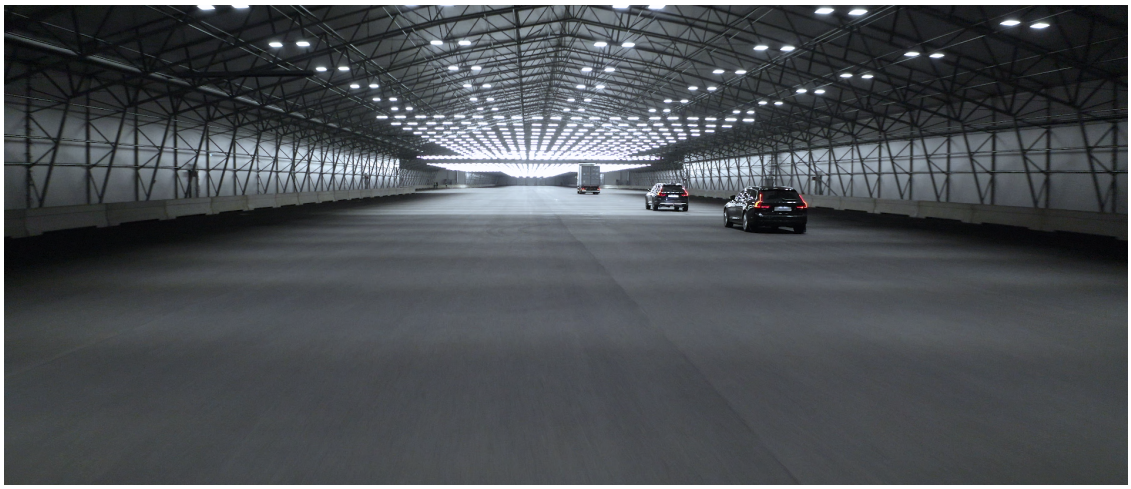
CHALMERS



Automate the video documentation of a Euro NCAP test by use of drones

EENX16-23-26

Bachelor's thesis in System Control

Johan Gustafsson
Christoffer Oresten
Teddy Sallén
Mårten Sanderöd
Leonard Smedenman
Axel Toresson

DEPARTMENT OF ELECTRICAL ENGINEERING

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2023
www.chalmers.se

Bachelor's thesis 2023

Automate the video documentation of a Euro NCAP test by use of drones

Johan Gustafsson
Christoffer Oresten
Teddy Sallén
Mårten Sanderöd
Leonard Smedenman
Axel Toresson

CHALMERS

Department of Electrical Engineering
Division of System Control
EENX16-23-26
Chalmers University of Technology
Gothenburg, Sweden 2023

Automate the video documentation of a Euro NCAP test by use of drones

Supervisor: Martin Fabian, Automation
Examiner: Knut Åkesson, Automation

Bachelor's Thesis 2023
Department of Electrical Engineering
Division of System Control
EENX16-23-26
Chalmers University of Technology
SE-412 96 Gothenburg

Cover: AstaZero Dry-Zone. From [1]. Reproduced with permission.

Typeset in LaTeX
Gothenburg, Sweden 2023 Automate video documentation of Euro NCAP tests by use of drones
EENX16-23-26
Department of Electrical Engineering
Chalmers University of Technology

Abstract

The objective of this project was to create an automated system for video documentation of Euro NCAP tests utilizing a camera-equipped drone. Specifically, the intention was to showcase the potential of using an autonomous drone to capture video footage of a car, or another vehicle undergoing a Euro NCAP test on the AstaZero testing grounds. The report details how an autonomous drone can be used to film these tests safely and effectively.

To achieve this objective, an Android drone application was developed which governs the drone's movement. The aim was to incorporate an object tracking algorithm, enabling the cameras on the drones to maintain direct focus on the vehicle being tested. This methodology aims to fortify the system's durability and ensure the camera records high-quality footage throughout the entire testing process. The tracking algorithm was however not finished on time resulting in the application only incorporating the object detection part.

The report also describes the challenges associated with using autonomous drones for filming, such as working with predetermined paths, staying connected to a control center and pulling live video feed to process it. To address these problems, several tests and adjustments were made to the drone's software.

Overall, this project represents an important stride forward in the automation of Euro NCAP tests and has the potential to significantly improve the efficiency and accuracy of these tests in the future.

Keywords: Autonomous drone, video documentation, Euro NCAP test, AstaZero, ATOS

Sammanfattning

Syftet med detta projekt var att skapa ett automatiserat system för videodokumentation av Euro NCAP-tester med hjälp av en kamerautrustad drönare. Mer specifikt är avsikten att utveckla en demonstration som visar en autonom drönares förmåga att följa och filma en bil eller ett annat fordon som genomgår ett Euro NCAP-test på företaget AstaZeros testområde. Genom rapporten beskrivs det hur en autonom drönare kan användas för att filma dessa tester på ett säkert och effektivt sätt.

För att uppnå detta mål utvecklades en Android-drönarapplikation som styr drönarens förflyttning, samt en objektföljningsalgoritm, vilket skulle möjliggöra att kameran på drönaren kan bibehålla fokus på fordonet som testas. Denna metod syftar till att stärka systemets robusthet och säkerställa att kameran fångar högkvalitativt videomaterial under hela testprocessen. Algoritmen färdigställdes dock inte i tid, applikationen innehåller därför endast objektigenkänning.

Rapporten beskriver också utmaningarna med att använda autonoma drönare för att filma, såsom att arbeta med förutbestämda rutter, bibehålla anslutning till ett kontrollcenter och bearbeta kameraströmmen. För att lösa dessa problem genomfördes flera tester och justeringar av drönaren och dess programvara.

Sammanfattningsvis är projektet ett viktigt steg framåt inom automatiseringen av Euro NCAP-tester och har potentialen att avsevärt förbättra effektiviteten och noggrannheten hos dessa tester i framtiden.

Nyckelord: Autonom drönare, videodokumentering, Euro NCAP test, AstaZero, ATOS

Acknowledgements

The group would like to express our gratitude to everyone who has contributed to the completion of this project. Firstly, we would like to thank our contact person at AstaZero, Victor Jarlow. Without his assistance and guidance this project would likely not be as successful.

Secondly we would like to extend our gratitude to the team from Penn State University, namely Daniel Dean, Brian Simons, Jamin Valick, Xinxheng Zhang, and Mohammed Alawadhi, for their valuable contributions. We feel privileged to have had the opportunity to learn from and collaborate with such exceptional students. Thirdly, we wish to convey our sincere appreciation to the supervisors of the Penn State University team, namely Nick Vlajic and Jason Moore, for their guidance and insights throughout the course of this project.

We would also like to extend our appreciation to our supervisor Martin Fabian for the helpful input he has provided during the project.

Furthermore we would like to thank Mikael Enelund for constantly being open for discussion, and continuously being most helpful regarding all the different aspects of conducting a project within a large group.

Thank you for all your contributions, sincerely

-EENX16-23-26

List of Acronyms

Below is a list of the acronyms used throughout this thesis:

API	Application Programming Interface
ATOS	Autonomous vehicle Testing Operating System
AZ	AstaZero
CNN	Convolutional Neural Network
DJI	Da-Jiang Innovations
EASA	European Union Aviation Safety Agency
GUI	Graphical User Interface
IDE	Integrated Development Environment
NCAP	New Car Assessment Programme
NPV	Net Present Value
PSU	Pennsylvania State University
RC	Remote Controller
RGB	Red Green Blue
RGBA	Red Green Blue Alpha
RISE	Research Institutes of Sweden
SDK	Software Development Kit
TDD	Test Driven Development
UAV	Unmanned Aerial Vehicle
XML	Extensible Markup Language

Contents

List of Figures

List of Tables

1
Introduction

With car safety and autonomous driving being ever more prominent in our society, as well as ambitious visions of zero car fatalities by leading car manufacturers such as Volvo Cars [2], there is a growing demand for data to support these efforts. Extensive testing and research are crucial for realizing these goals. A research facility for tests like these is located just outside of Borås, owned by Research Institutes of Sweden, and operated by AstaZero. Assessments of this nature are collectively known as the New Car Assessment Programme, with the European division simply being called Euro NCAP [3]. In addition to rating the safety performance of different vehicles, these tests also provide essential data for further development of car safety programs. To utilize the data to its maximum extent, it is essential to be able to revisit the tests, which requires accurate and reliable footage of the various test scenarios. An accurate depiction of the test and how the various vehicles compare are of essence. This is achieved when the tests are conducted in a consistent and standardized manner, implying that dynamic and autonomous footage capturing is applicable. This Bachelor thesis at Chalmers University of Technology will in joint venture with students from Penn State university develop software that enables automated video documentation of NCAP tests by the use of drones.

1.1 European New Car Assessment Programme

The European New Car Assessment Programme is a comprehensive initiative that evaluates the safety of vehicles that are about to be introduced to the market [3]. These tests are carried out globally in order to ensure accurate and unbiased assessments for the car being tested, leading to increased demand for safer cars. The vehicles are assigned a safety rating on a scale from 0-5 stars, zero stars implying that the vehicle is virtually unsafe. NCAP-tests assess the safety in different traffic scenarios, including safety for the driver, passengers, fellow road users and pedestrians. The tests were initially conducted merely in crash scenarios, but ever since the introduction of smarter cars the tests are also performed for crash avoidance and auto-pilot assist, with plans for conducting tests for fully automated vehicles in the coming decade [4].

To ensure the reliability and accuracy of the test results there is a need for documentation of the tests being conducted, including footage of the various scenarios. The footage of today is mostly static, either by use of stationary cameras or drones following a predetermined trajectory [5].

1.2 Objective

The Bachelor's thesis constituted a collaborative effort between students from Chalmers University of Technology in Gothenburg, and Pennsylvania State University in State college, PA. The joint venture aimed at creating automated video documentation of a Euro NCAP test, utilizing drones. The project was conducted on behalf of AstaZero, a subsidiary of Research institutes of Sweden (RISE). The final objective of the project was to deliver an application that, based on information pertaining to the moving vehicle, receives a pre-calculated path, which the drone follows while simultaneously recording the vehicle undergoing the Euro NCAP test. The position as well as the angle of the drones were in accordance with the Euro NCAP film and photo protocol, which ensured valid documentation that conformed to already established standards [5]. Furthermore, the mission provider expressed in the project description a need for conducting a cost-benefit analysis.

1.3 Problem Statement

AstaZero was at the time of writing this report, using an in-house developed program called ATOS, which is an abbreviation for Autonomous Vehicle Testing Operating System [6]. This program allows for starting and stopping of the vehicles that have been assigned predetermined paths. These paths have been created manually and the vehicles remain in their starting positions until ATOS triggers the execution command and the cars start moving. However, this approach requires continuous manual calculation of new paths for the drones whenever a new scenario is tested, which can be tedious and time consuming. The objective was therefore to develop an automated and adaptive solution for documentation of ongoing tests, which ultimately enabled Euro NCAP tests to be conducted at various locations with different scenarios that can be switched with ease and flexibility.

Euro NCAP provides detailed documentation of these tests. The tests are divided into different categories and the team's work will be used for tests regarding lane support systems [7], autonomous emergency breaking in car-to-car systems [8] and autonomous emergency breaking and lane support systems involving vulnerable road users [9]. Documentation of these tests follows certain rules and protocols [5]. These protocols are used to get a good understanding of the tests and which of them are a good fit for autonomous drone documentation. Some relevant parameters for determining this are speed, angles and number of cars present in the test. A suitable test for evaluating the functionality of the developed application, conducted at low speeds where the test car can be easily distinguished from other vehicles, can be found within these protocols.

AstaZero, the client, also had wishes for the project:

1. Design the solution in a way that is energy-efficient.

2. The system that the team is tasked with developing should be simple and easy to configure and use.
3. The system should support conducting tests on various sites.
4. Add a "dry run" function, i.e allow the actual test to run without the drone.
5. Run the real test at AstaZero's test facility.
6. Conduct a cost-benefit analysis of the utilization of drones for video documentation.

Given that the ultimate objective entails conducting a EuroNCAP test, it is imperative that these distinct functions and implementations are tested in a live setting, with the exception of the cost benefit analysis. The primary aim was to subject all functions to rigorous testing at the dedicated test track owned by AstaZero, thereby enabling the execution of a genuine test scenario.

1.4 Demarcation

Given that the framework for this project was established prior to the writing of this report, the only demarcations for the project were according to the project description as well as both hardware and software limitations. However, further demarcations were established in order to clarify the project parameters.

The client, AstaZero, expressed a wish that a charging/docking station be developed by the Mechanical engineering students, as the programming part may prove too challenging early on in the project. However, in the beginning of the project both the Chalmers and Penn-State groups concluded that the accumulated programming skills would be sufficient. The original, highly software-based task therefore remained the primary focus throughout the project. Both groups assessed the feasibility of also constructing a charging station for the drone, but as the project continued, the time frame was not sufficient and therefore the idea was discarded.

A drone of type DJI Mavic 2 Enterprise [10] was provided by the client AstaZero. Accordingly, the project was conducted based on the capabilities and limitations of the hardware, both in terms of flying and video capturing.

The challenge of developing an application, programming in Java, and communicating with drones was compounded by limited knowledge and little prior experience. This meant that more time was required for research, learning, and troubleshooting. These factors ultimately lead to less progress being made compared to working with extensive experience and knowledge.

Due to the limited experience in the subject and the restricted time frame, the focus was primarily to maximize the efficiency and proficiency of one single assessment from the Euro NCAP test protocol.

The drones of type DJI Mavic 2 Enterprise have limited battery and will only be able to film for about 15 minutes before they need to recharge. This presented a challenge since a test can be longer than 15 minutes if reruns are needed. Due to

the limited time-frame it was decided that instead of creating a solution for charging the drone automatically, the officials on the test track would just switch to another drone or new batteries.

The drone in question only has one camera and uses different sensors in collaboration with built in software to avoid obstacles. For high speeds tests, the built-in sports-mode could not be utilized due to it clashing with other built-functions like obstacle avoidance, hence the built in sports-mode was not used.

Furthermore, AstaZero provided an Android device that was used for the drone controller. Therefore, the application development was limited to Android devices.

1.5 Project outline

The main objective was to successfully utilize a DJI Mavic 2 Enterprise Zoom to capture a demonstration of an autonomous emergency braking test with vulnerable road users [9], as illustrated in Fig. 1.1.

