming Languages	React Native Typescript

Table 4.2: Mobile device specifications.

Model Specifications	
Audio Transcription	OpenAI Whisper1
Chat Completion	OpenAI Text-Davinci-003
Location	United States of America (US)

Table 4.3: Model specifications.

4.2 Testing Intent

As previously stated, the system uses audio transcription to insert the user's intent into the system. The system translates the user's intent into text using a voice recognition model, and the text is used later for drone mission generation. Testing the audio transcription component can provide information on whether audio transcription is a valid implementation choice for capturing the user's intent. This can be tested by measuring how correctly the audio transcription component can transcribe the user's audio recording into text. Another important measurement is measuring the latency of the audio transcription. Both of these tests are essential as the transcribed text has a direct impact on the mission generation as it is used in the prompt that generates drone missions.

In order to perform the audio transcription tests, the intent is captured by using the microphone of the mobile device. The microphone records the intent in the form of a voice command. The voice command can be seen below and takes approximately 1 minute to read. Two tests are conducted; a correctness test and an execution time test. The typical length of a voice command would be between 10-30 seconds. Therefore, the tests will be conducted by reading varying

durations of the voice command; **short** (10 seconds), **medium** (30 seconds), and **long** (1 minute). The correctness is measured per word by comparing the voice command that has been read, to the text that has been transcribed by the model. The latency is calculated by measuring the time it takes for the mobile application that has sent the request to it has received the response from the system. Each test is executed three times where the most correct answer is chosen. The voice command is as follows:

"*Fly to waypoint at GPS coordinate 69.667015, 19.609960, then to 69.66555, 19.605550, then return to home. After that take off and start recording a video, go to waypoint at GPS coordinate 69.667015, 19.609960, stop recording and then return to home. When you have returned home, fly in a circle and take a photo every 10 seconds, then return home.*"

The voice command is based on inputs that can be given to the model to generate a MAVLink file.

4.2.1 Correctness Test

The transcription's correctness is measured in percentages of transcribed words per sentence. For example, if the model transcribes "*Drive* to waypoint at GPS coordinate 69.667015, 19.609960" instead of "*Fly* to waypoint at GPS coordinate 69.667015, 19.609960". The model transcribed $\frac{7}{8}$ words in the sentence correctly because it transcribed every word correctly except it transcribed *Drive* instead of *Fly*. Transcription correctness is expressed using the following equation:

$$\text{Transcription Correctness} = \frac{\text{Correctly Transcribed Words}}{\text{Number of Words}} \tag{4.1}$$

$$\text{Transcription Correctness} = \frac{7}{8} = 87.5 \tag{4.2}$$

The transcription test is given a correctness score of 87.5% because the model transcribed 87.5% of the words correctly.

Model	Short	Medium	Long	Average
Whisper	100%	95%	90%	95%

Table 4.4: Transcription correctness results from the audio transcription model using short, medium, and long audio recordings.

The transcription correctness results from table 4.4 demonstrate that the audio transcription model can transcribe audio perfectly of short duration (10 seconds). At medium (30 seconds) audio duration, the correctness decreases to 95%, and with long (1 minute) audio duration, it decreases to 90%. The overall correctness of all tests demonstrates that the model has a 95% average transcription correctness.

4.2.2 Latency Test

The transcription's latency was measured in seconds for each recording duration.

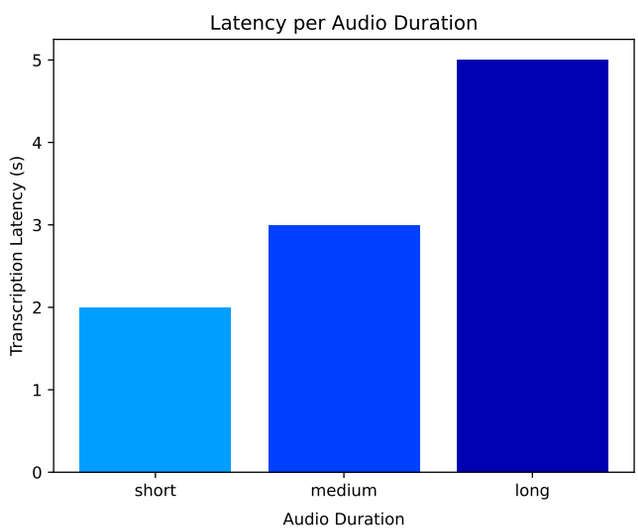


Figure 4.1: Transcription latency of varying audio duration.

The transcription latency results from figure 4.1 demonstrates that the latency for short (10 seconds) audio duration is 2 seconds, for medium (30 seconds) duration is 3 seconds, and for long (1 minute) duration, it is 5 seconds. The

results demonstrate an increase in transcription latency as the length of the audio increases.

