



Bachelor's Thesis

# Investigation of Visual Environmental Effects of Urban Air Vehicles

In partial fulfillment of the requirements for the degree  
Bachelor of Science  
at the department of Aerospace and Geodesy  
of the Technical University of Munich.

|                     |   |
|---------------------|---|
| <b>Advisor</b>      | Katharina Meinecke<br>Prof. Dr.-Ing. Agnes Jocher<br>Assistant Professorship of Sustainable Future Mobility |
| <b>Submitted by</b> | Cristina Molina Aragón  |
| <b>Submitted on</b> | Munich, September, 30, 2023   |

## **Abstract**

Urban Air Mobility (UAM) emerges as a promising solution to address transportation congestion in overpopulated megacities. Yet, there is a common concern regarding the prominent visual effects of these vehicles and the potential obstruction of clear skies. To address this, this thesis introduces a method to measure the visual impact of flying objects. Through specific Python algorithms, the research evaluates the visual annoyance caused by various Urban Air Vehicles and offers tools for introducing this new field of study. Furthermore, it proposes remote Visual Impact Assessments at multiple measurement stations using a specially configured Raspberry Pi. As part of the research project, an investigation was designed to explore the influence of a vehicle's position within the human Field of Vision (FOV) on its visual impact. Results show that the object's placement does not significantly change its perceived size and only its distance from the observer does. Additionally, a study on color contrast, which incorporates atmospheric scattering, has been conducted. This software-based approach does not need real-world scenarios, allowing for the examination of various Unmanned Air Vehicles colors under differing atmospheric conditions. Based on the investigated scenarios, an optimal color range for air vehicles intending to operate on sunny days with minimal visual disruption is presented.

# Acknowledgements

First and foremost, thanks to my tutor Katharina and her colleague Sila for giving me the opportunity to work on this challenging project and to Horyzn Team at TUM for letting me use their UAVs for my thesis.

Thank you to my parents and my sister Ana, for believing in me before I did in myself and for continuously providing strength and love. Thanks for taking care of me, even from so far away.

A big thank you to my colleagues and friends from Valencia and Munich with whom I have shared so many library hours, worries, and dreams. A special mention to Jaime, for his patience and kind words.

Lastly, thank you to my friend Ángela who has helped me so much in these four years of intense study, making the pursuit of Aerospace Engineering always worthwhile.

You all have been my team and without you, I would have never achieved this.

Gracias.

# Statement of Academic Integrity

I,

Last name: Molina Aragón

First name: Cristina

ID No.: 03770263

hereby confirm that the attached thesis,

Investigation of Visual Environmental Effects of Urban Air Vehicles

was written independently by me without the use of any sources or aids beyond those cited, and all passages and ideas taken from other sources are indicated in the text and given the corresponding citation.

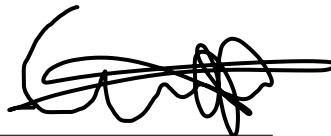
I confirm to respect the “Code of Conduct for Safeguarding Good Academic Practice and Procedures in Cases of Academic Misconduct at Technische Universität München, 2015”, as can be read on the website of the Equal Opportunity Office of TUM.

Tools provided by the chair and its staff, such as models or programs, are also listed. These tools are property of the institute or of the individual staff member. I will not use them for any work beyond the attached thesis or make them available to third parties.

I agree to the further use of my work and its results (including programs produced and methods used) for research and instructional purposes.

I have not previously submitted this thesis for academic credit.

Munich, September, 30, 2023



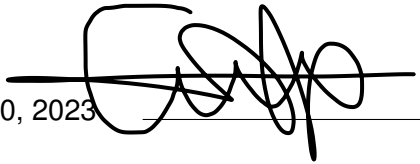
# Declaration for the transfer of the thesis

I agree to the transfer of this thesis to:

- Students currently or in future writing their thesis at the chair:
  - Flat rate by employees
  - Only after particular prior consultation.
- Present or future employees at the chair
  - Flat rate by employees
  - Only after particular prior consultation.

My copyright and personal right of use remain unaffected.

Munich, September, 30, 2023

A handwritten signature in black ink, consisting of several loops and a long horizontal stroke, positioned over a horizontal line.

# Contents

|   |            |
|---|------------|
| <b>Abstract</b>   | <b>ii</b>  |
| <b>Acknowledges</b>   | <b>iii</b> |
| <b>Statement of Academic Integrity</b>                                      | <b>iv</b>  |
| <b>Declaration for the transfer of the thesis</b>                           | <b>v</b>   |
| <b>List of Figures</b>  | <b>x</b>   |
| <b>List of Tables</b>   | <b>xi</b>  |
| <b>List of Acronyms</b>   | <b>xii</b> |
| <b>1 Introduction</b>   | <b>1</b>   |
| 1.1 Objective . . . . .   | 1          |
| 1.2 Procedural Method . . . . .   | 2          |
| <b>2 Theoretical Framework</b>  | <b>3</b>   |
| 2.1 Urban Air Mobility . . . . .  | 3          |
| 2.1.1 Introduction to Urban Air Mobility . . . . .                          | 3          |
| 2.1.2 Environmental Impacts of Urban Air Mobility . . . . .                 | 5          |
| 2.1.3 Social Acceptance of Urban Air Mobility . . . . .                     | 8          |
| 2.2 Factors Affecting the Visual Perception of Urban Air Vehicles . . . . . | 11         |
| 2.2.1 Estimation of Size . . . . .  | 12         |
| 2.2.2 Color Difference . . . . .  | 14         |
| 2.2.3 Atmospheric Scattering . . . . .                                      | 16         |
| 2.3 Regression model to estimate visual thresholds . . . . .                | 20         |
| 2.4 Human Field of View . . . . .   | 21         |
| <b>3 Methodology</b>  | <b>23</b>  |

|          |  |           |
|----------|--|-----------|
| 3.1      | Experimental Setup . . . . .   | 23        |
| 3.1.1    | Measurement Station . . . . .  | 23        |
| 3.1.2    | Raspberry Pi Camera Configuration . . . . .                                | 25        |
| 3.1.3    | Scenario Planning for Data Collection . . . . .                            | 27        |
| 3.2      | Determination of visual size . . . . .                                     | 30        |
| 3.2.1    | Detection and Recognition of the Air Vehicle . . . . .                     | 30        |
| 3.2.2    | Obtaining the Distance-to-pixel Ratio . . . . .                            | 32        |
| 3.2.3    | Obtaining the Camera-Vehicle Distance . . . . .                            | 33        |
| 3.2.4    | Applying Visual Size Formula . . . . .                                     | 34        |
| 3.3      | Determination of Absolute Color Difference . . . . .                       | 35        |
| 3.3.1    | Air Vehicle Isolation in Images Using GIMP . . . . .                       | 36        |
| 3.3.2    | Air Vehicle Isolation in Images Using Python . . . . .                     | 37        |
| 3.3.3    | Compute LAB Mean Values . . . . .  | 39        |
| 3.4      | Contrast Determination in the Presence of Atmospheric Scattering . . . . . | 40        |
| 3.5      | Digital Simulations of Scenarios using GIMP . . . . .                      | 42        |
| <b>4</b> | <b>Results and Discussion</b>  | <b>45</b> |
| 4.1      | Visual Size Assessment . . . . .   | 45        |
| 4.2      | Absolute Color Difference Variation using Flight Test Data . . . . .       | 52        |
| 4.3      | Color Contrast and Atmospheric Scattering Evaluation. . . . .              | 57        |
| 4.4      | Absolute Color Difference Simulation Results . . . . .                     | 59        |
| 4.5      | Limitations of the Study and Future Work . . . . .                         | 62        |
| <b>5</b> | <b>Conclusion</b>  | <b>64</b> |
| <b>A</b> | <b>Python Codes</b>  | <b>66</b> |
| A.1      | Raspberry Pi Camera Python Codes . . . . .                                 | 66        |
| A.1.1    | Raspberry Pi Camera - Code 1 . . . . .                                     | 66        |
| A.1.2    | Raspberry Pi Camera - Code 2 . . . . .                                     | 67        |
| A.1.3    | Raspberry Pi Camera - Code 3 . . . . .                                     | 68        |
| A.2      | Visual Size Python Codes . . . . .   | 69        |
| A.2.1    | Detection and scale computation - Code . . . . .                           | 69        |
| A.2.2    | Distance calculation - Code 1 . . . . .                                    | 71        |
| A.2.3    | Distance Calculation - Code 2 . . . . .                                    | 72        |
| A.2.4    | Computation of visual size - Code . . . . .                                | 72        |

|          |  |           |
|----------|--|-----------|
| A.3      | Absolute Color Difference Python Codes . . . . .                         | 73        |
| A.3.1    | Isolating object and background - Code . . . . .                         | 73        |
| A.3.2    | Mean LAB values and $\Delta E$ - Code . . . . .                          | 74        |
| A.4      | Cozman and Krotkov Formula Application Code . . . . .                    | 75        |
| <b>B</b> | <b>Flight Test Data Analysis</b>   | <b>76</b> |
| B.1      | Visual Size Assessment Procedure . . . . .                               | 76        |
| B.1.1    | Photographs with 0° orientation camera and 2m altitude . . . . .         | 76        |
| B.1.2    | Photographs with 0° orientation camera and 5m altitude . . . . .         | 77        |
| B.1.3    | Photographs with 30° orientation camera and 2m altitude . . . . .        | 77        |
| B.1.4    | Photographs with 30° orientation camera and 2m altitude . . . . .        | 78        |
| B.1.5    | Photographs with 0° orientation camera and 5m altitude (UAV 2) . . . . . | 79        |
| B.2      | Color Contrast Isolating object . . . . .                                | 79        |
| B.3      | Visual Size Numerical Results . . . . .                                  | 80        |



# List of Figures

|      |  |    |
|------|--|----|
| 1.1  | Procedural Method overview. . . . .  | 2  |
| 2.1  | CO <sub>2</sub> emissions by sector in the world from 1990 until 2020 . . . . .                    | 5  |
| 2.2  | Road transport emissions as a share of EU transport GHG emissions by mode in EU-27, 2019 . . . . . | 6  |
| 2.3  | Societal acceptance factors of UAM . . . . .   | 10 |
| 2.4  | Distopic landscape full of urban air vehicles . . . . .  | 10 |
| 2.5  | Visual size definition . . . . .   | 12 |
| 2.6  | Minimum resolution of human eye. . . . .   | 13 |
| 2.7  | Representation of LAB color space . . . . .  | 16 |
| 2.8  | Evaluation of the intensity of a ray along its path . . . . .                                      | 17 |
| 2.9  | Comparison of different levels of visibility . . . . .   | 19 |
| 2.10 | Vertical and horizontal FOV. . . . .   | 22 |
| 3.1  | Human neck flexion . . . . .   | 23 |
| 3.2  | Procedure to delete SSH Keys. . . . .  | 24 |
| 3.3  | UAVs provided by Horyzn. . . . .   | 28 |
| 3.4  | Scenario recreation seen from a lateral view. . . . .  | 29 |
| 3.5  | Scenario recreation seen from the view and marks on the ground. . . . .                            | 29 |
| 3.6  | Methodology to obtain visual size. . . . .   | 30 |
| 3.7  | Reference image. . . . .   | 32 |
| 3.8  | Methodology to obtain color contrast. . . . .  | 35 |
| 3.9  | Comparison of isolated UAV and background using GIMP. . . . .                                      | 37 |
| 3.10 | Comparison of isolated UAV and background using Python. . . . .                                    | 39 |
| 3.11 | Methodology to obtain atmospheric contrast. . . . .  | 41 |
| 3.12 | Sample images to calculate color contrast at different depths. . . . .                             | 42 |
| 3.13 | H-aero reference image . . . . .   | 43 |
| 3.14 | Simulation of boundary colors in H-aero using GIMP. . . . .  | 44 |
| 3.15 | Simulation of different blue colors in H-aero using GIMP. . . . .                                  | 44 |
| 3.16 | H-aero reference sky. . . . .  | 44 |
| 4.1  | Photographs at 3 m ground distance, 2 m altitude and 0° orientation. . . . .                       | 45 |
| 4.2  | Photographs at 20m ground distance, 2m altitude and 0° orientation. . . . .                        | 46 |
| 4.3  | Relation between real distance and visual size. . . . .  | 47 |
| 4.4  | Relation between lateral distance and visual size at 2 m altitude. . . . .                         | 48 |
| 4.5  | Relation between lateral distance and visual size at 5 m altitude. . . . .                         | 48 |
| 4.6  | Relation between dimensions and lateral distance at 2 m altitude (UAV 1). . . . .                  | 50 |
| 4.7  | Relation between dimensions and lateral distance at 5 m altitude. . . . .                          | 51 |
| 4.8  | UAV 1 with different backgrounds. . . . .  | 52 |
| 4.9  | Comparison of $\Delta E$ GIMP Method results. . . . .  | 54 |
| 4.10 | Difference in sky color due to atmospheric scattering - UAV 1. . . . .                             | 55 |
| 4.11 | Difference in sky color due to atmospheric scattering - UAV 2. . . . .                             | 55 |
| 4.12 | $\Delta E$ GIMP Method results for both UAVs. . . . .  | 56 |

|   |    |
|---|----|
| 4.13 Influence of distance and visibility on apparent contrast (UAV 1). . . . .                                   | 57 |
| 4.14 Influence of distance and visibility on apparent contrast (UAV 2). . . . .                                   | 58 |
| 4.15 Influence of higher distance and visibility on apparent contrast (UAV 2). . . .                              | 58 |
| 4.16 Influence of higher distance and visibility on apparent contrast (UAV 2). . . .                              | 59 |
| 4.17 Different colors and their associated apparent contrast at 10 m distance and<br>visibility of 16 km. . . . . | 61 |
|   |    |
| B.1 Photographs at 3m ground distance, 2m altitude and 0° orientation. . . . .                                    | 76 |
| B.2 Photographs at 10m ground distance, 2m altitude and 0° orientation. . . . .                                   | 76 |
| B.3 Photographs at 20m ground distance, 2m altitude and 0° orientation. . . . .                                   | 76 |
| B.4 Photographs at 10m ground distance, 5m altitude and 0° orientation. . . . .                                   | 77 |
| B.5 Photographs at 20m ground distance, 5m altitude and 0° orientation. . . . .                                   | 77 |
| B.6 Photographs at 3m ground distance, 2m altitude and 30° orientation. . . . .                                   | 77 |
| B.7 Photographs at 10m ground distance, 2m altitude and 30° orientation. . . . .                                  | 78 |
| B.8 Photographs at 20m ground distance, 2m altitude and 30° orientation. . . . .                                  | 78 |
| B.9 Photographs at 10m ground distance, 5m altitude and 30° orientation. . . . .                                  | 78 |
| B.10 Photographs at 20m ground distance, 5m altitude and 30° orientation. . . . .                                 | 79 |
| B.11 Photographs at 10m ground distance, 5m altitude and 0° orientation (UAV 2). . . . .                          | 79 |
| B.12 Case 1 isolating process. . . . .  | 79 |
| B.13 Case 2 isolating process. . . . .  | 79 |
| B.14 Case 3 isolating process. . . . .  | 80 |
| B.15 Case 4 isolating process. . . . .  | 80 |
| B.16 Case 5 isolating process. . . . .  | 80 |

# List of Tables

|     |  |    |
|-----|--|----|
| 2.1 | Categories of factors affecting visibility and visual contrast. . . . .      | 11 |
| 2.2 | Color representation systems. . . . .  | 15 |
| 2.3 | Observer's perception of color difference ( $\Delta E$ ). . . . .            | 16 |
| 2.4 | Linear-logit models for each model category. . . . .                         | 21 |
| 3.1 | Selected colors and their HTML notations. . . . .                            | 43 |
| 4.1 | Weather data on flight campaigns . . . . .                                   | 45 |
| 4.2 | Comparison of absolute color difference results for different cases. . . . . | 53 |
| 4.3 | $\Delta E$ results for different locations (UAV 1) using GIMP. . . . .       | 55 |
| 4.4 | $\Delta E$ results for different locations (UAV 2) using GIMP. . . . .       | 55 |
| 4.5 | Colors and their associated $\Delta E$ values. . . . .                       | 60 |
| 4.6 | Colors and their associated color contrasts. . . . .                         | 61 |
| B.1 | Visual size analysis results . . . . .                                       | 81 |

# List of Acronyms

|       |  |
|-------|--|
| CMY   | Cyan, Magenta, Yellow                          |
| EASA  | European Union Aviation Safety Agency          |
| EU    | European Union                                 |
| eVTOL | Electric Vertical Take-Off and Landing vehicle |
| FOV   | Field of View                                  |
| GHG   | Greenhouse Gas Emissions                       |
| GIMP  | GNU Image Manipulation Program                 |
| HSL   | Hue, Saturation, Lightness                     |
| HTML  | HyperText Markup Language                      |
| LAB   | Luminosity, a, b                               |
| NASA  | National Aeronautics and Space Administration  |
| PAV   | Personal Air Vehicle                           |
| RGB   | Red, Green, Blue                               |
| SBC   | Single-Board Computer                          |
| SSH   | Secure Shell                                   |
| UAM   | Urban Air Mobility                             |
| UAS   | Unmanned Aircraft System                       |
| UAV   | Unmanned Aerial Vehicle                        |
| UTM   | Urban Air Traffic Management                   |
| VIA   | Visual Impact Assessment                       |
| VNC   | Virtual Network Computing                      |
| VTOL  | Vertical Takeoff and Landing                   |

# 1. Introduction

By 2050, it is expected that 70% of the European and 80% of the North American population will reside in urban areas (United Nations, 2018). The rapid urbanization has created significant challenges in mobility and infrastructure expansion, leading to congestion, pollution, and conflicts that affect the environment and public health. Given these challenges, there is an urgent need to explore innovative urban transportation concepts. Urban Air Mobility (UAM) emerges as a promising solution to alleviate traffic congestion, offering the potential for a more efficient and sustainable mode of transport.

However, the successful integration of UAM into landscapes requires a comprehensive understanding of safety, noise, and visual disturbance. Predicting how this novel mobility concept will transform urban life is crucial to creating suitable vehicles and regulations that meet the needs and preferences of the public.

Among the many considerations to take into account, this thesis centres on the visual impact. It goes beyond mere appearances, playing a crucial role in shaping public perception and acceptance of this innovative transportation solution. Nonetheless, evaluating visual annoyance poses a unique challenge due to the diverse factors that shape it and its subjective characteristics.

## 1.1. Objective

The primary objective of this research is to determine the visual environmental impact caused by Urban Air Vehicles. This work proposes methods to measure such effects, which have never been explored previously. The initial step involves identifying the influential variables, followed by the development of an automated and efficient measurement method, by using software tools. To develop a Visual Impact Assessment of specific vehicles, scenarios will be simulated where the user observes a vehicle in flight. During these simulations, visual data will be collected using a Raspberry Pi<sup>1</sup>. Python will be employed to configure this module and incorporate it into an existing portable and adaptable measurement station.

---

<sup>1</sup>A Raspberry Pi Camera is a compact camera module designed for use with Raspberry Pi computers, which are single-board computers.

Considering societal acceptance and the constraints of Urban Air Mobility, evaluating its implications on the surrounding environment is crucial. The central research question is:

*How will the integration of urban air vehicles into megacity mobility systems affect the visual experience of the population?*

In order to address the central research question, the following three questions must be answered first:

1. *How can the visual impact of an object in flight be measured?*
2. *Which factors determine the extent of visual annoyance?*
3. *How can these factors be adjusted to mitigate the adverse effects of UAM?*

## 1.2. Procedural Method

This thesis starts with an introduction to UAM, highlighting its benefits and limitations. Subsequent chapters deal with the theoretical framework surrounding factors influencing visual impact and their effects on human perception. A detailed methodology to measure the visual influence of flying vehicles follows. Finally, the findings of the research are presented, together with conclusions and recommendations for further exploration in this domain. An overview is presented in Figure 1.1.



Figure 1.1.: Procedural Method Overview.

## 2. Theoretical Framework

This chapter provides an overview of the fundamentals of Urban Air Mobility, as well as the key factors influencing the human visual perception of these new vehicles.

### 2.1. Urban Air Mobility

This section reviews the theoretical framework of Urban Air Mobility, examining its current state of research, the challenges it must address before implementation, and the environmental and societal impacts it is anticipated to have.

#### 2.1.1. Introduction to Urban Air Mobility

The concept of Urban Air Mobility (UAM) is not new: it was first introduced in 1917 when Glenn Curtiss developed what he called a "flying car". While many inventors pursued similar dreams of flying automobiles, numerous technical limitations prevented them from achieving commercial viability (Cohen, Shaheen, and Farrar, 2021: 2). In the 1940s, helicopters began flying over urban areas and became the first vehicles with vertical take-off and landing (VTOL) capabilities. Yet, in the present day, helicopters are predominantly used for surveillance, emergency missions, and logistics but rarely for passenger transport due to factors such as accidents, high noise levels, and costs (Straubinger et al., 2020: 1).

With advancements in power electronics, electric aviation, unmanned systems, and high-performance batteries, numerous aerospace companies and startups have initiated the race to develop new aerial vehicles. In the mid-2000s, the idea of using drones for passenger transport began to be contemplated (NASA, 2001). By 2011, Thomas Senkel flew the *Volocopter VC1* prototype, which marked the first manned flight of an electric multicopter (Volocopter, 2011). The potential of distributed electric propulsion has led to the initiation of several UAM projects in European cities, and European Union Aviation Safety Agency (EASA) promises that within 5 to 10 years, Urban Air Vehicles will populate the city's skies (EASA, 2021a: 1).

According to EASA (2021a: 2), UAM has the potential to achieve significant benefits. This highlights the importance of developing a well-planned implementation strategy within society and are the following:

- Reducing standard city trip duration by 30 to 75%.
- Eliminating local air pollution caused by UAM operations.
- Significantly lowering the risk of fatal accidents during city travel compared to conventional road transport on a per passenger kilometer basis.
- Creating around 90,000 new job opportunities.

Additionally, UAM in the medical sector offers significant advantages by enabling rapid and efficient response in rescue missions, ensuring quick delivery of critical medical supplies, and facilitating the timely transportation of specialized medical personnel to areas in need. This approach not only reduces response times but also enhances the accessibility of vital medical resources in urban environments (Gillis et al., 2021: 417).

As this is a highly innovative field, no classification or regulation has been carried out by the relevant authorities. However, Straubinger et al. (2020: 3) proposes a classification of Urban Air Vehicles based on how they generate lift. It distinguishes between those with rotary-wing in cruise and those with fixed-wing in cruise. The latter are more efficient and faster during this phase of flight, and it states the requirements and important design drivers affecting the aircraft design for UAM.

On the other hand, there are other classifications based on the specific mission of the vehicle, such as Personal Air Vehicle (PAV)s like the two-passenger *Volocopter* by (Volocopter GmbH, 2017) or the *CityAirbus* for up to 4 passengers (CityAirbus, 2021). Furthermore, there are Air Taxi services available, offering on-demand aerial transportation for either individual passengers or small groups of travelers (Rajendran and Srinivas, 2020).

Furthermore, when discussing UAM, it is essential to consider not only the vehicles themselves but also all the necessary infrastructures for their implementation. This includes ground infrastructure, such as vertiports enabling Vertical Takeoff and Landing (VTOL) vehicles to take off and land in urban areas. Moreover, it deals with the establishment of a robust communication network to efficiently manage air traffic distribution (EASA, 2021a: 3). Therefore, a comprehensive set of regulations must be implemented to ensure the highest levels of safety and security during UAM operations within cities.



To summarize, the potential benefits of UAM are extensive and diverse. While the full extent of its transformative impact on urban air mobility is yet to be studied, the mobility of the 21st century will be shaped by the convergence of aviation and technological innovation in urban skies.

### 2.1.2. Environmental Impacts of Urban Air Mobility

Urban Air Mobility pretends to be a new transportation system and, therefore can potentially bring numerous environmental impacts never studied before. This section examines the environmental consequences of UAM, exploring both the positive and negative aspects, and underlines the importance of striking a balance between technical innovation and sustainability.

Concerning gas emissions, UAM vehicles are expected to be electric, reducing air pollution, especially compared to vehicles powered by fossil fuels. In 2019, *Energy Statistics Data Browser (2022)* estimated that around 8267 million tons of Greenhouse Gas emissions (GHG) were caused by the transportation sector, accounting for approximately 24% of the total emissions for that year worldwide. Figure 2.1 shows the progression of CO<sub>2</sub> emissions by sectors and time.

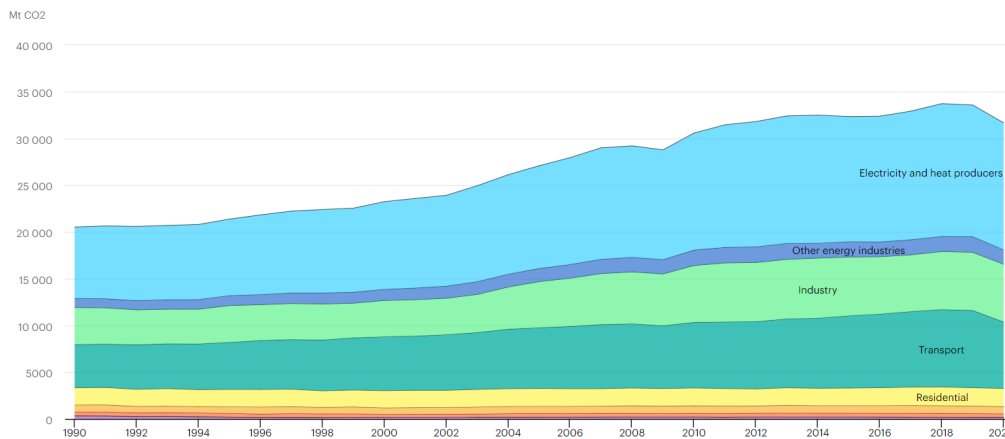


Figure 2.1.: CO<sub>2</sub> emissions by sector in the world from 1990 until 2020 (*Energy Statistics Data Browser 2022*).

In the EU, the transportation sector was responsible for about a quarter of total CO<sub>2</sub> emissions, of which road transport accounted for 71.1% (European Environment Agency, 2021: 17), as shown in Figure 2.2. Therefore, there is a need to reduce road congestion to reduce GHG emissions, and UAM will positively impact the environment by providing a sustainable solution.