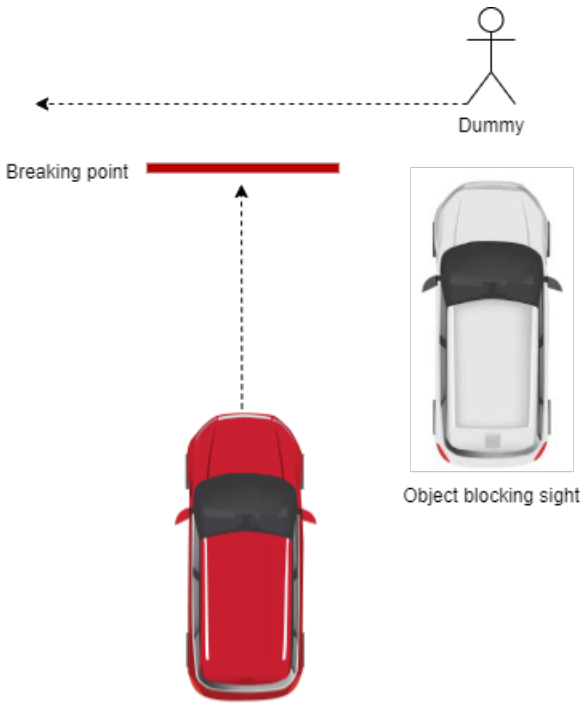


Figure 1.1: Illustration of the test, dummy walking in to the cars trajectory, red car is expected to emergency brake [5]. Source: Primary

To guarantee proper execution of the test, it was crucial to adhere precisely to the established protocol [5]:

- At the start of filming the car and the dummy representing a human should be in view with the dummy moving towards the camera. The camera should be positioned at an angle of approximately 45 degrees to the dummy motion.

When the car is about to brake, the view should start to zoom in until the car has stopped.
- For at least half of the runs required under media films, a drone must be used in replacement of this view. Start filming following the car. When the car approaches the target, both the car and the target should be in view. The angle should be as perpendicular as possible to the car and target in order to show the action.

To achieve the goal of this project, several sub-tasks were identified:

- Given the path of the object to be filmed, calculate a trajectory for the drone to document the test as outlined in [5].
- Transfer the calculated trajectories using an existing protocol.
- Develop a mobile application capable of receiving the calculated trajectories.
- Parse the trajectories to formats accepted by the drone.
- Execute the test and fly with the given trajectories.
- Use the video stream from the drone to identify and track selected object, keeping it in the center of the frame at all times.
- Develop a graphical user interface (GUI) to display information about the drone, the video feed and its current status.

2
Background

In this chapter, the foundation on which the implementation and results are built upon are presented. The chapter begins with an overview of the drone hardware, followed by an explanation of the software. This includes the drone's flight control, communication with ATOS, object detection and tracking as well as limitations pertaining to every component.

2.1 Hardware

In this section, the hardware used will be presented which is a drone, a mobile device and the test objects.

2.1.1 DJI Mavic 2 Enterprise Zoom

The unmanned aerial vehicle (UAV) that was used in this project was the DJI Mavic 2 Enterprise, developed by the Chinese technology company DJI, seen in Fig. 2.1.

Figure 2.1: Picture of the DJI Mavic 2 Enterprise. Source [11]. CC-BY 2.0

The Mavic 2 Enterprise is equipped with a 3-axis gimbal-stabilized camera that can capture high-quality 4K video and 12MP photos [10]. The gimbal is able to be controlled by both software and manual inputs to the remote controller. It also features omnidirectional obstacle sensing and avoidance with sensors positioned on the front, back, bottom, and sides, as well as a GPS and magnetometer sensor, which allows the drone to maintain a stable hovering position.

The Mavic 2 Enterprise has a maximum flight time of up to 31 minutes and a range of up to 10 km [10], making it a versatile and effective tool for various applications.

Yet, drone laws and regulations vary by country and region, and certain areas may require permits to fly beyond visual line of sight or certain distances. Therefore, strict adherence to local regulations and safety protocols was of paramount importance when operating the Mavic 2 Enterprise or any other UAV. More on the rules and regulations of drone flight can be found in Sec. 6.3.

The drone came with a remote controller (RC), which could be connected to a compatible smartphone with a designated application via a USB-cable, as seen in Fig. 2.2. This is required if a live feed from the drone's camera is desired. DJI offers its own application that offers drone operators a wide array of modes, settings, and options to customize and monitor the live feed from the drone. This application can be downloaded from either the App Store or Google Play. However, the group had to establish a connection with the existing in-house software ATOS, outlined in Sec. 2.2.1 to execute the tests. Since the ATOS software is not used outside of AstaZero, DJI offered no easy integration with ATOS. The scope of the project was therefore much more specific compared to what the demo application offers and accordingly, a new application needed to be developed.

Figure 2.2: Picture of the mobile device connected to the RC via an USB cable (highlighted in red). Source: Primary

2.1.2 Samsung A13

The Samsung Galaxy A13 is the mobile device developed by Samsung. The device comes equipped with the Exynos 850 chipset, which is a mobile processor that provides great computing capabilities [12] on mobile devices as well as facilitating fast

mobile internet. It is compatible with networks ranging from 2G to LTE with downlink speeds of up to 300Mbps and uplink speeds of up to 150Mbps. The model of the A13 device which was provided by AstaZero has the Android 12 operating system and comes with 64GB of internal storage, 4GB of RAM and a 5,000mAh battery.

In summary, the Samsung A13 mobile device is a suitable choice for the current application when not performing any other tasks in parallel. It is a reliable and cost-effective device with more than sufficient capabilities, whilst running the correct operating system for the project. See Fig. 2.3 for an external view of the A13 device.

Figure 2.3: Picture of the Samsung A13. Source: Primary

2.2 Software

This section will provide an overview of the software utilized in this project. The key components; ATOS, Linux and Ubuntu 20.04 as the operating system, along with essential applications such as Java, Android SDKs, and Android Studio. Additionally, the project incorporates DJI Waypointmission, AstaZero's application, object detection, a tracking algorithm, and a Graphical User Interface (GUI).

2.2.1 Autonomous Vehicle testing operating system (ATOS)

The core of each test is ATOS, short for Automatic Vehicle Testing Operating System, which follows the standards stated in the ISO22133 documentation [13] and creates trajectories for each object that is part of the test. An extensive comprehension of the ATOS software needed to be obtained, which included the functional requirements, software specifications and the communication protocol. The information can be obtained from the ISO22133 protocol as well as in the actual codebase [6]. ATOS is mainly written in C and C++ and developed on a Linux machine. The integration of ATOS in this project is illustrated in Fig. 2.4.

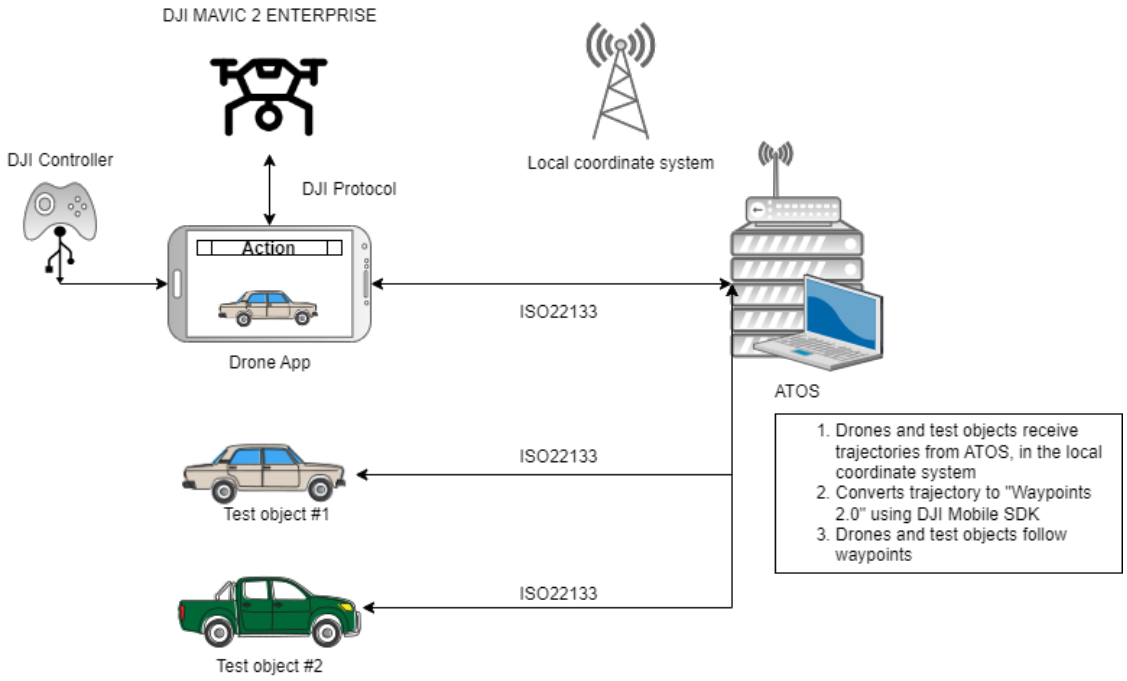


Figure 2.4: ATOS integration with other objects. Source: Primary

Fig. 2.5 displays the various states and transitions that ATOS sends to all objects during a test, each representing a specific action. In the “Init” state, each object is initialized but not connected to ATOS. When ATOS establishes a successful connection, the connected objects moves to the “Disarmed” state, meaning it is ready to receive telemetry data and begin mission planning. Once ATOS transitions to “Armed”, the objects receives the telemetry data and is set to execute the test. When “Armed”, ATOS can start the test and the state will transition to “Running”, which means that all objects follow their predetermined trajectories until each object reaches its endpoint. ATOS then moves to the “Normal Stop” state, indicating that the test has concluded. From ”Normal Stop,” the objects can transition back to the “Disarmed” state to wait on a new trajectory. Additionally, the “Disarmed” state can be reached from the “Emergency Stop” state, which acts as a kill switch to abort the test.

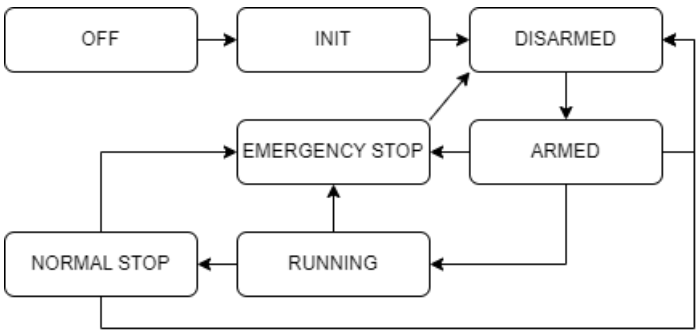


Figure 2.5: Illustration of relevant object states and their transitions. Source: Primary

Note that some states from the ISO22133 protocol are not shown in the figure, as they are not relevant to this project. One such state is the "Remote Ctl" state, which allows the operator to move the object freely without relying on predetermined telemetry, while still maintaining a connection to the control center.

2.2.2 AstaZero's application

AstaZero provided an application that they had begun working on [14]. The application included a DJI demo which was copied from DJI's publicly available demo applications [15]. This application provided the basic functionality to connect and communicate with the drone, as well as two different components. These two components utilized a map to place waypoints and fly the mission.

2.2.3 ISO DTS 22133 Communications Protocol

To conduct a thorough assessment of collision avoidance systems and perform various other tests, it was essential to carry out testing on specialized proving grounds. These proving grounds provided a controlled environment that ensured the safety of both the driver and other motorists.

In order to create realistic traffic scenarios during testing, the proving grounds had to utilize various targets that represented both vehicles and pedestrians. These targets helped simulate different scenarios that could occur on the road, enabling a comprehensive evaluation of the collision avoidance systems.

The ISO DTS 22133 standard [13] describes requirements, functionality, and protocol necessary for controlling multi target-carrier systems to create and monitor realistic traffic scenarios in a safe and effective manner. The ISO22133 is a standard designed to work with multiple manufacturers and different test objects. Each object holds a state that is controlled by ATOS, as described in Sec. 2.2.1, and the protocol is used to communicate this to all relevant objects. It also facilitates the communication of position, speed and heading from ATOS to the test object in question.

2.2.4 Linux and Ubuntu 20.04

As stated in Sec. 2.2.1, ATOS was developed using a Linux operating system, specifically Ubuntu 20.04. Linux and Windows have some fundamental differences making it impossible to compile the necessary projects aimed at Linux operating systems on Windows. Both the ATOS project and the Android application provided by AstaZero have dependencies on other Linux programs which is why an old version of the Ubuntu operating system was required.

2.2.5 Java, Android, SDKs and Android Studio

The application was built on top of one of the major mobile operating systems named Android. Android provides a framework for communicating with any mobile

device running the Android system making the development not tied to one specific device, but rather any device capable of running Android.

All Android applications are constructed using activities, which serve as the fundamental elements of the application's user interface. Activities act as individual screens or windows that facilitate user interaction within the application [16]. Each activity functions as an independent component, possessing its own life cycle. It can be initiated, halted, and terminated autonomously from other activities within the application. Activities also have the capability to communicate with one another, allowing for the sharing of information among all components of an application. Additionally, based on the provided information, new activities can be created when required.

For instance, within the confines of this project, one activity could be the first page displayed, e.g. the landing page, while another activity could be the interface that displays drone information. Together these different activities make up the application and communicate with each other.

The recommended option to develop Android applications is Android's own integrated development environment (IDE) called Android Studio. The program provides all the necessary tools for developing Android applications in Java. It also has support for designing user interfaces in both a manual way using the markup language XML or a via drag-and-drop functionality.

Android enables developers to use outside libraries, also called Software Development Kits (SDK). These external libraries allows the developer to use code written by other developers and use their frameworks. DJI provides their own SDK for communicating with their different products.

To access the functionalities of the DJI Mavic 2 Enterprise drone, a mobile application must be connected to the drone's remote controller (RC). DJI's framework for Android devices enables users to access functions such as reading drone states, initiating or terminating video recording, obtaining a live feed from the drone's camera, land at a given point and starting or stopping the motors. DJI's website [15] contains tutorials on how to implement various types of functions for the application. This has served as a foundation for learning how to communicate with the drone. Moreover, GitHub hosts a diverse range of projects of varying kinds that have been explored to acquire knowledge of the DJI Mobile SDK API.

2.2.6 DJI WaypointMission

The drone is able to fly on its own with different protocols provided by the DJI SDK. One of these protocols is the WaypointMission [17], which enables the developer to create a list of geographical locations to which the drone will fly. The developer can control the speed, heading, latitude, and longitude that the drone should have at each waypoint. It also offers a callback for different events such as mission start,

waypoint reached, and mission uploaded. This can be used to perform different actions based on the state of the mission.

2.2.7 Object detection

In order to ensure that the camera captures the vehicles at the optimal angle, it was imperative for the drones to accurately detect the location of the cars as the drones followed them.