4.3 Mission Generation Test

The system generates drone missions using the users intent. The user's intent is inserted into a prompt which a chat completion model uses to generate drone missions as MAVLink files. Measuring the chat completion model's ability to generate a MAVLink file based on intent is essential to understanding whether it is a valid implementation choice. This can be tested by measuring the latency of drone mission generation and the quality of the missions. However, it is challenging to evaluate the quality of the drone mission because the quality of a drone mission is relatively subjective. The mission generation test presents a framework for measuring the quality of the drone plan based on the user's input by assigning the model output a score. The model is provided with intent in the form of commands of varying complexity to discover the models intent understanding capabilities. The correctness and latency of the drone mission generation were measured by inserting commands of varying complexity (easy, medium, hard) into the following prompt:

> Generate a MAVLink mission file from the following starting position: 69.65780063537072, 19.61418948862267. Using the MAVLink Mission Plain-Text File Format. Using the following format:
>
> QGC WPL <VERSION> <INDEX> <CURRENT WP> <COORD FRAME> <COMMAND> <PARAM1> <PARAM2> <PARAM3> <PARAM4> <PARAM5/X/LATITUDE> <PARAM6/Y/LONGITUDE <PARAM7/Z/ALTITUDE> <AUTOCONTINUE>
>
> For example:
>
> QGC WPL 120
> 0 1 0 16 0 0 0 0 8.54800000000000004 47.3759999999999977 20 1
> 1 0 0 16 0 0 0 0 8.54800000000000004 47.3759999999999977 20 1
> 2 0 0 16 0 0 0 0 8.54800000000000004 47.3759999999999977 20 1
>
> Based on the provided starting position, create a MAVLink mission file that does the following [COMMAND HERE]. The altitude parameter should always be 20. GPS position should be precise. No waypoint should have identical GPS positions. GPS positions in the file should

be of the same length, fill the remaining length with 0. The file format should be in MAVLink Mission Plain-Text File format QGC WPL 120. Provide no explanation. Think step by step.

The tests were conducted using three different temperatures of the OpenAI Chat API; 0.0, 0.5, and 1.0. Temperature defines the randomness of the model and is a number between 0 and 1. A temperature of 0 will provide an almost deterministic response, while a temperature of 1 will return more creative responses. The model's output was evaluated based on correctness by a human on a scale (1-5) based on criteria listed below. The commands difficulty was categorized as easy, medium. Each test is executed three times, and the most correct answer is chosen.

4.3.1 Correctness Test

The correctness test measures the correctness of the model output using the evaluation criteria below. The test scores drone missions on a scale from 1 to 5, where 1 is the worst score, and 5 is the best score. It defines the correctness of the model output based on the model input and is evaluated by a human based on the following criteria:

1. **Formatting:** If the MAVLink file is formatted correctly.

2. **Commands:** Whether the MAVLink file contains the appropriate amounts of commands.

3. **Parameters:** If the command parameters correct.

4. **Understanding Intent:** Whether the output demonstrates an understanding of the intent based on the requested input.

The categories easy, medium, and hard, demonstrate varying degrees of complex commands which are used to measure the language models' chat/code ability to generate drone plans based on the users intent. The list below presents the categories with an example which are based on the prompt previously used for testing audio transcription. Easy inputs are commands that are short, precise, and similar to MAVLink commands. Medium inputs are commands which are sets of longer requests. Hard commands do not use MAVLink commands linguistics and contain more complex and nuanced requests that can be answered subjectively.

- **Easy:** Fly to GPS position X,Y then land.
- **Medium:** While recording fly to GPS position X,Y, then fly to X,Y, capture a photo, then return home.
- **Hard:** Fly 1000 meters north, and do a zig-zag pattern.

The evaluation process is illustrated in figure 4.2

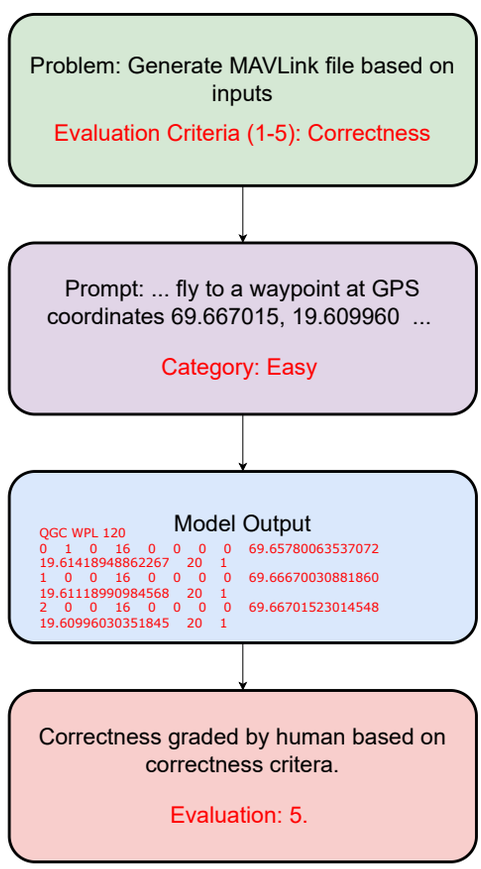


Figure 4.2: Model output evaluation process.

An example of a command of easy difficulty can for example be: Fly to GPS position at 69.662758, 19.850947, then return home. The command is inserted

into the prompt and sent to the model for chat completion. The response from the model is:

```
QGC WPL 120
0 1 0 16 0 0 0 0 69.65780063537072 19.61418948862267 20 1
1 0 0 16 0 0 0 0 69.66275800000000 19.850947000000000 20 1
2 0 0 20 0 0 0 0 0.000000000000000 0.000000000000000 20 1
```

The response shows that the file is formatted correctly according to the MAVLink standard. The file uses the commands 16 and 16 for the two starting waypoints, then uses the command 20 to return home. The first set of GPS coordinates are the starting coordinates, and the second GPS coordinates were the correct coordinates it was told to fly towards. Overall the MAVLink file is correctly formatted, uses the appropriate amounts of commands with correct parameters, and demonstrates an understanding of the user's intent. The correctness score is therefore 5/5.

The chat completion model Text-Davinci-003 is given 3 inputs from the three different categories using temperatures 0.0, 0.5, and 1.0. The test was executed to understand how the model performs using varying degrees of randomness. A command is inputted into the prompt and sent to the language model.