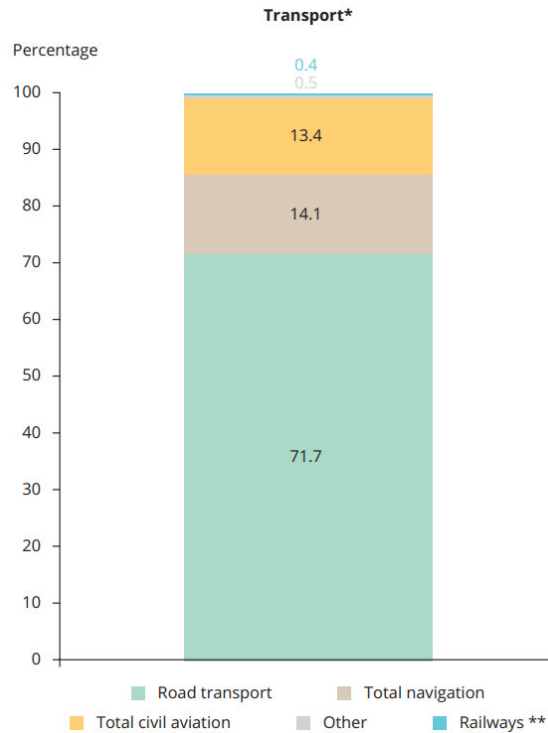


Figure 2.2.: Road transport emissions as a share of EU transport GHG emissions by mode in EU-27, 2019 (European Environment Agency, 2021: 18).

Kasliwal et al. (2019: 3) studied the environmental impact of Electric Vertical Take-Off and Landing vehicle (eVTOL)s in the United States in 2020. Their study found that an eVTOL with only the pilot on board resulted in a 35% reduction in GHG emissions compared to a gasoline-powered vehicle with a single occupant. However, the same study also revealed that the eVTOL would produce 28% more emissions than a battery-powered electric vehicle. Therefore, the key point is to increase the number of passengers in eVTOL flights to reduce emissions, enhance efficiency, and make UAM more competitive.

Even though electric aircraft for UAM seem promising and are generally quieter than traditional helicopters or airplanes, noise pollution is still a big concern. This concern increases during take-off and landing (see Section 2.1.3 for more details). Managing and reducing noise in natural environments is quite challenging because noise is a complex concept that does not have a precise definition. It involves multiple dimensions and variables, and its impact can vary greatly depending on the specific circumstances and surroundings. To properly analyze noise in this context, it is important to consider environmental and non-sound-related factors as well (Connors, 2019: 12).

Unlike conventional aircrafts, future air vehicles operate at lower altitudes and closer to urban areas, potentially annoying residents. When establishing the noise acceptance of a system, the level of “annoyance” is commonly used, and it is defined as “a global description of a stimulus that is clearly perceived as negative but without analysis of the cause of the negativity” by Connors (2019: 2). For instance, according to EASA (2021b: 3), noise annoyance is proportional to the degree of familiarity with the sound. Therefore, sounds commonly heard in urban environments at similar decibel levels tend to be more tolerable by humans. Moreover, noise is closely linked to depression and anxiety and has been investigated in commercial aviation (Beutel et al., 2016), but little research has been done on the acoustic impact of UAM.

Conversely, Urban Air Vehicles may have a negative impact on animals. According to EASA (2021b: 75), 62% of people are concerned about how drones can annoy wildlife. Notably, as more such vehicles occupy the skies, there is a rising concern about them colliding with birds and potentially upsetting ecological balances. Kwon, Kim, and Park (2017: 8) conducted a study on how can Unmanned Aircraft System (UAS) disturb wildlife and found that smaller UAS, electric engines and lawn-mower flight generally do not evoke disturbances to birds and can be comparable to the hazard caused by natural predators. However, when UAS collide directly and at close range with wildlife or sensitive structures such as nests, they have the potential to cause greater disturbance to wildlife.

The visual impact of UAM is a significant aspect to consider. Introducing the eVTOL aircraft into the urban airspace can potentially alter the skyline and traditional visual landscape of cities. As with acoustic pollution from Urban Air Vehicles, the visual annoyance aspect has received limited research attention and is intended to be the primary focus of this thesis. In one of the few articles addressing this topic, Kwon, Kim, and Park (2017) examined the general feeling towards drones. They found that possibly one of the causes that make Unmanned Aerial Vehicle (UAV)s visually disturbing is the obstruction of the field of view and the shadows they create. Very related to social acceptance, Yedavalli and Mooberry (2019) shows that 45% of respondents are concerned about visual pollution, considering the skies will be flooded with new and unfamiliar vehicles. The same study recommends a phased introduction rather than an immediate influx of numerous aircraft in the sky. Furthermore, a recent article by Thomas and Granberg (2023) found that a UAV is equally visually disruptive in urban and rural areas, indicating that the type of environment does not matter in this context.

According to Said et al. (2021: 99) and Thomas and Granberg (2023), visual pollution does not only involve environmental impacts but also has the potential to lead to health concerns, including:

- Increased stress and anxiety levels.
- Distraction and diminished concentration.
- Overwhelming volume of simultaneous data.
- Hazardous distractions.
- Diminished productivity.
- A negative emotional state and mood disorders.

### **2.1.3. Social Acceptance of Urban Air Mobility**

The social acceptance of UAM holds great significance in its integration into urban transportation systems. As using aerial transportation within cities gains importance, understanding the factors that drive social acceptance becomes crucial.

While it is true that UAM holds the promise of being a safe, reliable, and sustainable solution to alleviate road congestion in densely populated cities, social acceptance may serve as a potential barrier to the development of such vehicles. As unmanned aerial vehicles are increasingly prevalent in our daily lives and utilized in various contexts, such as search and rescue missions, aerial photography, and package delivery, many studies and surveys have been conducted to assess societal response.

A study by Hasan (2019: 23, 85) concluded that almost half of the respondents supported UAM as a delivery service, and more than 70% reported they would be comfortable with others using air taxi services in the future. However, significant concerns revolved around five categories: safety, privacy, loss of conventional jobs, environmental threats and noise, and visual disruption.

On the other hand, Yedavalli and Mooberry (2019: 3) conducted a survey on the perception of UAM in four geographies worldwide: Los Angeles, Mexico City, New Zealand, and Switzerland. They revealed that 44.5% of respondents expressed support for UAM and 41.4% thought it was safe or very safe. Back then, these surveys suggested that the initial impression of UAM was favourable.

EASA (2021b: 63) published a study titled "Study on the societal acceptance of Urban Air Mobility in Europe", which detailed the most important drivers of societal acceptance for UAM across different cultures and regions within the European Union, including perceived benefits and concerns. According to the report, 83% EU citizens initially demonstrated a positive attitude and genuine interest in UAM, perceiving it as a new and possible form of transportation. Furthermore, most individuals expressed willingness to experiment with UAM. However, it also concludes that when they want to contemplate the outcomes of possible UAM activities within their city, European Union residents desire to restrict their personal exposure to various risks. Among these risks, safety, noise levels, security, and environmental effects play an important role (EASA, 2021b: 3).

Given that it is a new field and its integration into the civilian world raises concerns, Hasan (2019: 37) has proposed an "Air Traffic and Fleet Operations Management and Community". This proposal outlines the regulations and responsibilities required for successful implementation. These include "Operator certification", which is intended to be an evolution of existing operator certifications, and "UAM traffic management and airspace integration", which must address Urban Air Traffic Management (UTM), along with the necessary technical specifications, operating protocols, and infrastructure required for these vehicles to coexist in the sky. Additionally, mention is made of "Noise requirements" that should be established by local and federal governments to define allowable noise levels. These are just the potential implementations to which a component related to the visual effects on urban and rural landscapes should be added.

EASA (2021b: 16) sets environmental impact as the fifth most significant challenge (making up 7.5% of the total challenges) when it comes to implementing UAM. The term "environmental impact" is commonly employed in the literature and encompasses a wide range of subjects, including noise, visual pollution, air pollution, land utilization, and the preservation of species. Regarding noise footprint and its concern to society, EASA (2021b: 76) concluded that 52% of the respondents were worried about noise pollution from drones, and 53% were concerned about this impact from air taxis. This is shown in Figure 2.3, which also shows that visual annoyance ranks among the top five factors causing concern.

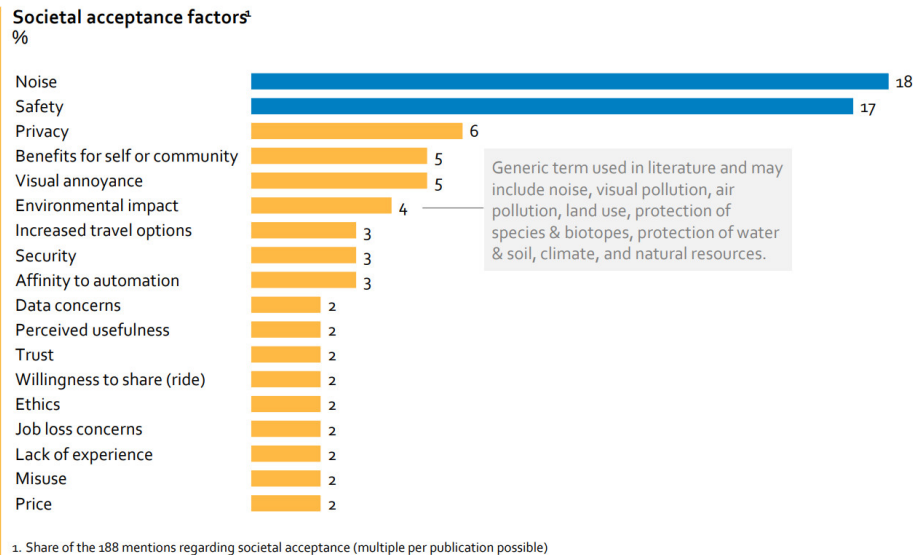


Figure 2.3.: Societal acceptance factors of UAM (EASA, 2021b: 18).

Based on EASA (2021b: 71), 19% of the people surveyed consider visual pollution as one of their top three concerns regarding delivery drones, while 16% have air taxis in their top three concerns. Additionally, the same research revealed that the visual impact, particularly on cultural heritage sites in historic European cities, is a significant concern for many (EASA, 2021b: 76).

Moreover, social scientists argue that drones can be seen as usurpers that take over people's right to the city and the air (as represented in Figure 2.4). In popular literature and the media, dystopian urban environments are often portrayed as spaces crowded with small aircraft, which could influence public opinion and the acceptance of UAMs in the real world (Shaw, 2016: 26). This perspective is similarly asserted by Chmielewski (2020), where it is argued that a visual pollution is a form of spatial disorder.

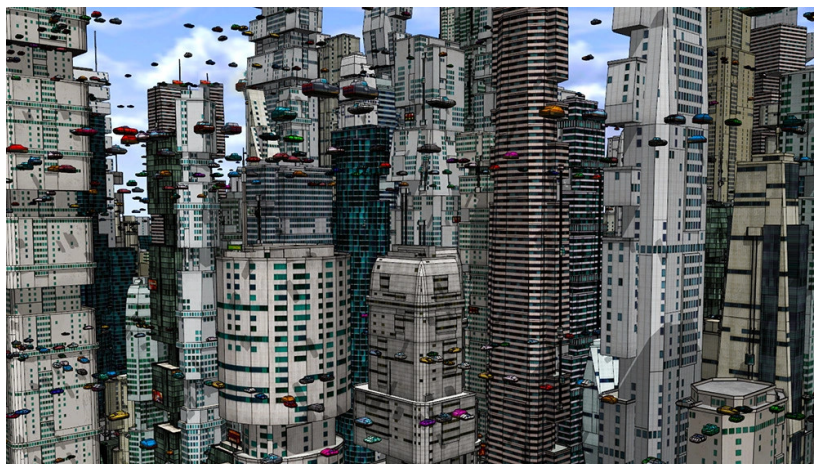


Figure 2.4.: Distopic landscape full of urban air vehicles (Empire, 2016).

Overall, comprehending the social acceptance of UAM is critical for its successful integration. Safety, noise, and visual nuisance significantly influence general beliefs. By thoroughly measuring and evaluating these factors, policymakers can address concerns related to UAM, promote its acceptance, and facilitate its harmonious integration into urban environments.

## 2.2. Factors Affecting the Visual Perception of Urban Air Vehicles

Measuring visual impact is intricate, with minimal research dedicated to the subject. However, earlier projects, such as wind turbines or solar panel installations, initiated the concept of Visual Impact Assessment (VIA) in the early part of the century. These assessments incorporated visual simulations, descriptions of potential impacts, and mitigation measures to alleviate visual repercussions. In the aforementioned instances, the landscape underwent permanent alteration. This contrasts Urban Air Vehicles, which are constantly moving, further complicating visual impact assessment.

Firstly, the difference between visual contrast and visual impact defined by Sullivan and Meyer (2014: 24) is crucial. Visual contrast refers to the changes observed by the viewer. Nevertheless, visual impact combines the observed changes and the resulting human response. Sullivan and Meyer (2014: 17) also outlined eight categories of factors affecting the visibility and visual contrast of an introduced object in a landscape, as summarized in Table 2.1.

Table 2.1.: Categories of factors affecting visibility and visual contrast.

| <b>Factor</b>                 | <b>Description</b>  |
|-------------------------------|---|
| Viewing geometry              | Refers to the viewer's spatial relationship with the object, incorporating both vertical and horizontal perspectives. |
| Distance                      | The space separating the viewer and the object.   |
| Object visual characteristics | The object's inherent visual properties.  |
| Lighting factors              | Pertains to sunlight distribution on the object.  |
| Backdrop                      | Visual background against which the element is seen.  |
| Atmospheric conditions        | Involves elements like haze and humidity.   |
| Viewer characteristics        | Factors such as individual visual acuity and experience.  |
| Viewshed limiting factors     | Variables that might compromise the precision of identifying visible areas from a particular location.                |

Within this thesis's framework, the sections related to size estimation will explore viewing geometry, distance, and object visual characteristics. Examining color contrast will assess lighting factors, backdrop, and even atmospheric conditions. The exploration of atmospheric scattering will cover atmospheric conditions. Lastly, an overview of the human field of view and its thresholds will study viewer characteristics.

### 2.2.1. Estimation of Size

When determining an object's visual effect, size is a critical component. Perceived size naturally changes with distance and viewing angle. For instance, viewing an object from above offers a complete view of its size and structure. Viewing the same item from a given angle in its lateral perspective may distort its proportions and alter its apparent size. Also, the further the objects move, the smaller they look and the less attention they capture.

Given these considerations, discussing visual size, also referred to as visual magnitude, is more convenient than the object's physical size. Visual size, denoted as  $S$ , refers to the proportion of the human field of view taken up by an object. It is determined by calculating the product of two angles,  $\alpha$  and  $\beta$ , which lie on the horizontal and vertical planes, respectively. Therefore, visual magnitude is an areal measurement shown in Figure 2.5 and is typically measured in square arc minutes ( $arcmin^2$ ).

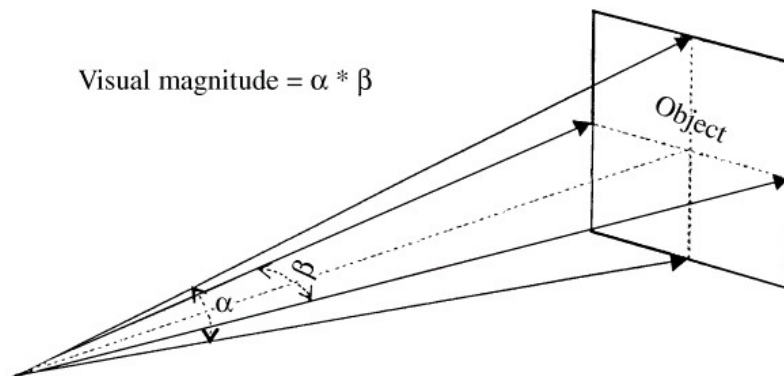


Figure 2.5.: Visual size definition (I. D. Bishop, 2002: 3).

Measuring angles  $\alpha$  and  $\beta$  can be a complex task. However, visual size can still be computed by obtaining and using the horizontal and vertical dimensions of the target object, as indicated in Equation 2.1 (Garnero and Fabrizio, 2015: 6). This equation considers the distance from the observer to the object. Additionally, the viewing geometry (i.e. the horizontal and vertical dimensions) and the object's visual characteristics are accounted for.



$$S = \beta \cdot \alpha = \left( \frac{180^2 \cdot 60^2}{\pi^2} \right) \cdot \frac{D \cdot H}{d^2}. \quad (2.1)$$

For a noticeable object, the apparent size must surpass a specific visual magnitude, also known as visual threshold. In order to assess the spatial resolution of the human eye, visual acuity is employed as a measurement tool. Visual acuity quantifies a person's ability to discern fine details. It is merely a size measurement and does not consider target contrast (see Section 2.2.2). This magnitude is generally measured under ideal lighting conditions, and a value between 0.8 and 1.6 arc minutes is considered a normal human acuity. Moreover, visual acuity also depends on the object structure, lines, lighting and aberrations of the observer's eye (Holladay, 2004: 3).

For two points to be perceived as distinct, at least one unstimulated cone must lie between two stimulated cones on the human retina (ALPF, 2022). Knowing that the distance between cones in the foveola is  $5\mu$  and that the focal length is 17 mm, the minimum resolution of a human eye with normal vision is one minute of an arc and is represented in Figure 2.6 (Caltrider, Gupta, and Tripathy, 2023).

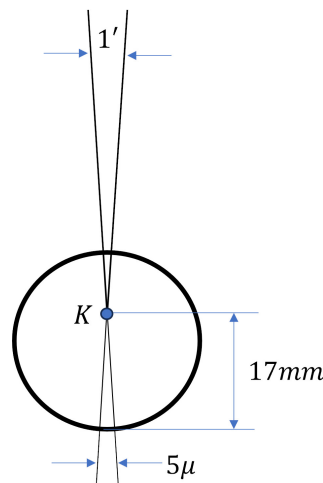


Figure 2.6.: Minimum resolution of human eye.

Although humans can perceive details of 1 minute of arc ( $1'$ ), they cannot identify them, and these details will not have a visual impact. To be recognized and have a noticeable visual relevance, an object must typically exceed a visual magnitude of approximately 5.5 minutes of arc ( $5.5'$ ). The visual magnitude required for recognition can vary depending on factors such as the type of object and the viewing conditions (Foley and Matlin, 2009).

On the other hand, the study conducted by I. D. Bishop (2002) investigates the perceived size of an object undergoing continuous shape changes, specifically a wind turbine. This wind turbine has a fixed structure and three rotating blades at a predetermined angular velocity. In the report, Bishop conducted an online survey where participants were asked to compare the size of objects in different positions. The objective of this survey was to gather information on how the size of the wind turbine was perceived based on its configuration. Based on the collected data, Bishop concluded that the most appropriate approach to determining the perceived size of the wind turbine was to consider both the size of the fixed part and the size of the moving part. Additionally, he proposed adding a rotation factor equivalent to 20% of the size of the moving part, thus acknowledging the visual impact of the rotating blades on size perception. The findings from this study provide valuable insights into how the perceived size of an object undergoing continuous shape changes can influence visual perception. These findings are also relevant for application in other contexts, such as the case of aerial vehicles that may have a transforming configuration if they possess blades.

### 2.2.2. Color Difference

The contrast between the introduced element in a landscape and its environment is a significant object-dependent variable when discussing visual impact and can be determined using image analysis. *Contrast* refers to the difference in lightness (or greyscale) between two points. When hue and saturation are also considered, the most appropriate term is *color difference* (Mokrzycki and Tatol, 2011: 4).

While the way individuals perceive the color of objects can differ and is influenced by factors such as lighting and atmospheric conditions, the color contrast between an object and its surroundings can still be determined by employing a relative difference calculation. This calculation considers representative observers and the specific lighting conditions present (Magill and Litton, 1986: 45-54).

To calculate the color difference, a color space model must first be chosen. A color model is a mathematical or conceptual representation in space that describes how colors can be created, mixed, and represented. A coordinate system is established to specify colors objectively and reproducibly, assigning them a numerical value or combination of numerical values. Different color descriptions are used in various applications, but the most common ones are described in Table 2.2.

Table 2.2.: Color representation systems (Skrok, 2022).

| System                           | Description   |
|----------------------------------|---|
| Red, Green, Blue (RGB)           | Colors are created by mixing different proportions of the primary colors, and each component is represented by a value ranging from 0 to 255, indicating the intensity of each primary color.   |
| Cyan, Magenta, Yellow (CMY)      | Colors are generated by subtracting amounts of cyan, magenta, and yellow from a white light source. Each component is represented by a value ranging from 0 to 100, indicating the amount of ink used.  |
| Hue, Saturation, Lightness (HSL) | Hue represents the chromatic position or hue, saturation indicates color purity, and lightness controls the amount of light. Hue values are typically represented on a 360-degree circle, while saturation and lightness values are represented on a scale from 0 to 100. |
| Luminosity, a, b (LAB)           | Luminosity ( $L$ ) indicates brightness, while the components $a$ and $b$ represent the color's position on the red-green and yellow-blue axes, respectively.   |

While the HSL model is intuitive, the preferred color description used in any VIA is LAB. This model is based on human visual perception, making it highly reliable. Moreover, LAB has been tested in landscape applications and has consistently shown superior performance, especially in scenarios with low contrast levels (I. D. Bishop, 2021).

As mentioned earlier, in the LAB color system, the  $L$  component represents the lightness of a sample, ranging from white to black. The variables  $a$  and  $b$  indicate the position of the sample on the red-to-green and blue-to-yellow axes, respectively. Therefore, using these three variables, the system creates a three-dimensional space with a specific color represented by its  $(L, a, b)$  coordinates as shown in Figure 2.7. The color difference ( $E$ ) between two colors is measured as the distance between their respective points,  $(L_1, a_1, b_1)$  and  $(L_2, a_2, b_2)$ , and is calculated using a three-dimensional application of the Pythagoras' theorem and seeing in Equation 2.2. Therefore,  $\Delta E$  represents how far are the two colors.

$$\Delta E_{ab}^* = \sqrt{(\Delta L^*)^2 + (\Delta a^*)^2 + (\Delta b^*)^2} \quad (2.2)$$

A standard observer may not always perceive the difference in color. Based on verified statistics (Mokrzycki and Tatol, 2011: 15), the thresholds described in Table 2.3 have been achieved for human vision, where  $\Delta E$  is a non-dimensional measure.

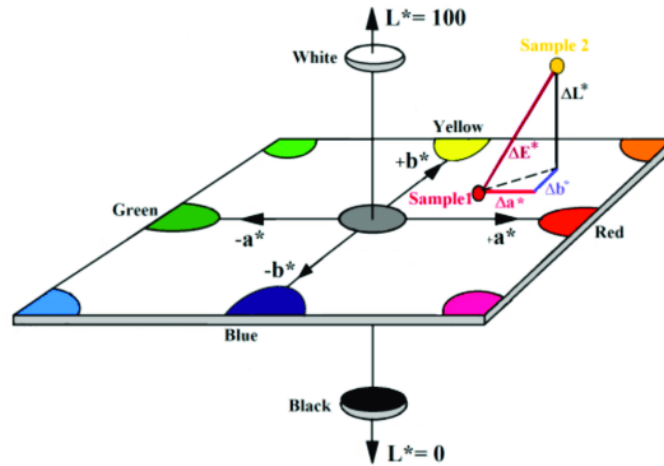


Figure 2.7.: Representation of LAB color space (Jnido, Ohms, and Viöl, 2019: 3).

Table 2.3.: Observer's perception of color difference ( $\Delta E$ ).

| Range                | Perceived Difference                                 |
|----------------------|--|
| $0 < \Delta E < 1$   | Observer does not notice the difference.             |
| $1 < \Delta E < 2$   | Only experienced observer can notice the difference. |
| $2 < \Delta E < 3.5$ | Unexperienced observer also notices the difference.  |
| $3.5 < \Delta E < 5$ | Clear difference in color is noticed.                |
| $\Delta E > 5$       | Observer notices two different colors.               |

### 2.2.3. Atmospheric Scattering

Atmospheric scattering is a natural occurrence arising from the interaction between sunlight and particles in the atmosphere. During daylight hours, the sky predominantly appears blue. However, during sunset, the color of the sky takes on a more reddish tone, particularly near the horizon. Rayleigh (1871) and Mie (1908) have described the behaviour of the light-particle interaction that produces these colors. Rayleigh scattering refers to the behaviour of light when it interacts with microscopic particles, which make up the bulk of particles in the atmosphere. On the other hand, Mie scattering is a phenomenon that describes the behaviour of light when it interacts with any type of particle. Rayleigh scattering is mainly responsible for the blue color of the sky, while Mie scattering is mostly used to describe the interaction with larger particles such as haze (Lopes and Ramires Fernandes, 2014).

Because of the considerable distance between the sun and the earth, it is presumed that light rays propagate in parallel and undisturbed through the vacuum until they reach the earth's atmosphere. Once they penetrate the atmosphere, these rays interact with atmospheric particles, with scattering being the most significant interaction in terms of color. Scattering occurs when a photon's electromagnetic field encounters the electric field of a particle in the atmosphere, causing the photon to be redirected in a different direction. Figure 2.8 shows the beam's path from entering the atmosphere to reaching the camera, which explains the phenomenon of scattering.

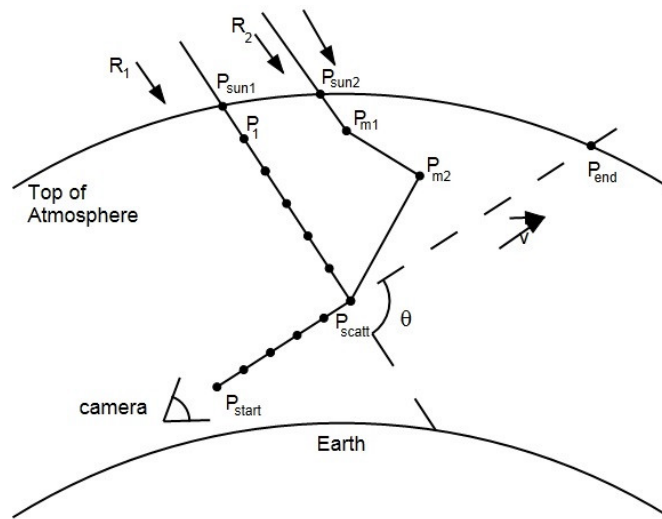


Figure 2.8.: Evaluation of the intensity of a ray along its path (Lopes and Ramires Fernandes, 2014: 2).

Figure 2.8 shows how some photons scatter in different direction and loose intensity once light from the sun and particles in the atmosphere interact between each other. While deflecting light into a specific direction (in this case, towards the camera) is called in-scattering, the loss of intensity is called out-scattering and represents the scattering of photons and their change in their original path. If the total intensity of light reaching a specific position wants to be calculated, all the in-scattering contributions following that direction must be summed up (Lopes and Ramires Fernandes, 2014: 2).

Particles in the atmosphere can be assumed to be spherical or very small, and therefore, light is scattered symmetrically concerning incident rays of light (Klassen, 1987). In this way, the portion of the light that is scattered will be a function of the angle between the incident ray of light and the emanating ray of light, represented by  $\theta$  in Figure 2.8 and the wavelength of light,  $\lambda$ . This phenomenon is commonly called the angular scattering function (Connors, 2019).