Object detection is a rapidly developing field of computer vision that has gained significant attention in recent years. At its core, object detection involves the automatic identification and classification of objects within digital images. This is accomplished through the use of sophisticated algorithms that are designed to recognize specific patterns and features within the image data.

According to Myeongsuk and Sanghoon [18], the process of image recognition is typically achieved using deep learning algorithms, such as convolutional neural networks (CNNs). These networks consist of several layers, including convolutional, pooling, and fully connected layers, which work together to extract and analyze features from the input image. The Single-shot Multibox Detection model [19] that TensorFlow Lite offers [20] does classification using convolutional layers instead of fully connected ones.

2.2.7.1 TensorFlow Lite

TensorFlow Lite is a specialized library designed specifically for mobile applications, offering pre-trained models for various scenarios, including object detection [20]. Given a mobile device's limited processing power compared to a traditional computer, TensorFlow Lite models are compressed versions of their normal TensorFlow counterparts, possessing less complexity. Thus, a TensorFlow Lite model trades a small amount of accuracy for significantly improved computational speed compared to a normal TensorFlow model, making it well-suited to our goal of running such a model on a mobile device.

2.2.7.2 Convolutional layers in CNNs

Convolutional layers are typically the first few layers in a CNN [21], meaning they are the first to receive the input image. When images are used as input, they are typically treated as matrices of pixel values. Pixels in grayscale images can be represented by a value between 0 and 1, where 0 is black, and 1 is white. The image is subsequently transformed into a numerical representation to which mathematical operations can be applied. In CNNs, said operation is the dot product between the filter and the grid of pixel values the filter applies to. The result of the dot product replaces the pixel value in the feature map produced by the CNN.

CNNs have a parameter called stride, which determines how many pixels the filter is moved by after it has been applied to a pixel. Given a stride of 1, the filter

will be applied to each pixel, with the filter first being applied to every pixel in the top row, followed by every pixel in the second top row. Given a stride of 2, the filter will be applied to every other pixel in the first row, then every other pixel in the third. The height and width of the output, a feature map, are thereby halved.

Pixels in color images can not be represented by a single value like pixels in grayscale images; they can, however, be represented by three values, the level of red, green, and blue present in the pixel. The image is therefore turned into three $w \cdot h$ matrices where w is the width, and h is the height of the image in pixels. The values in the first matrix indicate the level of red present in each pixel of the image, the values of the second matrix indicate the level of green present in each pixel, and the values in the third matrix indicate the level of blue present in each image. A three-dimensional filter can be used to apply a filter to the whole image, meaning all three matrices.

The outputted feature map of a CNN highlights a specific feature. In Fig. 2.6, diagonal features in the shown direction are highlighted in the resulting feature map. To highlight more complex patterns, feature maps can be used as inputs to other convolutional layers. Consecutive convolutional layers are usually used in CNN:s.

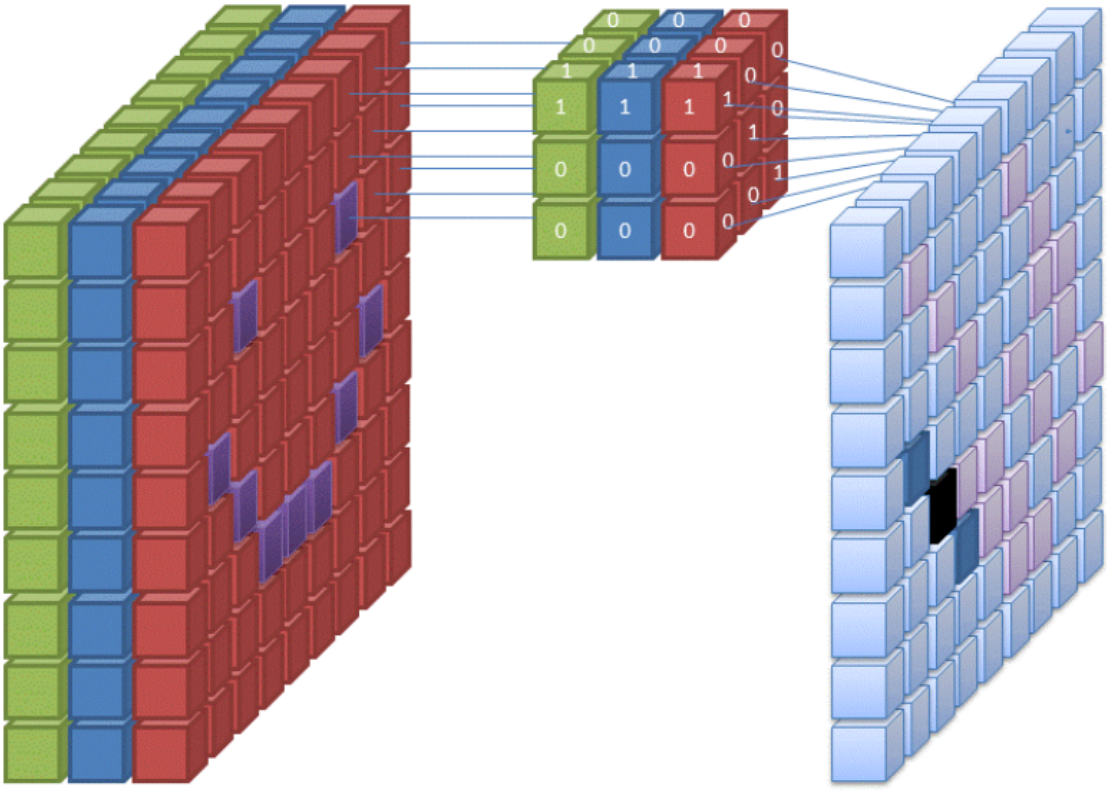


Figure 2.6: A convolutional layer is applied to an image, resulting in a feature map. Source: [22]. CC-BY SA 4.0

2.2.7.3 Pooling layers in CNNs

Various pooling layers exist, including max and average pooling layers, which operate similarly but employ distinct functions for selecting representative values [21]. Max pooling is a layer in CNNs often used in combination with convolutional layers. A max-pooling layer reduces the size of an image or matrix by splitting it into grids and representing each grid by the largest value in that grid. The features highlighted by previous convolutional layers are preserved by keeping the highest values of a grid. Since the image size gets reduced by the max pooling layer and certain features are highlighted, patterns of varying sizes can be detected. Filters applied to the image in its original resolution can detect local, smaller features, and filters applied to the image after a couple of max pooling layers can detect more global features since more of the image fits in the filter. Reducing the resolution of an image also boosts computing efficiency.

2.2.7.4 Single-shot Multibox Detection

A popular model used for object detection is Single-shot Multibox Detection (SSD) [19]. An SSD is a type of CNN used for object detection. The initial part of the SSD architecture consists of convolutional and max pooling layers. These can be derived from existing image classifier models. Since these layers use max pooling layers and the image's resolution is gradually decreased, the feature maps produced can highlight different features of differently sized objects. The next part of the SSD uses the feature maps generated by this first part of the SSD to predict bounding boxes, the rectangle surrounding a detected object. The bounding boxes are classified, and the model removes duplicate objects. Both the bounding box prediction and classification are done using convolutional layers. The output consists of a list of detected objects. Each object has a class, a size and position of the bounding box surrounding it, and a confidence score of how certain the model is of the prediction. An SSD is considered a good model for real-time object detection because of its computational efficiency and speed. The initial part of the SSD architecture, which can be derived from an existing image classifier, can be selected to achieve an optimal trade-off between accuracy and computational efficiency [23]

One model that is able to utilize the SSD is a stripped version of the MobileNet model [23]. This model contains many layers, but removing the fully connected layers leads to the model being a good option for image classification for mobile devices. MobileNet is a lightweight model that still performs comparably well and has substantially fewer parameters compared to alternative models, making it suitable for object detection on a mobile device.

2.2.8 Tracking Algorithm

Object detection is a crucial process for identifying specific objects within an image or video frame. However, object tracking takes this process to the next level by not only identifying the objects but also monitoring their movement over time. To facilitate this transition, one of the most widely-used approaches is implementing an

algorithm that can trace the identified objects from frame to frame. This is achieved by employing a tracking algorithm that uses the number of objects as well as their positions, heading and speed to track the different objects.

2.2.9 Graphical User Interface (GUI)

Developing the GUI for the application included designing a functional and informative interface. The interface was developed to be easy to use while also providing all the necessary information and controls for the operator and capturing the footage. Below are some scientifically based [24], as well as some project-based, considerations used for the GUI-development.

- **Creating a user centered interface:** The GUI should be intuitive and easy to navigate.
- **Include all necessary commands:** Evaluate what commands are needed for executing the program and make them easily accessible.
- **Create a distinct visual hierarchy:** Prioritize important information and make it easy for the user to locate the necessary information.
- **Real-time representation:** The GUI should provide users with the real-time footage captured by the drone, as well as real-time updates of the drones battery life etc.

2.3 Limitations

This section will delineate the limitations associated with the utilization of the drone hardware, the DJI SDK application with heavy focus in the WaypointMission software as well as the already existing code base.

2.3.1 Hardware

Research into the limitations of the drone presented in Sec. 2.1.1 was needed to obtain necessary information on the feasibility of achieving the performance requirements. The physical limitations does not hinder the project potential as much as the software limitations described in Sec. 2.3.2. This assumption is purely based on the fact that Euro NCAP tests chosen for this project do not create an environment that pushes the limits of the drone capabilities [25] from a hardware standpoint. Even though the drone hardware limitations, for this specific test, see Sec. 1.5, does not become a problem, it is worth noting that there exists Euro NCAP tests [26] with speeds that exceed the speed capability of the drone [25].

Furthermore, the computing power of the Samsung A13 phone presented in Sec. 2.1.2 will be significantly limited in terms of performance compared to a desktop or laptop computer. This limitation will become evident when performing tasks that require significant computational power, such as object detection and image recognition algorithms. Despite these limitations, with an appropriately streamlined software,

the Samsung A13 could still provide a sufficient level of computational power for on-the-go image recognition tasks, such as identifying cars or people.

2.3.2 Software

DJI offers a commercially available off-the-shelf way to both be able to track objects with the camera and the drone following it at the same time. However, the drone was not capable of running these software packages in parallel, making the built in object detection and tracking useless. This was because the implementation of the WaypointMission software was prioritized, as it facilitated the drone to easily follow predetermined paths as well as enabling maneuvering of the drone camera gimbal in parallel.

The WaypointMission software employed for flying pre-determined missions had certain constraints that limited the drone's range, restricting its flight radius to 500 m [17] or less. This is likely a safety precaution considering that the remote controller for the drone can operate within a range of around 10 km under ideal circumstances. The software's target tests at the time were geared towards filming within a radius of less than 500 m, which presented no issues. However, if AstaZero had intended to use the software for conducting tests at greater ranges, it would have been necessary to take this into consideration to ensure the feasibility of the test.

In addition, WaypointMission had a maximum limit of 99 predetermined waypoints per mission, which posed a significant constraint on the drone's accuracy for larger missions and required attention in the code. The trajectory provided by ATOS may consist of more than 99 points and had to be reduced accordingly; see Sec. 4.2.

The WaypointMissions were predetermined, static, and had to be uploaded to the drone before the test commenced. During the test no changes to heading, speed or position could be made automatically, and only the speed could be adjusted manually. This disabled the use of a systems control implementation to compensate for wind and/or similar interference. The WaypointMission class also offered an option to focus the gimbal on different points of interest at each waypoint. However, since information pertaining to the car's location was not given through the ISO DTS 22133 protocol, this was not an option.

2.3.3 Existing code base

The application provided by AstaZero was using an older Android SDK version, which posed some complications. The majority of modern object detection algorithms require newer versions of the SDK, which severely limited the available options for performing object detection and tracking. Updating the SDK used by the application was not an option since parts of the existing code relied on outdated functions.

3
Methods

The present chapter provides an outline of the methodology employed to automate the documentation process. The subsequent sections present a detailed description of the project setup within and between the teams from Chalmers and Penn State University, outlining how the information was gathered, and the methods utilized to validate the final product at the end of the project.

3.1 Project setup and structure

The team consisted of students from both Chalmers University of Technology and Penn State University. There were six students from Chalmers with majors in Mechanical Engineering, and Automation and Mechatronics, with three students in each major. There were also six students from Penn State with diverse majors, Mechanical Engineering (3), Computer Science (1), and Computer Engineering (2). This diversification in educational foundation provided complementing strengths that were applied during the project.

The primary objective for the Chalmers team was to develop an Android application. Said application would facilitate communication between the drone and the control system (ATOS), both supplied by AstaZero. However, the ATOS software lacked the capability to calculate drone trajectories and needed to be implemented. As outlined in Sec. 2.2.2, AstaZero had already developed an Android application with limited functionality with basic features that allowed communication with the drone. The objective was to keep developing this application to give it more features and automate it. The project's main objective was software development and did not involve any physical construction, as outlined in Sec. 1.3.

3.1.1 Work distribution between the two universities

As previously stated, the Bachelor's Thesis work was conducted as a joint venture between Chalmers and PSU. Due to the time difference between the two universities, it was crucial to clearly define the division of labor. This facilitated consistent and independent work from each group, eliminating the need for constant communication.

For the configuration to function properly, two pieces of software needed to be developed and integrated: ATOS and the drone application. This setup allowed

for a natural division of labor between the universities, as both the application and ATOS, which had already been developed by AstaZero, could be worked on independently.

Accordingly, the Chalmers team was solely responsible for developing the drone application, while the PSU team handled the orchestration of the test objects (ATOS) and path calculations. However, since both ATOS and the application were equally crucial for the success of the project, and equally crucial for the configuration to even operate, both groups had to be held accountable for upholding their part of the project. A more descriptive view over the distribution can be seen in Fig. 3.1 below, where AZ is short for AstaZero.