Temperature	Easy	Medium	Hard	Average
0.0	3.6	2.6	1	2.4
0.5	3.3	2.6	1	2.3
1.0	3.0	1.6	1	1.8

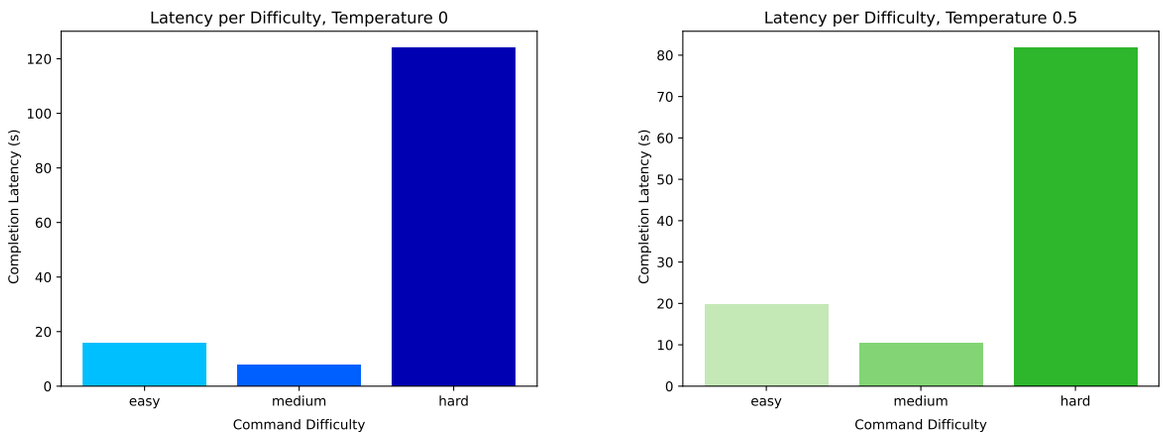
Table 4.5: Correctness results from varying degrees of temperatures.

The results from the correctness tests of varying degrees of temperature demonstrate that at temperature 0.0, the model's ability to generate drone plans decreases when increasing the difficulty of the requests. The same applies to temperatures of 0.5, and 1.0. The average score demonstrates shows that there is a 0.1 score difference between temperatures of 0.0 and 0.5. While there is a difference of 0.6 between a temperature of 0.0, and 1.0.

Latency Test

The latency was measured for each correctness test to measure the time the chat completion model takes to generate drone plans of varying difficulties.

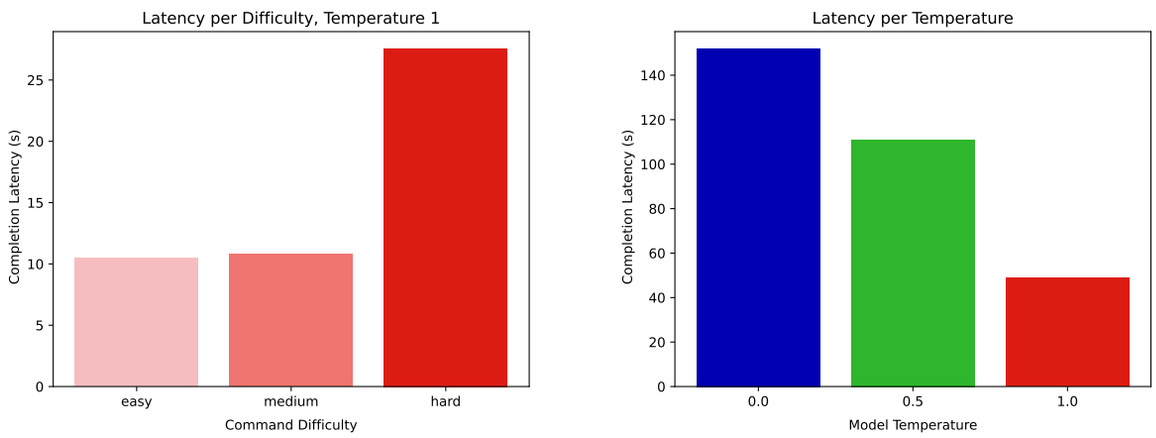
The latency of a request is defined as the time in seconds (s) it takes for a request to be sent from the mobile application until it has received a response. The latency test demonstrates the latency measurement of the chat completion model of varying temperatures when generating drone plans of easy, medium, and hard difficulty. In addition, the combined average of all tests is presented in a graph 4.4b.



(a) Average Latency per Difficulty of Model Temperature 0. **(b)** Average Latency per Difficulty of Model Temperature 0.5.

Figure 4.3: Average Latency per Command Difficulty of Model Temperature of 0.0, and 0.5

The test results from graph 4.3a demonstrate that commands with a temperature of 0.0 have a latency of 16 seconds with easy difficulty, which decreases to 8 seconds at medium difficulty and increases to 124 seconds. The test results from graph 4.3b demonstrate that commands with a temperature of 0.5 have a latency of 20 seconds with easy difficulty, which decreases to 10 seconds at medium difficulty and increases to 82 seconds.



(a) Average Latency per Difficulty of Model Temperature 1. **(b)** Average Latency per Difficulty across all Temperatures

Figure 4.4: Average Latency per Command Difficulty of Model Temperature of 1.0 and across all Temperatures

The test results from graph 4.4a demonstrate that commands with a temperature of 1.0 have a latency of 10.5 seconds with easy difficulty, which increases to 11 seconds at medium difficulty and increases to 27 seconds. The test results from graph 4.4b demonstrate the average latency per difficulty model temperature is 152 seconds for a temperature of 0.0, which decreases to 111 seconds for a model temperature of 0.5 and decreases to 48.9 seconds for a model temperature of 1.0. The overall trend is that the completion latency decreases with the increase in model temperature.