Nevertheless, according to Lopes and Ramires Fernandes (2014: 2), differences across wavelengths are minimal and become distinctly noticeable only over vast distances. Thus, it will be asserted that there is no dependency on wavelength. Finally, the formula computed by Cozman and Krotkov (1997) for the measured intensity of an object is referred to as the *Cozman and Krotkov formula*:

$$C' = C \cdot \exp(-\beta \cdot d + S \cdot (1 - \exp(-\beta \cdot d))). \quad (2.3)$$

Where  $C'$  is the measured intensity of an object,  $C$  is the intensity of the object without scattering,  $S$  is the sky intensity,  $\beta$  is the extinction coefficient (which increases as scattering increases), and  $d$  is the distance of the object from the viewer.

If the values for the intensity of the landscape and the object at a distance 0 are known ( $C_{1_0}$  and  $C_{2_0}$ ), the initial contrast at  $d = 0$  can be calculated. Knowing the possible range (256), it can be calculated as a percentage:

$$C_i = (C_{1_0} - C_{2_0}) \cdot \frac{100}{256}. \quad (2.4)$$

Therefore, the perceived contrast is:

$$C_d = C_i \exp(-\beta \cdot d). \quad (2.5)$$

Where  $C_d$  is the perceived contrast at depth  $d$ ,  $C_i$  is the initial contrast at  $d$  zero,  $d$  is the distance (m), and  $\beta$  is the extinction coefficient.

The extinction coefficient of light, denoted as  $\beta$ , is a scientific measure that characterizes diminished visibility. It quantifies the decline of light for each unit of distance due to the interplay of scattering and absorption by atmospheric gases and particulates from the light's source to its destination (Malm, 1983: 50). Although  $\beta$  values have been determined for various fog conditions, they may not universally apply to all experimental conditions (McCartney, 1976). Scattering varies significantly with the density and type of particles in the atmosphere, making the precise measurement of  $\beta$  complex.

According to the research conducted by (Chang, Song, and Liu, 2009: 3), visibility and extinction coefficient exhibit an inverse relationship, which is ruled by the *Koschmieder* constant, denoted as  $K$ . The value of  $K$  is influenced by factors such as the observer's eye sensitivity, the contrast between reference objects and the sky, and the presence of visual targets (Chang, Song, and Liu, 2009: 3). In their study of visibility in Chinese cities, they employed a specific value of  $K$ , namely 1.932, establishing the following relationship also used by I. D. Bishop (2019) when studying the impact of wind turbines:

$$\beta = 1.932/visibility. \quad (2.6)$$

As observed in Equation 2.6, visibility plays a vital role in estimating visual effects. Visibility refers to the clarity or distinctness with which objects can be seen. The presence of fine particles and gaseous air pollution in the atmosphere can create a visual phenomenon known as haze, which can obscure landscapes. While natural factors such as dust and wildfire smoke can contribute to its formation, it is predominantly caused by human-generated air pollution (Malm, 1983). The adverse effects of haze on visibility are attributed to its ability to scatter and absorb light within the atmosphere. Consequently, haze can limit the distance at which objects are detected and diminish the ability to perceive the vibrant colors, distinct shapes, and textures of a scenic vista, as compared in Figure 2.9. Therefore, visibility is shaped by both the tangible interactions of light with atmospheric gases and particles and the psychological perceptions of human viewers, and it is of high importance when assessing visual impact.



(a) Good level of visibility.



(b) Reduced level of visibility.

Figure 2.9.: Comparison of different levels of visibility (Town, 2023).

### 2.3. Regression model to estimate visual thresholds

Shang and Bishop (2000) investigated visual thresholds in landscape research in their paper titled “Visual thresholds for detection, recognition, and visual impact in landscape settings”. In this paper, they developed models that allow obtaining the visual thresholds by using different physical properties.

They defined the visual threshold as the minimum amount an individual can notice. That means, the threshold where one goes from not detecting something to detecting it. Additionally, they distinguished between detection, recognition, and visual impact. Detection refers to the ability to perceive a stimulus, recognition involves identifying its nature, and visual impact relates to its influence on a scene, landscape, or the observer. They considered visual size (explained in subsection 2.2.1) and visual contrast (explained in subsection 2.2.2) as variables. They also classified the objects based on their shape, using a transmission tower to represent a linear object shape and an oil refinery tank to represent a square-to-round object shape.

Following Dember (1960), they established that regular shapes are easier to perceive and provide a stronger stimulus to the observer. Consequently, they concluded that using regular shapes would provide more conservative results regarding threshold distances. In their report, they conducted two types of questionnaires for users:

- An uninformed detection test: where no information about the nature of the objects was provided, and participants were asked if they found any difference between two photos.
- An informed recognition test: where users were given information about the nature of the objects and the visual experience they would undergo, and they were directly asked if they detected an introduced element.

This led to four visual testings: uninformed detection, uninformed recognition, informed recognition and informed visual impact. Using the obtained results, they conducted a series of statistical models. In this case, it was a logistic regression analysis, as they aimed to predict whether an event would occur or not, and the probability of its occurrence for multiple variables. The linear logical model of this experiment is expressed as:

$$Z = B_0 + B_1(C \cdot S) + B_2(CD) + B_3(SH) + B_4(S) + B_5(C) \quad (2.7)$$



Where  $C$  represents visual contrast,  $S$  visual size,  $CD$  visual contrast direction,  $SH$  object type, and  $B_n$  the estimated coefficients obtained from the experiment.

Table 2.4.: Linear-logit models for each model category (Shang and Bishop, 2000: 9).

| Model Category         | Linear-logit Model   |
|------------------------|--|
| Uninformed detection   | $Z_{ud} = -16.02 + 0.0124(CS) + 12.75(D) + 1.525(H)$           |
| Uninformed recognition | $Z_{ur} = -7.56 + 0.0017(CS) + 5.014(D) - 1.623(H) + 0.037(S)$ |
| Informed recognition   | $Z_{ir} = -30.7336 + 0.0089(CS) + 27.971(D)$                   |
| Informed visual impact | $Z_{imp} = -19.7861 + 0.0045(CS) + 14.457(D)$                  |

Where  $SH$  and  $CD$  are binary indicator variables:  $SH = 1$  (tank);  $SH = 0$  (tower);  $CD = 1$  (positive contrast);  $CD = 0$  (negative contrast).

Once the values of  $Z$  have been obtained, the linear-logit model formula can be applied (Equation 2.8), expressing the probability that any individual detects an object. Finally, Shang and Bishop (2000) defined the impact threshold as “the physical condition of an object at which 50% of the viewers’ impact assessments, based on original and altered scenes shown side-by-side, exceed the middle position between low and high visual impact levels” and is defined by Equation 2.8.

$$P(event) = \frac{1}{1 + e^{-Z}} \quad (2.8)$$

## 2.4. Human Field of View

The Field of View (Field of View (FOV)) refers to the extent of visual area that can be perceived instantly, measured in degrees of angles. For humans, the instantaneous monocular (referring just to one eye) FOV is about  $100^\circ$  on the temporal side and about  $60^\circ$  on the nasal side, due to the nose position and the sclera and iris of the eye (Ellis et al., 2002: 9).

The binocular field refers to the area where the visual fields of both eyes overlap. It covers around  $120^\circ$  and varies based on the observer’s visual capability. Within this area exists the “central field”, which spans approximately the first 5 degrees from the fixation point. In this specific area, vision is more precise and detailed, due to the high concentration of photoreceptor cells in the fovea of the retina. Furthermore, there is the “Preferred Viewing Area”, which covers around  $30^\circ$  ( $15^\circ$  per eye) and an immediate FOV of  $70^\circ$  ( $35^\circ$  per eye).

Beyond these angles, the peripheral vision takes over (Torrejon, Callaghan, and Hagra, 2013: 8). All this data refers to the instantaneous field of view, which corresponds to the human FOV without any movement, neither of the eyes nor the head.

Humans generally have a total field of vision of slightly over 200° when combining the vision from both eyes and accounting for the overlap. This is depicted in Figure 2.10a, where the monocular FOV are not simply summed because there is an overlap. The combination of the temporal vision of one eye, the nasal vision of the other and the overlap determine the total FOV (Roberts and Osborne, 2019: 7).

On the other hand, the vertical field of view is also relevant. The upper limit is 50° from the standard line of sight, and the lower limit is 70°, as illustrated in Figure 2.10b. Summing both numbers, the normal eye vision field oscillates between 120° and 135° since the anatomy of the face constrains it. However, the limit of optimal color discrimination is located 25° upwards and 30° downwards (Torrejon, Callaghan, and Hagra, 2013: 8).

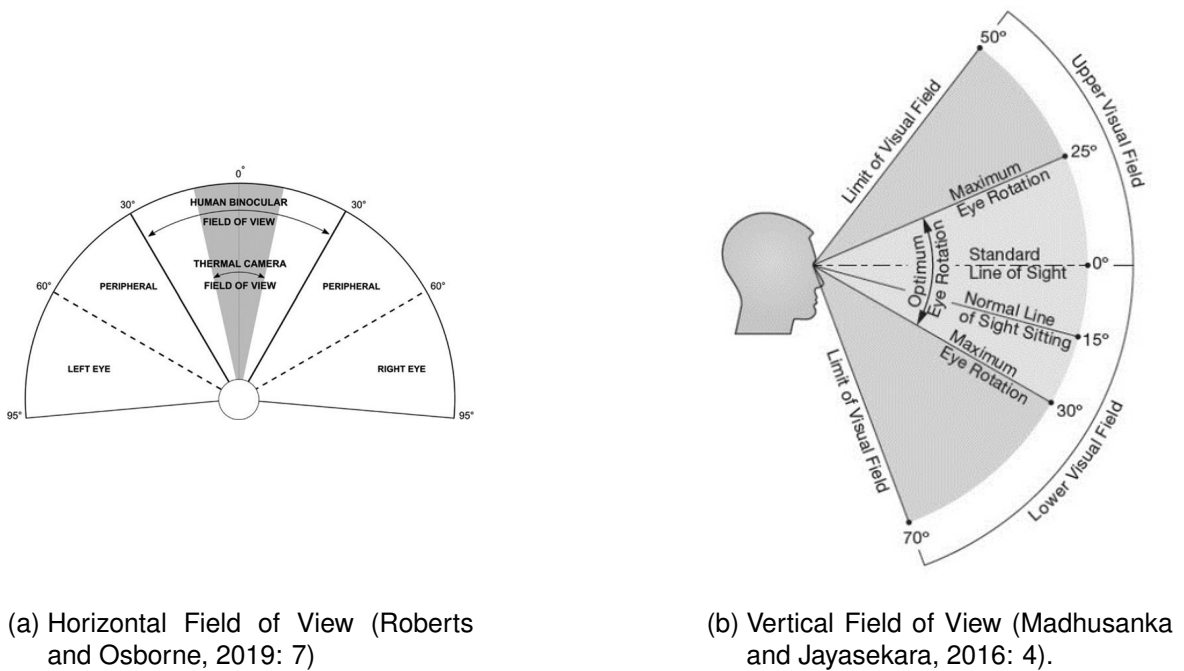


Figure 2.10.: Vertical and horizontal FOV.

While both the horizontal and vertical FOVs are important, the horizontal FOV is frequently highlighted, as it is considered more significant according to Thorpe Davis (1997). Moreover, any work that aims to reconstruct the visual field or present an item as “real” should focus on the binocular area, especially in contexts such as virtual reality and visual simulations, where virtual content is designed to interact with the consumer vision of the real-world (Wang and Cooper, 2022).

# 3. Methodology

This section describes the methodology implemented to systematically analyze the visual impact of different urban air vehicles in various locations. First, different scenarios are recreated, and the human vision field is visualized through a camera. Raspberry Pi Camera has been configured to provide a remote solution to control measurement campaigns from a central place. Finally, a methodology was developed, including techniques to analyze the previously introduced factors.

## 3.1. Experimental Setup

### 3.1.1. Measurement Station

A camera is used to visualize the human field of view. The measurement station consists of a tripod with a camera above it. The tripod's height has been fixed at 160 cm, as it falls within the range of the average height from feet to eyes for both men (163 cm) and women (151.5 cm), extracted from the anthropologic German study (Lange and Windel, 2017).

In order to assess the impact of viewing angles on the sky and the flying vehicle, a rotating support has been implemented. This support holds the camera and replicates human head movements. A human can flex their neck up to 60° backward and 60° forward, as shown in Figure 3.1. Therefore, it was decided to position the camera at 0° (completely vertical) to simulate the standard neck position. The camera was also orientated 30° backwards to recreate the human movement to focus most of the attention and vision on the sky.

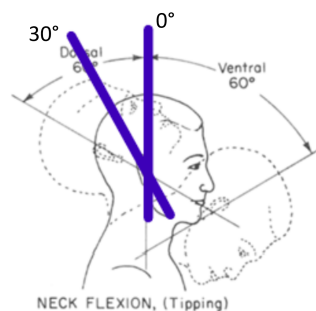


Figure 3.1.: Human neck flexion (Gilman, Dirks, and Hunt, 1979).

In the future, a Raspberry Pi Camera<sup>1</sup> will be used for data acquisition at different locations. To create a portable measurement station capable of conducting data collection campaigns at multiple locations controlled from a central place, remote access to the Raspberry Pi is essential. To access the Raspberry Pi remotely, the most suitable approach is through Secure Shell (SSH)<sup>2</sup>. To use the SSH protocol, the Raspberry Pi and the external computer must be connected to the same Wi-Fi network. Therefore, the Wi-Fi network's SSID, password, and country must be set during the OS installation on the SD card. Once the Raspberry Pi is connected to the same Wi-Fi network as the external computer, it can be accessed through the terminal using the user and password set during the OS installation (Raspberry-Pi, 2020).

In the event that the SD card has been booted with the same name and password and connected to the same PC, an error may occur that can be resolved by deleting the SSH Keys and accessing them again. To do so, the code shown in Figure 3.2 must be written in the terminal.

```
PS C:\Users\  
> ssh-keygen -R raspberrypi.local  
# Host raspberrypi.local found: line 1  
C:\Users\  
./ssh/known_hosts updated.  
Original contents retained as C:\Users\  
./ssh/known_hosts.old
```

Figure 3.2.: Procedure to delete SSH Keys.

After the external computer has recognized the Raspberry Pi connected via SSH, the Raspberry Pi Software Configuration Tool should be accessed through the terminal. From this interface, remote access using Virtual Network Computing (VNC) should be enabled, and it can be accessed from any other device by using RealVNC<sup>3</sup>.

This type of connection to the Raspberry Pi is called the "Headless Setup". However, users can configure the necessary settings by having direct access to the Raspberry Pi through a monitor, keyboard, and mouse, including enabling the RealVNC remote access right from the Raspberry Pi's terminal interface. Once this setup is completed, RealVNC can access the Raspberry Pi remotely.

Furthermore, a power bank will be available to power the Raspberry Pi, and a SIM card will be used to provide portable Wi-Fi connectivity. These will be inside a protective box on the ground and next to the tripod.

<sup>1</sup>The Raspberry Pi is a small Single-Board Computer (SBC). It is an accessible, compact, and affordable computer available in different models with varying processing capabilities.

<sup>2</sup>SSH is a protocol designed as a secure remote protocol using encrypted communication.

<sup>3</sup>RealVNC is a free software application that enables remote access and control of computers over a network.

### 3.1.2. Raspberry Pi Camera Configuration

The Raspberry Pi Camera must be controlled using Python, and specific codes have been developed to capture photos or videos while the vehicle is in flight. This will allow the user to choose which option is better depending on the desired factors to study. Moreover, taking videos has the advantage that screenshots can be made afterwards, selecting the specific moment when the data is relevant. Three codes have been developed for this purpose:

1. The first code captures photos in a row every 3 seconds. A `while True` loop has been created, which will run indefinitely until it is interrupted by the user pressing the `Ctrl+C` key combination as shown in Code 3.1. Complete code can be found in Appendix A.1.1.

Code 3.1: Loop to take photos every 3 seconds.

```
1     while True:
2         photo_filename = os.path.join(folder_path,
3         f'image_{timestamp}.jpg')
4         camera.capture(photo_filename)
5         print("Photo taken")
6         sleep(3)
```

2. The second code records a standard video that will start when the user wants to and end once `Ctrl+C` is pressed. While the videos are captured, the command window will show "Recording". The function is presented in Code 3.2, and the complete code can be found in Appendix A.1.2.

Code 3.2: Function to record videos.

```
1     def start_video_recording():
2         video_filename = os.path.join(folder_path,
3         f'video_{timestamp}.h264')
4         camera.start_recording(video_filename)
5         print("Recording")
```

3. The third code records a video of the vehicle while simultaneously taking a photo every 5 seconds. The code defines two functions for capturing images and recording videos. The code 3.3 summarizes how it starts a loop that captures images at regular intervals while simultaneously recording a video. The program stops execution upon receiving a `KeyboardInterrupt` exception, by pressing `Ctrl + C`. Complete code can be found in Appendix A.1.3.

Code 3.3: Function to simultaneously take photos and videos.

```

1  def start_recording(folder_path):
2      video_filename = os.path.join(folder_path,
3      f'video_{timestamp}.h264')
4      camera.start_recording(video_filename)
5      print("recording")
6      try:
7          while True:
8              interval = 5
9              capture_frame(folder_path, interval)
10             sleep(5)
11             print("photo taken")

```

Once any of the codes are executed, a preview is initiated, displaying exactly what the camera captures in real time on the monitor. This allows users to adjust the camera setting as needed manually. In the command window, users are asked to press `Enter` whenever the camera is ready to start making videos or photos. This is done in the lines of Code 3.4.

Code 3.4: Code to start and end the preview.

```

1  camera.start_preview()
2  input("Press enter to start recording")
3  camera.stop_preview()

```

After pressing `Enter`, users are asked to input all the relevant information about the specific data collection session (the command window shows "Relevant data for the .txt file: "). For instance, they can specify details such as date and time of the data collection, type of vehicle flight or the distance where the vehicle will take off. This will simplify the analysis of the collected media. After providing all relevant details, another press on `Enter` saves the input data into a .txt file named "data\_media" (refer to Code 3.5).

Code 3.5: Function to create a specific .txt file.

```

1  def save_txt_file(folder_path, content):
2      file_path = os.path.join(folder_path, "media_data.txt")
3      with open(file_path, "w") as file:
4          file.write(content)

```

Every code names the captured photos and videos with a timestamp representing the date and time of capture. These files are stored in a folder that is automatically created with its own date/time stamp, too. Notably, these folders will also contain the .txt file previously explained. This part of the codes is shown in Code 3.6.

Code 3.6: Creation of folder and timestamp

```

1  import os
2  from datetime import datetime
3
4  root_folder = "root folder path"
5  timestamp = datetime.now().strftime("image_%Y%m%d_%H%M%S")
6  folder_path = os.path.join(root_folder, timestamp)
7  create_folder(folder_path)

```

### 3.1.3. Scenario Planning for Data Collection

For comprehensive data collection, videos and photos will be taken to simulate different scenarios that might occur in real life.

UAVs will be flown at different distances and altitudes to analyze varying visual sizes. This approach allows for an exploration of how distances and its position in the FOV can influence the perceived size. The selected altitudes, however, depend on the particular urban vehicles being examined. For instance, smaller vehicles meant for package deliveries might fly closer to the population, while air taxis are projected to cruise at altitudes ranging between 100 m and 300 m (*Spain announces plans for flying taxi service in Barcelona 2020*). As such, the selected distances for study should align with the specific vehicle type.

On another note, as introduced in Section 3.1.1, a rotating device is positioned at the measurement station to emulate human neck flexion. To capture various dimensions of the vehicles in flight, the camera will be set at angles of 0° and 30°.

Various vehicles will be examined to observe different visual sizes and study the influence of color on visual impact and contrast against the background. A distinctive representative color for each UAV is anticipated.

The first UAV, referred to as UAV 1 throughout the thesis, has a distinctive white color and a structure similar to a commercial airplane. Its wingspan measures 128 cm, and its maximum vertical dimension is 28 cm. An illustration of this UAV can be found in Figure 3.3b. The second UAV, labelled as UAV 2, has a more complex design. Colored in black, it displays symmetry across both axes when viewed from a top perspective. This UAV measures 76 cm horizontally and 30 cm vertically. Figure 3.3a presents an image of it.

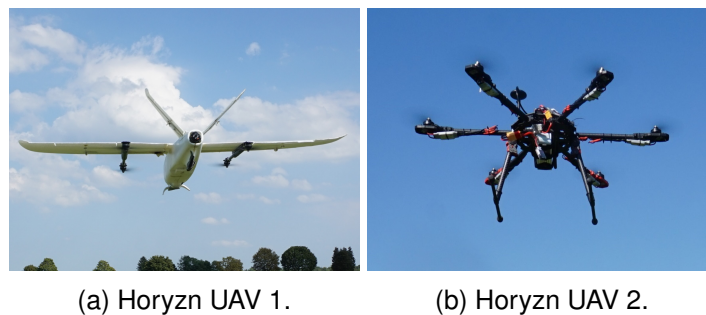


Figure 3.3.: UAVs provided by Horyzn.

Furthermore, capturing images and videos under varying weather and atmospheric conditions is essential. This range is crucial for simulating potential scenarios with diverse backgrounds and, most importantly, visibility levels. As discussed in the theoretical framework, visibility plays a significant role in visual impact.

The data capture scenario for the two studied UAVs seen from a lateral view is illustrated in Figure 3.4<sup>4</sup>. Additionally, the selected lateral distances are shown in Figure 3.7, seen from a top view. In this scenario:

1. **Take-off Positions:** Three distinct take-off positions are established at varying distances from the camera: 3 m, 10 m, and 20 m.
2. **Data Capture Process:**
  - a. The UAV initiates take-off from the first position and ascends to a flight altitude of 2 m.
  - b. Once at this altitude, two photographs of the UAV are captured.

---

<sup>4</sup>Neither of the scenario recreation figures show scaled dimensions.



- c. The UAV then moves laterally to the second position. When reached, photographs are taken again.
  - d. Similarly, the UAV moves laterally to the third position and photos are captured.
  - e. Without landing, the UAV adjusts its altitude to 5m and repeats the same path, with photographs being taken at each of the three positions.
3. **Repetition:** The above data capture process is repeated for each take-off position relative to the camera (i.e., 10 m and 20 m).
  4. **Camera Orientation:** The entire procedure is conducted twice – once with the camera oriented at  $0^\circ$  and once at  $30^\circ$ .
  5. **UAV tested:** The process is repeated for the two UAV tested.

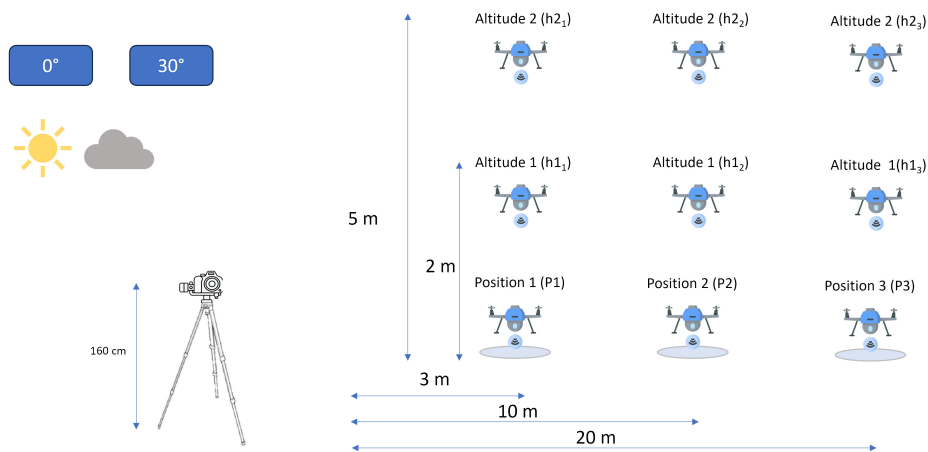


Figure 3.4.: Scenario recreation seen from a lateral view.

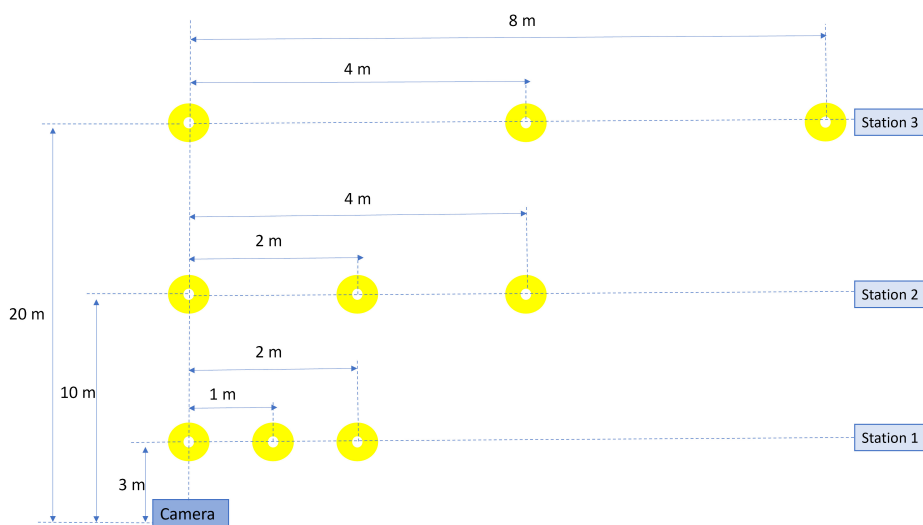


Figure 3.5.: Scenario recreation seen from the view and marks on the ground.

## 3.2. Determination of visual size

As explained in the theoretical framework, visual size is an important variable to consider when measuring an object's visual impact. It depends on the vertical and horizontal dimensions of the object. Therefore, it is evident that these dimensions also depend on the observed object proportion. When the object is seen from a lower perspective, one has a complete view of it, and it will occupy a different space in the visual field than if it is observed from the side.

The study aims to investigate how an object's position in relation to the observer and the viewing angle influence what is being observed. However, a methodology must first be developed to be able to measure this "visual size" from a picture using Python and the distance camera object. This is presented in Figure 3.6.

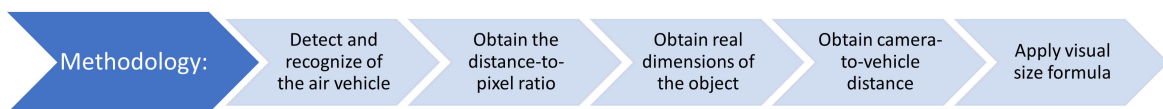


Figure 3.6.: Methodology to obtain visual size.

Firstly, detecting an object in the image and afterwards recognising it as an air vehicle is crucial. Once identified, its horizontal and vertical dimensions from that specific perspective must be calculated. However, from a picture, one can only derive the actual dimension of an object using a distance-to-pixel ratio. That means, how many centimeters a pixel occupies. This needs the use of a reference dimension within the image. Therefore, the third step involves calculating this scale. Determining the distance between the camera and the vehicle in the image is essential. Finally, with all these informations, the formula to compute the visual size of the vehicle can be implemented (Equation 2.1). Python will be employed in each step.

### 3.2.1. Detection and Recognition of the Air Vehicle

To detect the object within an image, the Python library `cvlib` (Ponnusamy, 2021) was employed together with created auxiliary functions. `Cvlib` is a superficial, high-level, easy-to-use open-source Computer Vision Python library available in GitHub, developed in 2021 by Arun Ponnusamy.

The function `image_object_detection` belongs to `cvlib` and enables the detection of common objects within an image. The function provides the coordinates of the bounding boxes, associated labels, and confidence levels for objects identified within the image. However, confidence levels are not needed in this case, and the function performs the following tasks:

1. The algorithm reads the image and uses `cvlib.detect_common_objects` to identify objects in it.

Code 3.7: Code for object detection.

```

1  def image_Object_Detection(image, scale_ratio=None):
2      img = cv2.imread(image)
3      bbox, labels, _ = cvlib.detect_common_objects(img)

```

2. From all the detected objects, it filters and selects only those labeled as 'airplane' or 'kite' and draws a rectangle around them:

Code 3.8: Code for filtering specific detected objects.

```

1  for (box, label) in zip(bbox, labels):
2      if label in ['airplane', 'kite']:
3          x1, y1, x2, y2 = box
4          cv2.rectangle(output_image, (x1, y1), (x2, y2),
5                          (0, 255, 0), 5)

```

3. Using the drawn rectangle, the algorithm calculates the horizontal and vertical distances the vehicle covers in the image, expressed in pixels. Also, the exact location of the introduced object in pixels is displayed (that means, the upper left corner and the right down corner of the rectangle):

Code 3.9: Code for calculation of pixel dimensions.

```

1  def calculate_distance(x1, y1, x2, y2):
2      horizontal_dist = x2 - x1
3      vertical_dist = y2 - y1
4      return horizontal_dist, vertical_dist

```

4. Lastly, the function returns the image with highlighted air vehicles, the total number of air vehicles detected in each image, and their coordinates and positions.

### 3.2.2. Obtaining the Distance-to-pixel Ratio

To accurately determine the actual size of a vehicle in an image, two key pieces of information are needed: the real-world dimension of the vehicle (such as its wingspan) and its corresponding dimension in pixels within the image (how long it appears in the image). These two measurements essentially create a scale that allows obtaining the vehicle's size in the image. By knowing the relationship between the real-world and pixel measurements, this scale can be applied to measure or determine the size of other objects within the same image

As explained in Section 3.1.1, the camera will remain fixed on a tripod, while the UAV will move horizontally from left to right in front of it, consistently maintaining the same altitude. The primary reference will be taken when the UAV is directly in front of the camera, providing a clear rear view of the object. If the vehicle has a wingspan, this should be fully visible, as shown in Figure 3.7.



Figure 3.7.: Reference image.

Hence, it becomes possible to establish a scale for reference by knowing the actual wingspan measurement in centimeters and understanding the number of pixels it covers in the picture. The `calculate_scale_ratio` function returns a ratio indicating how many centimetres correspond to a pixel in the image.

When comparing the sizes of vehicles, using the correct scale is essential. Therefore, the vehicle must maintain consistent horizontal and vertical distances as with other photographs where the obtained scale is applied. If the reference vehicle is at a different distance or height, the comparison will not be accurate and wrong results will be obtained. The appropriate scale function will allow the extrapolate of the actual horizontal and vertical dimensions for all examined images, and the change in dimensions is computed. Both procedures are presented in Code 3.10 and Code 3.11 respectively.

Code 3.10: Scale ratio function.

```

1  def calculate_scale_ratio(pixel_distance, real_distance_cm):
2  return real_distance_cm / pixel_distance

```

Code 3.11: Code for applying scale function.

```

1  if scale_ratio:
2      real_horizontal_distance = horizontal_dist * scale_ratio
3      real_vertical_distance = vertical_dist * scale_ratio
4      print(f'Real Dimensions: {real_horizontal_distance:.2f}
5          cm(horizontal),{real_vertical_distance:.2f}cm(vertical)')
6      dimensions.append([real_horizontal_distance,
7          real_vertical_distance])

```

The object's detection and the scale computation have been developed in the same Python code available in Appendix A.4.

### 3.2.3. Obtaining the Camera-Vehicle Distance

To calculate the distance from a camera to a vehicle, two methods have been developed based on the available data:

1. In the first method, at the moment the picture was taken, the coordinates of both the camera and the vehicle are known.
2. In the second method, the vehicle first ascends from a position known relative to the camera and rises to a specific height. After reaching that height, the UAV travels a predetermined number of meters laterally (see Section 3.1.3 for more details).

Knowing the latitude and longitude coordinates where the camera is located and the exact coordinates of the vehicle when the photo is taken, it is possible to obtain the horizontal distance between the two points. Given their longitudes and latitudes, the Haversine formula calculates the distance between two points on the Earth's spherical surface. This is a scenario calculated in Equation 3.1, where  $R$  is the Earth's radius (6371 km<sup>5</sup>).

---

<sup>5</sup>For short distances like in this case, the Earth is assumed to be perfectly spherical being 6371 km its radius.

$$\begin{aligned}
 a &= \sin^2\left(\frac{\Delta\text{lat}}{2}\right) + \cos(\text{lat}_1) \cdot \cos(\text{lat}_2) \cdot \sin^2\left(\frac{\Delta\text{lon}}{2}\right), \\
 c &= 2 \cdot \text{atan2}\left(\sqrt{a}, \sqrt{1-a}\right), \\
 d &= R \cdot c.
 \end{aligned}
 \tag{3.1}$$

Once the horizontal distance is calculated, and knowing the altitude at which the vehicle flies and the tripod's height on which the camera is positioned, one simply needs to apply Pythagoras' theorem to determine the distance between the object and the observer. The application of the function is shown in Code 3.12, but the complete code is available in Appendix A.6.

Code 3.12: Application of Haversine formula.

```

1 for lat2, lon2 in positions_obj2:
2     dist_hor = haversine_distance(lat1, lon1, lat2, lon2)
3     dist = math.sqrt(dist_hor**2 + dist_ver**2)
4     distances.append(dist)
5     print(f"Distance from the camera to the air vehicle
6         {len(distances)}:", dist, "meters")

```

For the second approach, Pythagoras' theorem is used twice. First, to find the ground distance considering the distance between the camera and the starting position of the vehicle as one leg of a triangle and the lateral movement of the vehicle as the other leg. Then, for the following calculation, the previously found ground distance is used as one leg, and the difference between the vehicle's flying height and the camera's height as the other leg. This will give the total distance from the camera to the vehicle's new position. The complete developed code is shown in Appendix A.7.

### 3.2.4. Applying Visual Size Formula

Finally, the Visual Size Formula (Equation 2.1) can be used, making sure that both the distance between the camera and the vehicle and the dimensions of the object are in the same units. To do so, the created `process_images` function will manage the primary logic.

Initially, a reference image will be defined, and a list of all other images to be processed will be created. It will import the functions created in Section 3.2.1 to detect the vehicle in the reference photo. Additionally, it will import the function `calculate_scale_ratio` explained in Section 3.2.2 to calculate the scale ratio. These same functions will also be used to detect the air vehicles in each loaded image and apply the calculated ratio. The functions `havarsine_distance` or `calculate_distance` will also be imported into the main processing flow.

The visual size formula (Equation 2.1) is employed to get the real dimensions of all the loaded pictures. This code is presented in Appendix A.8, and a summary is in Code 3.13.

Code 3.13: Code for obtaining visual size.

```

1 beta = (180**2 * 60**2) / math.pi**2
2 for (H, D), d in zip(dimensions, distances):
3     alpha = H * D / (d*100)**2
4     S = beta * alpha
5     S_values.append(S)
6 for i, S in enumerate(S_values):
7     print(f"Visual size {i + 1} = {S}")

```

### 3.3. Determination of Absolute Color Difference

To compute the color difference between two objects in an image, one must calculate each object's LAB values within the CIELAB color model. For this purpose, the method shown in Figure 3.8 is proposed.

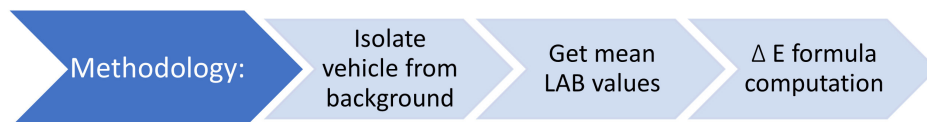


Figure 3.8.: Methodology to obtain color contrast.

### 3.3.1. Air Vehicle Isolation in Images Using GIMP

Several image processing software facilitates object selection within an image. For instance, Bishop (2000) utilized Adobe Photoshop to assess the visual impact of wind turbines. For the purposes of this thesis, GIMP 2.10.34<sup>6</sup> was employed. This software features the "Fuzzy Select Tool", analogous to the famous "Magic Wand Tool" in Photoshop. This tool enables rapid selections based on color ranges, and its threshold can be adjusted to determine how strict this color selection is. The following steps must be followed:

1. **Photograph Insertion:** The process starts by importing the desired photograph into GIMP.
2. **Using the "Fuzzy Select Tool":** This tool is used to highlight the UAV
3. **Copying the Selection:** The chosen portion of the image is copied to the clipboard.
4. **Creating a New Project:** To maintain the same dimensions as the original image and ensure consistency, a new project within GIMP is initiated.
5. **Setting the Background to Transparency:** This is achieved by navigating through GIMP's menu options: 'File » New » Advanced Options » Filled with: transparency.' The transparency setting is essential when superimposing one image onto another while eliminating the original background.
6. **Pasting the Copied Selection:** The previously copied UAV selection is pasted into the new project.
7. **Preparing for Export and Analysis:** The edited image is ready for further processing, analysis, or export. This edited image will be employed for analysing the color difference.

Consequently, the background must also be isolated from the object to get the LAB values later. After selecting the UAV, this selection is inverted (Right click » Select » Invert) to isolate the background. Another new project starts with a transparent background; the selection is pasted here and then exported.

The more complex the object is, the more challenging its selection becomes, and finer details of the object may be lost. However, this loss of detail is not critical since the primary objective is to obtain the mean LAB values, emphasizing the focus on the 'mean' aspect.

---

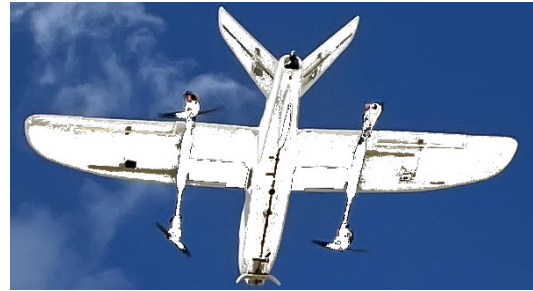
<sup>6</sup>GIMP (GNU Image Manipulation Program) is an open-source and free image editing software.



Below is an example (Figure 3.9): the first image displays the isolated UAV, while the second shows the isolated background.



(a) Isolated UAV.



(b) Isolated Background.

Figure 3.9.: Comparison of isolated UAV and background using GIMP.

### 3.3.2. Air Vehicle Isolation in Images Using Python

In an effort to improve efficiency and automate the previously described methodology, a Python-based approach was considered. The primary aim is to eliminate the need to use GIMP and manually analyze each picture. To achieve this, a Python code has been developed using the OpenCV library<sup>7</sup>. The process involves loading an image, separating the object from its background and saving both pictures transparently.

The developed code facilitates image segmentation by converting the input image to grayscale and allowing the user to set a threshold value (as done in GIMP approach). It categorizes pixels based on their grayscale intensity relative to the threshold: pixels above the threshold are represented as white (255), indicating the object, while pixels below it are set to black (0), indicating the background. Subsequently, two new images, mirroring the original's dimensions, are generated, featuring transparent backdrops. One image highlights the object of interest. The other image showcases the background. Therefore, establishing a correct threshold is essential. This relevant part of the algorithm is shown in Code 3.14.

<sup>7</sup>OpenCV (Open Source Computer Vision) is an open-source library that contains tools and algorithms for image processing and computer vision

Code 3.14: Code for obtaining visual size.

```

1 def segment_image(image_path, save_object_path,
2   save_background_path, threshold_value=100):
3     image = cv2.imread(image_path, cv2.IMREAD_COLOR)
4     gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
5     _, thresholded = cv2.threshold(gray, threshold_value, 255,
6     cv2.THRESH_BINARY)
7     mask = thresholded == 255
8     height, width, channels = image.shape

```

Once segmented, the isolated background and isolated vehicle are pasted into the transparent new images and saved in a .png format. Code 3.15 represents this procedure and the complete one is included in Appendix A.9.

Code 3.15: Code for obtaining visual size.

```

1   transparent_obj_img = np.zeros((height, width, 4),
2   dtype=np.uint8)
3   transparent_obj_img[:, :, :3] = image
4   transparent_obj_img[mask, 3] = 255
5   cv2.imwrite(save_object_path, transparent_obj_img)
6   transparent_bg_img = np.zeros((height, width, 4),
7   dtype=np.uint8)
8   transparent_bg_img[:, :, :3] = image
9   transparent_bg_img[~mask, 3] = 255
10  cv2.imwrite(save_background_path, transparent_bg_img)

```

The following example showcases the capabilities of the developed Python code for image segmentation. The first image exhibits the isolated UAV, while the second image presents the background isolated from the scene, all accomplished using the Python code. This automated approach provides a convenient solution for object isolation, but it may exhibit slightly less precision than manual segmentation using image processing software. However, this minor trade-off in precision is offset by the significant time-saving advantage it offers. Users can simply provide the base image, eliminating the need for meticulous manual selection of relevant image components.

In summary, both the manual GIMP procedure and the automated Python code yield successful results. The GIMP procedure is better for precision but requires more manual effort. However, the Python code offers an automated solution that simplifies the segmentation process, balancing efficiency and precision.



Figure 3.10.: Comparison of isolated UAV and background using Python.

### 3.3.3. Compute LAB Mean Values

Several methods were considered when aiming to calculate an image's LAB mean values. Initially, the idea was to obtain these values using GIMP. The software allows the user to get the color description of each pixel by selecting the 'Colour picker' tool (represented by an eyedropper icon). This tool allows users to derive color values in different color models, including CIELAB. However, GIMP does not offer an option to get the mean value among all the non-transparent pixels in the image.

Given GIMP's limitations, Python is an efficient alternative for this task. As a result, a code was developed where both images are loaded. The function 'get\_mean\_lab\_values' (Code 3.16) stands as the central part of the code and performs the following tasks:

1. Opens and displays the loaded image.
2. Converts the image into a NumPy matrix that will allow the correct manipulations.
3. Filters out the non-transparent pixels in the image by checking the alpha channel.
4. Converts the image matrix from RGB to LAB model.
5. Computes the mean L, A, and B values of the non-transparent pixels.

Code 3.16: Function 'get\_mean\_lab\_values'.

```

1 def get_mean_lab_values(image_path):
2     img = Image.open(image_path)
3     img_arr = np.asarray(img)
4     non_transparent_pixels = np.where(img_arr[..., -1] > 0)
5     img_arr = img_arr[:, :, :3]
6     lab_img = color.rgb2lab(img_arr)
7     L_mean = np.mean(lab_img[non_transparent_pixels][:, 0])
8     A_mean = np.mean(lab_img[non_transparent_pixels][:, 1])
9     B_mean = np.mean(lab_img[non_transparent_pixels][:, 2])
10    return L_mean, A_mean, B_mean

```

This function is applied to the two pictures, and accordingly, the formula to calculate the color difference is utilized (Equation 2.2). This part of the script is shown in Code 3.17, and the complete algorithm is available in Appendix A.3.2.

Code 3.17: Color difference equation

```

1 E_abs = ((L_mean_background - L_mean_object) ** 2 +
2 (A_mean_background - A_mean_object) ** 2 +
3 (B_mean_background - B_mean_object) ** 2) ** 0.5
4 print("Absolute colour difference:", E_abs)

```

### 3.4. Contrast Determination in the Presence of Atmospheric Scattering

To calculate the contrast of an object and its background, considering the effects of atmospheric scattering, the *Cozman and Krotkov* (equation 2.5) formula should be used. I. D. Bishop (2002) employed this formula to model visual thresholds at varying depths and under different haze conditions without needing an actual scenario image. As a result, applying the *Cozman and Krotkov formula* when estimating contrast from an actual image is unnecessary, given that a specific distance and haze apply in such cases. Nevertheless, this formula proves relevance for computing color contrast without the need to visit the flight field for measurements under various weather conditions.

As discussed in Chapter 2.2.3, knowing the initial color difference is crucial. I. D. Bishop (2002) suggests capturing a photograph very close to the object of interest (in his case, a wind turbine). This approach helps establish a condition for the “absolute color difference” at zero depth, after which the *Cozman and Krotkov formula* can be applied to study contrast at diverse distances and on different haze conditions.

The recommended methodology (represented in Figure 3.11) involves capturing a very close photograph of the studied vehicle and another one of the sky. The background color can vary depending on lighting, visibility and weather conditions. Bishop recommends using the worst-case scenario, which means a very sunny day when objects in the sky are most discernible. Subsequently, a Python code has been developed to process these photographs. In this algorithm, both images are loaded and converted to grayscale, and their mean luminosity is computed, as I. D. Bishop (2002: 6) proposes.

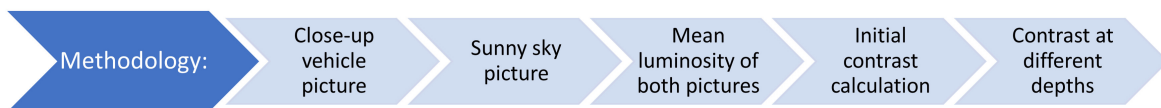


Figure 3.11.: Methodology to obtain atmospheric contrast.

In order to get the mean lightness, the same function as in Code 3.16 is employed, but in this case, only taking the Lightness value. This process allows for the calculation of the initial contrast, i.e., contrast without the influence of atmospheric scattering. Finally, the Python code integrates the *Cozman and Krotkov formula*, which can be employed when the distance and visibility factors are known using Equation 2.6.

Visibility is conventionally measured at meteorological stations using an optical instrument called a visimeter. These devices employ optical techniques to determine the maximum distance at which a reference object is visible from an observation point under specific atmospheric conditions. The results are commonly expressed in kilometres and can be obtained from the websites of meteorological services (Time and Date AS, 2023).

For both UAVs tested, photographs were taken very near to the most representative color of the object and no modifications to brightness or contrast were made on the camera. These images are presented in Figure 3.12a and Figure 3.12b for each respective UAV. On the other hand, due to the optimum weather on the data collection campaign day, it was possible to capture a clear sky photograph from which the lightness could also be computed.

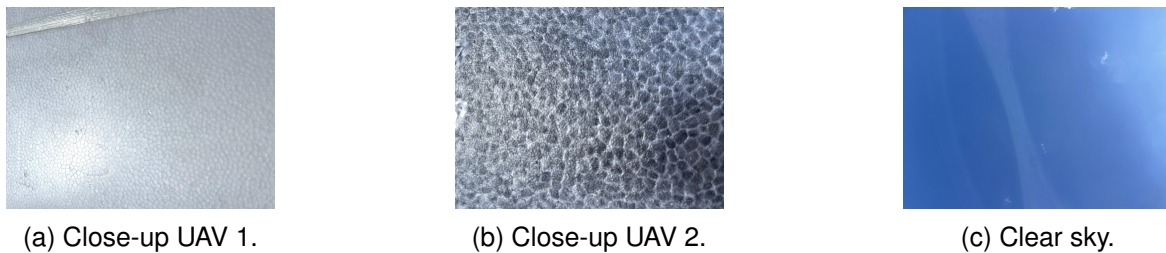


Figure 3.12.: Sample images to calculate color contrast at different depths.

After calculating the initial contrast, the contrast for varying depths and visibilities can be computed. In this way, the influence of the distance between the observer and the UAV can be studied. Moreover, different visibility scenarios can be recreated. These will vary based on factors like haze and atmospheric conditions. This allows for a study to be conducted using these parameters and even by altering the colors of the UAVs. The Python code formulated for this purpose can be found in Appendix A.4.

### 3.5. Digital Simulations of Scenarios using GIMP

Due to logistical limitations and challenges in accessing various scenarios and vehicles needed for this thesis, the decision was made to utilize digital simulations generated through image processing software. These simulations were conducted to recreate diverse scenarios and analyze the influence of vehicle colors without physically fabricating them.

The simulations were created using GIMP. The H-aero zero vehicle served as the primary model to assess the influence of color contrast. This is a sustainable, ecological, and dynamic UAV designed for earth observation, live event recording, surveillance tasks, and more. The vehicle has a shape similar to a balloon and a dimension of 200 x 80 cm. Its distinctive kite shape, as shown in Figure 3.13, and vibrant colors make it particularly relevant for this thesis. The manufacturer mentions that vehicle color depends on the availability and emphasizes the importance of studying the difference in color contrast that each color can produce (H-Aero, 2023). Therefore, it is vital to investigate the potential differences in color contrast that each color can produce. For this purpose, Figure 3.13 served as a reference, and the vehicle was isolated from the background using the procedure described in Section 3.3.2. The color of the vehicle was then changed using the Bucket Fill Tool<sup>8</sup> in GIMP allows a selected area to be filled with a specific color.

<sup>8</sup>The bucket fill tool in GIMP is used for filling areas of an image or selection with a specific color or pattern. It is represented by an icon that looks like a paint bucket.



Figure 3.13.: H-aero reference image (H-Aero, 2023).

Colors chosen for this study were green, red, blue, yellow, white and black, representing the boundaries of the LAB color space model. Furthermore, as the absolute color difference represents the distance between two colors in a 3D representation, it was also essential to assess colors close to the sky color. Here, this difference was minimal, potentially leading to reduced visual annoyance.

To utilize the Bucket Fill Tool, colors need to be specified in the HTML notation<sup>9</sup>. The colors chosen for this study and their respective HTML notations are represented in Table 3.1 and are defined as described in Codes (2023).

Table 3.1.: Selected colors and their HTML notations.

(a) Basic Colors.

| Color  | HTML Notation |
|--------|---------------|
| Green  | #00FF00       |
| Red    | #FF0000       |
| Blue   | #0000FF       |
| Yellow | #FFFF00       |
| White  | #FFFFFF       |
| Black  | #000000       |

(b) Blue Shades.

| Color            | HTML Notation |
|------------------|---------------|
| Blue Sky         | #87ceeb       |
| Corn Flower Blue | #6495ed       |
| Light Blue       | #87cefa       |
| Steel Blue       | #4682b4       |
| Color Sky        | #71BCE1       |
| Gray Sky         | #BCC8C6       |

<sup>9</sup>HTML color codes are sets of six hexadecimal digits representing the amounts of red, green, and blue in a color (#RRGGBB).

The first simulations are presented in Figure 3.14, where the primary colors are chosen to be analyzed and in Figure 3.15, the blue shades.

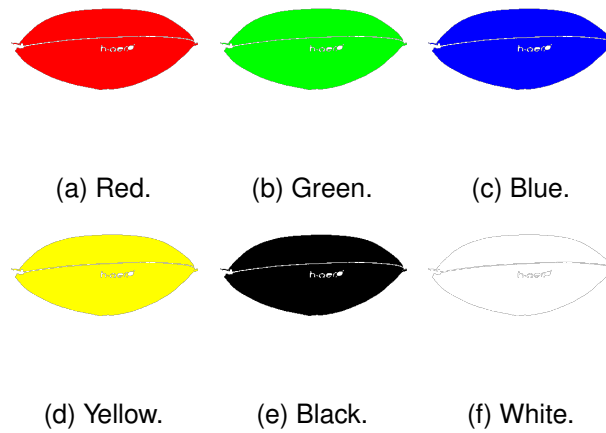


Figure 3.14.: Simulation of boundary colors in H-aero using GIMP.

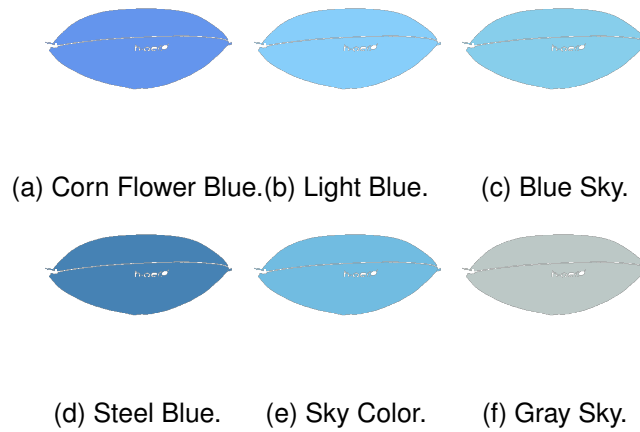


Figure 3.15.: Simulation of different blue colors in H-aero using GIMP.

These figures can then be compared with the average LAB values of the background of the reference image. This background, isolated using the methodology discussed in Section 3.3.1, is depicted in Figure 3.16.

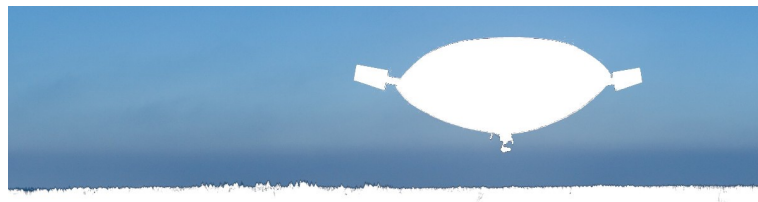


Figure 3.16.: H-aero reference sky.



## 4. Results and Discussion

Data collections were carried out on July 4th, August 22nd, and 23rd. During these three days, the weather conditions were optimum, with visibilities up to 16 km and mostly sunny, as shown in Table 4.1. This section presents the data obtained and the results assessed after applying the methodology previously detailed for the different variables under study.

| Date        | Wind        | Humidity | Visibility | Temperature |
|-------------|-------------|----------|------------|-------------|
| July 4th    | 13 km/h SW  | 84%      | 16 km      | 22 °C       |
| August 22nd | 10 km/h SSW | 72%      | 16 km      | 27 °C       |
| August 23rd | 12 km/h N   | 54%      | 16 km      | 29 °C       |

Table 4.1.: Weather data on flight campaigns (Time and Date AS, 2023).

### 4.1. Visual Size Assessment

For the visual size evaluation, data collection for the test on August 23rd and white Horyzn vehicle (UAV 1) is employed. Photographs were taken following the structure represented in Figure 3.4 and Figure 3.5.

Annex B.1.1 contains pictures when the drone was at an altitude of roughly 2 m above the ground. The drone was positioned directly over marked spots on the ground and the camera is oriented at a 0°. These images highlight, with a green-colored rectangle, the area of the detected UAV, which has been used to determine the exact dimensions of it.

As shown in Figure 4.1, which corresponds to a ground distance of 3 m and a lateral distance of 2 m, the entire vehicle structure is not visible. This means that part of the UAV is outside the camera's field of view; thus, this result is not considered.

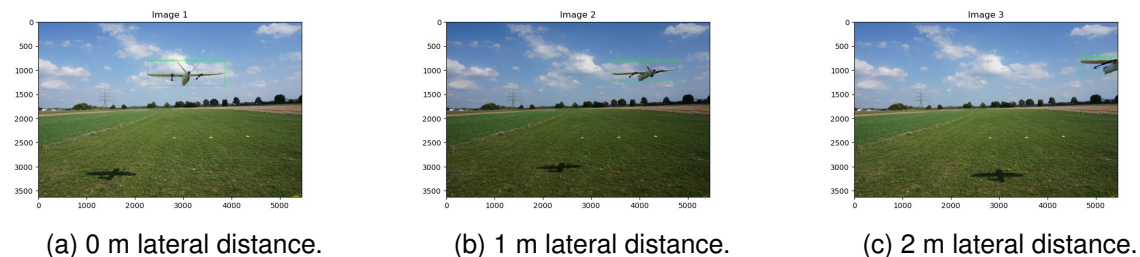


Figure 4.1.: Photographs at 3 m ground distance, 2 m altitude and 0° orientation.

Furthermore, the algorithms developed in Python fail to detect the vehicle when positioned 20 m away horizontally from the camera. The UAV is so far away that the algorithm cannot distinguish details. However, zooming and cropping the image makes detection possible because all the attention is focused on a specific part of the picture. This is why Figure 4.2 displays a closer view of the object. The procedure does not alter the precision of dimension measurements, as the image's pixel resolution is also adjusted proportionally.

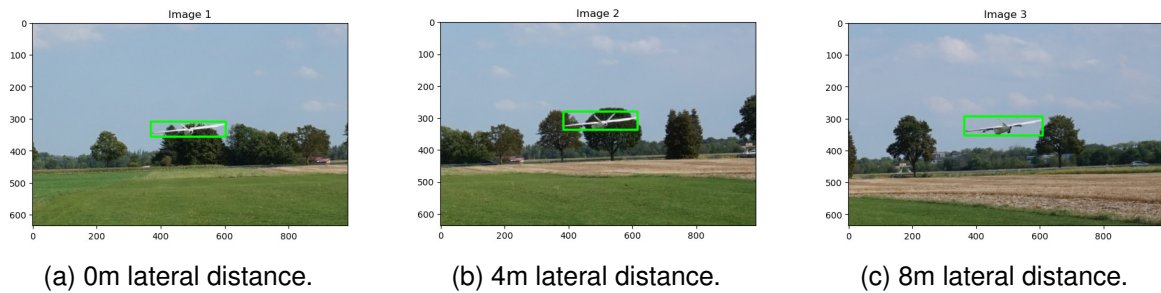


Figure 4.2.: Photographs at 20m ground distance, 2m altitude and  $0^\circ$  orientation.

Annex B.1.2 contains photographs taken and processed when the UAV is at an altitude of approximately 5 m, and the camera is again oriented at a  $0^\circ$ . Images corresponding to a horizontal distance of 3 m are not included, as, in that case, the UAV is outside the camera's field of view. On the other hand, annexes B.1.3 and B.1.4 display images captured at altitudes of 2 m and 5 m, respectively, but this time with the camera tilted approximately  $30^\circ$ . Once again, specific images must be manually zoomed to allow the drone's detection by the algorithm. Images of the UAV at horizontal distances of 3 m and altitudes of 5 m are not included for the same reasons previously mentioned.

After collecting and processing the data using Python, the exact dimensions of the UAV are determined for each position, and the distance between the object and the camera is calculated using the "Second Approach" method (see Section 3.2.3 for more details). Later on, the visual size of the object is established. Table B.1, included in Appendix B.3, shows all the variables that have been calculated to study visual size and the factors that influence it. These variables include: camera rotation angle, flight altitude, ground distance, lateral distance, distance between the camera and the object, centimeter-pixel ratio and horizontal and vertical dimension of the vehicle. Hereafter, these will be analysed in more detail.

Figure 4.3 displays a scatter plot generated in Python<sup>1</sup>, illustrating the relationship between the distance from the "observer" to the UAV and the visual size, for different altitudes and camera angles. In this graph, the visual size of every position is shown.

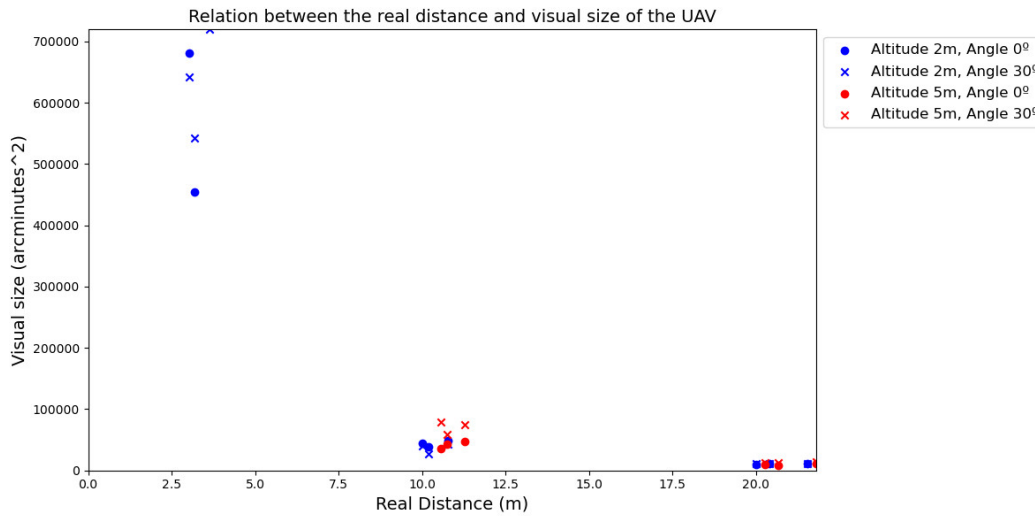


Figure 4.3.: Relation between real distance and visual size.

The results shown in Figure 4.3 agree with expectations: the visual size decreases as the distance between the observer and the object increases. This means that the further away it is, the less space it will occupy in the human field of vision; therefore, its impact will be smaller. In this same graph, all the results can be grouped into 3 main groups: one representing an observer-vehicle distance of approximately 3 m, another around 11 m and the last 21 m. This corresponds to the three stations, and therefore, it can be seen that the distance does not vary significantly when only the lateral distance and height are modified, and the ground distance is kept constant. The visual size results obtained for a ground distance of 3 m turned out to be extremely high (between 454339.022 arcmin<sup>2</sup> and 719504.461 arcmin<sup>2</sup>). This was predictable since, at that distance, the UAV is so close to the user that it practically occupies their entire field of view. This is merely a simulation; such a situation is not expected in a natural environment. The oscillation is less significant in the case of a ground distance of 10 m. There, the maximum visual size is 78437.495 arcmin<sup>2</sup>, and the minimum is 27350.631 arcmin<sup>2</sup>. The same happens with a ground distance of 20 m, where the values oscillate between 8937.488 arcmin<sup>2</sup> and 14609.914 arcmin<sup>2</sup>. This slight variation in visual size suggests now that the position of the UAV within the FOV does not significantly affect its perceived size, contrary to initial beliefs. While this observation provides a preliminary conclusion, it will be further examined in detail throughout this section.

<sup>1</sup>Every graph shown in "Results and Discussion" Section has been meticulously created using Python.

Important to note is that, for the same altitude and horizontal distance (resulting in the same actual distance), and when varying the rotation angle, the visual size maintains a similar order of magnitude, and no substantial changes are observed. This suggests that when the neck is bent to look directly at the sky and considering the specific structure of the analyzed UAV, the relationship between its dimensions is similar and thus its annoyance is also similar. Only at very small distances do relevant differences appear. This conclusion has been achieved after analysing Figure 4.3 and the different results represented by an "x" or point.

Two graphs are included that depict the relationship between the visual size and lateral distance. Figure 4.4 illustrates this relationship for a height of 2 m, while Figure 4.5 for 5 m. These figures are designed to explore how the perceived size of an object changes when it is located at different horizontal locations within the human field of vision. Moreover, a linear regression line has been drawn to study the tendencies.

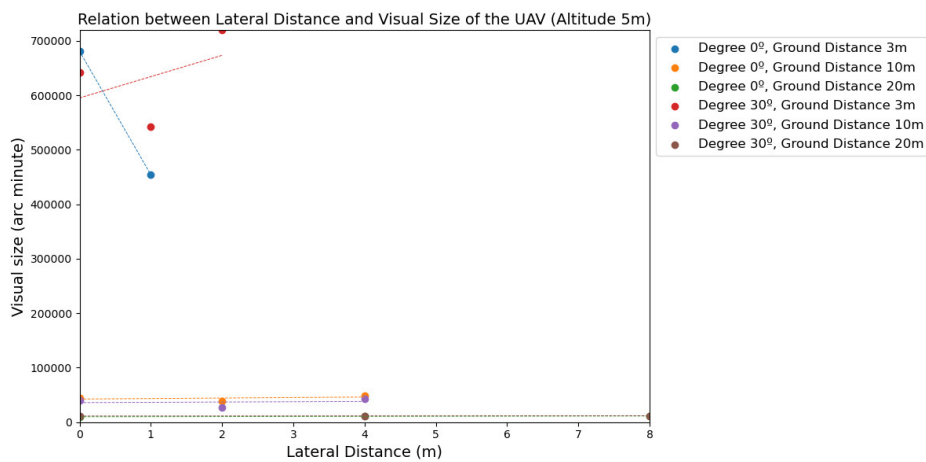


Figure 4.4.: Relation between lateral distance and visual size at 2 m altitude.

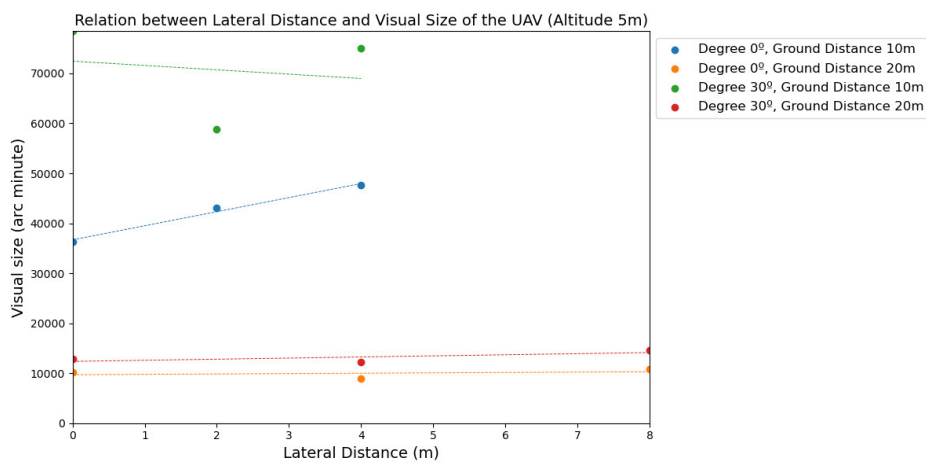


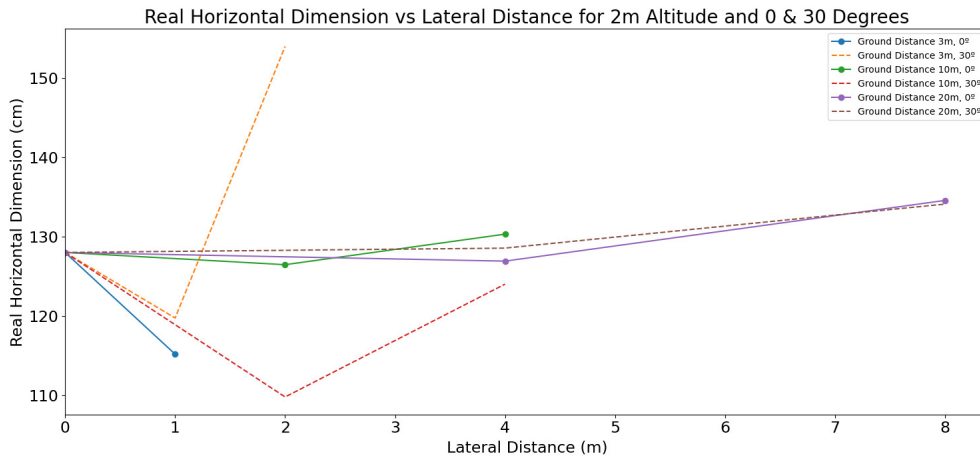
Figure 4.5.: Relation between lateral distance and visual size at 5 m altitude.

In Figure 4.4, the red points represent the scenario where the camera has no rotation ( $0^\circ$ ), and the ground distance is 3 meters. In this case, the visual size varies between 680951.137 and 454339.022 arcmin<sup>2</sup>, depending only on the lateral distance. The respective regression line illustrates how visual size decreases as lateral distance increases. The case where the camera has a  $30^\circ$  rotation angle at the same 3-meter ground distance is depicted in red. Here, the visual size again varies significantly, with values of 641324.887, 542072.726, and 719504.461 arcmin<sup>2</sup>. Notably, the regression line in this scenario exhibits a negative slope. For the rest of the cases, the visual size results do not vary significantly, and they follow an approximately horizontal regression line when the lateral distance is changed.

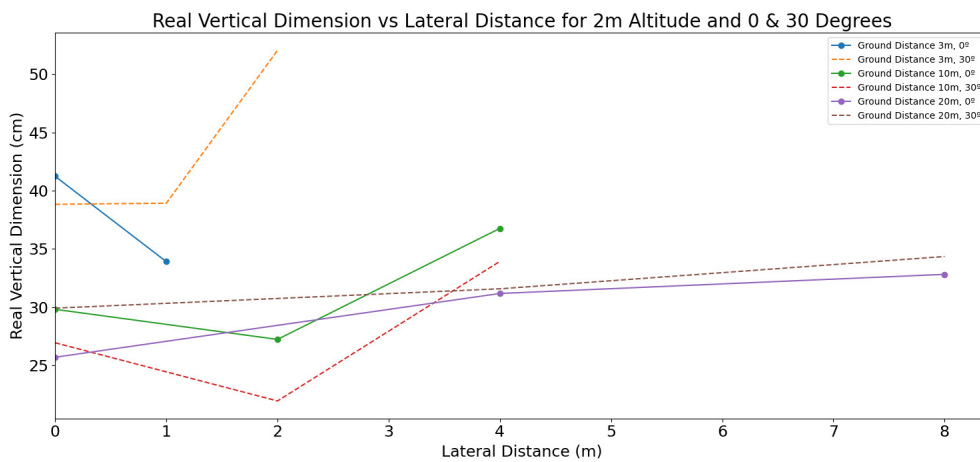
Observing Figure 4.5, one might perceive a significant change in visual size in the green and blue cases (i.e., when the ground distance is 10 m). However, the values on the vertical axis of Figure X are one order of magnitude lower than those in Figure 4.4. This indicates that the visual size results do not vary dramatically in this case. Once again, as the ground distance increases, the visual size values become more similar when only the lateral distance is changed. This aligns with the conclusions drawn from Figure 4.4.

Therefore, after analyzing both graphs, only abrupt changes are evident in the initial tests, where the distance between the user and the object is minimal and unrealistic. This scenario is not possible due to security reasons since the vehicle cannot fly that close to the observer. For the rest of the cases, the vehicle's position inside the human field of view does not change significantly its visual perception, and the conclusion achieved from Figure 4.3 is confirmed.

During the data analysis, an observation was made that while the visual size decreased with increasing distance, the actual dimensions did not consistently exhibit a decreasing trend as distance increased. To better understand the relationship between the dimensions and the variation in lateral distance, the following linear graphs are introduced (Figures 4.6 and 4.7). There, "real horizontal dimension" and "real vertical dimension" refer to the horizontal and vertical dimensions of the vehicle in those specific locations, measured in centimeters.



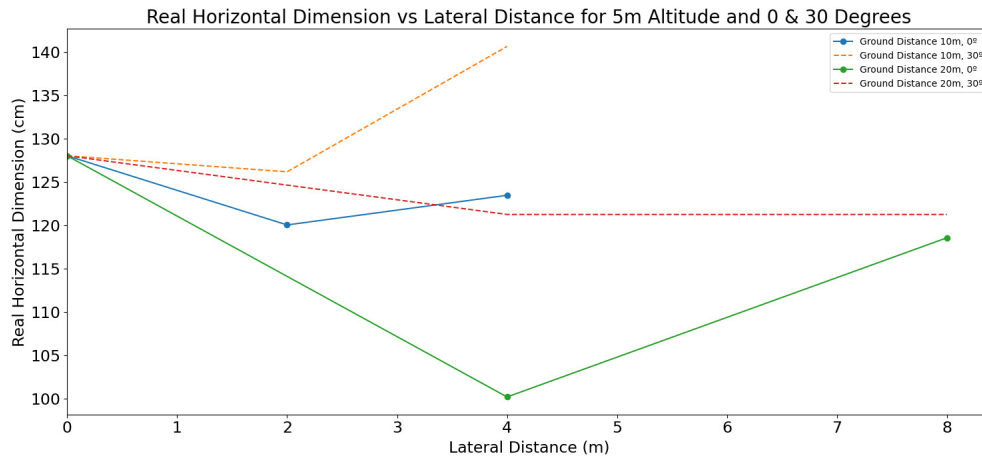
(a) Horizontal dimension vs lateral distance.



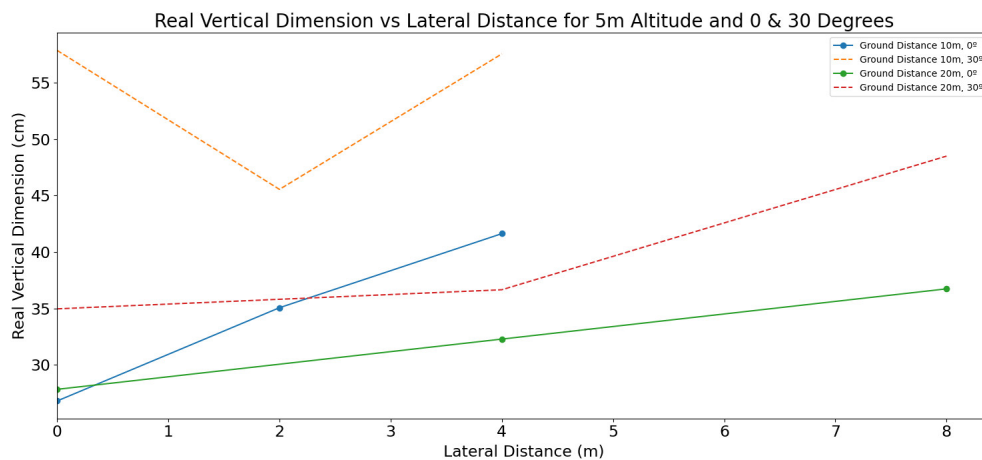
(b) Vertical dimension vs lateral distance.

Figure 4.6.: Relation between dimensions and lateral distance at 2 m altitude (UAV 1).

In these figures, the absence of a linear trend in both the vertical and horizontal dimensions with increasing distance is evident; fluctuations in perceived dimensions are observed. Take, for instance, the violet points denoting the perceived vertical dimension when the ground distance is 20 m, and the rotation angle is  $0^\circ$  (Figure 4.7). In this scenario, the vertical dimension measures 25.71 cm at 0 m lateral distance, 31.18 cm at 4 m, and 32.82 cm at 8 m lateral distance. This variability in perceived dimensions can be primarily attributed to the angle from which the UAV observes the object. Certain object parts become more prominent depending on this angle, altering their perceived proportions. For example, a lateral view of the UAV will emphasize its profile over its front view. These observations help explain why the visual size, resulting from the combination of dimensions and distance, does not show a significant decrease as the distance increases. It shows the significance of considering perspective and angles when interpreting visual data in the context of distance.



(a) Horizontal dimension vs lateral distance.



(b) Vertical dimension vs lateral distance.

Figure 4.7.: Relation between dimensions and lateral distance at 5 m altitude.

On another note, to calculate the scale, the image corresponding to a lateral distance of 0 m was always used as a reference for each ground distance and altitude, where the UAV's wingspan is fully visible. However, consideration of potential human errors should include instances where the drone, during piloting, did not maintain a perfectly horizontal position, potentially introducing inaccuracies in the scale. Additionally, wind conditions play an important role in these cases.

Moreover, while the distance calculation provides a reasonable initial estimation, it might not be entirely accurate. This is due to potential discrepancies in the altitude and lateral distances used in the calculations and the ones really achieved by the UAV pilot. To address this limitation, the "First Approach" that uses the exact geographical coordinates of both the camera and the vehicle might get more accurate results.

As a final point, the study of the visual size of UAV 2 has not been estimated because it was considered irrelevant. This is due to its symmetrical structure along both axes when viewed from above and the results obtained for UAV 1. This means that, regardless of its position in the visual field, as long as the drone flies horizontally, it will maintain proportional horizontal and vertical dimensions. Appendix B.1.5 shows an example of this.

## 4.2. Absolute Color Difference Variation using Flight Test Data

An analysis is conducted on the contrast between UAV 1 (colored white and physically similar to a commercial airplane) against diverse backgrounds. Six photographs, represented in Figure 4.8, are chosen to assess the influence of varying meteorological conditions. Notably, not only does the background change, but the UAV's color also alters depending on the light's angle in each instance, resulting in distinct shadow patterns on the structure. Additionally, several of these photos are cropped versions where the UAV was situated at a significant distance from the camera, explaining the difference in quality among them.

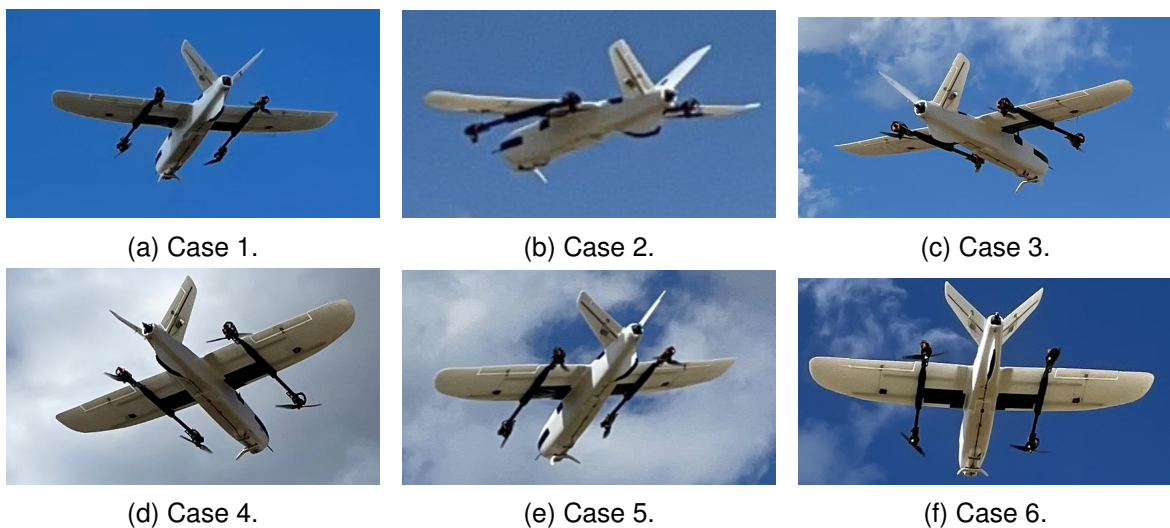


Figure 4.8.: UAV 1 with different backgrounds.

As delineated in the methodology section, isolating the object to obtain two distinct images is the preliminary step in studying the absolute color difference between an object and its background. Within that section, two strategies were proposed: one utilizing the GIMP software, and the other deploying a Python algorithm specifically developed for this research.



As previewed in Section 3.3, the method employing GIMP yields more exact results but needs an individual manual analysis of each photo. Conversely, the Python algorithm is faster but less precise. This difference can be clearly observed in Appendix B.2, where the images previously introduced in Figure 4.8 are shown together with the outcomes derived from both methodologies.

The outcomes from the Python algorithm, elaborated upon in Section 3.3.3 for applying the absolute color difference<sup>2</sup> formula after the object-background separation are catalogued in Table 4.2. This table presents the results of both methodologies. A relevant point to consider is that improper object isolation may leave portions of it still in the background image, causing the algorithm to misinterpret these pixels. This calculation results in inaccurate average LAB values, influencing the final results, as observed from the differences between the columns.

|        | $\Delta E$ Python Method | $\Delta E$ GIMP Method |
|--------|--------------------------|------------------------|
| Case 1 | 52.593                   | 48.235                 |
| Case 2 | 15.548                   | 35.213                 |
| Case 3 | 45.755                   | 42.200                 |
| Case 4 | 46.046                   | 47.027                 |
| Case 5 | 28.216                   | 31.247                 |
| Case 6 | 46.368                   | 36.963                 |

Table 4.2.: Comparison of absolute color difference results for different cases.

Table 2.3 indicates that  $\Delta E$  differences less than 2 are not visible to the human eye. Using this value as a reference, the difference between the results obtained using the two approaches is especially big in cases 2 and 6. Thus, it has been decided to use the data from the GIMP method over the Python code due to its higher precision when isolating the object.

Analyzing the results obtained from the GIMP method and presented in the bar chart of Figure 4.9, the lowest contrast corresponds to the image with a cloud-covered background, Case 5 (being its value 31.247 and using GIMP approach). This reflects that the difference in color is smaller when the background is white. It is consistent with expectations since the drone is white and its color closely resembles that of clouds more than a clear sky. Indeed, the highest contrast, and therefore the most significant impact, occurs when the sky exhibits a very intense blue color (Case 1 with  $\Delta E = 48.235$ ).

<sup>2</sup> $\Delta E$  is an adimensional measure since L, a and b components of the CIELAB Model also are.

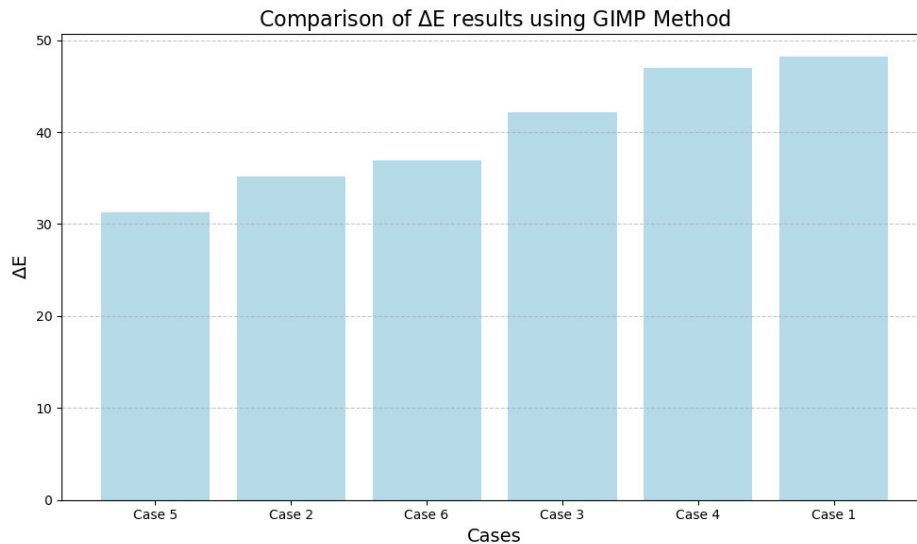


Figure 4.9.: Comparison of  $\Delta E$  GIMP Method results.

Analyzing Figure 4.9, Case 2 also features a sky with lighter tones, resulting in low contrast (35.213). However, Case 4 is surprising. Although the drone is positioned against a white background, its color difference is among the highest. The drone's hue appears considerably darker in this image than in others. This could be attributed to the lighting at that moment, the angle at which the photo was taken, or even the camera's resolution. Cases 4 and 1 show similar color differences, even though they initially seem distinct from each other. This phenomenon can be attributed to variations in the background color and the object's perceived colour in each case. In the CIELAB color space model, the object color and background color positions are markedly different for these two cases—however, the Euclidean distance metric yields similar values for both of them.

In Figure 4.10, three images of UAV 1 are presented, taken from the same position with just seconds between each shot. These images are specifically chosen to illustrate the noticeable variation in the sky's color at that moment. Near the horizon, the sky exhibits a lighter hue, while at the top of the image, its blue is more vibrant. This chromatic variation is explained by the phenomenon of atmospheric scattering, by which small particles and molecules in Earth's atmosphere scatter sunlight in all directions (see Section 2.2.3 for more detail). The  $\Delta E$  results are also presented in Table 4.3, where the difference is notable.



Figure 4.10.: Difference in sky color due to atmospheric scattering - UAV 1.

|            | Case 7 | Case 8 | Case 9 |
|------------|--------|--------|--------|
| $\Delta E$ | 43.560 | 15.672 | 9.288  |

Table 4.3.:  $\Delta E$  results for different locations (UAV 1) using GIMP.

In Cases 7, 8, and 9, significantly different  $\Delta E$  results are evident despite consistent atmospheric conditions and the same vehicle. When the UAV is near the horizon, the color contrast is reduced because the sky appears whiter and more similar to the vehicle's color. However, the magnitude of the color difference is unexpectedly large. Conducting a dedicated analysis of color differences in the specific locations where the vehicle is expected to fly becomes essential for deeper insights. This analysis will be closely linked to the UAV's flying altitude

The same comparison is carried out with UAV 2 provided by Horyzn and the same background and time. Figure 4.11 shows the three studied photographs. Again, results shown in Table 4.4 show big differences in color contrast.

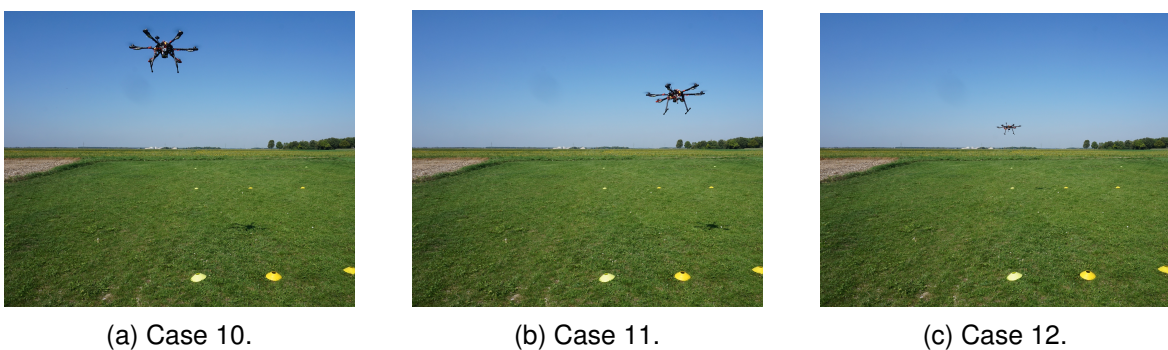


Figure 4.11.: Difference in sky color due to atmospheric scattering - UAV 2.

|            | Case 10 | Case 11 | Case 12 |
|------------|---------|---------|---------|
| $\Delta E$ | 64.787  | 65.433  | 57.891  |

Table 4.4.:  $\Delta E$  results for different locations (UAV 2) using GIMP.

The color contrast results for cases 10, 11, and 12 do not exhibit as much variation as observed with the white UAV. This can be attributed to the black color of UAV 2 being positioned further away from any blue shade in the CIELAB color space model (refer to Figure 2.7 in Section 2.2.2). Therefore, location in the sky field is less crucial for darker vehicle colors compared to lighter ones.

Results from Table 4.3 and 4.4 also provide a basis for comparing the two UAVs. These UAVs have entirely different colors, one being black and the other white, representing the two extremes of the Lightness parameter in the LAB color space model. While blue is also an extreme value for the 'b' component of this color space, it represents a primary color. On the other hand, the sky does not have such an intense blue shade, making it closer to white. This explains why UAV 1 exhibits significantly less color contrast with its background than UAV 2. Refer to Figure 4.12 for a visual representation of this comparison.

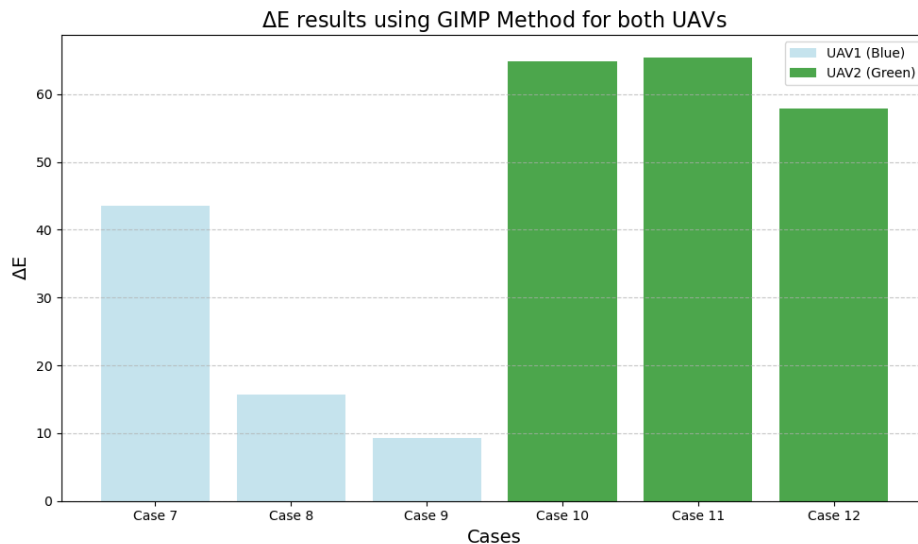


Figure 4.12.:  $\Delta E$  GIMP Method results for both UAVs.

Thus, it can be concluded that not only is the color of the UAV significant, but also its position within the human field of vision, as well as atmospheric conditions and the background. On a sunny day, due to atmospheric scattering, the sky appears lighter closer to the horizon. Consequently, vehicles with light colors will have a reduced visual impact in these areas compared to higher positions in the visual field.

### 4.3. Color Contrast and Atmospheric Scattering Evaluation.

In this section, apparent color contrast is calculated and discussed, taking into account atmospheric scattering and using only an initial condition. As explained in Section 3.4, this approach requires only a close-up photograph of the object and a photograph of a clear sky. For this analysis, an image of both Horyzn UAVs are used as samples (Figure 3.12 detailed in Section 3.4).

Using the code detailed in Section 3.4, the influence of different distances and visibilities on color contrast for these UAVs is explored. A visibility of 16 km, corresponding to the day of the Flight Test, was selected. Additionally, visibilities of 9 km, representing a partly sunny day, and 7 km, indicating a cloudy day, are analyzed.

Distances ranging from 10 meters to 40 meters (from the observer to the flying object) are initially chosen to calculate the associated color contrast in percentage terms. The results for these distances can be found in Figure 4.13 for UAV 1 and in Figure 4.14 for UAV 2. However, within this range of distance, there appears to be no significant contrast difference. In fact, the difference in apparent contrast of UAVs between a distance of 10 m and 40 m is only 0.0389% when visibility is 16 km, 0.069% with 9 km of visibility, and 0.088% with 7 km, all of which are completely irrelevant. For UAV 2, these values change to 0.027%, 0.071%, and 0.091%, respectively, for the same visibility magnitudes and the same conclusion is driven.

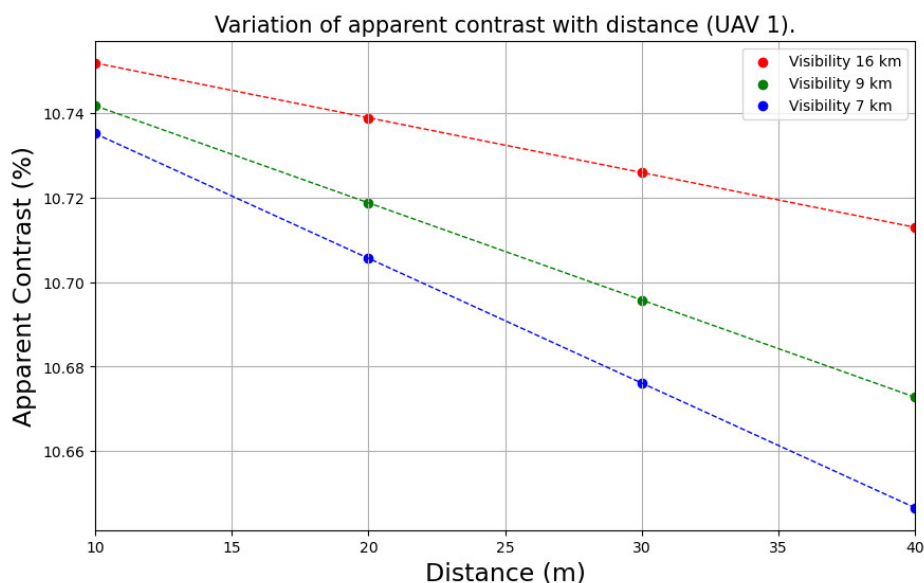


Figure 4.13.: Influence of distance and visibility on apparent contrast (UAV 1).

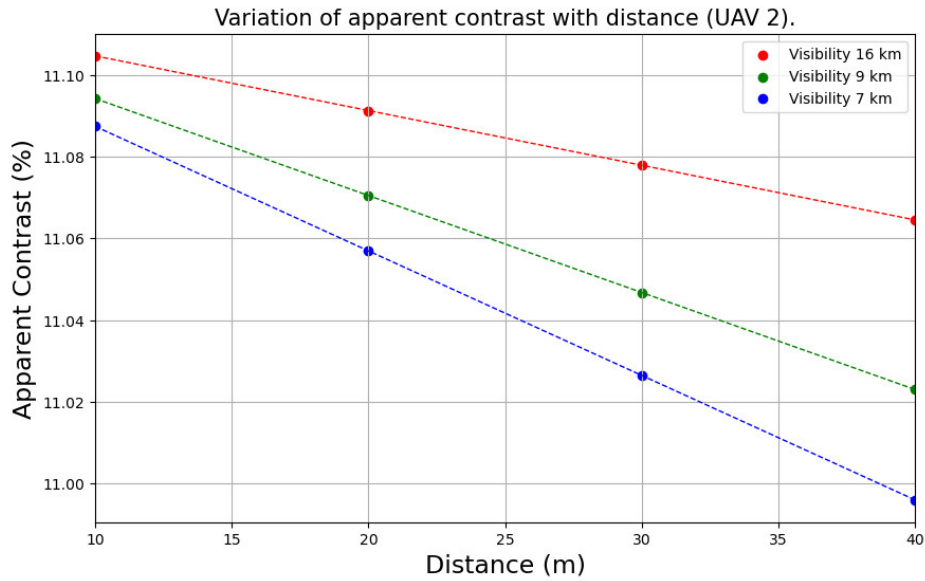


Figure 4.14.: Influence of distance and visibility on apparent contrast (UAV 2).

The evaluation is then expanded to include distances of 1 km, 2 km, and 3 km. Given the small size of the selected UAVs, these distances may not offer realistic scenarios since UAVs would likely remain unseen at such lengths and, therefore, have no significant impact on observers. Nonetheless, these results can be extrapolated to more oversized vehicles with the same colours. The results are illustrated in Figures 4.15 and 4.16 for each UAV.

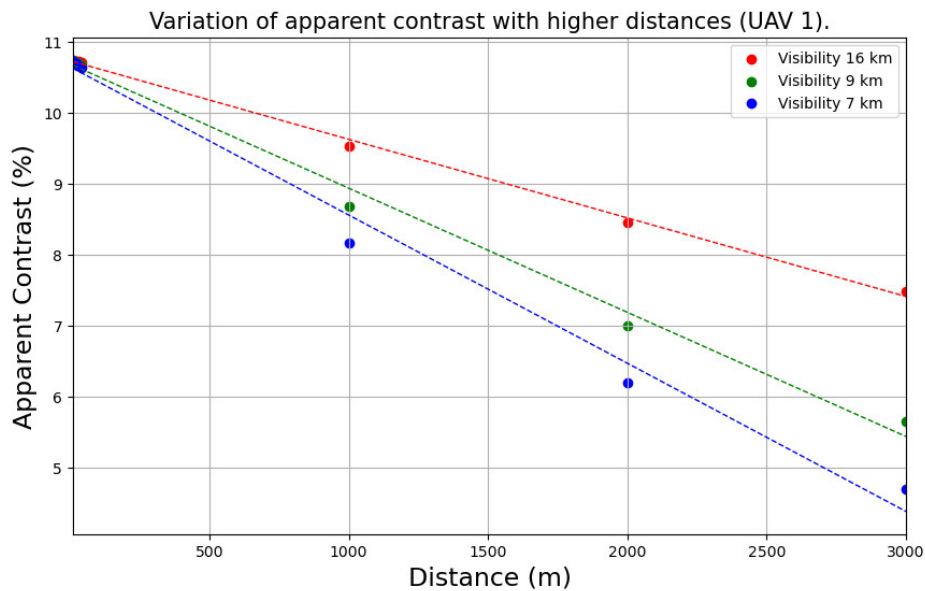


Figure 4.15.: Influence of higher distance and visibility on apparent contrast (UAV 2).

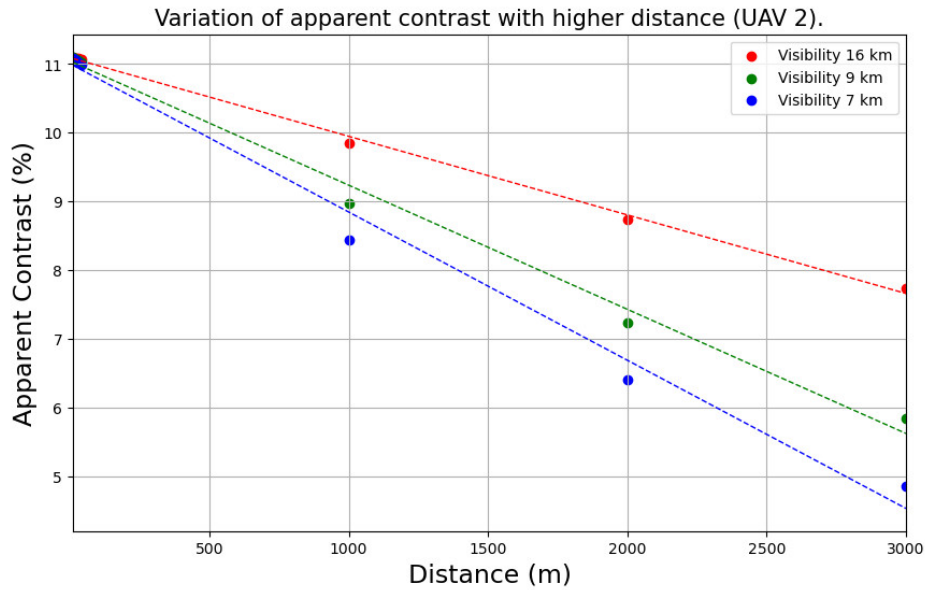


Figure 4.16.: Influence of higher distance and visibility on apparent contrast (UAV 2).

Figure 4.15 and Figure 4.17 display a more comprehensive range of apparent contrast values on the vertical axis. At these greater distances, a noticeable decrease in contrast becomes evident as the range increases. When examining UAV 1 and considering a 3 km increase in distance, the apparent contrast decreases by 3.258% for 16 km visibility, 5.088% for 9 km visibility, and 6.032% for 7 km visibility. Similarly, for UAV 2, the contrast reduction is 3.365%, 5.255%, and 6.229% for the same 3 km increase in distance and corresponding visibilities. Compared to the earlier cases, the significant change in apparent contrast as distance increases becomes clear.

Interestingly, the contrast percentages for both UAVs show remarkable similarity. This is somewhat unexpected, given that white and black colors are far from each other in the CIELAB space model. A potential reason for this outcome might be the use of the difference in lightness rather than  $\Delta E$  (see Section 3.4 for more details). In conclusion, it might be beneficial to apply a similar methodology emphasizing the absolute color difference instead of lightness for future research.

#### 4.4. Absolute Color Difference Simulation Results

As detailed in Section 3.5, a different color for the H-aero vehicle was relevant to simulate to study their influence on the absolute color difference. This allows to analyze color contrast without real photographs.

The basic colors represent the boundaries of the CIELAB color space model. The results, obtained from the codes developed and explained in Section 3.3.3, are presented in Table 4.5a. Analyzing the results, the notably large values become evident. As said before, the color difference is determined using a clear sky background contrasted against the most distant points in the chosen color space. Since  $\Delta E$  represents this distance, the results show meaningful values.

On another note, various shades of blue are applied to the vehicle, attempting to emulate colors like the background being "Gray Sky" color chosen to simulate a cloudy day with haze. Table 4.5 shows the corresponding absolute colour difference results.

From Table 4.5b, it is clear that the values are substantially smaller than those in Table 4.5. This suggests that the selected UAV colors are more similar to the background, thus making it challenging to discern differences, consequently diminishing visual annoyance. Notably, "Blue Steel" and "Sky Color" exhibit minimal  $\Delta E$  values.

Therefore, it can be concluded that to minimize the visual impact of a vehicle flying in the sky, its representative color has to be as similar to the background as possible. However, as elaborated in the Theoretical Framework, sky does not always maintain a similar hue due to the scattering of light coming from the sun. Apart from the reddish color of sunrise and sunset or the grays on cloudy days, clear skies can also exhibit various shades of blue. Atmospheric aerosols can influence these tones, sometimes even giving the sky a white appearance during a sunny day.

Table 4.5.: Colors and their associated  $\Delta E$  values.

| (a) Basic Colors. |            | (b) Blue Shades. |            |
|-------------------|------------|------------------|------------|
| Basic Color       | $\Delta E$ | Blue Shade Color | $\Delta E$ |
| White             | 47.136     | Steel Blue       | 10.797     |
| Black             | 64.984     | Sky Color        | 15.492     |
| Blue              | 121.494    | Light Blue       | 21.024     |
| Red               | 125.095    | Blue sky         | 21.935     |
| Yellow            | 125.757    | Corn Flower Blue | 28.580     |
| Green             | 137.830    | Gray Sky         | 31.079     |



Even though this procedure shows relevant results, these are just simulations where atmospheric scattering has not been considered. In order to get color contrast as a percentage and have atmospheric scattering in mind, the approach shown in Section 3.4 must be followed. The results of applying the *Cozman and Krotkov formula* are shown in table 4.4 when the distance is 10 m and the visibility 16 km. Additionally, a lineal graph shows the apparent contrast of each color under these conditions.

Table 4.6.: Colors and their associated color contrasts.

| Color      | Initial Contrast (%) | Apparent Contrast (%) |
|------------|----------------------|-----------------------|
| Red        | 2.643                | 2.640                 |
| Blue       | 10.825               | 10.811                |
| Green      | 10.832               | 10.819                |
| Yellow     | 14.505               | 14.488                |
| White      | 15.623               | 15.604                |
| Black      | 23.440               | 23.412                |
| Blue Corn  | 0.750                | 0.749                 |
| Blue Steel | 2.946                | 2.942                 |
| Sky Color  | 5.010                | 5.004                 |
| Blue Sky   | 7.500                | 7.491                 |
| Blue Light | 7.702                | 7.693                 |
| Gray       | 7.676                | 7.667                 |

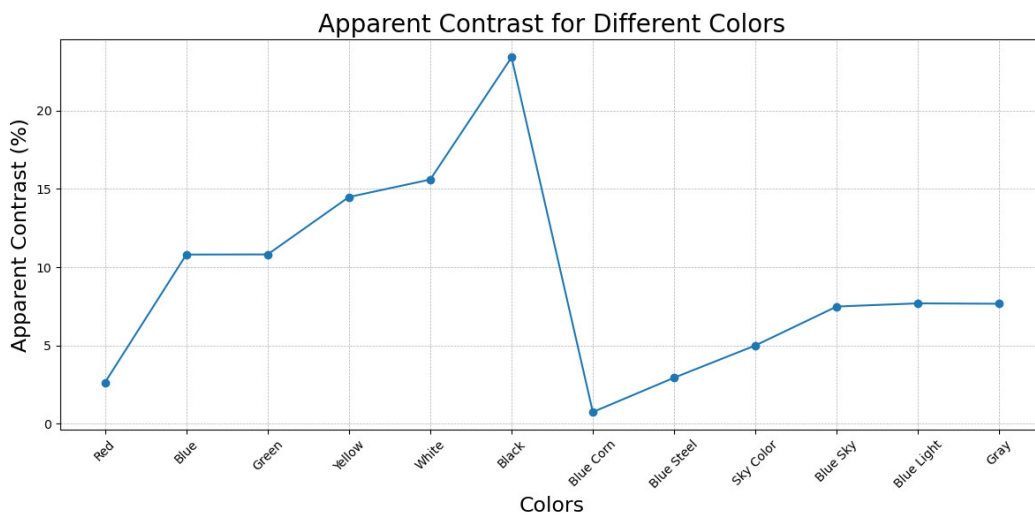


Figure 4.17.: Different colors and their associated apparent contrast at 10 m distance and visibility of 16 km.

Nevertheless, Table 4.6 presents inconsistent results. While the 'Blue Corn' color is set as the one generating the least visual impact, it is worth noting that red is indicated as the second least annoying. This conclusion may not appear intuitive, considering the big difference between the sky color and red. Additionally, the primary colors blue, green and yellow contrast less with the transparent sky background than white. This is not obvious since blue, green, yellow and white are limit points in the CIELAB space model representation, but the blue sky color was thought to be closer to white than the rest of the shades. Conversely, black is rated as the most annoying color, which aligns with its extreme position in the CIELAB space model representation. On another note, the difference between initial and apparent contrast is minimal and sometimes imperceptible. One might question the need to calculate apparent contrast based on this observation. However, these findings contrast with the conclusions of I. D. Bishop (2002: 8), where the difference was consistently at least 0.1%, and atmospheric scattering always played a significant role in color difference.

As a result, determining the suitability of this method for obtaining the visual contrast of a flying object is in question. As previously discussed in Section 4.3, the algorithm should be adjusted to account for differences in color contrast and not just variations in lightness.

### **4.5. Limitations of the Study and Future Work**

There are several aspects that this study does not cover due to the no existing research about Visual Impact Assessment of UAM available and the given time. Moreover, some aspects had to be simplified in order to make this research possible.

One major limitation was the inability to utilize the Raspberry Pi Camera for the data campaign. The Measurement Station was not ready in time, requiring the use of a standard camera as a substitute. Future work could benefit from the Raspberry Pi module, offering a more automated data collection method. However, a disadvantage could be the poorer quality of the images. The effect on data analysis should have to be investigated in detail.

A further limitation was the reliance on the flight vehicle's known altitudes, ground, and lateral distances during the data campaign to calculate the distances between the UAV and the viewer. This was done instead of utilizing the specific UAV coordinates. Since the UAV's position was manually controlled, ensuring consistency was sometimes challenging. The wind conditions and difficulty piloting specific UAVs play an essential role.

For improved accuracy in future studies and if the data is available, specific coordinates are recommended.

Regarding possible future work, the Python code developed for object detection exhibited challenges, mainly when the UAV was significantly distant. Future efforts could focus on improving this algorithm and the isolating code to have more precise results.

For UAV 1, visual size evaluations were conducted only from its tail perspective. Other viewpoints could get different outcomes. Thus, future research could investigate how the visible face of the UAV influences its perceived size. Furthermore, assessing the visual size of UAV 2 would confirm conclusions regarding symmetry.

Lastly, while the two UAVs served as foundational references for this study's methodology, extending the assessments to UAVs of varied sizes, structures, and colors would be beneficial to come to more comprehensive conclusions. Once this is done, the regression model to estimate visual thresholds developed by Shang and Bishop (2000) could also be applied since it was not possible in this thesis due to the obtained results.

## 5. Conclusion

This thesis examines the impact of Urban Air Vehicles on the human field of view and visual experience. Factors such as FOV obstruction, perceived size, and distinctive color contrast have been identified as key determinants of social acceptance. This research introduces an innovative Visual Impact Assessment of Urban Air Vehicles approach. Unlike previous studies that relied on image processing software like GIMP and Adobe Photoshop, this study pioneers an automated methodology primarily utilizing Python—an approach previously unexplored in this context.

Three distinct Python codes were developed to configure a Raspberry Pi Camera for use in a portable measurement station. These codes enable data acquisition at multiple locations, all managed centrally. They offer users the capability to capture photos, videos, or both simultaneously, as well as to efficiently record and organize pertinent data.

The algorithms formulated for evaluating the influence of the different variables demonstrated efficiency and simplicity of implementation. Codes applying absolute color difference equation, *Cozman and Krotkov* and distance calculation were notably successful. However, certain challenges emerged when trying to recognize distance objects, requiring manual zoom adjustments. Similarly, the process to isolate the vehicle from its background was only not successful when the image quality was not optimal.

In terms of visual size, this study has yielded pertinent conclusions. Firstly, as anticipated, visual magnitude decreases with distance. Contrary to expectations, the visual size showed minimal variance when Horyzn UAVs were used as a reference. This occurred even as the UAV maintained a consistent altitude and moved solely in a horizontal plane with different camera orientations. The results obtained from the employed UAVs indicate that its visual size only changes by approximately 15% inside the FOV but by around 300% when its flying distance is doubled. Furthermore, altitude does not appear to be a relevant factor when estimating visual size, except when the flight height is so substantial that the vehicle exits the FOV, in which case it no longer impacts human perception. This finding suggests that the UAV's position within the primary human field of view is insignificant; only its distance from the observer influences it.

Regarding color contrast, the research concludes that the similar the object color is to the background color, the closer these two colors are in the CIELAB color space and therefore, the minimum contrast and impact. Nevertheless, the variable sky shades, influenced by factors such as atmospheric scattering, weather conditions, and lighting, significantly affect this perception. When considering only the color difference, a range of colors is proposed as the least distracting (from #4682b4 to #71BCE1 in the HTML color system) for flying on a clear and sunny day. Additionally, the importance of understanding the flying parameters is emphasized, as the ideal vehicle color may vary depending on its expected position in the sky.

This thesis also assessed the *Cozman and Krotkov formula's* application, which considers atmospheric scattering and different depths. The results revealed inconsistencies, mainly when applied to UAVs with contrasting colors. Previous literature often used the difference in lightness to address apparent contrast. However, it was concluded that using the Euclidean distance between colors would yield a more logical outcome in such cases.

To summarize, this research has demonstrated that visual size and color contrast are closely linked to the vehicle's specific characteristics, implying that findings might differ based on its specific dimensions and intended flying parameters. Most importantly, this study introduces the first methodology for measuring the visual effects of such Urban Air Vehicles. It sets the foundation for further investigations, allowing for a wide range of possibilities.

# A. Python Codes

## A.1. Raspberry Pi Camera Python Codes

In this appendix are the developed Python codes that allow for the remote control of a Raspberry Pi Camera intuitively.

### A.1.1. Raspberry Pi Camera - Code 1

Code A.1: Raspberry Pi Camera code to take pictures in a row.

```
1 from picamera import PiCamera
2 from time import sleep, time
3 import os
4 from datetime import datetime
5
6 def create_folder(path):
7     try:
8         os.makedirs(path)
9     except OSError as e:
10        print(f"Error creating folder {e}")
11
12 def save_txt_file(folder_path, content):
13     file_path = os.path.join(folder_path, "media_data.txt")
14     with open(file_path, "w") as file:
15         file.write(content)
16
17 def take_photos():
18     root_folder = "root folder path"
19     timestamp = datetime.now().strftime("image_%Y%m%d_%H%M%S")
20     folder_path = os.path.join(root_folder, timestamp)
21     create_folder(folder_path)
22     user_content = input("Relevant data for the .txt file: ")
23     save_txt_file(folder_path, user_content)
24
25     try:
26         while True:
27             timestamp = datetime.now().strftime("%Y%m%d_%H%M%S")
28             photo_filename = os.path.join(folder_path,
29             f'image_{timestamp}.jpg')
30             camera.capture(photo_filename)
31             print("Photo taken")
32             sleep(3)
33     except KeyboardInterrupt:
34         camera.close()
35         print("Interrupted")
36 camera = PiCamera()
37 camera.resolution = (1600, 900)
38 #camera.resolution = (2592, 1944) #Maximum resolution for photos
39 camera.start_preview()
40 input("Press enter to start taking pictures")
41 camera.stop_preview()
42 take_photos()
```

### A.1.2. Raspberry Pi Camera - Code 2

Code A.2: Raspberry Pi Camera code to take videos.

```

1 from picamera import PiCamera
2 from time import sleep
3 import os
4 from datetime import datetime
5
6 camera = PiCamera()
7
8 def create_folder(path):
9     if not os.path.exists(path):
10        os.makedirs(path)
11
12 def save_txt_file(folder_path, content):
13     file_path = os.path.join(folder_path, "media_data.txt")
14     with open(file_path, "w") as file:
15         file.write(content)
16
17 def start_camera_preview():
18     camera.start_preview()
19     input("Press enter to start recording")
20     camera.stop_preview()
21
22 def start_video_recording():
23     root_folder = "root folder path"
24     timestamp = datetime.now().strftime("video_%Y%m%d_%H%M%S")
25     folder_path = os.path.join(root_folder, timestamp)
26     create_folder(folder_path)
27     user_content = input("Relevant data for the .txt file: ")
28     save_txt_file(folder_path, user_content)
29     video_filename = os.path.join(folder_path, f'video_{timestamp}.h264')
30     camera.start_recording(video_filename)
31     print("Recording")
32
33     try:
34         while True:
35             pass
36     except KeyboardInterrupt:
37         camera.stop_recording()
38         camera.close()
39         print("Interrupted")
40
41 start_camera_preview()
42 start_video_recording()

```

### A.1.3. Raspberry Pi Camera - Code 3

Code A.3: Raspberry Pi Camera code to simultaneously take pictures and videos.

```

1 from datetime import datetime
2 from picamera import PiCamera
3 from time import sleep, time
4 import os
5
6 def create_folder(path):
7     try:
8         os.makedirs(path)
9     except OSError as e:
10        print(f"Error creating folder {e}")
11
12 def save_txt_file(folder_path, content):
13     file_path = os.path.join(folder_path, "media_data.txt")
14     with open(file_path, "w") as file:
15         file.write(content)
16
17 def capture_frame(folder_path, interval):
18     timestamp = datetime.now().strftime("%Y%m%d_%H%M%S")
19     image_filename = os.path.join(folder_path, f'capture_{timestamp}.jpg')
20     camera.capture(image_filename)
21
22 def start_recording(folder_path):
23     timestamp = datetime.now().strftime("%Y%m%d_%H%M%S")
24     folder_name = datetime.now().strftime("video_capture_%Y%m%d_%H%M%S")
25     folder_path = os.path.join('root folder path', folder_name)
26     create_folder(folder_path)
27     user_content = input("Relevant data for the .txt file: ")
28     save_txt_file(folder_path, user_content)
29     video_filename = os.path.join(folder_path, f'video_{timestamp}.h264')
30     camera.start_recording(video_filename)
31     print("Recording")
32     try:
33         while True:
34             interval = 5
35             capture_frame(folder_path, interval)
36             sleep(interval)
37             print("Photo taken")
38     except KeyboardInterrupt:
39         camera.stop_recording()
40         camera.close()
41         print("Interrupted")
42
43 camera = PiCamera()
44 camera.resolution = (1600, 900) #closest resolution to human eye
45 #camera.resolution =(1920,1080) #maximum resolution for videos
46
47 camera.start_preview()
48 input("Press enter to start recording")
49 camera.stop_preview()
50
51 folder_path = 'root folder path'
52
53 start_recording(folder_path)

```



## A.2. Visual Size Python Codes

This section includes the complete Python Codes developed to compute the visual size of an object within an image.

### A.2.1. Detection and scale computation - Code

Code A.4: Detection and scale computation (Part 1).

```

1
2 import cv2
3 import cvlib
4 from cvlib.object_detection import draw_bbox
5 import matplotlib.pyplot as plt
6
7 def calculate_distance(x1, y1, x2, y2):
8     horizontal_dist = x2 - x1
9     vertical_dist = y2 - y1
10    return horizontal_dist, vertical_dist
11
12 def calculate_scale_ratio(pixel_distance, real_distance_cm):
13    return real_distance_cm / pixel_distance
14
15 dimensions = []
16
17 def image_Object_Detection(image, scale_ratio=None):
18    img = cv2.imread(image)
19    bbox, labels, _ = cvlib.detect_common_objects(img)
20
21    output_image = img.copy()
22    airplane_box = None
23
24    for (box, label) in zip(bbox, labels):
25        if label in ['airplane', 'bird']:
26            x1, y1, x2, y2 = box
27            cv2.rectangle(output_image, (x1, y1), (x2, y2),
28                (0, 255, 0), 5)
29
30            horizontal_dist, vertical_dist = calculate_distance
31                (x1, y1, x2, y2)
32            print(f'{{label}} at ({{x1}}, {{y1}}) to ({{x2}}, {{y2}}):')
33            print(f'Distance (horizontal): {{horizontal_dist}} pixels')
34            print(f'Distance (vertical): {{vertical_dist}} pixels')
35
36            if not airplane_box:
37                airplane_box = (x1, y1, x2, y2)
38
39            if scale_ratio:
40                real_horizontal_distance = horizontal_dist * scale_ratio
41                real_vertical_distance = vertical_dist * scale_ratio
42                print(f'Real Dimensions: {{real_horizontal_distance:.2f}}cm
43                    (horizontal),{{real_vertical_distance:.2f}} cm (vertical)')
44                dimensions.append([real_horizontal_distance,
45                    real_vertical_distance])

```

Code A.5: Detection and scale computation (Part 2).

```

1
2     output_image = cv2.cvtColor(output_image, cv2.COLOR_BGR2RGB)
3
4     return output_image, sum(1 for label in labels if label in
5     ["airplane", "kite"]), airplane_box
6
7 def process_images():
8     main_image = 'main image path'
9     other_images = [
10        'studied image paths'
11    ]
12
13    main_output, main_count, airplane_detected_box =
14    image_Object_Detection(main_image)
15    print(f"Reference image: {main_count} air vehicle(s) detected")
16
17    if airplane_detected_box:
18        x1, _, x2, _ = airplane_detected_box
19        ref_horizontal_dist, _ = calculate_distance(x1, 0, x2, 0)
20    else:
21        print("No 'airplane' detected in main_image!")
22        ref_horizontal_dist = 0
23
24    scale_ratio = calculate_scale_ratio(ref_horizontal_dist, 128)
25    print(f"Scale: 1 pixel = {scale_ratio:.2f} cm")
26
27    plt.imshow(main_output)
28    plt.title('Reference Image')
29    plt.show()
30
31    for idx, img in enumerate(other_images, 1):
32        output_image, detected_count = image_Object_Detection(img,
33        scale_ratio=scale_ratio)[0:2]
34        print(f"Image {idx}: {detected_count} air vehicle(s) detected")
35
36        plt.imshow(output_image)
37        plt.title(f'Image {idx}')
38        plt.show()
39
40 if __name__ == "__main__":
41     process_images()

```

## A.2.2. Distance calculation - Code 1

Code A.6: Distance calculation using Haversine Equation (Approach 1).

```

1 import math
2
3 def haversine_distance(lat1, lon1, lat2, lon2):
4     r = 6371
5     lat1_rad = math.radians(lat1)
6     lon1_rad = math.radians(lon1)
7     lat2_rad = math.radians(lat2)
8     lon2_rad = math.radians(lon2)
9
10    delta_lat = lat2_rad - lat1_rad
11    delta_lon = lon2_rad - lon1_rad
12
13    a = math.sin(delta_lat/2)**2 + math.cos(lat1_rad)
14    * math.cos(lat2_rad) * math.sin(delta_lon/2)**2
15    c = 2 * math.atan2(math.sqrt(a), math.sqrt(1 - a))
16
17    distance = r * c * 1000
18    return distance
19
20    lat1 = #latitude of the camera position
21    lon1 = #longitude of the camera position
22
23    positions_obj2 = ['different UAV coordinates']
24
25    height_object = #air vehicle height in meters
26    height_camera = #camera height in meters
27    dist_ver = height_object - height_camera
28
29    distances = []
30
31    for lat2, lon2 in positions_obj2:
32        dist_hor = haversine_distance(lat1, lon1, lat2, lon2)
33        dist = math.sqrt(dist_hor**2 + dist_ver**2)
34        distances.append(dist)
35    print(f"Distance from the camera to the air vehicle {len(distances)}:
36    ", dist, "meters")

```

### A.2.3. Distance Calculation - Code 2

Code A.7: Distance calculation using Pythagoras' Theorem twice (Approach 2).

```

1 import math
2
3 def calculate_distance(dist_cam_land_0, height, dist_lat):
4     dist_cam_land = math.sqrt(dist_cam_land_0**2 + dist_lat**2)
5     return math.sqrt(dist_cam_land**2 + height**2)
6
7 dist_cam_land_0 = # distance taking off-camera
8 height = # flight height
9 height_camera = 1.60
10 real_height = height-height_camera
11 dist_lats = [lateral1, lateral2, lateral3, lateral4]
12
13 dist0=math.sqrt(dist_cam_land_0**2 + real_height**2)
14 print(f"Distance from the camera to the air vehicle 0:", dist0, "meters")
15
16 distances = []
17
18 for dist_lat in dist_lats:
19     dist = calculate_distance(dist_cam_land_0, real_height, dist_lat)
20     distances.append(dist)
21     print(f"Distance from the camera to the air vehicle for lateral
22         distance from the landing position {dist_lat} m: {dist:.2f} meters")

```

### A.2.4. Computation of visual size - Code

Code A.8: Visual size formula application.

```

1 import math
2 from detection_scale import dimensions, process_images
3 from obtain_distance import distances
4
5 process_images()
6
7 S_values = []
8
9 beta = (180**2 * 60**2) / math.pi**2
10
11 for (H, D), d in zip(dimensions, distances):
12     alpha = H * D / (d*100)**2
13     S = beta * alpha
14     S_values.append(S)
15
16 for i, S in enumerate(S_values):
17     print(f"Visual size {i + 1} = {S}")

```

### A.3. Absolute Color Difference Python Codes

In this appendix are presented the needed Python algorithms to isolate the air vehicle from the sky and compute the specific absolute color difference formula.

#### A.3.1. Isolating object and background - Code

Code A.9: Isolate object and background.

```

1 import cv2
2 import numpy as np
3
4 def segment_image(image_path, save_object_path, save_background_path,
5   threshold_value=100):
6     image = cv2.imread(image_path, cv2.IMREAD_COLOR)
7     gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
8     _, thresholded = cv2.threshold(gray, threshold_value, 255,
9     cv2.THRESH_BINARY)
10    mask = thresholded == 255
11    height, width, channels = image.shape
12
13    transparent_obj_img = np.zeros((height, width, 4), dtype=np.uint8)
14    transparent_obj_img[:, :, :3] = image
15    transparent_obj_img[mask, 3] = 255
16    cv2.imwrite(save_object_path, transparent_obj_img)
17
18    transparent_bg_img = np.zeros((height, width, 4), dtype=np.uint8)
19    transparent_bg_img[:, :, :3] = image
20    transparent_bg_img[~mask, 3] = 255
21    cv2.imwrite(save_background_path, transparent_bg_img)
22
23 image_path = 'image path'
24 output_object_path = 'output object path'
25 output_background_path = 'output background path'
26 segment_image(image_path, output_object_path, output_background_path)

```

### A.3.2. Mean LAB values and $\Delta E$ - Code

Code A.10: Absolute color difference computation.

```

1
2 from PIL import Image
3 import numpy as np
4 from skimage import color
5 import matplotlib.pyplot as plt
6
7 ref_image_path = 'reference image path'
8 ref_image = Image.open(ref_image_path)
9 plt.imshow(ref_image)
10 plt.show()
11
12 def get_mean_lab_values(image_path):
13     img = Image.open(image_path)
14     plt.imshow(img)
15     plt.show()
16     img_arr = np.asarray(img)
17     non_transparent_pixels = np.where(img_arr[..., -1] > 0)
18     img_arr = img_arr[:, :, :3]
19     lab_img = color.rgb2lab(img_arr)
20     L_mean = np.mean(lab_img[non_transparent_pixels][:, 0])
21     A_mean = np.mean(lab_img[non_transparent_pixels][:, 1])
22     B_mean = np.mean(lab_img[non_transparent_pixels][:, 2])
23     return L_mean, A_mean, B_mean
24
25 object_path = 'object image path'
26 background_path = 'background image path'
27
28 L_mean_object, A_mean_object, B_mean_object =
29 get_mean_lab_values(object_path)
30 print("L_mean_object:", L_mean_object)
31 print("A_mean_object:", A_mean_object)
32 print("B_mean_object:", B_mean_object)
33
34 L_mean_background, A_mean_background, B_mean_background =
35 get_mean_lab_values(background_path)
36 print("L_mean_background:", L_mean_background)
37 print("A_mean_background:", A_mean_background)
38 print("B_mean_background:", B_mean_background)
39
40 E_abs = ((L_mean_background - L_mean_object) ** 2 +
41 (A_mean_background - A_mean_object) ** 2 +
42 (B_mean_background - B_mean_object) ** 2) ** 0.5
43 print("Absolute colour difference:", E_abs)

```

## A.4. Cozman and Krotkov Formula Application Code

Here is presented the formulated algorithm to apply Cozman and Krotkov Formula that enables to study color contrast at different depths and visibilities taking into account atmospheric scattering.

Code A.11: Raspberry Pi Camera code to simultaneously take pictures and videos.

```

1 from PIL import Image
2 import numpy as np
3 from skimage import color
4 import matplotlib.pyplot as plt
5 import math
6
7 def get_mean_luminosity(image_path):
8     img = Image.open(image_path)
9     plt.imshow(img)
10    plt.show()
11    img_arr = np.asarray(img)
12    non_transparent_pixels = np.where(img_arr[..., -1] > 0)
13    img_arr = img_arr[:, :, :3]
14    lab_img = color.rgb2lab(img_arr)
15    L_mean = np.mean(lab_img[non_transparent_pixels][:, 0])
16    return L_mean
17
18 object_path = 'object image path'
19 background_path = 'background image path'
20
21 L_mean_object_0 = get_mean_luminosity(object_path)
22 print("L_mean_object_initial:", L_mean_object_0)
23
24 L_mean_background_0 = get_mean_luminosity(background_path)
25 print("L_mean_background_initial:", L_mean_background_0)
26
27 visibility = #visibility value in meters
28 d = # distance between object and observer in meters
29 c_i = abs((L_mean_object_0 - L_mean_background_0)) * (100/256)
30 print(f"Initial Contrast is {c_i} %")
31 beta = 1.932 / visibility
32 print(f"Beta is {beta} ")
33 c_d = (c_i)*math.exp(-beta*d)
34 print(f"The contrast at a distance of {d} m is {c_d} %")

```

# B. Flight Test Data Analysis

## B.1. Visual Size Assessment Procedure

In this appendix, photographs of UAV 1 taken during the Flight Campaign on the 23rd of August are presented. These images have been processed using Python to estimate variations in visual size. Additionally, an example of UAV 2 visual size measurement is also presented.

### B.1.1. Photographs with 0° orientation camera and 2m altitude

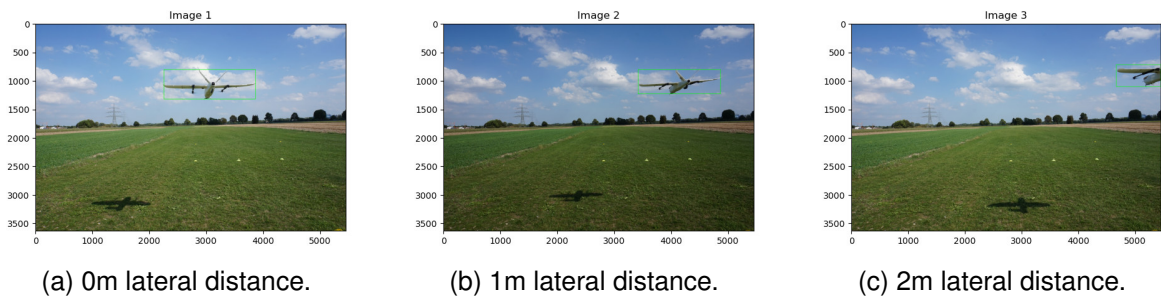


Figure B.1.: Photographs at 3m ground distance, 2m altitude and 0° orientation.

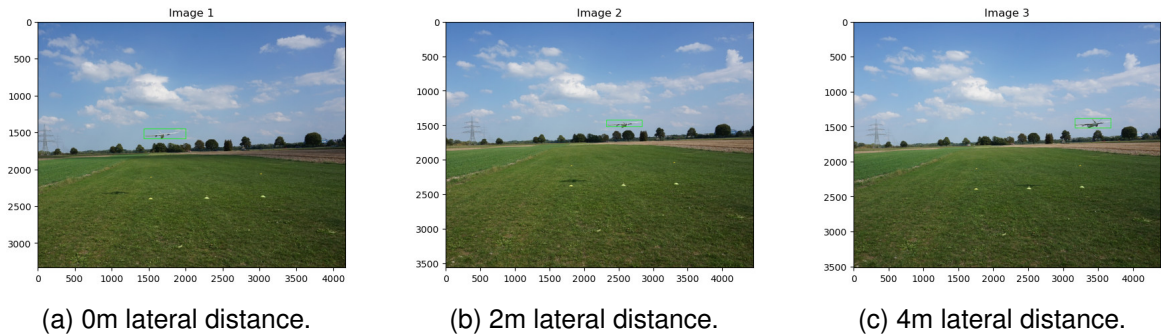


Figure B.2.: Photographs at 10m ground distance, 2m altitude and 0° orientation.

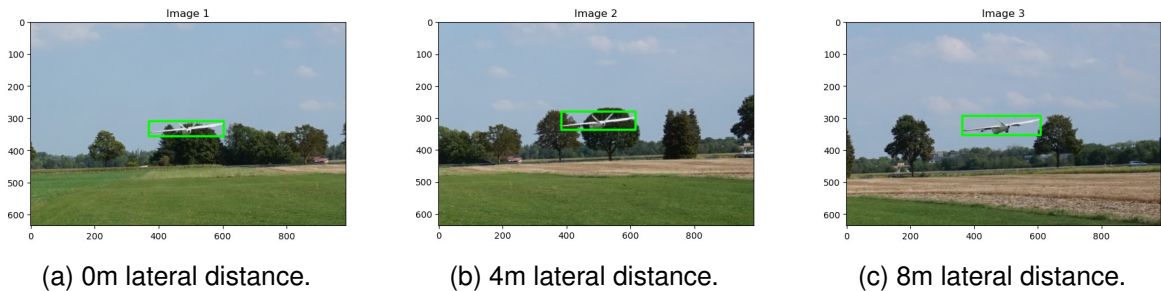


Figure B.3.: Photographs at 20m ground distance, 2m altitude and 0° orientation.



**B.1.2. Photographs with 0° orientation camera and 5m altitude**

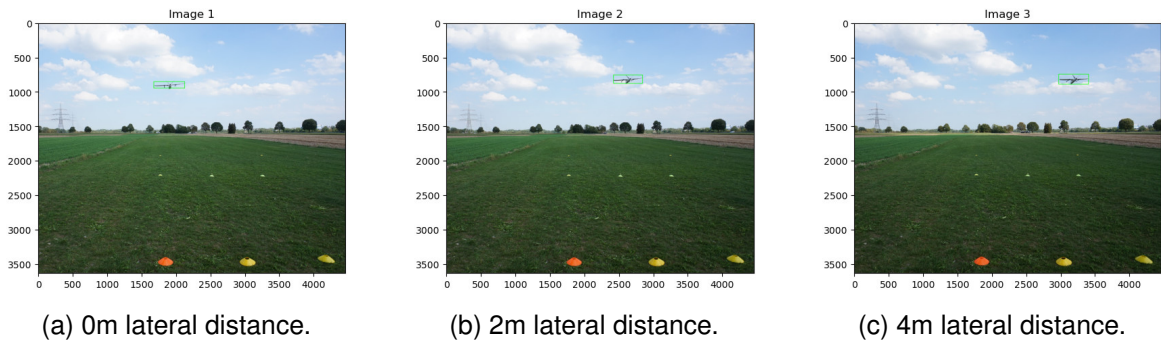


Figure B.4.: Photographs at 10m ground distance, 5m altitude and 0° orientation.

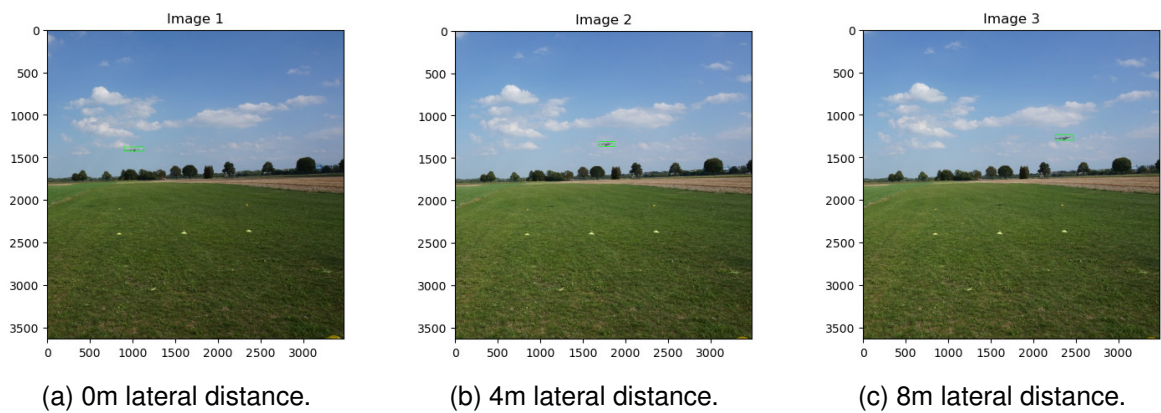


Figure B.5.: Photographs at 20m ground distance, 5m altitude and 0° orientation.

**B.1.3. Photographs with 30° orientation camera and 2m altitude**

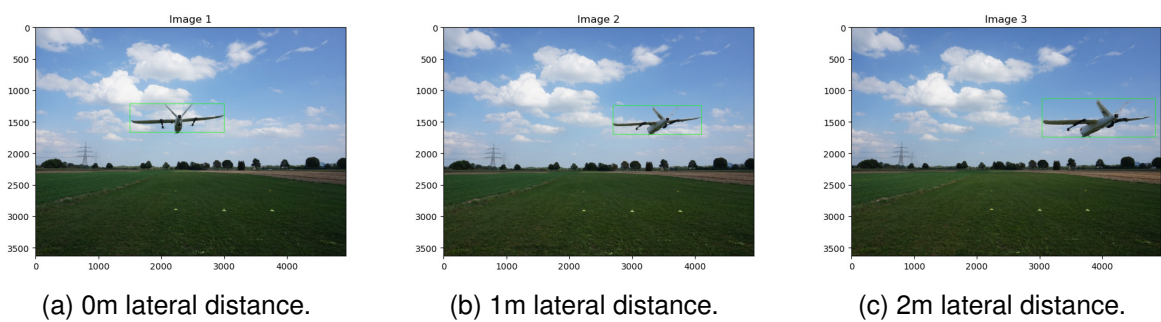


Figure B.6.: Photographs at 3m ground distance, 2m altitude and 30° orientation.

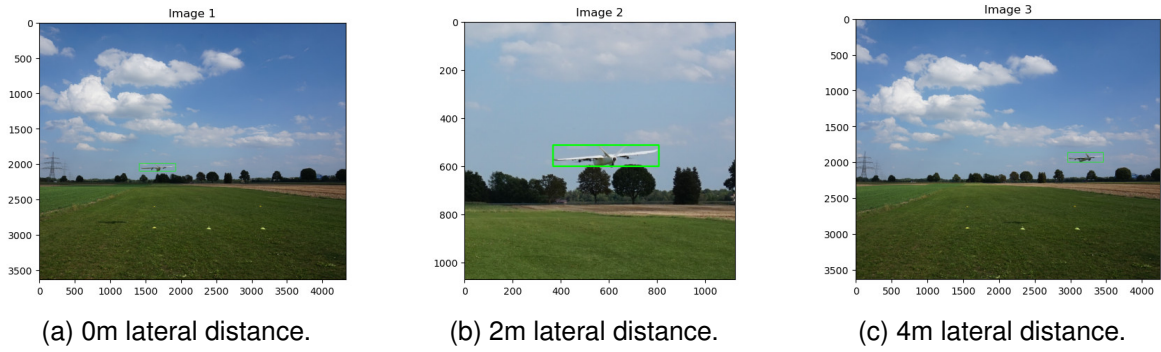


Figure B.7.: Photographs at 10m ground distance, 2m altitude and  $30^\circ$  orientation.

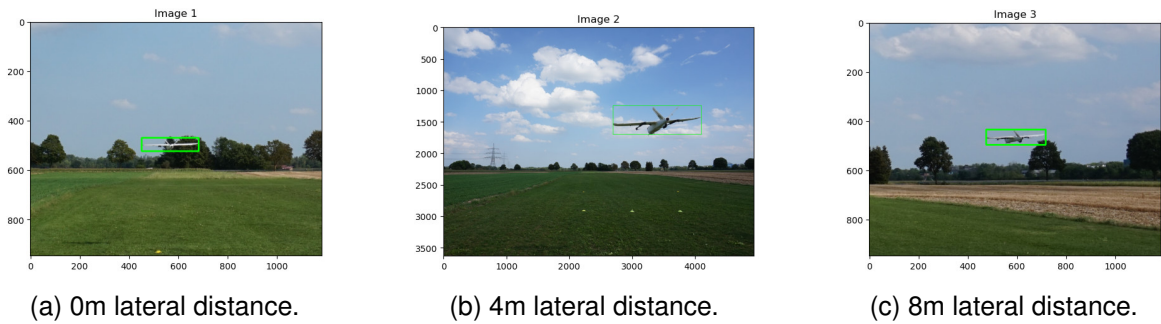


Figure B.8.: Photographs at 20m ground distance, 2m altitude and  $30^\circ$  orientation.

**B.1.4. Photographs with  $30^\circ$  orientation camera and 2m altitude**

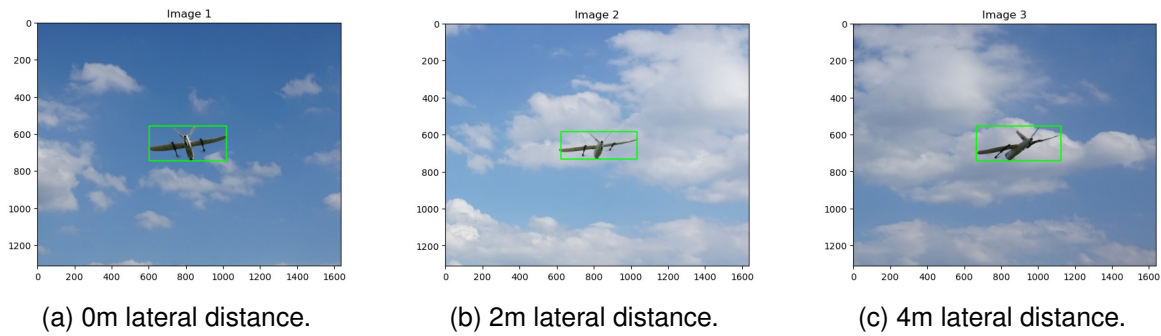


Figure B.9.: Photographs at 10m ground distance, 5m altitude and  $30^\circ$  orientation.

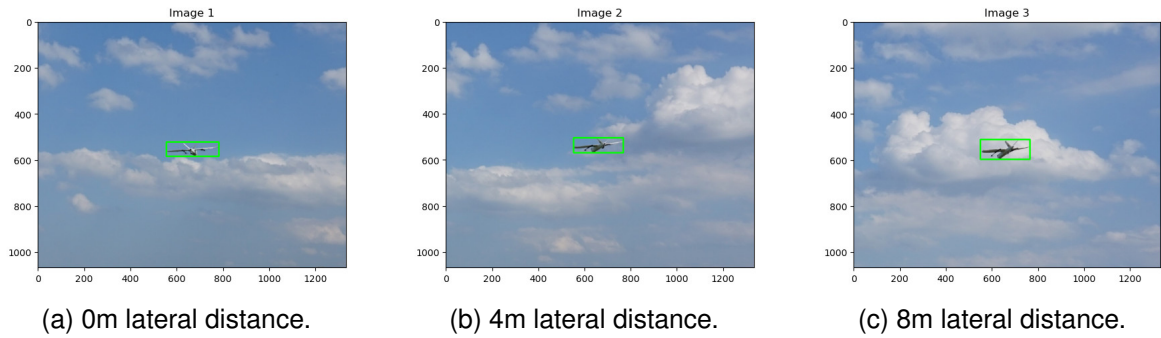


Figure B.10.: Photographs at 20m ground distance, 5m altitude and 30° orientation.

**B.1.5. Photographs with 0° orientation camera and 5m altitude (UAV 2)**

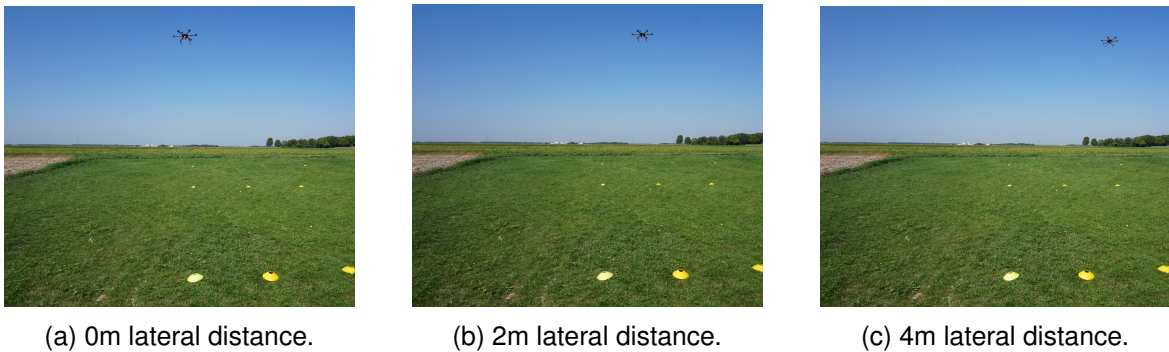


Figure B.11.: Photographs at 10m ground distance, 5m altitude and 0° orientation (UAV 2).

**B.2. Color Contrast Isolating object**



Figure B.12.: Case 1 isolating process.



Figure B.13.: Case 2 isolating process.

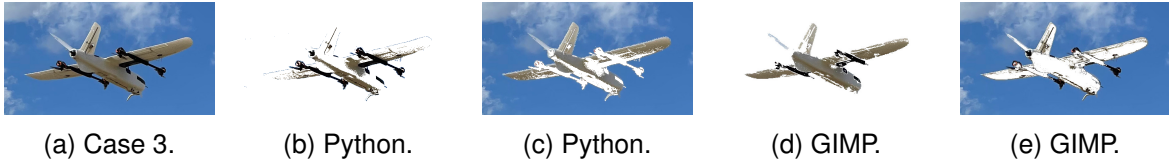


Figure B.14.: Case 3 isolating process.

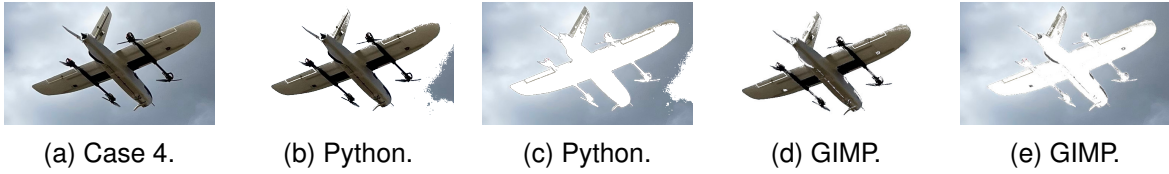


Figure B.15.: Case 4 isolating process.

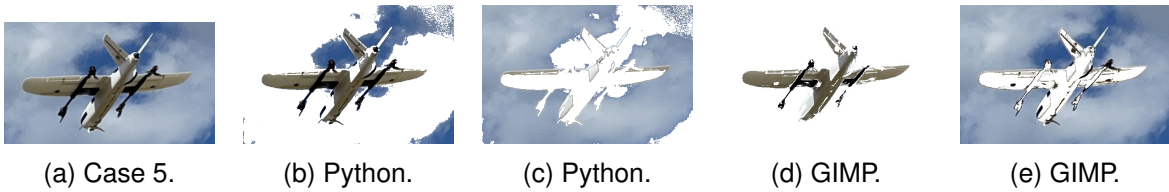


Figure B.16.: Case 5 isolating process.

### B.3. Visual Size Numerical Results

| Degrees | Altitude (m) | Ground Dist (m) | Lateral Dist (m) | Distance (m) | Scale (cm/px) | Horizontal Dimension (cm) | Vertical Dimension (cm) | Visual Size (arcmin <sup>2</sup> ) |
|---------|--------------|-----------------|------------------|--------------|---------------|---------------------------|-------------------------|------------------------------------|
| 0       | 2            | 3               | 0                | 3.03         | 0.08          | 128.00                    | 41.23                   | 680951.1326                        |
| 0       | 2            | 3               | 1                | 3.19         | 0.08          | 115.18                    | 33.91                   | 454339.0215                        |
| 0       | 2            | 10              | 0                | 10.01        | 0.26          | 128.00                    | 29.82                   | 45030.00918                        |
| 0       | 2            | 10              | 2                | 10.21        | 0.26          | 126.46                    | 27.24                   | 39091.20124                        |
| 0       | 2            | 10              | 4                | 10.78        | 0.26          | 130.31                    | 36.76                   | 48730.00016                        |
| 0       | 2            | 20              | 0                | 20.00        | 0.55          | 128.00                    | 25.71                   | 9718.875774                        |
| 0       | 2            | 20              | 4                | 20.40        | 0.55          | 126.91                    | 31.18                   | 11236.69249                        |
| 0       | 2            | 20              | 8                | 21.54        | 0.55          | 134.56                    | 32.82                   | 11244.8751                         |
| 0       | 5            | 10              | 0                | 10.56        | 0.29          | 128.00                    | 26.8                    | 36336.30078                        |
| 0       | 5            | 10              | 2                | 10.75        | 0.29          | 120.02                    | 35.06                   | 43038.25922                        |
| 0       | 5            | 10              | 4                | 11.29        | 0.29          | 123.44                    | 41.62                   | 47599.37655                        |
| 0       | 5            | 20              | 0                | 20.29        | 0.56          | 128.00                    | 27.83                   | 10227.67018                        |
| 0       | 5            | 20              | 4                | 20.68        | 0.56          | 100.17                    | 32.28                   | 8937.487823                        |
| 0       | 5            | 20              | 8                | 21.81        | 0.56          | 118.54                    | 36.73                   | 10820.07456                        |
| 30      | 2            | 3               | 0                | 3.03         | 0.09          | 128.00                    | 38.83                   | 641324.887                         |
| 30      | 2            | 3               | 1                | 3.19         | 0.09          | 119.74                    | 38.92                   | 542072.7259                        |
| 30      | 2            | 3               | 2                | 3.63         | 0.09          | 153.97                    | 52.03                   | 719504.4608                        |
| 30      | 2            | 10              | 0                | 10.01        | 0.25          | 128.00                    | 26.95                   | 40698.62898                        |
| 30      | 2            | 10              | 2                | 10.21        | 0.25          | 109.79                    | 21.96                   | 27350.63111                        |
| 30      | 2            | 10              | 4                | 10.78        | 0.25          | 124.01                    | 33.93                   | 42812.60642                        |
| 30      | 2            | 20              | 0                | 20.00        | 0.55          | 128.00                    | 29.92                   | 11311.38568                        |
| 30      | 2            | 20              | 4                | 20.40        | 0.55          | 128.55                    | 31.58                   | 11530.44984                        |
| 30      | 2            | 20              | 8                | 21.54        | 0.55          | 134.10                    | 34.35                   | 11729.59695                        |
| 30      | 5            | 10              | 0                | 10.56        | 0.31          | 128.00                    | 57.85                   | 78437.49544                        |
| 30      | 5            | 10              | 2                | 10.75        | 0.31          | 126.15                    | 45.54                   | 58751.51669                        |
| 30      | 5            | 10              | 4                | 11.29        | 0.31          | 140.62                    | 57.54                   | 74959.09588                        |
| 30      | 5            | 20              | 0                | 20.29        | 0.56          | 128.00                    | 34.96                   | 12849.91866                        |
| 30      | 5            | 20              | 4                | 20.68        | 0.56          | 121.23                    | 36.65                   | 12282.04654                        |
| 30      | 5            | 20              | 8                | 21.81        | 0.56          | 121.23                    | 48.49                   | 14609.91144                        |

Table B.1.: Visual size analysis results

# Bibliography

- United Nations (2018). *World Urbanization Prospects*. URL: <https://population.un.org/wup/Download/> (cit. on p. 1).
- Cohen, Adam P., Susan A. Shaheen, and Emily M. Farrar (2021). "Urban Air Mobility: History, Ecosystem, Market Potential, and Challenges". In: *IEEE Transactions on Intelligent Transportation Systems* 22.9, pp. 6074–6087. DOI: 10.1109/tits.2021.3082767 (cit. on p. 3).
- Straubinger, Anna et al. (2020). "An Overview of Current Research and Developments in Urban Air Mobility – Setting the Scene for UAM Introduction". In: *Journal of Air Transport Management* 87, p. 101852. DOI: 10.1016/j.jairtraman.2020.101852 (cit. on pp. 3, 4).
- NASA (2001). *SATS: A bold vision: NASA-led technology development aimed at increasing mobility, access for smaller communities*. FS-2001-03-59-LaRC. URL: <https://www.nasa.gov/centers/langley/news/factsheets/SATS.html> (cit. on p. 3).
- Volocopter (Oct. 2011). *Manned First Flight Writes Aviation History*. Karlsruhe, Germany in October 2011. URL: <https://www.volocopter.com/newsroom/manned-first-flight-writes-aviation-history/> (cit. on p. 3).
- EASA (2021a). *Urban Air Mobility (UAM) - Frequently Asked Questions*. Tech. rep. European Union Aviation Safety Agency (EASA). URL: [https://www.easa.europa.eu/sites/default/files/dfu/uam\\_-\\_faqs.pdf](https://www.easa.europa.eu/sites/default/files/dfu/uam_-_faqs.pdf) (cit. on pp. 3, 4).
- Gillis, Dominique et al. (Sept. 2021). "Urban Air Mobility: A State of Art Analysis". In: pp. 411–425. ISBN: 978-3-030-86959-5. DOI: 10.1007/978-3-030-86960-1\_29 (cit. on p. 4).
- Volocopter GmbH (2017). *Volocopter*. URL: <http://www.volocopter.com> (visited on 07/20/2017) (cit. on p. 4).
- CityAirbus (2021). *Technology*. URL: <https://www.airbus.com/en/newsroom/press-releases/2021-09-airbus-reveals-the-next-generation-of-cityairbus> (visited on 09/21/2021) (cit. on p. 4).
- Rajendran, Sivaprasad and Sushma Srinivas (2020). "Air taxi service for urban mobility: A critical review of recent developments, future challenges, and opportunities". In: *Transportation Research Part E: Logistics and Transportation Review* 143, p. 102090. DOI: 10.1016/j.tre.2020.102090 (cit. on p. 4).
- Energy Statistics Data Browser* (2022). Accessed on July 3, 2023. International Energy Agency. URL: <https://www.iea.org/data-and-statistics/data-tools/energy-statistics-data-browser> (cit. on p. 5).
- European Environment Agency (2021). *Decarbonising road transport – the role of vehicles, fuels and transport demand*. Tech. rep. European Environment Agency. URL: <https://www.eea.europa.eu/publications/transport-and-environment-report-2021> (visited on 06/29/2023) (cit. on pp. 5, 6).
- Kasliwal, A. et al. (Dec. 2019). "Role of flying cars in sustainable mobility". In: *Nature Commun.* 10.1, p. 1555. DOI: 10.1038/s41467-019-09344-7 (cit. on p. 6).
- Connors, M. M. (2019). *Factors that Influence Community's Acceptance of Noise: An Introduction for Urban Air Mobility*. Tech. rep. NASA Ames Research Center. URL: <https://ntrs.nasa.gov/api/citations/20190032256/downloads/20190032256.pdf> (cit. on pp. 6, 7, 17).
- EASA (2021b). *Study on the societal acceptance of Urban Air Mobility in Europe*. Tech. rep. European Union Aviation Safety Agency (cit. on pp. 7, 9, 10).

- Beutel, M. E. et al. (2016). "Noise Annoyance Is Associated with Depression and Anxiety in the General Population - The Contribution of Aircraft Noise". In: *PLoS ONE* 11.5, e0155357. DOI: 10.1371/journal.pone.0155357 (cit. on p. 7).
- Kwon, J., Y. Kim, and Park (2017). "Applying LSA text mining technique in envisioning social impacts of emerging technologies: the case of drone technology". In: *Technovation*. DOI: 10.1016/j.technovation.2017.01.001 (cit. on p. 7).
- Yedavalli, Pavan and Jessie Mooberry (2019). "An Assessment of Public Perception of Urban Air Mobility (UAM)". In: *Airbus UTM: Defining Future Skies*, pp. 1–28 (cit. on pp. 7, 8).
- Thomas, Kilian and Tobias A. Granberg (2023). "Quantifying Visual Pollution from Urban Air Mobility". In: *Drones* 7.6. ISSN: 2504-446X. URL: <https://www.mdpi.com/2504-446X/7/6/396> (cit. on pp. 7, 8).
- Said, Mohamed et al. (Aug. 2021). "Visual pollution manifestations negative impacts on the people of Saudi Arabia". In: *International Journal of ADVANCED AND APPLIED SCIENCES* 8, pp. 94–101. DOI: 10.21833/ijaas.2021.09.013 (cit. on p. 8).
- Hasan, Shahab (2019). "Urban Air Mobility (UAM) Market Study". In: *NASA*, pp. 1–148 (cit. on pp. 8, 9).
- Shaw, I. (2016). "The urbanization of drone warfare: policing surplus populations in the dronopolis". In: *Geograph. Helv.* DOI: 10.5194/gh-71-19-2016 (cit. on p. 10).
- Chmielewski, S. (2020). "Chaos in motion: Measuring visual pollution with tangential view landscape metrics". In: *Land* 9, p. 515. DOI: 10.3390/land9120515. URL: <https://www.mdpi.com/2073-445X/9/12/515> (cit. on p. 10).
- Empire, Understanding (2016). *The Urbanization of Drone Warfare: Policing Surplus Populations in the Dronopolis*. URL: <https://understandingempire.wordpress.com/2016/02/03/the-urbanization-of-drone-warfare-policing-surplus-populations-in-the-dronopolis/> (cit. on p. 10).
- Sullivan, Robert and Mark Meyer (Aug. 2014). *Guide To Evaluating Visual Impact Assessments for Renewable Energy Projects*. DOI: 10.13140/2.1.3216.5767 (cit. on p. 11).
- Bishop, I. D. (2002). "Determination of Thresholds of Visual Impact: The Case of Wind Turbines". In: *Environment and Planning B: Planning and Design* 29.5, pp. 707–718. DOI: 10.1068/b12854. URL: <https://doi.org/10.1068/b12854> (cit. on pp. 12, 14, 40, 41, 62).
- Garnero, G. and E. Fabrizio (2015). "Visibility analysis in urban spaces: a raster-based approach and case studies". In: *Environment and Planning B: Planning and Design* 42.4, pp. 688–707. DOI: 10.1068/b130119p (cit. on p. 12).
- Holladay, Jack T. (Feb. 2004). "Visual acuity measurements". In: *Journal of Cataract & Refractive Surgery* 30.2, pp. 287–290. DOI: 10.1016/j.jcrs.2004.01.014 (cit. on p. 13).
- ALPF (2022). *Uncorrected and Corrected Visual Acuity*. <https://www.alpfmedical.info/visual-acuity/basic-knowledge-itx.html>. Accessed June 5 2023 (cit. on p. 13).
- Caltrider, D., A. Gupta, and K. Tripathy (2023). "Evaluation of Visual Acuity". In: *StatPearls* (cit. on p. 13).
- Foley, H. J. and M. W. Matlin (2009). *Sensation and Perception*. New York: Pearson College Division (cit. on p. 13).
- Mokrzycki, Wojciech and Mirosław Tatol (2011). *Color difference Delta E - A survey*. [https://www.researchgate.net/publication/236023905\\_Color\\_difference\\_Delta\\_E\\_-\\_A\\_survey](https://www.researchgate.net/publication/236023905_Color_difference_Delta_E_-_A_survey) (cit. on pp. 14, 15).
- Magill, A. W. and R. B. J. Litton (1986). "A color measuring system for landscape assessment". In: *Landscape J.* 5.1, pp. 45–54 (cit. on p. 14).
- Skrok, Daniel (2022). *What are Color Modes?* Accessed: 03-05-2023. URL: <https://www.interaction-design.org/literature/article/what-are-color-modes> (cit. on p. 15).

- Bishop, I. D. (2021). "Analysis and visualization of temporal variation in visual impacts". In: *Landscape and Urban Planning* 210, p. 104068. DOI: 10.1016/j.landurbplan.2021.104068. URL: <https://doi.org/10.1016/j.landurbplan.2021.104068> (cit. on p. 15).
- Jnido, Ghiath, Gisela Ohms, and Wolfgang Viöl (July 2019). "Deposition of TiO<sub>2</sub> Thin Films on Wood Substrate by an Air Atmospheric Pressure Plasma Jet". In: *Coatings* 9, p. 441. DOI: 10.3390/coatings9070441 (cit. on p. 16).
- Rayleigh (Feb. 1871). "XV. On the light from the sky, its polarization and colour". In: *Philosophical Magazine Series 4* 41.271, pp. 107–120 (cit. on p. 16).
- Mie, G. (Jan. 1908). "Beitrage zur optik trüber medien, speziell kolloidaler metallösungen". In: *Ann. Phys.* 330.3, pp. 377–445 (cit. on p. 16).
- Lopes, Diogo and António Ramires Fernandes (Nov. 2014). "Atmospheric Scattering -State of the Art". In: (cit. on pp. 16–18).
- Klassen, R. V. (July 1987). "Modeling the effect of the atmosphere on light". In: *ACM Transactions on Graphics* 6.3, pp. 215–237 (cit. on p. 17).
- Cozman, F and E Krotkov (June 1997). "Depth from scattering". In: *Proceedings of Computer Vision and Pattern Recognition*. IEEE Computer Society, Washington, DC, pp. 801–806 (cit. on p. 18).
- Malm, William (1983). "Introduction to Visibility". In: *National Park Service, Air and Water Quality Division, Air Research Branch*, pp. 1–70 (cit. on pp. 18, 19).
- McCartney, E. J. (1976). *Optics of the Atmosphere*. New York: John Wiley and Sons, Inc. (cit. on p. 18).
- Chang, D., Y. Song, and B. Liu (2009). "Visibility trends in six megacities in China 1973–2007". In: *Atmospheric Research* 94.2, pp. 161–167. DOI: 10.1016/j.atmosres.2009.05.006. URL: <https://doi.org/10.1016/j.atmosres.2009.05.006> (cit. on p. 19).
- Bishop, I. D. (2019). "The implications for visual simulation and analysis of temporal variation in the visibility of wind turbines". In: *Landscape and Urban Planning* 184, pp. 59–68. DOI: 10.1016/j.landurbplan.2018.12 (cit. on p. 19).
- Town, Mountain (2023). *I-70 EB closed at MM-255*. Accessed: 09-06-2023. URL: <https://mymountaintown.com/forum/46-scanner-emergency-info-weather-forecasts/332060-i-70-eb-closed-at-mm-255-left-lane-wb-closed-due-to-accident> (cit. on p. 19).
- Shang, Haidong and Bishop (2000). "Visual thresholds for detection, recognition and visual impact in landscape settings". In: *Journal of Environmental Psychology* 20, pp. 125–140 (cit. on pp. 20, 21, 63).
- Dember, W. N. (1960). *The Physiology of Perception*. New York: Holt, Rhinehart and Winston (cit. on p. 20).
- Ellis, Stephen et al. (Oct. 2002). "Augmented Reality in a Simulated Tower Environment: Effect of Field of View on Aircraft Detection". In: (cit. on p. 21).
- Torrejon, Alfonso, Victor Callaghan, and Hani Hagraas (Nov. 2013). "Panoramic Audio and Video: towards an Immersive Learning experience". In: (cit. on p. 22).
- Roberts, Beth and Juliet Osborne (May 2019). "Testing the efficacy of a thermal camera as a search tool for locating wild bumble bee nests". In: *Journal of Apicultural Research* 58, pp. 1–7. DOI: 10.1080/00218839.2019.1614724 (cit. on p. 22).
- Madhusanka, Achintha and Buddhika Jayasekara (Dec. 2016). "Design and Development of Adaptive Vision Attentive Robot Eye for Service Robot in Domestic Environment". In: DOI: 10.1109/ICIAFS.2016.7946529 (cit. on p. 22).
- Thorpe Davis, E. (1997). "Visual Requirements in HMDs: What can we see and what do we need to see?" In: *Head-mounted displays: designing for the user*. Ed. by J.E. Melzer and K. Moffit. New York: Mc Graw-Hill, pp. 208–252 (cit. on p. 22).



- Wang, Minqi and Emily A. Cooper (Nov. 2022). "Perceptual Guidelines for Optimizing Field of View in Stereoscopic Augmented Reality Displays". In: *ACM Trans. Appl. Percept.* 19.4. DOI: <https://doi.org/10.1145/3554921> (cit. on p. 22).
- Lange, Karl-Heinz and Klaus Windel (2017). *Kleine Ergonomische Datensammlung*. 16th. Colonia: Editorial de Colonia (cit. on p. 23).
- Gilman, Samuel, Donald Dirks, and Steven Hunt (Jan. 1979). "Measurement of head movement during auditory localization". In: *The Journal of the Acoustical Society of America* 11, pp. 37–41. DOI: 10.3758/BF03205429 (cit. on p. 23).
- Raspberry-Pi (2020). *Raspberry Pi Remote Access Documentation*. Accessed: 15-06-2023. URL: <https://www.raspberrypi.com/documentation/computers/remote-access.html> (cit. on p. 24).
- Spain announces plans for flying taxi service in Barcelona (Nov. 2020). Accessed: [2023-08-15]. The Guardian. URL: <https://www.theguardian.com/world/2020/nov/05/spain-announces-plans-flying-taxi-service-barcelona> (cit. on p. 27).
- Ponnusamy, Arun (2021). *cvlib Python Library*. URL: <https://www.github.com/arunponnusamy/cvlib> (cit. on p. 30).
- Time and Date AS (2023). *Weather in Garching, Bavaria, Germany: Historic Data*. Accessed: May-July-August 2023. URL: <https://www.timeanddate.com/weather/@2922582/historic> (cit. on pp. 41, 45).
- H-Aero (2023). *Zero Plus*. Accessed: [15th August 2023]. URL: <https://h-aero.com/en/products/zero-plus> (cit. on pp. 42, 43).
- Codes, HTML Color (2023). *Blue Color Codes*. Accessed: 12-08-2023. URL: <https://html-color.codes/color-names> (cit. on p. 43).