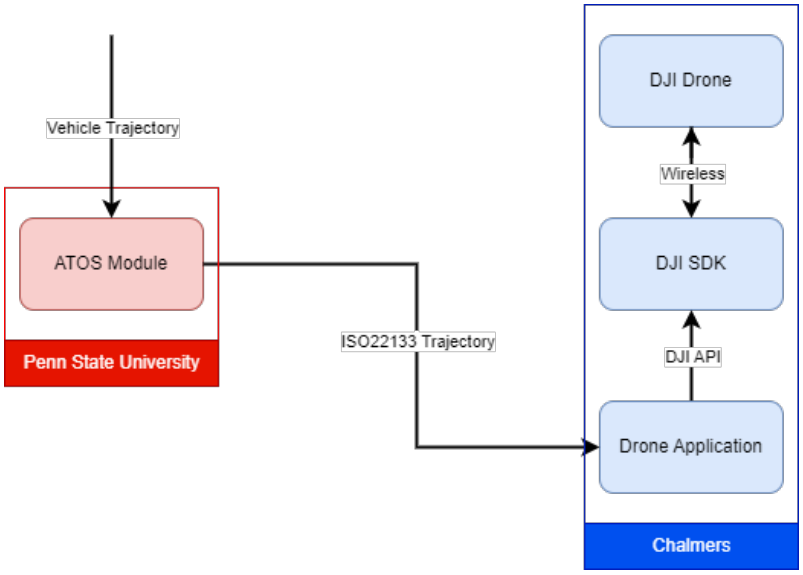


Figure 3.1: Illustration of the work distribution between the universities. Source: Primary

PSU were tasked with determining the precise flight trajectory, dictating the drone's specific positioning at designated times, commonly referred to as trajectories. According to ISO22133 [13], trajectories are a collection of waypoints to define a path of a single test object during test, added by a time stamp for each waypoint (location and time) and test-object dynamics (e.g. yaw, velocity, acceleration). In line with Sec. 1.5 of the project outline, PSU are responsible for providing the accurate trajectory required to capture footage at the appropriate zoom level and angle. Hence, the primary task of the Chalmers team entails enabling the drone to precisely navigate its designated trajectory, while tracing the car with the drone's camera.

3.2 Test driven development

Throughout the project, the software development methodology known as Test-Driven Development (TDD) [27] was a source of inspiration. TDD is a process that follows an iterative development cycle and involves the creation of tests before writing the code itself. As a component of the agile software development approach,

TDD emphasizes continuous integration, testing, and deployment.

In order to ensure the robustness of each subsystem of the drone application, it was crucial to continuously test them. This approach helped to facilitate a smoother integration process, while also providing ease in identifying any subsystem failures. TDD is based on five fundamental steps which is detalied in the following list and Fig. 3.2:

1. Think of what the next step of the code development is. Thereafter, add a test pertaining to the problem at hand, as detailed as possible.
2. Do not write any code which purpose is to solve the issue, i.e ensure that the test fails. The test failure provides insight in what needs to be done.
3. Write new code which purpose is to pass the test.
4. Ensure that the test succeeds
5. Refactor the code, i.e make it more efficient without altering functionality

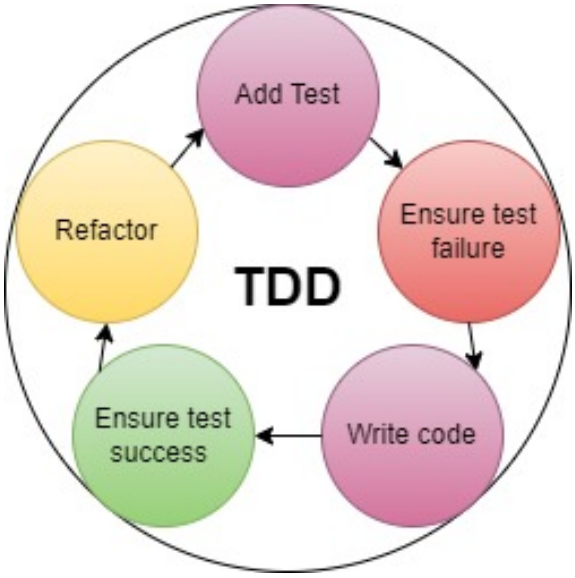


Figure 3.2: Illustration of test driven development. Source: Primary

Throughout the project, the application of TDD has proved to be highly advantageous in the development of the application, with a specifying predefined goal for each function. However, there are some major disparities between the development method applied in the project and TDD. Instead of actually writing tests in the code, actual physical tests were conducted for each improvement of the code. Since the software development concerns drone flight and camera-gimbal movement, ensuring that a test fails is, needless to say, not a great idea. Nevertheless, TDD's modular approach to code development and ease for step-by-step improvement was of immense help and contributed to making the code easier to maintain.

3.3 Validation

Validation assessments were conducted on various implementations of the application, with a specific focus on flight control and object detection. The results of these assessments formed the basis for the project outcomes and the ultimate demonstration at AstaZero's test track.

3.3.1 Flight control

To ensure that the application lived up to the wishes and demands from AstaZero, the drone and ATOS was extensively used in testing component implementations during the project, utilizing the TDD philosophy. The validation tests also laid ground for the results of the project and a demonstration scheduled to take place at AstaZero's test track.

The same baseline for the majority of the tests conducted regarding flight control and WaypointMission. In the ATOS program, a virtual road with the length of 25 m was set up with an origin at a specific spot, chosen in longitudinal and latitudinal coordinates within ATOS. The complexity of the test was then gradually increased to see how different parts worked together. The following list showcases the most important tests conducted:

- Achieve automatic takeoff and landing without ATOS control
- Achieve automatic takeoff and landing with ATOS control
- Run predetermined WaypointMission from application without ATOS control
- Run predetermined WaypointMission from application with ATOS control
- Generate trajectory in ATOS, convert it to WaypointMission and run test with ATOS control.

To ensure that the drone would not pose any danger to anyone, the EmergencyStop function was thoroughly tested with multiple different scenarios. Tests were conducted where the EmergencyStop was issued from both the GUI and the ATOS program in different states of the mission such as the uploading and running phases. To prevent any damage, the EmergencyStop function only stops the drone and the operators had to land the drone manually.

3.3.2 Object detection

In order to ascertain whether the model was suitable for the project, tests were conducted to ensure desirable object detection performance. Given the limited prior experience in object detection, fundamental tests were set up to ensure gradual and steady progress in development. As object detection with a mobile device is a well-documented area, the initial development and testing was done with a camera on a mobile device. The tests were the following:

- Setting up the model on the supplied mobile device

- Film non moving objects
- Film moving cars and persons
- Test model output frequency

Concurrently, a test was conducted on the drone to prepare for the integration of object detection. As the application solely retrieves the position of the selected object, it was necessary to confirm whether the camera angle could be adjusted during a WaypointMission. The test demonstrated that it was feasible to adjust the drone's gimbal while simultaneously executing a WaypointMission.

4
Implementation

In order to achieve the project objective, it was necessary to apprehend a clear and unified understanding of how the different software components would interact with each other. The contributions made by this project should able to run alongside the existing project provided by AstaZero. To facilitate this, the application can be found in a file called ChalmersDemo.java [14] which is an Android activity that the user of the application can decide to navigate to. The user can easily navigate between these two implementations depending on the current use state.

4.1 Re-usability of existing software

As previously mentioned in Sec. 2.2.2, the AstaZero team had previously developed an Android application that could be utilized. The code is currently open-source, and contributions were made to the existing platform rather than creating a new one. As a result, certain code functions critical to the success of the project had already been created.

4.1.1 IsoDrone

To establish a connection between ATOS and the drone, the file IsoDrone [14] provides a Java class that can be initiated to create a drone object able to communicate over the ISO22133 protocol as presented in Sec. 2.2.3. All test objects needs to report the status of the object every X seconds, decided in the ATOS configuration. The need to convert flight telemetry data into ISO-formatted strings were redundant, as this was already handled by the IsoDrone object. Instead, class variables could simply be changed to adjust current speed, heading, coordinates, and similar. The IsoDrone object then automatically transmitted this over the ISO22133 protocol to ATOS.

Moreover, the protocol enabled the drone object to identify the current state of ATOS. This allowed the application to set up triggers that would activate when the state changed.

4.1.2 Local coordinates to Latitude and Longitude

Since ATOS worked with a local coordinate system, the location coordinates had to be converted to latitude and longitude coordinates in the Android application.

The function handling the conversion was already present in the Waypoint1Activity file [14] and works by first reducing the amount of points and then creating a square matrix A with dimensions equal to the number of waypoints in the trajectory, and an array b with the same length as A's dimensions. It then populates A and b with values based on the trajectory information.

To find the radii for each waypoint, which will be used to define the radius of a circular area around each waypoint in which the drone will fly, the Android application utilizes LU decomposition. LU decomposition is a numerical method that decomposes a matrix into the product of two matrices: a lower triangular matrix (L) and an upper triangular matrix (U).

In the context of the waypoint conversion, LU decomposition is applied to the matrix A. By decomposing A into L and U, the Android application can efficiently solve the system of linear equations represented by the matrix equation $A * x = b$.

The LU decomposition allows for efficient solving of systems of equations, as it simplifies the process of finding the radii for each waypoint. Once the decomposition is performed, the solver can easily solve the equations for the radii, providing the necessary information to define the circular flight areas for the drone.

Finally, the method created parameters for each waypoint in the trajectory. It uses a method to convert the waypoint's Cartesian coordinates to geographic coordinates. It then sets various parameters, including the heading, altitude, and speed of the drone at each waypoint, as well as the radius calculated earlier.

4.1.3 Ramer-Douglas-Peucker Algorithm

With the WaypointMission limitations presented in Sec. 2.3.2, the AstaZero team made use of the Ramer-Douglas-Peucker algorithm which is used for reducing the number of points in a curve while preserving its shape [28]. This implementation made it possible to use trajectory files that normally would be composed of too many points to be run by the DJI WaypointMission software.

The algorithm divides the line recursively, starting with a line segment between the first point and last one. The first and last points are automatically selected to be retained, it then identifies the point that is the maximum distance from the line segment. If the point is more than a predefined tolerance away from the line segment, it must be selected as a new start- and endpoint, otherwise the point can be discarded without impacting the shape of the curve. The algorithm recursively calls itself on the two newly created segments and the process is repeated for each new line segment until all the points are outside of the given tolerance. Fig. 4.1 visualizes the algorithm. The result is a new curve with fewer points that still closely approximates the original one.

This algorithm was extensively employed in the deployment of curved trajectories to

test objects that communicate over the ISO22133 protocol, primarily drones. However, it posed a challenge when deploying trajectories that only comprised points in a straight line, as it tended to reduce too many points. The subsequent section outlines a solution to this issue.

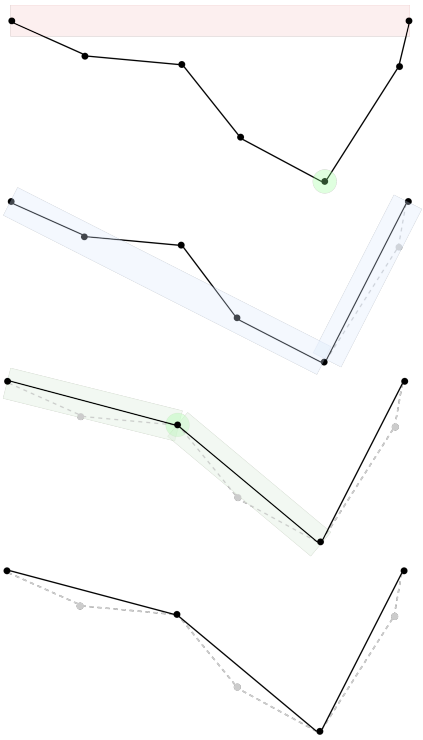


Figure 4.1: Illustration of the Ramer-Douglas-Peucker Algorithm. Source: Primary

4.2 Addition to Ramer-Douglas-Puecker algorithm

The vast majority of the Ramer-Douglas-Puecker implementation produced by the AstaZero team was used with only minor modification and additions. The problematic aspect of the algorithm was that it reduced too many points when presented with a trajectory in a straight line, leaving too few points in the trajectory object to be sent to the next part of the point reduction implementation. The solution to this problem was to develop a simple collinearity check of the trajectory before applying the algorithm. In cases where all points in the trajectory lie on a straight line, the Ramer-Douglas-Peucker algorithm becomes overly complex and is therefore unnecessary. Moreover, adding a tolerance was deemed unnecessary as any straight segments in the programmatically generated trajectories would not deviate from the pre-defined path unless it was an intended curvature. Consequently, adding a tolerance would only introduce unnecessary complexity.

tog bort to

The algorithm's addition functioned by extracting the first two points of the input to establish a reference line for all subsequent points. Thereafter, one of the algorithm's loops was employed to iterate through each point in the input. During

each iteration, if the cross product of the vector generated by the first and current points and the reference line equaled zero, the current point was determined to be collinear with the reference line. This calculation was done for every point in the input trajectory and if any resulting vector from the current point would fail to produce a cross product equal to zero, a *isCollinear* variable is set to false. This would disable the superseding of the Ramer-Douglas-Puecker algorithm and allow it to run as intended. Algorithm 1 describes the addition to the Ramer-Douglas-Puecker algorithm in pseudocode.

Algorithm 1: Pseudocode of the addition to the existing Ramer-Douglas-Peucker algorithm

```
1  Function douglasPuecker(traj, resultTraj):
2      ** Algorithm starts **;
3      if size of traj < 3 then
4          return traj;
5      end
6      isColinear ← true;
7      x1, y1 ← traj[0];
8      x2, y2 ← traj[size of traj − 1];
9      for i ← 1 to size of traj do
10         ** First part of Ramer Douglas Peucker algorithm runs **;
11         px, py ← traj[i];
12         crossProduct ← (x2 − x1) · (py − y1) − (y2 − y1) · (px − x1);
13         if crossProduct ≠ 0 then
14             isColinear ← false;
15         end
16         ** Original Ramer Douglas Peucker algorithm continues **;
17     end
18     if size of traj > max number of points and isColinear then
19         multiplier ← ⌊size of traj / max number of points⌋;
20         for i ← 0 to size of traj-1 do
21             resultTraj ← resultTraj + [traj[i]];;
22             i ← i + multiplier;;
23         end
24         resultTraj ← resultTraj + traj[size of traj];
25         return resultTraj;
26     end
```

4.3 Main Activity

As stated in Sec. 2.2.5, activities are the basic building blocks of any Android application, including the AstaZero application. Each activity in the provided application represents a unique screen with its own graphical user interface, consisting of various elements like buttons, text views, images, and input fields. Every activity is responsible for managing its own tasks, such as establishing a connection between a

remote controller and a drone.

The utilized approach facilitated the creation of a custom activity, named ChalmersDemo.java, which comprises the majority of the code employed in this project. Accompanying ChalmersDemo.java, an associated XML file, activity_chalmers_demo, contained all the visual details for the activity, as presented in the subsequent section. This approach ensured that the code provided by AstaZero for other functionalities remained unaffected, as it has its own activities.