4.4 Evaluation

The experiments were conducted in order to evaluate the system's ability to translate intent into drone mission plans. This was done by measuring the correctness and latency of the system's key components. The system's key components for translating intent into drone mission plans are the voice recognition and chat completion models. Their correctness and latency were measured to evaluate the system's ability to translate intent.

4.4.1 Evaluating Intent Capture

The system's ability to capture the user's intent was measured by testing the voice recognition models' correctness and latency. This was tested in order to evaluate whether it should be used in the system. The correctness test measured the audio transcription model's ability to transcribe the user's intent correctly expressed as an audio recording. The results demonstrate that the transcription model can accurately transcribe audio recordings of short, medium, and long recordings, with an average of 95% correctness. As the audio duration increases, the model transcribes less accurately. This is to be expected as the longer the audio duration, the more difficult it is for the model to transcribe audio correctly because it increases the chance of transcribing incorrectly. The latency results demonstrate that the model uses between 2-5 seconds to transcribe audio of short, medium, and long duration. A transcription latency of 5 seconds for 1-minute audio recordings is impressive when taking transcription correctness of 95%, and the location of the servers in the US into account. In addition, a typical voice command has a duration of 10-30 seconds which the model has demonstrated it can reliably transcribe. The overall results from the intent capture tests demonstrate that the voice recognition model can accurately capture the user's intent with low latency. Based on the results it is recommended to use voice commands of low duration in order to provide the system the most accurate translation of the intent.

4.4.2 Evaluating Mission Generation

The system generates a mission based on the user's intent. This is done by inserting the intent into a prompt sent to a chat completion model. The model uses intent in the form of audio transcriptions to generate a drone mission as a MAVLink file. Measuring the chat completion model's ability to generate a MAVLink file based on intent is essential to understanding whether it is a good implementation choice. The evaluation process scores the correctness of the MAVLink file based on inputs of varying complexity. While the score is subjective, it captures crucial elements of the model's ability to generate MAVLink files, such as correct formatting, appropriate amounts of command and corresponding parameters, and the overall ability to express an understanding of the intent. The results from the correctness tests demonstrate that the model can generate drone mission plans provided commands of easy complexity but struggles with commands of medium and hard complexity. This is to be expected as the chat completion model is a general-purpose language model designed for completing text such as emails and is not trained explicitly to generate MAVLink files. The results also demonstrate that as the temperature of the

model increases, the correctness of the drone plans decreases. This is because the model the language model will choose more random and unlikely tokens when generating a response. The latency results demonstrate that the model's latency decreases with commands of easy to medium complexity, then increases dramatically with commands of hard complexity. This could be because the definition of an easy or medium complex command is different for the chat completion model than for humans. The latency test across all temperatures demonstrates that the lower the temperature, the higher the latency. This is because the chat completion models spend more time calculating the correct tokens when the temperature is lower to gain the most accurate results. The overall results from the mission generation evaluation are that the model can reliably generate drone missions with commands of easy complexity but cannot generate any drone missions with commands of increased complexity reliably. Based on the results it is recommended to use commands of low complexity in order to most accurately translate the intent.

4.5 Summary

This chapter described how the system was evaluated. This included the details of the experimental setup. It described how the system's ability to generate drone missions based on intent was tested using correctness and latency tests for audio transcription and mission generation. The results of the intent capture tests demonstrate that the voice recognition model can accurately capture the user's intent with low latency, and use voice commands of low duration in order to provide the system with the most accurate translation of the intent. The results from the mission generation evaluation are that the model can reliably generate drone missions with commands of easy complexity but cannot generate any drone missions with commands of increased complexity reliably. Based on the results it is recommended to use commands of low complexity in order to most accurately translate the intent.

5 Discussion

This chapter will discuss the findings from the thesis. The first chapter discusses the technical feasibility of the system and some design and implementation choices. The second chapter discusses the system evaluation, and the third chapter discusses the viability of the system for first responders. The final chapter discusses future work.

5.1 Technical Feasability

The first sub-problem of the thesis questions the technical feasibility of an IBR system for automatic mission creation for drones:

> Sub-problem 1: Is it technically feasible to create a prototype of the IBR system?

Based on the system implementation, it can be concluded that it is technically feasible to create a prototype of an IBR system for automatic mission creation. The system prototype provides the following functionalities: user input, communication management, mission generation, and a programmable drone. This has been implemented by recording the intent of the user using a mobile appli-

cation on a mobile device. A Rust REST API for managing mission generation communication. A Python REST API for programming the drone. A drone mission generator using a voice recognition model and a chat completion model. And an ANFAI AI as a programmable drone that can execute drone missions. The IBR system is able to capture the intent of the user and translate it to a drone mission plan in the form of a MAVLink file displayed on a screen, which can be executed on the drone.

5.1.1 User Input

A problem with using audio recordings to capture the user's intent is that the language used to dictate the drone's actions through voice commands is unclear. While it helps to use MAVLink commands, the experiments have shown that the chat completion model struggles to understand complex commands. A handbook for creating commands should be made based on the inputs from the evaluation to give the users a clear understanding of how they should generate commands.

5.1.2 Communication Management

The communication management functionality was split into two servers because one was needed to manage drone mission generation and the other to program the drone. This was done to separate the concerns of the servers, and since the mission generation functionality was implemented as a Rust Server, the other drone communication functionality needed to be implemented in Python to connect with the drone using Olympe. Another implementation strategy would have been to implement the functionality as one server in Python, or implement both functionalities in one server. OpenAI has released the openai [30] Python library that provides the same features and more as asyncopenai [5]. Ultimately the decision to use Rust and, therefore two servers was taken because Rust provides high performance and memory safety, which is essential in applications that rely heavily on performance.

Generating drone mission plans based on intent inserted into prompts is challenging because it is difficult to get concrete data on whether the potential for the chat completion model has been reached. This is because the quality of the output of the model is subjective. This thesis has attempted to provide data that demonstrate the correctness of the model output. Still, the criteria that are used to evaluate the correctness vary considerably based on the task it attempts to execute. One of the ways to test whether the prompt is successful is to create

automated tests which can be run after a new prompt has been designed. This would allow the tester to view whether the prompt is successful on various tasks efficiently. A problem with this approach is that different variations of MAVLink files lead to the same outcome and needs to be verified manually, which is time-consuming.

Ultimately, the prompt was designed by following OpenAIs guidelines on prompt design [32] and modified through trial and error. The prompt uses a static GPS starting position because Olympe does not provide a way to get the GPS position of the drone using Olympe. A solution to this would have been to use the GPS position of the mobile device. However, this would mean that the mobile device needed to be near the drone, which would limit the use cases. The prompt is then given the MAVLink format from the MAVLink websites [22] and a complete example of a MAVLink file. The prompt could have included more variations but also examples of how the chat completion model should respond to the MAVLink files from commands.

5.1.3 Mission Generation

The speech recognition model was integrated into the system using OpenAIs API because it allowed speech-to-text transcriptions to be interpreted by the language model to generate MAVLink files. The model provides a high level of accuracy and has a low execution time. It is also open-source and be implemented locally, but OpenAIs API was used instead due to hardware constraints. Google also provides Speech to Text service with a paid API [51], which can be integrated into the mobile application. An alternative to this service is that they offer asynchronous transcriptions, which can lower the time it takes to receive a transcription. An open-source alternative is react-native-voice, an open-source react-native library for transcriptions. Ultimately, Whisper was chosen because it is cutting-edge, has excellent performance, and is easy to integrate.

Text-Davinci-003 was chosen as the chat completion model because it can generate drone missions based on intent. The model is easily integrated into the application. Other alternative language models provided by OpenAI was GPT-3.5-turbo. However, it is a chatbot, and not a chat completion model. It could have been used by providing the chatbot with instructions and examples. The advantage of using it is that it can remember context, and be told what works and doesn't. In addition, it is 10x less expensive per 1k tokens. The advantage of using Text-Davinci-003 is that it can be fine-tuned, but this involves creating a dataset. Another alternative is Bard AI [55] which is a chatbot similar to ChatGPT developed by Google. It is able to generate MAVLink files, but was not

used because it is currently unavailable in Europe probably due to regulatory issues with GDPR [50].

5.1.4 Drone Programmability

At the time of writing this thesis, there are currently no commercially available 4G drones other than Parrots ANAFI AI. However, other drone engineers have created their own customized 4G drones [12]. This involves customizing already existing drones, or building one from scratch and attaching a 4G modem. In addition to 4G connectivity, the ANAFI AI is included in the Parrot SDK ecosystem, which allows developers to develop software for controlling the drone. There is also a community of other developers who can offer help and inspiration. The ecosystem provides documentation and a development forum where developers can ask for help. This has allowed the implementation of the drone into the system, allowing MAVLink commands to be uploaded to the drone.

The problem with Parrots SDK is that it provides many libraries, but lacks features and documentation documentation. In addition, the drone itself and Olympe are very unreliable with connections. It is difficult to install and set up correctly, and the overall development experience is bad. An attempt to allow 4G MAVLink uploads was unsuccessfully made due to a firmware update which caused problems for developers. A problem with the drone's hardware is that it has a low battery time (30 minutes) and therefore needs to be charged often, limiting its range. It is also limited by weather conditions, as storms or very high winds can limit its ability to fly. Another problem is that the drone is not equipped with nighttime vision, or thermal cameras, which makes it inconvenient to use at night when there is not much daylight.

5.2 System Performance

The second sub-problem of the thesis questions the performance of the system, and whether it can reliably generate missions:

> Sub-problem 2: What is the system performance, and can the system reliably generate missions?

The experiments were conducted in order to evaluate the system's ability to

translate intent into drone mission plans. This was done by measuring the correctness and latency of the system's key components. The results from the system evaluation demonstrate that the system is able to capture intent reliably with an average correctness of 95% with latency between 2-5 seconds, and that the system the chat completion model can reliably generate drone missions with commands of easy complexity but cannot generate any drone missions with commands of increased complexity reliably. Based on the results it is recommended to use commands of low complexity in order to most accurately translate the intent.

5.3 System Viability

The third sub-problem of the thesis questions whether an IBR system is a viable solution for first responders during PPDR missions.

Sub-problem 3: Is an IBR system a viable solution?

The first and second sub-problems were concerned with whether a prototype of the system could be implemented and its performance, while sub-problem 3 questioned the viability of one such system. The evaluation has shown that a large language model is able to generate drone missions based on the user intent if the complexity of the command conveyed by the user is low. Otherwise, the system fails to generate the drone mission. An essential functionality of the system is the ability to inspect the drone mission plan to evaluate whether the intent has been translated correctly. This makes an IBR system a viable solution for first responders during PPDR missions because it can automate the mission planning process.

5.3.1 Potential

An IBR system exhibits promise and a large capacity for transformative impact not only in the field of PPDR, but other fields that utilize drones. Regarding PPDR, the government can systematically plan the positioning of the drones in areas with high occurrences of natural disasters such as floods or avalanches. The drones can be placed in drone docking stations where they can be recharged automatically with induction charging. The system can either be completely autonomous and automatically deploy a drone to surveil areas impacted by natural disasters, or it can be instructed using voice commands

or maps where the drones should be deployed. The decision-making process of the drone does not need to use a general language but can be trained to generate drone missions based on natural language specifically. This can allow the drone to recommend routes, optimize routes, and also coordinate with other drones in order to improve coverage and efficiency. While acknowledging the system's potential, it is important to recognize that its actualization is still a long way off. The system can generate MAVLink files from commands of low complexity but struggles with more complex commands.

5G cellular technology has the potential to revolutionize PPDR systems by allowing drones to be used by providing low latency and high bandwidth. Network slicing enables the allocation of dedicated network slices for drones, which ensures reliable and stable connectivity. Edge computing can be utilized to bring computing closer to the location of the drone. This can be done dynamically based on the drone's position. An example is if a drone has to fly large distances, and the computation can be moved based on its position. Another benefit of 5G technology's performance is that the performance enables BVLOS flights. Drones can be operated with precise control and navigation from completely different locations, like a centralized operations room for disasters. This can allow the drones to venture into areas which it is normally incapable of reaching.

5.3.2 Ethical Concerns

One of the biggest problems with large language models is hallucination. The language model will provide information confidently even though the information is not present in the training data. An example of this is when the model is asked to provide sources to authors that do not exist. This mistake is easily discovered by searching for them. However, this is a big problem when the answer is not easily found. OpenAI mitigates this in ChatGPT with moderation. The chatbot will inform the user that it might be unaware of a question and that it is only trained on data until September 2021. The problem with this is that GPT can still provide harmful or unsafe information. This can be used to spread misinformation or incite racism. An example of this was when a Norweigan high school student asked ChatGPT to provide examples of relevant Norwegian heroes after the year 2000, and ChatGPT provided examples containing among others the Norwegian Terrorist Anders Bering Breivik [16].

5.3.3 Privacy & Security

Privacy led to ChatGPT being banned in Italy because it broke the GDPR rules on data privacy [48]. After OpenAI changed the way it stored data, the ban was later lifted. This is concerning because it is easy to provide language models with confidential information. This is a risk to an individual's personal privacy, but even more so in the professional industry where the secrecy of business information is crucial. Any business utilizing language models should take adequate steps to prevent confidential information from being leaked into language models.

An advantage of involving a human in the decision-making process is that the responsibility of the drone mission is placed on the human. In case an accident happens; for example, if the drone crashes and causes property damage, the person who verified the MAVLink file and uploaded it to the drone is responsible. The involvement of a human in the decision-making problem makes it more secure, however this does not mean that humans cannot make mistakes.

5.4 Future Work

This thesis presents a prototype of an IBR system that provides some basic functionalities that are not completed. There are many viable approaches to expanding the system. Some of them are further discussed.

5.4.1 User Input

Once the markers have been rendered, the map screen should update the map view to render the markers. In addition, the markers should indicate what order the drone will traverse the waypoints. This could have been done by numbering the markers. Another addition is to allow the markers to be moved, allowing the operator to quickly adjust the makers if the proposed MAVLink file is almost correct but requires some change. Moving markers would dynamically change the corresponding coordinates in the MAVLink file. However, this was not in the scope of the project because there are already similar mobile applications that provide this feature such as FreeFlight of QGroundControl.

When recording a voice command, the mobile application does not indicate that the recording has started or stopped. This is fixed by changing the button's color or background to red to indicate that the recording has started. Then change

the color back to the original color to indicate that the recording has stopped. It could also display a counter showing the recording time, and should also allow the user to play it back to confirm that the audio recording was recorded as intended. Another useful feature would be to allow the user to view the transcription in the recording screen, and allow the transcription to be edited in case it was wrong.

The MAVLink screen should provide information on whether the MAVLink upload to the drone succeeded. This feature needs to be integrated with the drone. Any errors that the drone throws should be handled by Olympe, and if it can't be handled by the system it should be propagated to the user, providing them with an indication of what steps to take to solve the problem. The mobile application could contain an additional tab for the drone status. This would include displaying metadata about the drone, such as the name, serial number, and more importantly connectivity status and GPS position. Because currently, the system does not provide an indication of the connectivity, meaning that you do not know in advance if the MAVLink upload will succeed.

5.4.2 Drone Mission Generation

The servers responsible for providing drone mission generation functionality do not have any security, meaning any user can connect to the API locally. This also means HTTP requests between the client and the server are unencrypted. An improvement would have been to implement HTTPS on the server. Another feature to improve security would be to implement an accounting system. This would make it easier for moderators to view anyone doing anything wrong, but more importantly, allows the user to store their prompts, flight plans, and MAVLink files.

OpenAI supports fine-tuning of the chat completion model Text-Davinci-003 [33]. This provides the user the chance to improve their model. This can allow for higher-quality results without the use of prompting. Additionally, this provides the user with lower latency requests. Overall this can lower the token usage which lowers the cost. Because the system relies heavily on the prompt that is provided in the chat completions model it is important to test using variations of prompts in order to understand how to get the most out of the model. Research should also be put into how the commands to the drones should be spoken as it can have a great effect on the outcome of the file.

A critical functionality that was not implemented was the ability to upload drone missions using 4G. This was not done because of a firmware update on

the SkyController which broke the 4G connection functionality which would have provided the user the option to upload mission plans in remote areas and BVLOS using 4G.

5.4.3 Experiments

Some experiments that could have been conducted are an end-to-end system test and a user test. The end-to-end system test could have provided data on the performance of the entire system. This could have included testing using 4G in the field, to learn how the system interacts with the real world. A user test could have been conducted to gather data on whether the system is able to understand different users' intent.

5.5 Summary

This chapter discussed the findings from the thesis and answered the sub-problems. Based on the system implementation, it was concluded that it is technically feasible to create a prototype of an IBR system for automatic mission creation. It was concluded based on the evaluation of the performance of the system that the model can reliably generate drone missions with commands of easy complexity but cannot generate any drone missions with commands of increased complexity reliably. Based on the results it is recommended to use commands of low complexity in order to most accurately translate the intent. It was also stated that an IBR system is a viable solution for first responders during PPDR missions because it can automate the mission planning process. The future works mentioned some approaches to expanding the system such as numbering the markers on the map, adding 4G support, and conducting an end-to-end test on the system.

6

Conclusion

The research problem of the thesis was defined as:

> Research Problem: IBR systems for controlling autonomous drones using 5G technology can aid first responders during PPDR missions.

The presented design and implementation of the system prove that IBR prototypes are technically feasible. The results from the system evaluation demonstrate the potential for controlling drones autonomously using IBR systems. The ability to inspect drone mission plans to confirm whether the intent has been translated correctly makes the system a viable solution for rescue operations. Drones are used today for PPDR operations using a designated pilot. This is a potential waste of resources and could be done more efficiently with the use of autonomous drones. The advancement of 5G provides the infrastructure that allows IBR systems to control drones autonomously to aid first responders during PPDR missions. This thesis concludes that IBR systems for controlling autonomous drones using 5G technology can aid first responders during PPDR missions.

Bibliography

[1] *3GPP – The Mobile Broadband Standard*. URL: https://www.3gpp.org/.

[2] *7.6*. URL: https://developer.parrot.com/docs/olympe/index.html.

[3] *Actix*. URL: https://actix.rs/.

[4] Sultan Alain. *5G System Overview*. Aug. 2022. URL: https://www.3gpp.org/technologies/5g-system-overview.

[5] *async_openai - Rust*. URL: https://docs.rs/async-openai/latest/async%5C_openai/.

[6] *Audio*. URL: https://docs.expo.dev/versions/latest/sdk/audio/.

[7] Widodo Budiharto et al. "Mapping and 3D modelling using quadrotor drone and GIS software." In: *Journal of Big Data* 8 (Mar. 2021). DOI: 10.1186/s40537-021-00436-8.

[8] Canonical. *Ubuntu 22.04.2 LTS (Jammy Jellyfish)*. URL: https://releases.ubuntu.com/jammy/.

[9] *Civil Aviation Authority – Norway*. URL: https://luftfartstilsynet.no/en/.

[10] Jacob Devlin et al. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: 1810.04805 [cs.CL].

[11] "Docker: Accelerated, Containerized Application Development." In: *Docker* (Apr. 2023). URL: https://www.docker.com/.

[12] The Drone Dojo. *MAKE A 4G LTE RASPBERRY PI DRONE WITH LIMITLESS RANGE | The Ultimate BVLOS Guide*. Aug. 2021. URL: https://www.youtube.com/watch?v=IokyotAGbJI.

[13] Afra Domeke, Bruno Cimoli, and Idelfonso Tafur Monroy. "Integration of Network Slicing and Machine Learning into Edge Networks for Low-Latency Services in 5G and beyond Systems." In: *Applied Sciences* 12.13 (2022). ISSN: 2076-3417. DOI: 10.3390/app12136617. URL: https://www.mdpi.com/2076-3417/12/13/6617.

[14] *EUR-Lex - 32019R0947 - EN - EUR-Lex*. URL: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX%5C%3A32019R0947.

[15] *Expo*. URL: https://expo.dev/.

[16] Jonathan Falk. “ChatGPT foreslo Anders Behring Breivik som «norsk helt».” In: (Feb. 2023). URL: https://www.vg.no/nyheter/i/WRkK5K/chatgpt-foreslo-anders-behring-breivik-som-helt.

[17] Air Force. *MQ-1B Predator*. URL: https://www.af.mil/About-Us/Fact-Sheets/Display/Article/104469/mq-1b-predator/.

[18] Telecom Forum. *LTE/4G architecture and its components functionality*. May 2020. URL: https://www.youtube.com/watch?v=d-UbwYLM08E.

[19] Shivi Garg and Niyati Baliyan. “Comparative analysis of Android and iOS from security viewpoint.” In: *Computer Science Review* 40 (2021), p. 100372. ISSN: 1574-0137. DOI: https://doi.org/10.1016/j.cosrev.2021.100372. URL: https://www.sciencedirect.com/science/article/pii/S1574013721000125.

[20] Abdul Hafeez et al. “Implementation of drone technology for farm monitoring & pesticide spraying: A review.” In: *Information Processing in Agriculture* 10.2 (2023), pp. 192–203. ISSN: 2214-3173. DOI: https://doi.org/10.1016/j.inpa.2022.02.002. URL: https://www.sciencedirect.com/science/article/pii/S2214317322000087.

[21] *Introduction - 7.6*. URL: https://developer.parrot.com/docs/airsdk/general/overview.html.

[22] *Introduction · MAVLink Developer Guide*. URL: https://mavlink.io/en/.

[23] *JavaScript With Syntax For Types*. URL: https://www.typescriptlang.org/.

[24] Takeshi Kojima et al. “Large Language Models are Zero-Shot Reasoners.” In: *ArXiv* abs/2205.11916 (2022).

[25] *MapView*. URL: https://docs.expo.dev/versions/latest/sdk/map-view/.

[26] *Mel*. URL: https://www.sfu.ca/sonic-studio-webdav/handbook/Mel.html.

[27] Kara Murphy. “Parrot’s new Anafi AI drone features 4G connectivity and an insect-inspired design.” In: (June 2021). URL: https://www.dpreview.com/news/9329060411/parrot-new-anafi-ai-drone-features-4g-connectivity-and-an-insect-inspired-design.

[28] Alexander Nordby and Geir Randby. “Norsk mikrodrone er blitt milliardsuksess.” In: *NRK* (Mar. 2019). URL: https://www.nrk.no/innlandet/norsk-mikrodrone-er-blitt-milliardsuksess-1.14472536.

[29] *Open5GCore*. URL: https://www.open5gcore.org/open5gcore/fundamental-core-functionality.

[30] *openai*. May 2023. URL: https://pypi.org/project/openai/.

[31] *OpenAI API*. URL: https://platform.openai.com/overview.

[32] *OpenAI API*. URL: https://platform.openai.com/docs/guides/completion/introduction.

[33] *OpenAI API*. URL: https://platform.openai.com/docs/guides/fine-tuning.

[34] *Overview - 7.6*. URL: https://developer.parrot.com/docs/groundsdk-android/overview.html.

[35] Parrot. *About Parrot | Parrot*. URL: https://www.parrot.com/en/about-parrot.

[36] Parrot. *Open Source Drone Software | Parrot drones*. URL: https://www.parrot.com/us/open-source-drone-software.

[37] Parrot. *Parrot ANAFI Ai | The 4G robotic UAV | Technical documentation*. URL: https://www.parrot.com/us/drones/anafi-ai/technical-documentation/.

[38] *Parrot unveils the first 4G connected robotic UAV - Modern Power Systems*. URL: https://www.modernpowersystems.com/news/newsparrot-unveils-the-first-4g-connected-robotic-uav-8875782.

[39] Parrot-Developers. *GitHub - Parrot-Developers/openflight-ios: OpenFlight is the open-source core of FreeFlight7*. URL: https://github.com/Parrot-Developers/openflight-ios.

[40] STL Partners. *5G Standalone vs Non-standalone: deployment models - STL Partners*. Nov. 2022. URL: https://stlpartners.com/articles/telco-cloud/5g-deployment-models-standalone-vs-non-standalone/.

[41] *Phone 11 Pro - Tekniske spesifikasjoner (NO)*. URL: https://support.apple.com/kb/SP805?locale=no_NO.

[42] Marzena Polka, Szymon Ptak, and Łukasz Kuziora. "The Use of UAV's for Search and Rescue Operations." In: *Procedia Engineering* 192 (Dec. 2017), pp. 748–752. DOI: 10.1016/j.proeng.2017.06.129.

[43] *Pricing*. URL: https://openai.com/pricing.

[44] *Production-Grade Container Orchestration*. URL: https://kubernetes.io/.

[45] *QGC - QGroundControl - Drone Control*. Feb. 2020. URL: http://qgroundcontrol.com/.

[46] Alec Radford et al. *Robust Speech Recognition via Large-Scale Weak Supervision*. 2022. arXiv: 2212.04356 [eess.AS].

[47] *React Native · Learn once, write anywhere*. URL: https://reactnative.dev/.

[48] Adi Robertson. "ChatGPT returns to Italy after ban." In: (Apr. 2023). URL: https://www.theverge.com/2023/4/28/23702883/chatgpt-italy-ban-lifted-gpdp-data-protection-age-verification.

[49] *Rust Programming Language*. URL: https://www.rust-lang.org/.

[50] Ben Schoon. “Google Bard isntt available in any European Union countries and Canada.” In: (May 2023). URL: https://9to5google.com/2023/05/11/google-bard-european-union/.

[51] *Speech-to-Text: Automatic Speech Recognition | Google Cloud*. URL: https://cloud.google.com/speech-to-text.

[52] App Store. *Expo Go*. Aug. 2015. URL: https://apps.apple.com/us/app/expo-go/id982107779.

[53] App Store. *FreeFlight*. May 2010. URL: https://apps.apple.com/us/app/freeflight/id373065271.

[54] Vinod Thomas. *Global Increase in Climate-Related Disasters*. Nov. 2015. URL: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2709331.

[55] *Try Bard, an AI experiment by Google*. URL: https://bard.google.com/?hl=en.

[56] *Unreal Engine | The most powerful real-time 3D creation tool*. URL: https://www.unrealengine.com/en-US.

[57] Ashish Vaswani et al. “Attention Is All You Need.” In: *CoRR* abs/1706.03762 (2017). arXiv: 1706.03762. URL: http://arxiv.org/abs/1706.03762.

[58] *Welcome to Flask — Flask Documentation (2.3.x)*. URL: https://flask.palletsprojects.com/en/2.3.x/.

[59] *Welcome to Python.org*. May 2023. URL: https://www.python.org/.

[60] *What is Parrot Sphinx - 2.11*. URL: https://developer.parrot.com/docs/sphinx/index.html.