4.4 Graphical user interface

The GUI was developed continuously during the project, going through different iterations. The goal with the GUI was to make it as easy and as intuitive as possible, limiting the number of buttons and information shown on the screen, seen in Fig. 4.2. The relevant information was determined to be:

- The video feed of the camera with detected objects highlighted
- Latitude, longitude and altitude
- Drone state
- Mission information
- IP address of the phone
- Buttons for starting, stopping, landing and resetting the missions

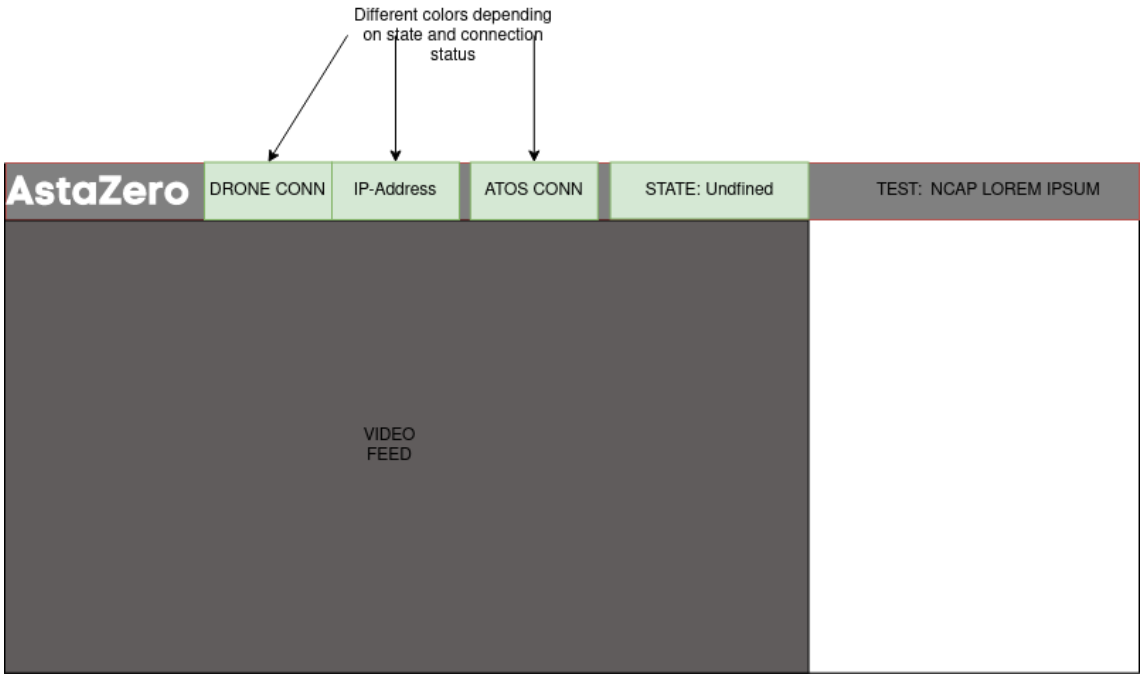


Figure 4.2: The first design idea for the GUI. Source: Primary

This was implemented using XML and relies on a RelativeLayout hierarchy that contains different views as can be seen in the project [14].

The GUI implementation was developed using two different strategies. The initial strategy was to create two separate applications, one for object tracking and another for receiving trajectory data and flying the drone. This was quickly merged into one development process and went through different iterations, while maintaining the general layout. Parts were added and removed as the knowledge base increased with continuous input from the client, ultimately resulting in the final version of the GUI as presented in 5.1.

4.5 DJI Flight Controller

The flight aspect of the project is one of two main parts, the other one being object tracking presented in Sec. 4.7.

4.5.1 Updates from the flight controller

As presented in Sec. 2.2.5, the Android application makes heavy use of DJI's mobile SDK as it is the primary way for third-party software to interact with the drone. The foundation of this interaction is built upon the flight controller object [29] which facilitated the interaction between the drone's different flight-enabling systems.

The DJI SDK constructor was used to initialize the flight controller, and a callback function for the controller was specified at the same time. This callback function provided the application with relevant information on every update, which occurred 10 times per second. This information included the drone's GPS position in terms of latitude and longitude, the strength of the GPS signal, and the current altitude. During the callback, a method was called to update the IsoDrone's flight data. This method utilized the flight controller object multiple times in various ways. One of its primary uses was to create, deploy, and activate WaypointMissions. The update flight data method also integrated ATOS into the Android application. An illustration of ATOS integration is provided in Fig. 4.3.

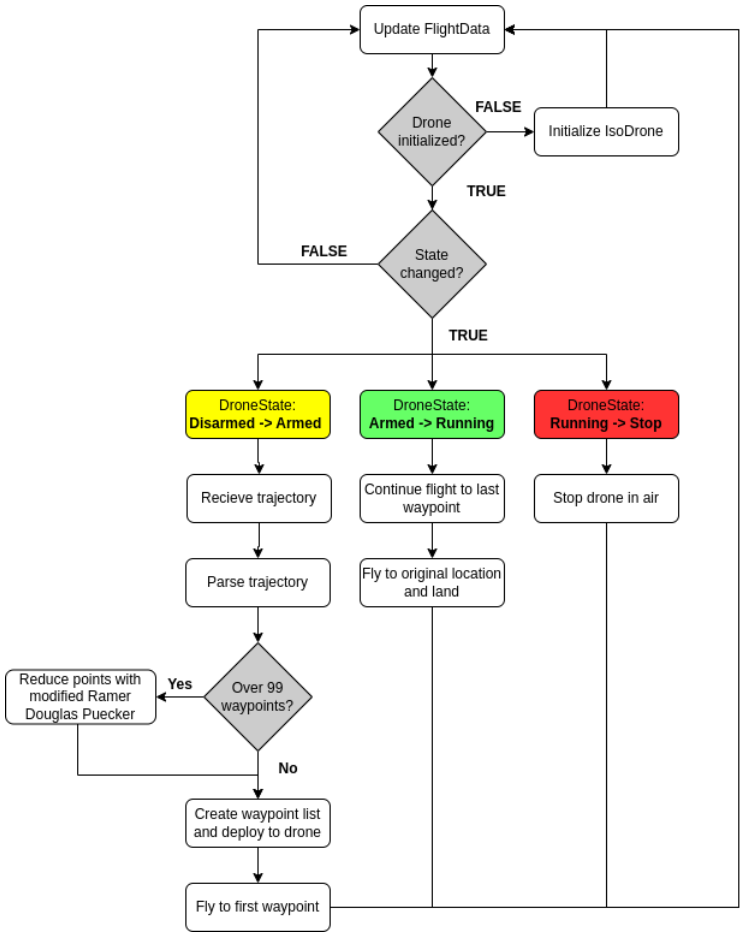


Figure 4.3: Flight data update flow chart. Source: Primary

When the state of the IsoDrone changes from Disarmed to Armed, the trajectory is passed through the application and verified to be in the right format, as can be seen in Fig. 4.3. If the trajectory is too large, it is reduced by the algorithm described in Sec. 4.1.3 and Algorithm 1. When the trajectory is correct, the application transforms the trajectory into global latitudinal and longitudinal coordinates through a coordinate conversion script developed by AstaZero. The application then transmits the trajectories to the drone via a DJI protocol from the remote controller connected to the Android device and the drone flies to the first waypoint. At the first waypoint, the drone stops and hovers in its place. Following this, the test commences when the ATOS state is changed to "Running", and the drone as well as all other test objects execute their pre-determined paths.

4.5.2 Updating the GUI

Updating the user interface was crucial for providing real-time information about the drone's location, altitude, and GPS signal strength as well as showing the test track personnel what is currently being filmed. The GUI had to be updated continuously, and this was achieved by also sending information to the GUI when receiving the callback presented in Sec. 4.5. The GUI also used information such as IP address and ATOS state from the IsoDrone object, as well as the current waypoint the drone is

flying towards. This facilitated the real-time updates needed by the drone operator to make sure the application and drone was running the way it was intended to be run.

4.6 Object detection

Object detection was only implemented using the mobile camera and not the drone camera due to a couple of obstacles described in Ch. 7.

As mentioned in Sec. 2.2.7, TensorFlow Lite was used for object detection in the project, mainly because it had the fast classification that was needed during the test. The image recognition is heavily inspired by an open source project [30] that uses a TensorFlow Lite model to recognize objects.

The application utilized a pre-trained SSD model which uses the convolutional layers of MobileNet v.1.0. The ssd_mobilenet_v1_coco model [31] was selected and is visualized in A.1 and A.2. As described in Sec. 2.2.7.4, an SSD offers computational efficiency while retaining competitive accuracy, making it a perfect fit for a real-time object detection running on a mobile device. Despite the model looking intricate and excessively detailed, the number of parameters is relatively low for the task at hand. The model uses convolutions layers to reduce the number of parameters compared to alternative models such as the Inception V2 and VGG16[23] which uses 10 million and 14 million parameters respectively. The MobileNet only uses 3.2 parameters. The relatively low amount of parameters enhances the speed of the model, making it the best fit for the task at hand [21].

The last part of the model's name, "coco", implies that the model has been trained on the COCO dataset [32]. The COCO dataset is a free website which provides over 200 thousand labeled images to train models on. The model used is able to detect around 90 objects, including persons, cars, bicycles and many more. The selection of this particular model was based on its ability to detect a wide variety of objects, thus rendering it versatile in numerous situations. The Euro NCAP test comprises evaluations where the objects to be detected could be other than cars, including trucks and bicycles. Having a model that can recognize diverse objects offers the test operator the liberty to track any object on the premises with minimal code modifications.

The camera feed from the mobile device was transmitted to the application in a RAW format, which is an unprocessed image data format [33], and subsequently decoded and displayed on the screen. As the video was displayed, a frame from the feed was simultaneously sent to be processed in the background and converted from RAW to RGBA format in a bitmap. The RGBA color model is a three-channel, red-green-blue color representation enriched by an additional alpha channel [33]. This transformation was necessary because the chosen model required the input image to be in the RGBA format. The model output has different attributes for each detected object, which are:

1. Index - The index of the identified object
2. Label - The type of the object
3. Confidence - How sure the model is about the predicted object, 0-100 %
4. Rectangle - Coordinates for drawing a rectangle over the detected object

Since the tests would be conducted in controlled conditions with only a few other objects, several tests were carried out and a confidence threshold value of 50% was found to be optimal.

Consequently, the model was configured to detect people and cars only, and the results were filtered to display only the detected instances of these objects. However, if there arises a requirement to detect other types of objects, such as trucks or buses, the code can be easily modified accordingly. If an object was detected with a confidence score lower than the threshold, it was ignored. As the function iterated through the detected objects, it also annotated the video stream with labeled rectangles and their respective confidence scores. An illustration describing the image recognition process can be seen in Fig. 4.4.

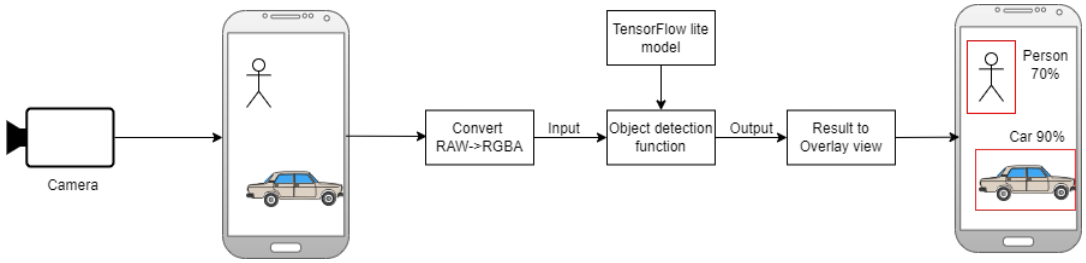


Figure 4.4: Flowchart describing the object detection sequence. Source: Primary

4.7 Object tracking

As previously mentioned in Sec. 2.2.8, object tracking needs to be implemented with the use of object detection to track chosen vehicle. But as stated in the previous section, object detection is only implemented on the mobile camera and not the drone camera, where the object tracking algorithm was supposed to be. Since the adjustments needed for the tracking algorithm relied on the movement of the camera gimbal, it was decided to not pursue a solution until the object detection algorithm was fully functional on the drone camera.

5
Results

This results in this chapter are based on the validation assessments outlined in Sec. 3.3. It includes paragraphs and figures detailing the updated graphical user interface, flight control process, and object detection.

5.1 Graphical User Interface

The updated GUI was incorporated, which included several additional buttons, while still maintaining the same general layout as depicted in Fig. 4.2. The objective was to ensure that the GUI fulfilled specific requirements, such as user-friendliness, visual hierarchy, real-time presentation, and inclusion of all necessary commands. To achieve this, the application employed a single activity that displayed all information on the screen. This design helped users avoid the hassle of navigating through various activities and accidentally pressing the wrong buttons. Furthermore, since the mobile device was connected to the controller in landscape mode, the application was locked to this orientation.

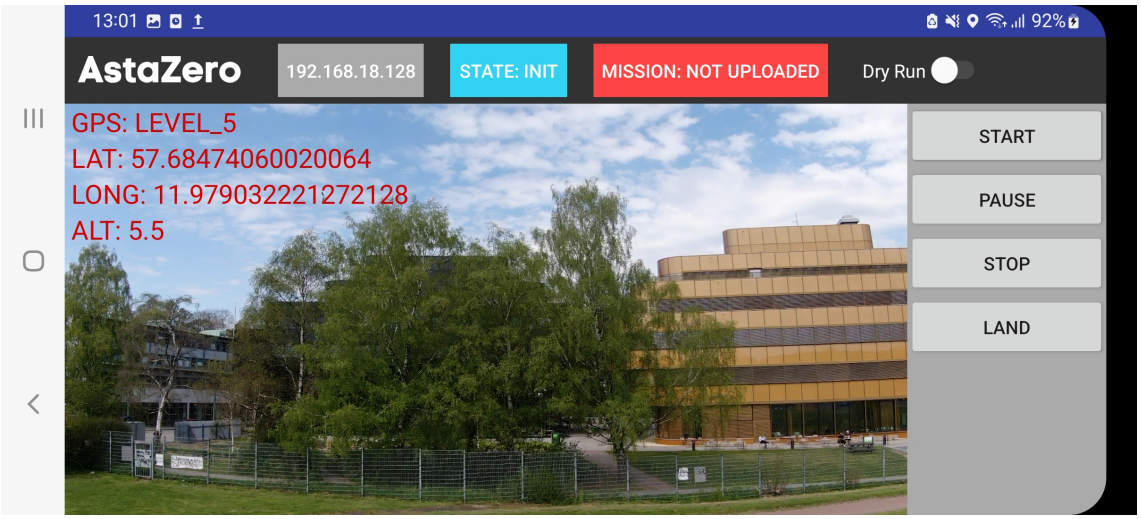


Figure 5.1: GUI visualization while flying manual flight. Source: Primary

In the left center of the screen, the live video feed from the drone can be found, catching the user's attention as it is the main focus of the program. The less relevant data, such as longitudinal and latitudinal coordinates, as well as GPS and

altitude, is displayed above the video feed in red to make it stand out. This information constantly changes and is only relevant if the program needs to be debugged. The header consists of three different text boxes and a slider button, showing vital information to the operator. ATOS works with IP addresses to connect to different test objects and therefore it is relevant to show which IP address the drone currently possesses. The state of the drone is shown in the middle text box with visual information being given in both text and color, shown in Fig. 5.1, Fig. 5.2 and Fig. 5.3. Depending on what state the drone is in, the text box will change color according to the same color scheme used in the ATOS control interface. This gives a quick and clear indication if the drone is in the intended state. Similarly, the third text box also changes color depending on what state WaypointMission currently is in. When uploading the mission, it changes to cyan and switches to green when the mission is executing.

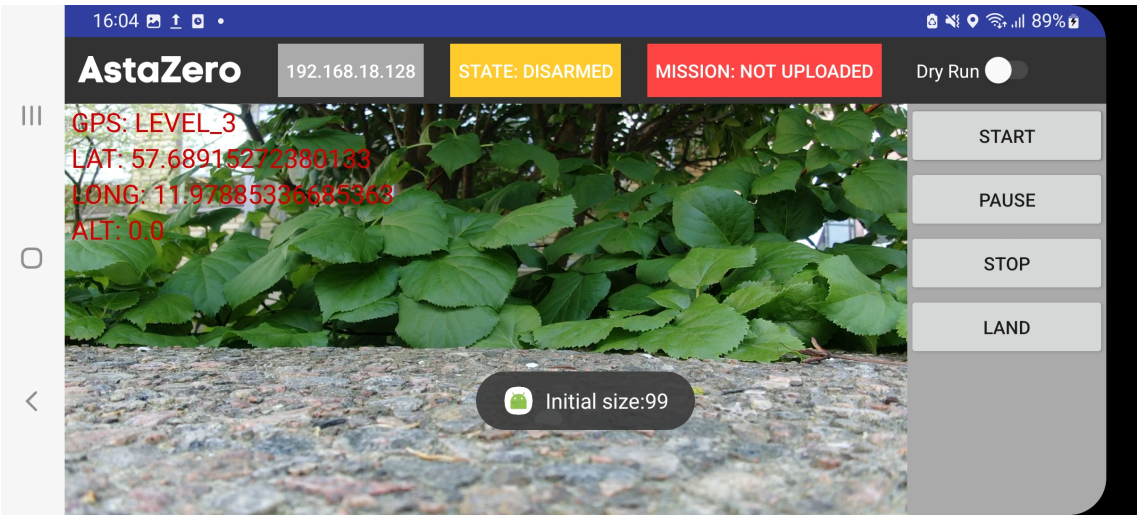


Figure 5.2: GUI when in the ”Disarmed” state and receiving trajectory. Source: Primary

There is also an indication of how many waypoints the mission consists of and what waypoint the drone is currently heading to or currently uploading, as shown in Fig. 5.3. As also shown in the figure, if the drone has not completed all steps and is ready for the running phase, the GUI will show an error message and not execute the mission. The “Dry run” slider enables the operator to decide if the drone should be part of the test or not. The slider is not changeable when a test is executing, it can only be switched when the drone is in the “Disarmed” or “Armed” modes.

The buttons added on the right-hand side are responsible for controlling the functionality of the drone, including initiating an uploaded WaypointMission, pausing it, stopping it, and landing. This provides an efficient means for an operator to promptly halt a mission if any anomalies are detected or if an emergency arises. Moreover, it facilitates developers’ interaction with the application while testing new features, ensuring that the drone can always be halted and landed if necessary.

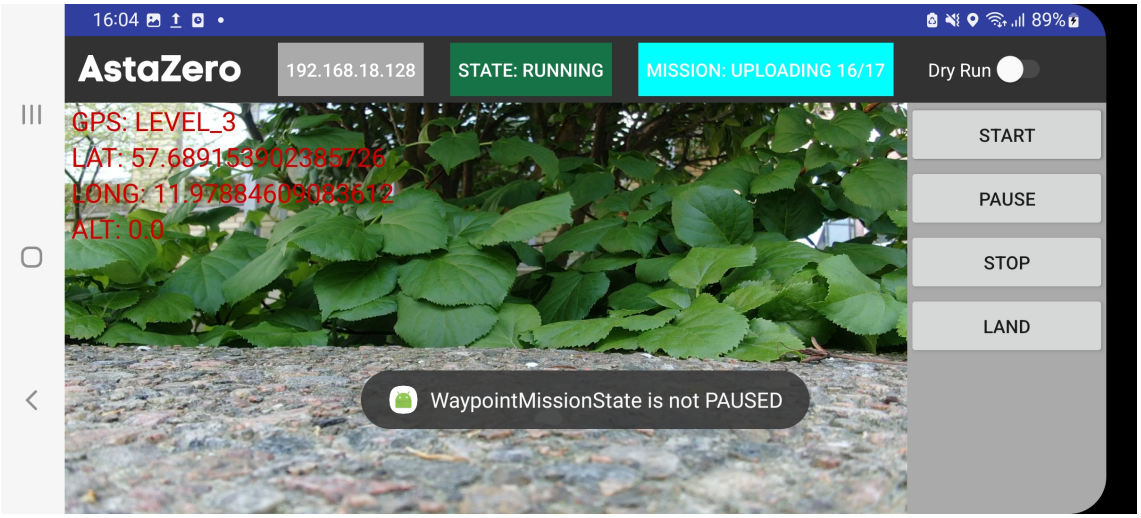


Figure 5.3: GUI when the drone is in the ”Running” state and has not finished uploaded all waypoints. Source: Primary

5.2 Flight control

The foundation for the flight control, as outlined in Sec. 3.3.1, paved the way for the drone to run the desired Euro NCAP tests via ATOS. To do this, the drone must first connect to the ATOS server. As the application handles the ISO-object in the background, establishing the connection is relatively simple. All that is needed is to ensure that the server device and the mobile device are on the same network, configure the relevant files, and provide ATOS with the IP address of the mobile device. Once ATOS starts running, it will automatically connect to the given IP address and begin communicating with the ISO-object in each test object via the ISO DTS 22133 protocol mentioned in Sec. 2.2.3. When the connection is established, all test objects can be initiated and their states switched to “Disarmed” upon connection to ATOS.

After establishing a connection with ATOS and the drone, the application can receive trajectories, modify, convert and deploy them to the drone. If the trajectories consist of too many points, they will be correctly reduced by Algorithm 1. A small popup window on the mobile device screen will show the number of waypoints in the trajectory prior to reduction, and another popup will indicate the number of waypoints remaining after the reduction. The application then converts the trajectories from the ATOS format to the DJI WaypointMission format, preparing them for deployment to the drone.

Upon initiation of ATOS and the transition from the “Init” state to the “Armed” state, the flight path is able to be uploaded to the drone, and afterward take off autonomously. The drone will then ascend to an altitude specified within ATOS, and proceeded towards the test origin where the first waypoint is located. This procedure was thoroughly tested from multiple positions surrounding the test origin, and the drone consistently arrived at the origin, awaiting the “Run” command of

the test before executing the remainder of the mission.

Once ATOS transitions all test objects to the "Running" state, the Android application issues a command to resume the paused WaypointMission and from a hovering position, the drone proceeds to execute the remaining waypoints on the flight path. Throughout all states of the test, the Android application continuously updates the GUI with all relevant information, as described in the previous section. When the mission is finished, the application is capable of autonomously flying the drone back to the position from where it took off.

In summary, the drone has the ability to autonomously take off, navigate its flight path, and land with precision. All of this is achieved and managed through ATOS, without requiring any external input to correctly execute the supplied trajectories at the right time.

5.3 Object detection and tracking

The current version of the application has the capability to capture data from the mobile device's camera, input it to the model and display the output on top of the camera feed on the screen of the mobile device running it. The application can capture data from the drone's camera and display the camera feed on the mobile device running the application, but it cannot do both simultaneously. When running the application on a mobile device, the captured data is initially in the RAW format which is subsequently converted to RGBA to enable it to be used as an input to the object detection model. The outcome of the object detection process is then displayed on an overlay view on the screen, which presents the results generated by the model. An example of the results displayed on top of the camera feed can be seen in Fig. 5.4.

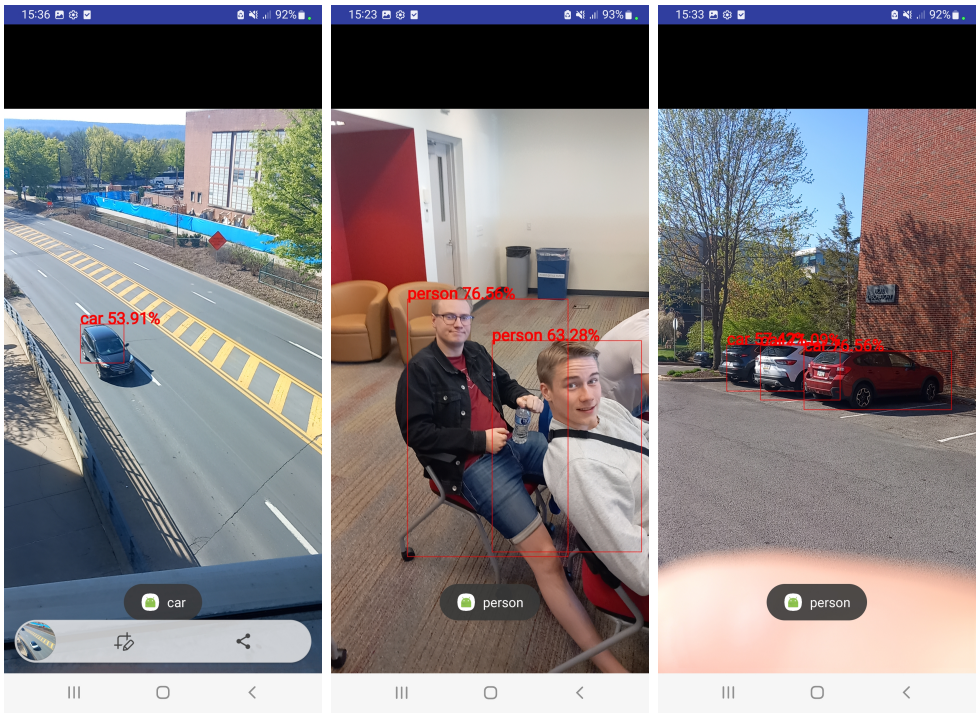


Figure 5.4: Screenshots from the mobile device. The pop up messages were not fully developed and are sometimes inaccurate. Source: Primary

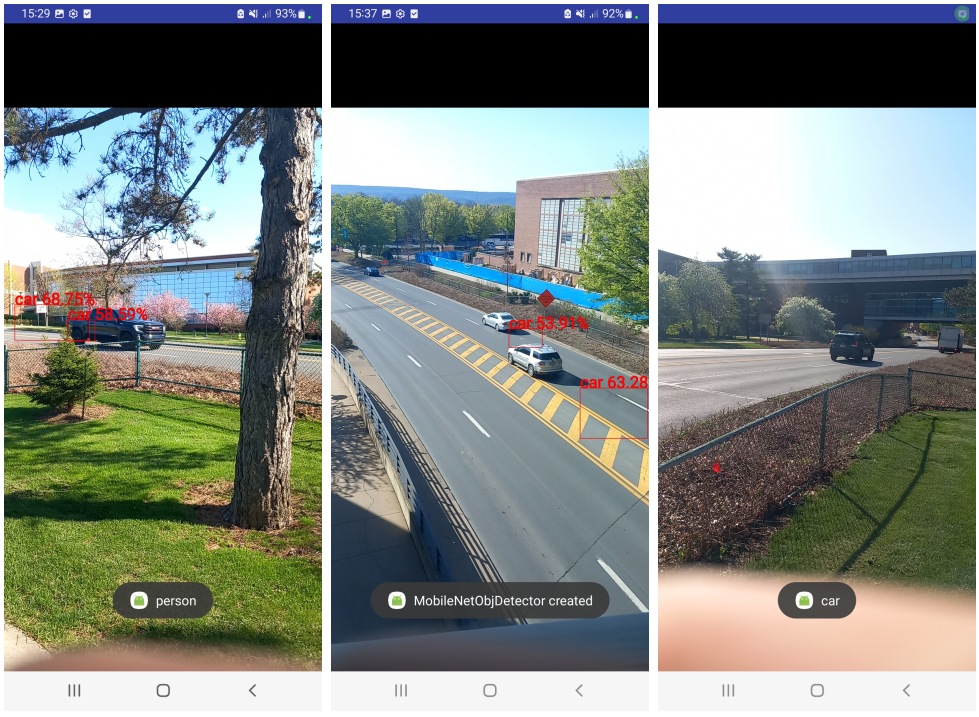


Figure 5.5: Screenshots from the mobile device. The pop up messages were not fully developed and are sometimes inaccurate. Source: Primary

Other tests, presented in Sec. 3.3.2 were also conducted. The model was successful in detecting every non-moving car that it was tested on, an example is shown in

Fig. 5.4. However, larger vehicles such as trucks, buses, and sometimes vans were not detected. This was expected since the model was designed to output only persons and cars. Next, moving cars and people was filmed, which revealed some limitations of the model. Although the model was able to handle moving objects, it struggled with certain scenarios. Fig. 5.5 shows the resulting issues. Finally, the model output frequency was tested to determine how often the model produced results to ensure that the model was quick enough for its intended use. The frequency and subsequent delay of the model can be seen in the picture in the middle of Fig. 5.5. The model averaged four outputs per second.

To implement the object detection algorithm using the drone's camera feed, it was first tested whether the gimbal on the drone could be controlled freely during a WaypointMission. This happened to be the case and the drone flew a simple WaypointMission while the gimbal was automatically controlled with predetermined movements.

Problems arised when trying to implement object detection on the drone camera. To process video and display it, the DJI SDK offers a class named DJICodecManager [34] which offers relevant functions for video encoding and decoding. When trying to work with the DJICodecManager and pull live video data from the stream, the video was formatted in a YUV format [35]. YUV is an image format that separates the luminance (Y) component, representing brightness, from the chrominance components (U and V), representing color information. In this format, the Y, U, and V components are stored as three separate planes within a single array. The class was not able to display the video on screen while also feeding the camera feed to the object detection algorithm. An attempt to decode the YUV data and transform it to a format accepted by the object detection model was undertaken, without success. Following that the object detection on the drone camera was not implemented, the object tracking algorithm was put on hold until the issue was solved.

5.4 Cost-benefit analysis

As previously stated there was a requirement for conducting a cost-benefit analysis for the incorporation of the drones and the developed drone application.

The current utilization of drones for filming the different tests at AstaZero's facility is mostly stationary. The integration of the developed drone app, together with the existing software ATOS, will allow for dynamic filming. Whether the possibility of dynamic filming is beneficial, both from a financial standpoint and in terms of time however, is something that needs to be investigated.

When conducting a cost-benefit analysis there are a lot of minor parameters which can be taken into account, and a lot of different viewpoints that could change depending on the chosen approach. This analysis is dissected into the major and more clearly stated points of affect. The main cost for the project would be the cost of (at least) one drone of type DJI Mavic Enterprise ZOOM, which is roughly

20000SEK [36].

However, depending on adopted approach, this expense could be considered a sunk cost, as the drones are already paid for, regardless of whether AstaZero chooses to utilize the project group's developed app or maintain the current configuration of stationary footage.

In addition, maintenance expenses such as unforeseen repairs and charging requirements should also be considered. Furthermore there are the labor costs, information regarding how many workers are needed to conduct the tests are not specified and vary from time to time. However, a deduction of at least one manpower position is reasonable due to the app-enabled automatic flying capability of the drones, which eliminates the need for an individual human operator per drone, unlike the current configuration. However, considering that the development of the app is completely free for the client in terms of labor costs, the NPV-analysis will neglect salary costs for developers, making it non-applicable for use outside of this specific project.

The implementation and subsequent development of the application will ensure the production of consistent and dynamic footage over an extended period of time. In the event that the trajectories for the test objects and the drones remain unchanged, this will establish standardized scenes for all future tests conducted. As a result, this will provide a clearer overview of the different tests and facilitate a more straightforward comparison of the various vehicles based on the footage.

Furthermore, if AstaZero chooses to continue to use DJI as their drone provider, and assuming that future versions of drones remain similar to the current models, the application will continue to serve as a useful tool for documentation of Euro NCAP tests without becoming obsolete.

This configuration also reduces the need for human interaction, it also eliminates the risk for human errors while footage is being captured, making it feasible that the lifetime for the drones will be maximised. Combining all these considerations, the potential upside of this configuration heavily outweighs the minuscule potential downsides.

In strictly monetary terms, the annual savings of the implementation can be calculated with data provided by the manager at the AstaZero facility through an interview. According to the facility-manager there are roughly 300 tests annually that includes the use of drones. The test itself in effective time takes five minutes, however, with estimated preparation-time it is rather around 15 minutes. The stated hourly cost for each worker was said to be 350 SEK. Taking these factors into consideration, the annual saved amount of the implementation totals to 26250 kr.

5.4.1 Net present value analysis

In order to quantify the costs and savings for implementing the drone configuration in monetary terms, there is a need to define the usual annual costs for the drones and for labor. AstaZero is an non-profit organization, which means that they are unlikely to have a discount rate for alternative investments. The NPV-analysis will therefore only take inflation into account when calculating the value.

All the data for the NPV-analysis are taken from an interview with the testing manager at the AstaZero facility. However, it is based on assumptions and is shown to give an overview for the monetary benefits of implementing the drone-app-configuration. The analysis is only taking 5 years into consideration, meaning that the net present value in fact could be a lot larger, provided that it does not become obsolete.
The following parameters are taken into consideration for the NPV:

- Cost of one singular drone: 20000 SEK
- Saved amount annually for a reduction of one manpower: 26250 SEK
- Estimated annual maintenance costs: 1500 SEK
- Estimated future inflation rate: 4% (which is way higher than the mean value over the last few decades, which is less than 2%)

Table 5.1: Five-year Net Present Value Analysis

Net Present Value Analysis [SEK]						
Year	0	1	2	3	4	5
Drone investment	-20000					
Saved amount		26250	26250	26250	26250	26250
Maintenance costs		-1500	-1500	-1500	-1500	-1500
Net value (NV)		24750	24750	24750	24750	24750
Inflation rate		4%	4%	4%	4%	4%
NV adjusted for inflation	-20000	23798	22883	22003	21156	20343
Net present value = $\sum$ NV = 90183 kr						

6
Considerations

Drones are generally considered to be environmentally friendly, there are however certain risks that come with drones and vehicle testing.

6.1 Ethical aspects

Although the project did not have a direct impact on the general public's life in an ethical aspect or in a socio-economic point of view, there were still some important considerations to keep in mind. At the time, AstaZero performed manual calculations and transmissions of paths, requiring man-hours. Upon completion of the project, the work-load was likely to decrease, leading to a shift in labor distribution. Although the elimination of manual adjustments could lead to a reduced demand for staff, it was deemed to be a non-feasible outcome. Contrarily, the development of the application would serve as a foundation for further software development, implying an increase in labor demand. Additionally, there would be a need for maintenance and quality control, creating even more job opportunities.

In terms of ethical aspects, there were a few more things to keep in mind. The project included video documentation, and it was crucial to abide by the laws and relevant guidelines in this regard. Furthermore, during the NCAP tests, there could be vehicles and classified information that should not be filmed. Therefore, careful consideration had to be given to the implementation of the drone app to ensure that the captured footage was limited only to the desired vehicle and did not include any sensitive or restricted information.

6.2 Environmental aspects

As with many electric appliances or vehicles, most drones are powered by lithium batteries. The production of lithium batteries comes with many environmental problems, such as a lot of water being used when mining for lithium, which dries surrounding land and severely harms the local ecosystem [37]. Potential leakage that has been known to occur in these mines can poison the surrounding wild life [38]. However, the implementation of drones is not unique to our project, our goal is to automate the drones instead of manually flying them. To truly consider environmental aspects of the project, one should consider alternatives to drones capturing the footage from the tests. Since the implementation of drones, footage from a camera moving alongside a car is possible. This footage can probably provide better

material to analyse a car's performance in certain tests. Alternatives to having a drone move alongside a car being tested include filming from some other vehicle. Few vehicles are however as light as drones, and if the battery is charged with electricity from renewable sources, the environmental impact of using the drone can be considered almost negligible.

6.3 Laws and rules

The operation of drones, is subject to varying rules and regulations across different countries. In Ch. 1.3, the problem statement declares AstaZero's objective of expanding drone flight capabilities beyond its Swedish test track. However, accomplishing this goal presents a formidable challenge due to the regulatory framework. Considering that the project pertains to European testing, the focus is solely on complying to regulations within Europe. The increasing expansion of drones necessitates constant adaptation of regulations, further compounded by the divergent rules prevailing in each country. Consequently, conducting comprehensive research on the regulatory requirements becomes vital when contemplating tests in countries other than Sweden. Additionally, it is essential to note some general rules applicable to flying drones, particularly within the context of private property, as AstaZero is unlikely to undertake Euro NCAP tests outside a designated test track.

6.3.1 Overview of Drone Regulations

The operation of drones is governed by a huge number of regulations that differ from one country to another. These regulations are established at the national level and may be supplemented by international standards, such as those implemented by the European Union Aviation Safety Agency (EASA) [39], to ensure consistency across Europe.

6.3.2 Dynamic Nature of Regulations

The increasing popularity of drones has led to a continuous evolution of regulatory frameworks. As a result, regulations related to drones undergo frequent updates, making it imperative for all parties involved to remain informed about the latest requirements. Moreover, due to the divergent regulations across countries, it is crucial to conduct thorough research to ensure conformity when planning drone tests outside Sweden. It is worth noting that although AstaZero primarily focuses on conducting tests on private property, certain general rules should still be taken into consideration.

6.3.3 General Rules for Flying Drones on Private Property

When operating a drone on private property, the following rules should be complied with according to EASA [39] and Transportstyrelsen [40]:

- Registration: Prior to flight, it is necessary to register the drone on the official website of the transport department or the relevant regulatory authority.
- Online Course Completion: Drone operators are required to complete an online course covering the essential knowledge and guidelines for safe and responsible drone operation.
- Altitude Limitations: Drone flights should not exceed a maximum altitude of 120 m or 400 feet, ensuring compliance with airspace regulations and safety protocols.
- Minimum Pilot Age: The pilot operating the drone must be at least 16 years of age, demonstrating maturity and proficiency in UAV handling. By adhering to these general rules, AstaZero can ensure responsible drone operations while considering the specific regulations of the country where tests are conducted.

In conclusion, the operation of drones is governed by diverse and ever-changing regulations across different countries. AstaZero's ambition to expand drone flight capabilities beyond its Swedish test track presents a significant challenge due to the regulatory framework. With a focus on European testing, compliance with regulations within Europe is of utmost importance. The increasing popularity of drones necessitates continuous adaptation of regulations, and comprehensive research is crucial when planning tests in countries other than Sweden. Observing general rules, including registration, online course completion, altitude limitations, and minimum pilot age, ensures responsible drone operations. By considering these factors, AstaZero can navigate the complex regulatory landscape and conduct successful drone tests.

7
Discussion

A fully developed application could be substantially beneficial for AstaZero while performing the assessments included in the Euro NCAP portfolio. Implementation of a fully finished composition of the application and the ATOS-module would guarantee accurate, reliable, and most of all, consistent footage of the various test scenarios. The standardized approach for footage-capturing would also simplify the comparison between vehicles that are tested, which of course is of essence when grading them. Furthermore, as the cost-benefit analysis concluded in Sec. 5.4, the implementation is also financially advantageous and beneficial in terms of time-consumption for AstaZero.

The sheer scale and complexity of this project have posed significant challenges that required numerous problem-solving endeavors throughout the process. As previously mentioned, this project has been conducted in collaboration with six students from Penn-State University in State College, PA. The six hour time-difference and the diverse schedules between the groups and the members within the group, has in some instances been challenging. Without the division of labor between the two groups from the different universities, this would likely have been a larger obstacle. However, with a project of this nature, dividing the tasks led to a limited influence on the other group's output, which implicates the need of holding both groups accountable in case of any failure to fulfill their designated responsibilities, which unfortunately was the case.

The ATOS-module which was developed by the PSU-team was not finished on time for their graduation which ultimately led to that the strictly necessary ATOS-module could not be implemented in the fully-fledged package. Due to the Chalmers team's limited experience with the ATOS-module, the team could not simply continue where the Penn State-team left off. As previously mentioned, the ATOS-module sends the calculated trajectories for the drones, making it completely invaluable for the success of the project. The fact that this was not yet finished by the end of the project made it impossible to deliver a fully finished and integrated application. However, the Chalmers team has utilized other parts of ATOS to create static trajectories which are not calculated relative to a object. This made it possible to test the application developed by the team and does in fact simulate how it was intended to work, all thanks to the ISO22133 standard. The standard implicates that all data transmitted to the drone over the protocol will always be in the same format, making it easy to switch to the module PSU was responsible for to deliver the trajectories if it was to be finished, either by another project or by employees at AstaZero.

Moreover, the team encountered multiple challenges in their efforts to obtain accurate footage of the test. Initially, the application utilized the ActiveTrackOperator [41] from the DJI SDK and several weeks were spent attempting to incorporate this package into the application. However, it was later discovered that the WaypointMission and ActiveTrackOperator packages were incompatible and could not be run simultaneously. More research should have been conducted before committing to a singular solution, as this setback impeded the progress of the object detection development for several weeks as alternative approaches were explored.

Additionally, as the entire process relies on a mobile device with limited computational power, the application was restricted to only employ a streamlined and lightweight image recognition package that demands minimal processing power. Although the Samsung A13 performs well in certain areas, it cannot compete with a laptop or workstation, which can support more intensive software. If AstaZero was not satisfied with the performance of the model, another model could be used instead. There are many models within TensorFlow Lite which are trained on the same COCO dataset as mentioned in 4.6. Some models are even faster but not as accurate as the model used in this project.

As previously mentioned object tracking was not accomplished due to multiple reasons. Firstly, there was limited prior knowledge in object detection and no experience with object tracking. Secondly, as stated previously, not enough research was conducted on the compatibility of different DJI SDK components, especially the ActiveTrackMission in relation to WaypointMission. Thirdly, since the application that was provided by AstaZero runs on an old version of the Android SDK the options for object tracking packages were slim as mentioned in Sec. 2.3.3. Lastly, due to an unfortunate accident where the drone crashed into a wall, it became impossible to continue testing new features. To be able to test different parts of the application, the mobile device needs to be connected to the remote controller which also needs a connection to the drone. Without an active connection, the application will not run and contributed to the reason why object detection was not completed on the drone and why object tracking still has not been implemented.

It is important to acknowledge the ethical and safety considerations associated with employing drones in safety testing. One such concern is the potential breach of privacy that may arise if drones are employed to record video footage of multiple tests being conducted simultaneously, or in areas where people may not anticipate surveillance. Moreover, there exists a plausible risk to safety in the event of a drone malfunction or collision with a foreign object during a test.

All in all, the Chalmers team is satisfied with their accomplishments, even though a fully functional solution can not be delivered to AstaZero. The application can serve as a foundation for future work by either AstaZero or another Bachelor's thesis project, adding even more functionality to simplify the documentation of Euro NCAP tests using drones.

8
Future work

The current system has limited capabilities and there are plenty of opportunities for future work to improve and expand upon this technology.

One area which requires further development is the task not completed, the object detection and tracking algorithm. Essentially, there are two approaches to this challenge. The first approach involves parsing the YUV data correctly and displaying it on the screen while finding a way to still being able to send it to the object detection algorithm. The second approach entails not utilizing the YUV callback at all and instead attempting to implement a different callback or receiver in order to display the camera view on the screen. Further testing and research about the capabilities and limitations of the DJI SDK is needed to determine the best course of action.

As detailed in Sec. 2.3.3, the selection of compatible object detection packages was limited by the outdated Android SDK version. Nonetheless, TensorFlow Lite was identified as a promising solution owing to its fast response time and acceptable accuracy levels. Our experiments revealed that the camera's capability to track the selected object was fundamental, thereby emphasizing the significance of prioritizing response speed over accuracy.

In light of this observation, any changes of model should focus on maximizing speed rather than enhancing accuracy by incorporating a bulkier and more sophisticated model. This strategy will ensure that the application remains agile and effective on any Android mobile device, while concurrently fulfilling its primary goal of tracking objects. Currently, the project uses a single drone to capture video footage of Euro NCAP tests. While this provides valuable information, it is limited by the fact that it can only capture footage from one angle at a time. By using three drones instead of one, it would be possible to capture footage from multiple angles simultaneously, providing a more comprehensive view of the test or if the test instructions would change.

Theoretically, a test could be documented using any number of drones, where each drone is linked to a unique instance of the application running on separate mobile devices. The problem lies within the ATOS module that the PSU team was responsible for, which currently is not designed to create different trajectories for three different drones. Since this project only focused on the mobile application, it lied outside the scope of this project but could be a subject for future work.

9
Conclusion

The drone application developed for Euro NCAP test documentation has the potential to save costs and enhance efficiency at AstaZero by reducing human errors and interventions. The approach presented in this report improves testing accuracy and sustainability, making the test process more reliable and streamlined.

AstaZero expressed their desire to find a solution to address the drones limited battery life. This desire was mainly in relation to the battery's capacity being unable to document several tests back to back. The solution was to add the "Dry Run" button since documenting the unofficial rehearsal tests were redundant. This lowers the amount of flight time which saves battery.

The application could potentially utilize other means of saving battery, such as using the zoom function instead of visiting every waypoint in a mission. This means that the drone could fly on a higher altitude with a lower speed and mostly utilizing the camera to capture the desired angles. However, since the ISO22133 protocol lacks the capability of transmitting information about zoom or the location of other objects, this is hard to achieve.

However, some technical challenges need to be addressed to improve the performance of the system. The application is not capable of automatically adjusting the camera-gimbal for capturing the relevant video frames. This means that an operator still needs to control the gimbal, but not the actual flying. This leads to better performance and smoother video capturing compared to how the tests were captured before, but does not remove the need for a person operating the drone.

In conclusion, many wishes and subtasks presented has been addressed and solved, making the application a contender instead of manual flight and operation of the drone. Even though some challenges still are unsolved, this project has demonstrated the ability to use drones for automating documentation of Euro NCAP tests and has made an significant contribution the existing code base at AstaZero.

The team's recommendation to AstaZero is to publish the continuation of this project in next year's Bachelor thesis repertory to guarantee the finalization of the application and the ATOS module, if AstaZero does not intend on completing it themselves. Although the remaining work to attain complete functionality may not be vast, there is always room for improvement.

A

Appendix 1

Figure A.1: Part 1 - Object detection model used in the application

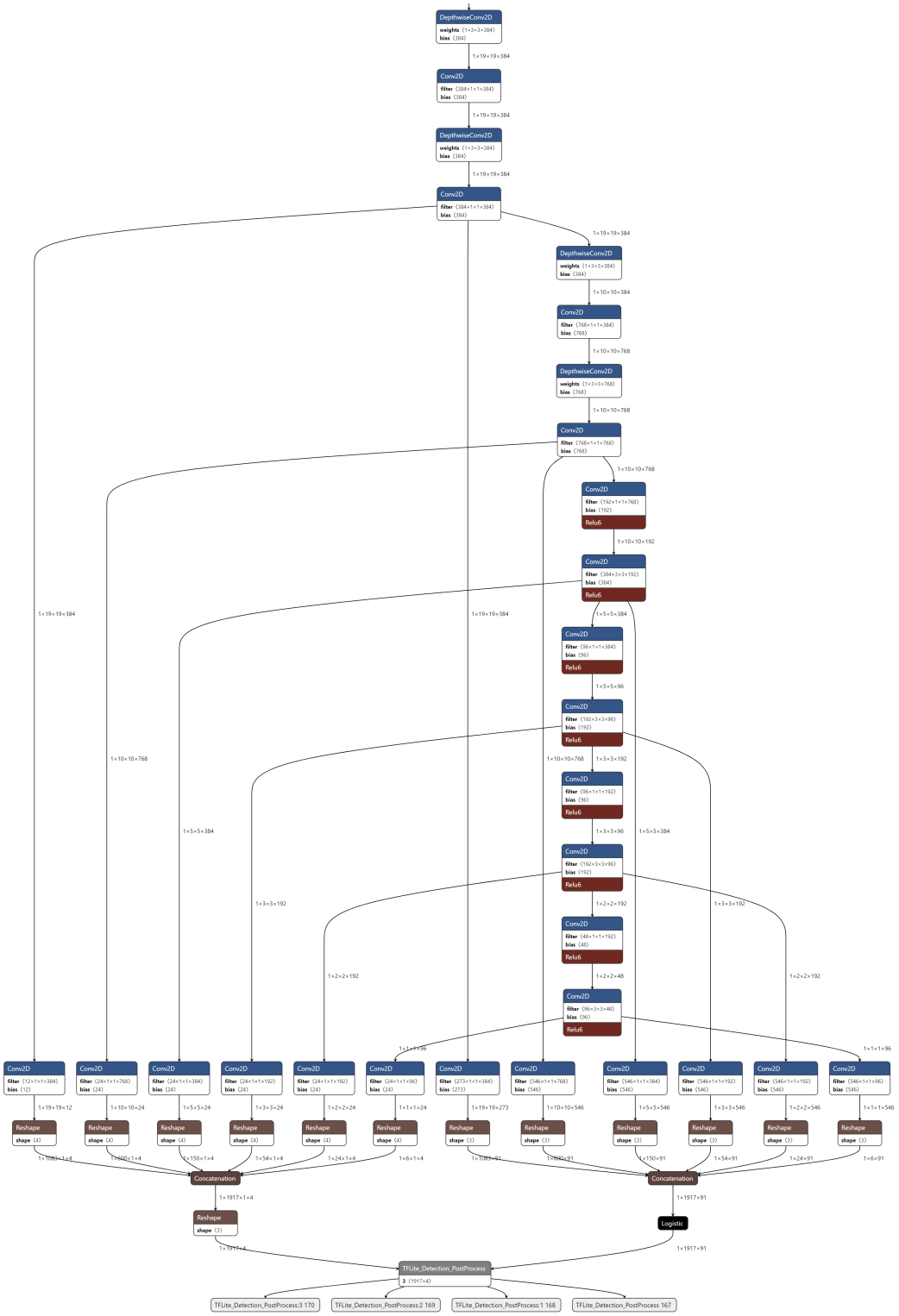


Figure A.2: Part 2 - Object detection model used in the application

References

[1] AstaZero, *About us.* [Online]. Available: https://www.astazero.com/en/about-us/ (visited on 05/05/2023).

[2] V. C. Corporation, *Aiming for zero*, 2020. [Online]. Available: https://group.volvocars.com/company/safety-vision (visited on 05/15/2023).

[3] E. NCAP, *The european new car assessment programme.* [Online]. Available: https://www.euroncap.com/en (visited on 03/11/2023).

[4] E. NCAP, "Euro ncap vision 2030: A safer future for mobility," Nov. 2022. [Online]. Available: https://www.euroncap.com/en/press-media/press-releases/euro-ncap-vision-2030-a-safer-future-for-mobility/ (visited on 01/31/2023).

[5] E. NCAP, "Film & photo protocol," Nov. 2021. [Online]. Available: https://cdn.euroncap.com/media/67257/euro-ncap-film-and-photo-protocol-v131.pdf (visited on 01/31/2023).

[6] AstaZero, *Atos: Ros2 based platform for coordinating tests of autonomous vehicles and their surrounding systems.* Mar. 2023. [Online]. Available: https://github.com/RI-SE/ATOS (visited on 04/29/2023).

[7] E. NCAP, "Test protocol-lane support systems implementation," Nov. 2022. [Online]. Available: https://cdn.euroncap.com/media/75440/euro-ncap-lss-test-protocol-v42.pdf (visited on 01/31/2023).

[8] E. NCAP, "Test protocol-aeb car-to-car systems implementation," Nov. 2022. [Online]. Available: https://cdn.euroncap.com/media/75439/euro-ncap-aeb-c2c-test-protocol-v411.pdf.

[9] E. NCAP, "Test protocol-aeb/lss vru systems implementation," Nov. 2022. [Online]. Available: https://cdn.euroncap.com/media/75436/euro-ncap-aeb-lss-vru-test-protocol-v43.pdf.

[10] DJI, *Mavic 2 - dji*, 2018. [Online]. Available: https://www.dji.com/se/mavic-2 (visited on 02/07/2023).

[11] M. Verc, *Close-up of dji mavic 2 zoom drone*, Sep. 2018. [Online]. Available: https://www.flickr.com/photos/149561324@N03/44418213012 (visited on 05/12/2023).

[12] Samsung, *Exynos 850 | mobile processor | samsung semiconductor global*, 2023. [Online]. Available: https://semiconductor.samsung.com/processor/mobile-processor/exynos-850/ (visited on 05/11/2023).

[13] *Iso/ts 22133:2023 - road vehicles — test object monitoring and control for active safety and automated/autonomous vehicle testing — functional requirements, specifications and communication protocol*, 2023. [Online]. Available: https://www.iso.org/standard/78970.html.

[14] AstaZero, *A demo for using dji mobile sdk to create a waypoint mission app using google map.* [Online]. Available: https://github.com/RI-SE/Android-GSDemo-GoogleMap (visited on 05/15/2023).

[15] DJI, *Mobile-sdk-tutorials.* [Online]. Available: https://github.com/DJI-Mobile-SDK-Tutorials (visited on 05/15/2023).

[16] A. Developers, *Activity.* [Online]. Available: https://developer.android.com/reference/android/app/Activity (visited on 04/28/2023).

[17] DJI, *Dji mobile sdk documentation | waypointmission*, 2022. [Online]. Available: https://developer.dji.com/api-reference/android-api/Components/Missions/DJIWaypointMission.html (visited on 05/04/2023).

[18] M. Pak and S. Kim, "A review of deep learning in image recognition," *Proceedings of the 2017 4th International Conference on Computer Applications and Information Processing Technology, CAIPT 2017*, vol. 2018-January, pp. 1–3, Mar. 2018. DOI: 10.1109/CAIPT.2017.8320684.

[19] W. Liu, D. Anguelov, D. Erhan, *et al.*, *Ssd: Single shot multibox detector*, 2016. DOI: 10.1007/978-3-319-46448-0_2. [Online]. Available: https://doi.org/10.1007\%2F978-3-319-46448-0_2.

[20] TensorFlow, *Post-training quantization | tensorflow lite.* [Online]. Available: https://www.tensorflow.org/lite/performance/post_training_quantization (visited on 05/15/2023).

[21] R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: An overview and application in radiology," *Insights into Imaging*, vol. 9, pp. 611–629, 4 2018, ISSN: 1869-4101. DOI: 10.1007/s13244-018-0639-9. [Online]. Available: https://doi.org/10.1007/s13244-018-0639-9.

[22] Cecbur, *Convolutional neural network with color image filter.* [Online]. Available: https://openverse.org/image/8e83a90d-12e3-402f-a00a-b66f600cbd10?q=neural\%20network.

[23] e. a. Jonathan Huang, *Speed/accuracy trade-offs for modern convolutional object detectors*, 2017. (visited on 05/15/2023).

[24] I Antcheva, R Brun, F Rademakers, and G. Switzerland, "Guidelines for developing a good gui,"

[25] DJI, *Mavic 2 enterprise series user manual*, Apr. 2021. [Online]. Available: https://dl.djicdn.com/downloads/Mavic_2_Enterprise/20210413/Mavic_2_Enterprise_Series_User_Manual-EN.pdf (visited on 01/31/2023).

[26] E. NCAP, "Test protocol-speed assist systems," 2017.

[27] TestDriven.io, *What is test-driven development?* [Online]. Available: https://testdriven.io/test-driven-development/ (visited on 04/04/2023).

[28] D. Douglas and T. K. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," University of Ottawa/Simon Fraser University, British Columbia, Dec. 1973, pp. 112–122.

[29] DJI, *Mobile sdk flightcontroller.* [Online]. Available: https://developer.dji.com/api-reference/android-api/Components/FlightController/DJIFlightController.html?search=flightcon&i=0& (visited on 05/15/2023).

[30] mrinalTheCode, *Android app using tensorflow lite api for object detection.* [Online]. Available: https://github.com/mrinalTheCoder/ObjectDetectionApp (visited on 05/12/2023).

[31] TensorFlow, *Tensorflow 1 detection model zoo*, 2021. [Online]. Available: https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf1_detection_zoo.md#mobile-models (visited on 05/12/2023).

[32] C. Consortium, *Common objects in context*. [Online]. Available: https://cocodataset.org/#home (visited on 05/12/2023).

[33] E. T. W. Menasco and H. S. E. LLP, *Method, device, and storage medium for converting image from raw format to rgb format*, Oct. 2022. [Online]. Available: https://eds.s.ebscohost.com/eds/detail/detail?vid=7&sid=7b3e4dd2-262f-4493-bf40-8070e5274b3a\%40redis&bdata=JnNpdGU9ZWRzLWxpdmUmc2NvcGU9c2l0ZQ\%3d\%3d#db=edspgr&AN=edspgr.11470294 (visited on 05/23/2023).

[34] DJI, *Dji mobile sdk documentation | djicodecmanager*. [Online]. Available: https://developer.dji.com/api-reference/android-api/Components/CodecManager/DJICodecManager.html (visited on 05/12/2023).

[35] test, *Rgb to yuv format conversion and inverse conversion method and circuit for depth packing and depacking*, Mar. 2019. [Online]. Available: https://eds.s.ebscohost.com/eds/detail/detail?vid=6&sid=8ed0222f-b410-4a00-8cf0-aee5fe6197da\%40redis&bdata=JnNpdGU9ZWRzLWxpdmUmc2NvcGU9c2l0ZQ\%3d\%3d#db=edspgr&AN=edspgr.10242646 (visited on 05/23/2023).

[36] D. Stockholm, *Dji mavic*. [Online]. Available: https://djistockholm.se/kategori/dji-mavic-main/ (visited on 05/12/2023).

[37] B. Sophie, *Explainer: The opportunities and challenges of the lithium industry*, Dec. 2020. [Online]. Available: https://dialogochino.net/en/extractive-industries/38662-explainer-the-opportunities-and-challenges-of-the-lithium-industry/ (visited on 01/31/2023).

[38] A. Katwala, *The spiralling environmental cost of our lithium battery addiction*, May 2018. (visited on 01/31/2023).

[39] P. office of the European Union, "Commission delegated regulation (eu) 2019/945 on unmanned aircraft systems and on third-country operators of unmanned aircraft systems," *EUR-Lex*, 2019. [Online]. Available: http://data.europa.eu/eli/reg_del/2019/945/2020-08-09.

[40] Transportstyrelsen, *Drones – unmanned aircraft*. [Online]. Available: https://www.transportstyrelsen.se/en/aviation/Aircraft/drones--unmanned-aircraft/ (visited on 05/12/2023).

[41] DJI, *Dji mobile sdk documentation | activetrackoperator*. [Online]. Available: https://developer.dji.com/api-reference/android-api/Components/Missions/DJIActiveTrackMissionOperator.html (visited on 05/12/2023).

DEPARTMENT OF ELECTRICAL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS