UNIVERSIDADE
BEIRA INTERIOR

Design of a Collision Avoidance System for Small Fixed-wing UAVs

(Versão final após defesa)

Diogo Joaquim Santos Ferreira

Dissertação para obtenção do Grau de Mestre em

Engenharia Aeronáutica

(Mestrado Integrado)

Orientador: Prof. Doutor Kouamana Bousson

Covilhã, dezembro de 2025

Declaração de Integridade

Eu, Diogo Joaquim Santos Ferreira, que abaixo assino, estudante com o número de inscrição 47507 de Engenharia Aeronáutica da Faculdade de Engenharia, declaro ter desenvolvido o presente trabalho e elaborado o presente texto em total consonância com o **Código de Integridades da Universidade da Beira Interior**.

Mais concretamente afirmo não ter incorrido em qualquer das variedades de Fraude Académica, e que aqui declaro conhecer, que em particular atendi à exigida referenciação de frases, extratos, imagens e outras formas de trabalho intelectual, e assumindo assim na íntegra as responsabilidades da autoria.

Universidade da Beira Interior, Covilhã 30/12/2025.

Assinado por: **Diogo Joaquim Santos Ferreira**
Num. de Identificação: 14858908
Data: 2025.12.30 11:28:24+00'00'

Acknowledgements

First and foremost, I would like to extend my sincere thanks to my supervisor, Professor Kouamana Bousson, for his guidance, availability, and patience throughout this journey.

To all my friends, for sharing this path with me over the past years.

To Sofia and Zé, for never stopping believing in me.

To Tati, for all the love, care, and patience.

Finally, to my parents, for their unconditional love, constant support, and the values they have instilled in me. This achievement is as much yours as it is mine.

Abstract

This dissertation addresses the development and evaluation of a conflict detection and resolution algorithm inspired by the principles of the Traffic Collision Avoidance System (TCAS) — with the reference to TCAS concerning specifically its conflict detection logic and separation thresholds — and tailored to the operation of Unmanned Aerial Vehicles (UAVs). The proposed approach employs Model Predictive Control (MPC) to generate evasive trajectories that preserve flight safety and efficiency while ensuring conflict-free operation.

To properly evaluate the algorithm, the simulation environment was designed so that conflict situations are always present. Specifically, two aircraft are initially assigned colliding trajectories, deliberately creating a reference conflict. In addition, three extra aircraft are randomly generated within a predefined range that reflects the maximum distance at which inter-aircraft information exchange is considered feasible. This ensures that the evasive trajectory is shaped not only by the conflicting pair but also by the surrounding simulated traffic. The stochastic placement of additional aircraft guarantees diversity across scenarios, enabling a more comprehensive and rigorous validation of the algorithm.

The results show that the algorithm successfully modifies the evasive aircraft's trajectory to avoid conflicts while accounting for nearby traffic to prevent the creation of new ones. Although computational cost remains a relevant limitation, the study confirms the feasibility of applying MPC-based strategies to UAV conflict resolution in three-dimensional scenarios. Overall, the work establishes a solid foundation for future research, including extensions to multi-aircraft encounters, integration of more realistic environmental conditions, and improvements in computational efficiency.

Keywords

UAV conflict resolution, Collision avoidance, Traffic Collision Avoidance System (TCAS), Model Predictive Control (MPC), Air traffic simulation, Evasive manoeuvres, Autonomous systems

Resumo Alargado

O crescimento acelerado da utilização de veículos aéreos não tripulados (UAVs) em diferentes áreas, desde missões de vigilância e monitorização até ao transporte de pequenas cargas e operações em ambiente urbano, tem vindo a evidenciar de forma clara a necessidade de sistemas de gestão de tráfego capazes de garantir a segurança em espaço aéreo partilhado. A introdução massiva destes veículos em corredores aéreos que, até agora, estavam reservados a aviação convencional levanta desafios adicionais: elevada densidade de tráfego, incerteza nas trajetórias e limitações na capacidade de deteção e reação dos UAVs. Neste contexto, a presente dissertação focou-se no desenvolvimento e avaliação de um algoritmo de prevenção de colisões, inspirado na lógica de deteção e nos limites de separação definidos em sistemas TCAS existentes, especificamente adaptado à operação de UAVs. O objetivo central passa por assegurar a deteção precoce de conflitos e a execução de manobras de evasão seguras, consistentes e operacionalmente viáveis.

O trabalho iniciou-se com a definição clara do problema de colisão, considerando cenários tridimensionais com cinco aeronaves em simultâneo. Duas destas seguiam trajetórias definidas pelo utilizador, ao passo que as outras três foram geradas de forma estocástica fora da zona de conflito, mas sempre respeitando margens mínimas de separação. No que diz respeito às trajetórias geradas pelo utilizador, estas foram construídas de forma determinística a partir de sequências de waypoints (latitude, longitude e altitude), interpolados no tempo e no espaço, de modo a criar deliberadamente situações de conflito de referência. Já as trajetórias estocásticas resultaram da atribuição aleatória de condições iniciais às aeronaves adicionais, sujeitas a restrições de separação mínima, permitindo introduzir variabilidade nos cenários sem comprometer a segurança. Esta abordagem integrada assegurou, por um lado, a presença consistente de conflitos a resolver e, por outro, a validação da consciência situacional, dado que as manobras evasivas tiveram de ser avaliadas face ao risco de originar novos conflitos. Assim, os cenários de teste obtidos foram simultaneamente controlados e reprodutíveis, mas com diversidade suficiente para aproximar a validação do algoritmo a condições operacionais mais realistas.

O núcleo do algoritmo é composto por dois módulos principais. O primeiro é o módulo TCAS, responsável pela deteção de conflitos através da monitorização contínua de três métricas fundamentais: a distância horizontal mínima projetada (CPA_h), a separação vertical (Δh) e o tempo até à aproximação mais próxima (τ). A decisão de adotar o nível de sensibilidade mais restritivo entre as aeronaves traduziu-se numa monitorização conservadora, assegurando a emissão atempada de alertas em todos os instantes críticos. O segundo módulo é o de resolução de conflitos, que assenta num modelo de controlo preditivo. A formulação incluiu de forma explícita os limites físicos e operacionais da aeronave, permitindo que as soluções encontradas fossem sempre exequíveis. O método SLSQP foi selecionado por lidar diretamente com restrições e saturações, garantindo convergência

estável. A função de custo foi cuidadosamente estruturada: penalizações de primeira e segunda ordem nas variações dos ângulos de rumo (χ) e subida/descida (γ), uma penalização adicional sobre inversões de sinal em γ, e ainda um termo de seguimento da trajetória nominal, de modo a limitar desvios excessivos.

Os resultados obtidos confirmaram a eficácia da solução nos dois cenários analisados: *head-on* e ortogonal. Em todas as simulações, o MPC respeitou os limites impostos e conseguiu recuperar separações seguras. O parâmetro τ evoluiu de forma consistente para valores próximos de zero, refletindo a aproximação entre aeronaves, enquanto CPA_h e Δh aumentaram significativamente durante as fases de evasão. Em ambos os cenários, observaram-se manobras complementares: movimentos laterais, destinados a ampliar o CPA_h, e variações verticais, que maximizaram Δh. Estas últimas revelaram-se determinantes, dado que, de forma geral, a resolução assentou mais em ajustamentos verticais do que horizontais. Este comportamento reflete a própria estrutura da função de custo, que penaliza de forma mais expressiva grandes desvios laterais à trajetória inicialmente planeada.

O facto de o algoritmo considerar, para além das duas aeronaves em conflito, a posição de outras três aeronaves adicionais garantiu que todas as decisões fossem tomadas de forma consciente em relação ao tráfego circundante. Esta capacidade de manter consciência situacional, ausente no TCAS convencional, constitui um dos principais contributos do presente trabalho. A utilização de um horizonte preditivo recedente, com truncagem na aplicação dos controlos (aplicando apenas os primeiros passos em cada ciclo), reforçou a robustez face a incertezas de previsão, atenuando comportamentos excessivamente antecipatórios e assegurando maior estabilidade ao longo do processo. Por fim, a lógica de retorno à trajetória demonstrou ser eficaz e consistente: a aeronave evasora foi conduzida até um ponto de convergência definido permitindo a reintegração na rota nominal sem manobras bruscas ou instabilidades.

Em síntese, a presente dissertação propõe e valida um algoritmo de prevenção de colisões baseado em MPC que, ao contrário do TCAS convencional, considera permanentemente múltiplas aeronaves em simultâneo. O sistema revelou ser capaz de alterar as condições de voo da aeronave evasora de modo a evitar conflitos, mantendo a segurança em relação ao tráfego envolvente e assegurando trajetórias operacionalmente plausíveis. Os resultados evidenciam quatro aspetos centrais: (i) a capacidade de deteção conservadora e atempada de conflitos, (ii) a produção de soluções de controlo suaves e respeitadoras dos limites físicos, (iii) a consistência do regresso à trajetória original e (iv) a integração de múltiplas aeronaves no processo de tomada de decisão.

Apesar dos resultados promissores, subsistem limitações que devem ser alvo de trabalho futuro. Entre estas destacam-se a necessidade de converter a implementação para uma linguagem de programação mais eficaz do ponto de vista computacional, a inclusão de modelos mais realistas de sensores, capazes de reproduzir imperfeições e atrasos na deteção, e a validação em ambientes experimentais com restrições de tempo real. Estes de-

senvolvimentos são essenciais para aproximar a solução proposta da sua aplicação efetiva em sistemas de gestão de tráfego aéreo com UAVs, contribuindo para a integração segura destes veículos em espaços aéreos cada vez mais complexos.

Contents

Acknowledgements iv

Abstract v

Resumo Alargado vii

Contents xi

List of Figures xv

List of Tables xvii

List of Algorithms xix

Acronyms and Abbreviations xxi

1 Introduction 1

1.1 General Context 1

1.1.1 Geometric Approaches 1

1.1.2 Potential Field Approaches 2

1.1.3 Path Planning Approaches 2

1.1.4 Optimization-Based Approaches 3

1.1.5 Sampling-Based Approaches 4

1.1.6 Vision-Based Approaches 5

1.1.7 Model Predictive Control (MPC) Approaches 6

1.2 Literature Review 6

1.2.1 CAS for Stationary Obstacles 7

1.2.2 CAS for Moving Obstacles 9

1.2.3 Regulatory Standards 11

1.3 Limitations 13

1.4 Objectives of the Dissertation 13

1.5 Thesis Structure 14

2 Models for Air Collision Avoidance 15

2.1 Traffic Collision Avoidance System 15

2.1.1 Conflict Definition in TCAS 16

2.1.2 Threshold Values Adopted for Conflict Detection 17

2.2 Flight Dynamics Model 18

2.2.1 Aircraft Displacement in the ENU Reference Frame 18

2.2.2 Transformation of Displacement from ENU to ECEF 18

2.3 Model Predictive Control (MPC) 19

2.3.1 Fundamental Concepts 20

2.3.2 Bounds, Constraints and Optimization Algorithm 21

3 Model Predictive Control Based Air Collision Avoidance 25

3.1 Algortithm Flowchart 25

3.2 Simulation Input Data 26

3.3 Interpolation of Predefined Trajectories 27

3.3.1 Continuous Trajectory Generation from Waypoints 27

3.3.2 Finite Difference Approach for Velocity Estimation 27

3.4 Random Generation of Aircraft for Conflict Scenario Simulation 28

3.4.1 Definition of Initial Positions for Random Aircraft 28

3.4.2 Random Aircraft Simulation 32

3.4.3 Considerations Regarding the Simulation Approach 33

3.4.4 Timing of Aircraft Generation 34

3.5 Traffic Collision Avoidance System Algorithm 34

3.6 Model Predictive Control Algorithm 35

3.6.1 Control Vector 36

3.6.2 Bounds 37

3.6.3 Constraints 37

3.6.4 Optimization and Iterations 38

3.6.5 Application of Control Inputs 39
3.6.6 Cost Function 40
3.6.7 Post-Processing and Criteria for MPC Re-run 47
3.6.8 Returning to the Planned Trajectory 48

4 Simulation Methodology and Results **51**
4.1 Constraints Definition 51
4.2 Trajectory Definition 52
4.2.1 Head-on Encounter 52
4.2.2 Orthogonal Encounter 54
4.3 Results and Sensitivity to the Prediction Horizon 56
4.3.1 Results 57

5 Conclusions and Future Work **77**
5.1 Limitations 79
5.2 Future Work 80

Bibliography **81**

List of Figures

2.1 Illustration of model predictive control. 20

2.2 Flowchart of the MPC optimization process. 21

3.1 Algorithm flowchart illustrating the main simulation steps. 25

3.2 Comparison between the Earth-Centered Earth-Fixed (ECEF) and the East-North-Up (ENU) reference frames. 33

4.1 Trajectory defined for the head-on encounter scenario. 54

4.2 Trajectory defined for the orthogonal encounter scenario. 56

4.3 CPA_h resulting from applied controls across MPC iterations — Head-on scenario. 58

4.4 Vertical separation resulting from applied controls across MPC iterations — Head-on scenario. 59

4.5 τ resulting from applied controls across MPC iterations — Head-on scenario. 60

4.6 Resulting aircraft trajectory for the head-on scenario. 61

4.7 Evolution of heading angle χ across successive MPC iterations — Head-on scenario. 61

4.8 Evolution of flight path angle γ across successive MPC iterations - Head-on scenario. 62

4.9 Evolution of altitude across successive MPC iterations - Head-on scenario. 62

4.10 Evolution of speed across successive MPC iterations — Head-on scenario. . 63

4.11 Comparison of horizontal separation parameter CPA_h for different numbers of prediction steps P — Head-on scenario. 65

4.12 Comparison of vertical separation Δh for different numbers of prediction steps P — Head-on scenario. 65

4.13 Comparison of parameter τ for different numbers of prediction steps P — Head-on scenario. 66

4.14 CPA_h resulting from applied controls across MPC iterations — Orthogonal scenario. 68

4.15 Δh resulting from applied controls across MPC iterations — Orthogonal scenario. 68

4.16 τ resulting from applied controls across MPC iterations — Orthogonal scenario. 69

4.17 Resulting aircraft trajectory for the orthogonal scenario. 70

4.18 Evolution of heading angle χ across successive MPC iterations — orthogonal scenario. 70

4.19 Evolution of flight path angle γ across successive MPC iterations - Orthogonal scenario. 71

4.20 Evolution of altitude across successive MPC iterations - Orthogonal scenario. 71

4.21 Evolution of speed across successive MPC iterations — Orthogonal scenario. 72

4.22 Comparison of horizontal separation parameter CPA_h for different numbers of prediction steps P — Orthogonal scenario. 73

4.23 Comparison of vertical separation parameter Δh for different numbers of prediction steps P — Orthogonal scenario. 74

4.24 Comparison of τ parameter for different numbers of prediction steps P — Orthogonal scenario. 75

List of Tables

1.1 TCAS II thresholds by own altitude and sensitivity level (SL). 13

3.1 Example of predefined trajectories for aircraft 0 and 1. 26

3.2 Example of physical and dynamic limits for each aircraft. 26

3.3 Weights associated with the cost function terms considered in the MPC-based conflict resolution algorithm. 47

4.1 Head-on Trajectory Data of Aircraft 0 53

4.2 Head-on Trajectory Data of Aircraft 1 53

4.3 Orthogonal Trajectory Data of Aircraft 0. 55

4.4 Orthogonal Trajectory Data of Aircraft 1. 56

4.5 Execution time of the algorithm for different prediction steps - Head-on scenario. 67

4.6 Execution time of the algorithm for different prediction steps - Orthogonal scenario. 75

List of Algorithms

1 Iterative resolution process of the SLSQP optimization algorithm. 22
2 Iterative resolution process of the BFGS optimization algorithm. 23

Acronyms and Abbreviations

ACAS	Airborne Collision Avoidance System
ADS-B	Automatic Dependent Surveillance–Broadcast
ALIM	Altitude Limit – Vertical Threshold for Corrective Resolution Advisory
APF	Artificial Potencial Field
BFGS	Broyden–Fletcher–Goldfarb–Shanno Algorithm
CAS	Collision Avoidance System
CPA	Closest Point of Approach
DMOD	Horizontal Distance Modification Parameter
ECEF	Earth-Centered Earth-Fixed
ENU	East-North-Up
GPS	Global Positioning System
ICAO	International Civil Aviation Organization
MPC	Model Predictive Control
RA	Resolution Advisory
RCZ	Relative Conflict Zone
RTT	Rapidly-Exploring Random Trees
SARPs	Standards and Recommended Practices
SL	Sensitivity Level
SLSQP	Sequential Least Squares Quadratic Programming
TA	Traffic Advisory
TCAS	Traffic Alert and Collision Avoidance System
TVTHR	Time (Vertical) Threshold – Reduced Time to Co-altitude Threshold
UAV	Unmanned Aerial Vehicle
ZTHR	Z Threshold – Vertical Threshold for Resolution Advisory

Chapter 1

Introduction

The increasing use of Unmanned Aerial Vehicles (UAVs) in commercial, civilian, and military applications has expanded operational scenarios, increasing the complexity of airspace management and the likelihood of collisions. Therefore, collision prevention has become a priority in the research and development of UAV technologies, ensuring safety and efficiency in autonomous missions. The goal is to highlight the relevance of implementing Traffic Collision Avoidance Systems (TCAS) in UAVs, improving recent approaches, and increasing the application scenarios.

1.1 General Context

Collision avoidance systems can be classified into multiple categories, each with specific advantages depending on the environment and performance requirements. The main approaches include the following.

1.1.1 Geometric Approaches

Geometric approaches rely on kinematic and geometric relationships between aircraft to predict conflicts and design avoidance manoeuvres. By analysing parameters such as relative position, velocity vectors, and heading angles, these methods assess whether trajectories will converge within a predefined safety threshold. A common enabler is *Automatic Dependent Surveillance-Broadcast* (ADS-B), which provides continuous state updates in real time.

These methods are computationally simple and intuitive, but their reliance on nominal extrapolation means that uncertainties in measurements or intent are not explicitly considered. As shown by Paielli and Erzberger [1], trajectory prediction errors grow significantly with time—particularly in the along-track direction due to wind uncertainties—so ignoring them can lead to false alarms or delayed detections. Moreover, these methods are usually applied pairwise, which limits scalability in multi-aircraft scenarios, where global strategies—though more demanding—tend to be more robust.

1.1.2 Potential Field Approaches

The *Potential Field* method is one of the earliest and most widely studied strategies for UAV collision avoidance, due to its conceptual simplicity and low computational cost. The basic idea is to model the UAV as a particle navigating within a virtual field composed of attractive forces (pulling towards the goal) and repulsive forces (pushing away from obstacles and other aircraft). The resulting trajectory emerges from the balance of these forces, creating smooth paths that avoid collisions.

1.1.2.1 Principles of the Potential Field

In practice, the potential field method defines an artificial potential function $U(q)$, where q represents the UAV configuration (e.g., position and orientation). Attractive potentials guide the UAV to the target, while repulsive potentials increase near obstacles. The control action is then derived from the gradient of the potential, as shown in Equation 1.1.

$$F(q) = -\nabla U(q) \tag{1.1}$$

Despite its elegance, the method suffers from well-known issues such as local minima and oscillations in narrow corridors, which can trap the UAV or lead to inefficient trajectories. As noted by Batinovic *et al.* [2], these drawbacks are inherent to the conventional APF approach, and significant research has focused on mitigating them. Their work, for instance, introduces a rotational component to the repulsive force to overcome local minima, demonstrating that while the APF method is attractive for its simplicity and low computational burden, enhancements are often required to ensure robust performance in complex environments.

1.1.3 Path Planning Approaches

Path planning methods focus on finding a collision-free route from the origin to the destination, given knowledge of the environment. These approaches are particularly useful when the environment can be represented as a graph or grid, enabling the use of systematic search algorithms. They are effective in static or slowly changing scenarios but may struggle in highly dynamic environments, where real-time adaptability is required [3].

1.1.3.1 Principles of Path Planning

The environment is typically represented as a *configuration space*, where feasible and infeasible regions are clearly defined. Search algorithms explore this space to determine

optimal routes according to criteria such as minimum distance, fuel efficiency, or minimal risk.

1.1.3.2 Dijkstra's Algorithm

Dijkstra's algorithm provides guaranteed shortest paths by systematically exploring nodes in increasing order of distance. While robust, it has high computational requirements for large state spaces, limiting its direct application in UAV operations with real-time constraints [4].

1.1.3.3 A* Algorithm

The A* algorithm enhances Dijkstra's method by incorporating heuristics (such as the Euclidean distance to the goal) to guide the search more efficiently. This makes A* particularly suitable for UAV path planning, striking a balance between computational tractability and near-optimal results [3].

1.1.4 Optimization-Based Approaches

Collision avoidance can be formulated as an optimization problem, where the objective is to minimize a cost function, such as path length or travel time, while ensuring safety constraints. Metaheuristics like Genetic Algorithms (GA) and Particle Swarm Optimization (PSO) have been successfully applied to UAV trajectory planning, providing flexible and effective solutions in complex environments [5, 6, 7].

1.1.4.1 Principles of Optimization Algorithms

Optimization algorithms are typically inspired by natural processes and operate iteratively, refining candidate solutions until convergence. Their strength lies in handling nonlinear, high-dimensional problems where analytical methods are intractable, making them suitable for UAV path planning [5].

1.1.4.2 Classical Optimization Methods

Before the rise of metaheuristic algorithms, collision avoidance problems were primarily addressed using deterministic optimization techniques. These methods rely on mathematical programming formulations, such as linear programming (LP), quadratic programming (QP), or nonlinear programming (NLP), to directly minimize an objective function under explicit constraints. Typical objectives include minimizing flight time, traveled

distance, or fuel consumption while ensuring safe separation from obstacles and other aircraft. Deterministic methods often employ well-established techniques such as gradient-based optimization, sequential quadratic programming (SQP), or interior-point methods to iteratively refine solutions [5].

Their main strength lies in providing mathematically rigorous solutions with guaranteed convergence when the problem is convex, which makes them effective in structured or low-complexity environments.[5]

1.1.4.3 Genetic Algorithms (GA)

Genetic algorithms (GAs) are inspired by Darwinian evolution, using operators such as selection, crossover, and mutation to evolve a population of solutions. In UAV navigation, GAs have been successfully applied to collision-free path planning in complex environments by effectively balancing exploration and exploitation of the search space. Improved versions of GAs, such as multi-domain inversion GA, have shown enhanced convergence speed, robustness, and efficiency in generating collision-free trajectories for unmanned vehicles [6]. These characteristics make GAs particularly effective in solving highly non-convex optimization problems associated with UAV path planning.

1.1.4.4 Particle Swarm Optimization (PSO)

Particle Swarm Optimization (PSO) is inspired by the collective behavior of bird flocks and fish schools. In PSO, each particle represents a potential solution and adjusts its trajectory based on both its own best experience and the knowledge shared by the swarm. This social interaction enables efficient exploration of continuous, nonlinear search spaces. PSO has demonstrated strong performance in trajectory planning tasks due to its relatively low computational cost, simplicity of implementation, and effectiveness in avoiding local optima [7]. These features make it a valuable alternative to evolutionary algorithms such as GAs in UAV collision avoidance.

1.1.5 Sampling-Based Approaches

Sampling-based motion planners generate random samples and connect them via collision checks to incrementally build a roadmap or tree of feasible motions, instead of explicitly constructing the obstacle set in configuration space; this strategy has been shown to be effective in high-dimensional state spaces [8, 9].

1.1.5.1 Principles of Sampling-Based Approaches

These methods incrementally build a tree or graph by connecting randomly sampled configurations to existing nodes, while ensuring feasibility through collision detection. This allows planning even in complex scenarios, such as those involving nonholonomic and kinodynamic constraints, without the need to explicitly compute connections between arbitrary states [8].

1.1.5.2 Rapidly-Exploring Random Trees (RRT)

Rapidly-Exploring Random Trees (RRT) were introduced as a data structure tailored to efficiently explore large and complex state spaces, with a strong bias towards unexplored regions [8]. This makes them particularly effective for problems with nonholonomic and dynamical constraints. Although RRTs are probabilistically complete, the paths they produce are generally suboptimal. To overcome this, Karaman and Frazzoli proposed RRT*, which incorporates a rewiring procedure that iteratively improves previously found paths. Unlike the basic RRT, RRT* is proven to be asymptotically optimal, meaning that the solution converges almost surely to the optimum as the number of samples increases, while maintaining computational complexity within the same order as RRT [9].

1.1.6 Vision-Based Approaches

Vision-based approaches leverage onboard cameras and optical sensors to perceive the environment and detect potential collisions. These systems are particularly advantageous for small UAVs, where payload and energy constraints limit the use of heavier sensors like radar or LiDAR. By processing visual data, UAVs can detect obstacles, estimate their motion, and plan avoidance manoeuvres.

1.1.6.1 Principles of Vision-Based Approaches

The core idea of vision-based approaches is to transform raw visual input into actionable information for navigation. This typically involves three main steps: feature extraction, obstacle detection, and motion estimation. The resulting data is then integrated into an internal representation of the environment, such as an occupancy grid, which supports trajectory planning and collision avoidance.

1.1.6.2 Stereo Vision

Stereo vision uses two or more cameras with a fixed baseline to infer depth through disparity estimation. Compared to RGB-D sensors, stereo cameras are lighter, passive, and

more suitable for outdoor UAV applications, where long-range perception is required. Hrabar [10] demonstrated that stereo vision can be effectively applied to UAVs, enabling the construction of 3D occupancy maps and dynamic path replanning in cluttered environments.

1.1.7 Model Predictive Control (MPC) Approaches

Model Predictive Control (MPC) represents a more advanced control paradigm, where future UAV states are predicted over a finite horizon, and control inputs are optimized at each step to avoid conflicts. Unlike reactive methods, MPC accounts for both current and predicted conflicts, making it well-suited for dense traffic scenarios and free-flight concepts [11].

1.1.7.1 Principles of Model Predictive Control

MPC solves an optimization problem at each control step, minimizing a cost function that penalizes deviations from the intended path, excessive control effort, or proximity to other aircraft. Constraints ensure that minimum separation distances and performance limits are respected. By continuously updating predictions with new sensor data, MPC provides adaptive and anticipatory collision avoidance [11].

Collision avoidance methods comprise a broad spectrum of strategies, each designed to meet specific operational demands. Approaches range from geometric principles and potential fields to advanced optimization algorithms and sampling-based techniques, with each offering distinct advantages and inherent trade-offs. Vision-based systems and Model Predictive Control (MPC) further expand the toolbox, enabling robust performance in dynamic and uncertain environments. Taken together, these methods provide a comprehensive foundation for enhancing UAV navigation, promoting safety, efficiency, and adaptability in increasingly congested and autonomous airspaces.

1.2 Literature Review

Theoretical concepts and previous studies highlight significant limitations in the CAS currently implemented in UAVs. These limitations often arise from computational constraints, reliance on idealized assumptions, and challenges in adapting to dynamic and unpredictable environments. As UAV applications continue to expand, particularly in complex operational scenarios, these shortcomings become increasingly evident, posing risks to safety and efficiency.

Addressing these limitations requires rethinking traditional approaches and exploring new perspectives that can better accommodate the specific demands of UAV operations.

By focussing on these critical gaps, this work aims to propose innovative strategies that enhance the effectiveness and scalability of CAS for UAVs, offering solutions that align with the evolving needs of this rapidly growing field.

Two approaches can be considered when discussing CAS. A more simplified approach involves the fact that the obstacles to be avoided are stationary, meaning they do not exhibit any type of movement. A second approach complements the first and introduces movement to the obstacles, where in this case, the obstacles can be either predefined or detected by the system afterward.

1.2.1 CAS for Stationary Obstacles

1.2.1.1 Definition of Obstacles

The detection and resolution of conflicts in the case of a stationary obstacle involves a much simpler approach, as any route can effectively resolve the conflict since the obstacle does not exhibit any movement. This resolution principle also serves as a bridge for addressing moving obstacles later, as the obstacle detection algorithm (if present) retains its foundation, and the impact of the alternative route on the UAV's final destination must be taken into account to determine the optimal solution to the problem.

In terms of defining the position of obstacles, they can be specified through their coordinates, as observed in *Toledo et al.* (2007) [12], or through action zone mapping, which is usually associated with missions in controlled or even restricted areas, as discussed by *Aldao et al.* (2020) [13].

In the first approach [12], the Flight Management System (FMS) and the remaining onboard systems of a light aircraft were used to calculate the feasible trajectory to circumvent the imposed obstacles. Additionally, the physical limitations of the aircraft to perform the maneuvers are also taken into account. The resolution of various conflicts is achieved using Machine Learning to ensure trajectory convergence.

An alternative to imposing obstacles through coordinates arises by making it possible to obtain a mapping of the flight zone through three-dimensional point clouds with the aid of LiDAR sensors or other similar alternatives [13]. With this method, it is still necessary to add a system capable of acquiring position data to compare with the position data of the obstacles or sensors/cameras capable of detecting the obstacles. Mapping the obstacles can still be important for optimising the corrected trajectory.

1.2.1.2 Detection of Obstacles

Obstacle detection is one of the most important aspects when addressing CAS. In a generic sense, detection is nothing more than the direct comparison between the operated vehi-

cle's route and an object that is or may be on the same path.

Two possible approaches to obstacle detection should be highlighted. A more passive approach is based on imposing the coordinates of existing obstacles, *Toledo et al.* (2007) [12], which are then compared with the instantaneous position of the vehicle.

On the other hand, in *Lin et al.* (2018) [14], an active monitoring system is adopted, which, based on onboard sensors, such as ultrasonic or infrared sensors, monitors potential obstacles from the aircraft within a limited field of view. If a potential obstacle presents a distance greater than the required minimum separation distance, it is classified as a real threat. This approach is also used in *Guo et al.* (2021) [15].

Recently, LiDAR technology has also been adopted for detection under these conditions, generally being a more expensive approach, *Aldao et al.* (2020) [13].

Although not explicitly presented, there is a more active variation of the approach described in *Toledo et al.* (2007) [12], where an obstacle continuously transmits its coordinates to a system capable of interpreting and comparing them with the vehicle in question.

1.2.1.3 Obstacle Discretization

The discretization of obstacles represents a very important factor due to the dependence of the alternative trajectory on the discretization. In *Guo et al.* (2020) [15], the discretization of conflicts during flight is explored through sensors such as infrared cameras, photoelectric radars, laser rangefinders or acoustic and optical processors.

The information from each of the possible sensors to be adopted must then be used to manipulate the objects in two or three dimensions effectively. With the points obtained from the sensors, an adjustment algorithm (least squares method) is necessary to optimise and calculate the geometric shape in question. In the 2D case, it is sufficient to fit a circle that contains all the points obtained, whereas in the 3D case, the fitting is done with the help of standard convex bodies such as cylinders, cones, or hemispheres.

guoIn a similar yet simplified approach, *Aldao et al.* (2020) [13] obtained information about obstacles using LiDAR sensors, capable of providing point clouds that facilitate the acquisition of the obstacle's geometric shape itself.

1.2.1.4 Alternative Trajectory

When a collision is detected, measures must be taken to avoid it, particularly an alternative route that moves the aircraft away from the collision path with an obstacle. At this point, it is necessary to consider not only the guarantee that the conflict is resolved but also to ensure that the alternative route has the least possible impact on the final destination.

Typically, a relative conflict zone (RCZ) is defined, which specifies the area where a colli-

sion might occur. In *Ouyang et al.* (2022) [16], in the static case, this conflict zone corresponds to the area where the minimum separation distance from the obstacle is guaranteed. Given the previously established zone, a cost function is defined that not only minimizes the deviation from the pre-established route but also minimizes the energy impact by optimizing speed variation. Subsequently, the trajectory is smoothed using cubic splines.

An alternative approach, presented in *Aldao et al.* (2020) [13], and more suitable for controlled environments, is based on 3D mapping of the mission area and the use of LiDAR sensors or stereoscopic cameras. Using this data, the alternative trajectory is calculated based on an optimization problem aimed at minimizing the deviation time and distance. This solution recalculates a new trajectory, confirming whether it maintains the previously detected issue or not, always ensuring the safety distance.

1.2.2 CAS for Moving Obstacles

The approach for the CAS with moving obstacles shares many similarities with the approach without any movement assigned to the obstacle, as presented in *Section 1.2.1*.

1.2.2.1 Obstacles Detection

The nature of the problem leads to a point where it no longer makes sense to discuss conflict enforcement due to the presence of a movement associated with an obstacle. This fact can be explained by the fact that it is now necessary to detect or at least consider the movement of the potential conflict-causing object.

In *Shim* (2009) [17], a typical approach was taken in these cases through the use of laser scanning for obstacle detection. The choice fell on this type of laser because it does not require much range.

In another approach, *Bousson* (2008) [18], obstacle detection is based on their position coordinates. With knowledge of the coordinates, leveraging the presence of GPS systems and similar technologies, proximity between the two aircraft is then measured, with a circular safety distance of $d > 0$ centered on the geometric center of each aircraft. Similarly, in *Chen et al.* (2020) [19], obstacle detection is carried out using a conflict detection module, which, based on position data, velocities, heading, and others, identifies possible conflicts by predicting the future flight paths of the aircraft. This prediction is made in pairs of aircraft, similarly to all others.

It is also possible to combine the use of passive systems (cameras or sensors) with active systems (flight data usage). This approach, present in *Džunda et al.* (2022) [20], is used to extend the detection range in order to reduce the collision risk. Flight data from ADS-B systems or others allow continuous monitoring and a subsequent prediction of whether

the chosen route will conflict with another. Passive systems, in addition to adding redundancy, especially at shorter distances, where they even provide higher precision—also increase safety by compensating for the lack of active systems in other aircraft.

In *Yu and Park* (2023) [21], when the relationship between the UAV and the obstacle satisfies a potential detection condition, the position of the obstacle is obtained through a sensor, and its future trajectory is predicted using the Kalman Filter. The filter is initialized with sensor data, assumed to be noise-free. In this way, the aircraft can estimate the position, velocity, and acceleration of the obstacle, making it possible to assess the conflict.

1.2.2.2 Obstacle Discretization

The discretization of obstacles must be carried out in a completely generic manner so that safety regulations can be respected in any situation. In *Roelofsen et al.* (2016) [22], a sphere with radius r is defined to delimit the safety area. A collision is defined when the distance between two aircraft is less than $2r$.

On the other hand, in *Becker et al.* (2012), obstacles are modeled as vertical cylinders with different diameters and heights, with these dimensions being determined based on the geometry of the obstacle. This discretization is performed by the UAV itself using onboard sensors.

1.2.2.3 Path Change Priority

Taking obstacles as aircraft and considering the presence of two or more simultaneously, it is important to establish criteria that define which aircraft should alter their route. This definition of criteria can be carried out in various ways and is highly dependent on the scenario under study.

In *Bousson* (2008) [18], the priority of each aircraft is defined by a priority index I_i, calculated based on the proximity of the aircraft to its destination point.

Complementarily, in *Shim* (2009) [17], a cost function was defined, penalising deviations from the predefined trajectory as well as the energy expenditure required for such a change, with this energy being related to the aircraft's geometry. The presence of deviation penalties both in terms of distance and time is an extremely important factor given the finite nature of available energy.

In *Lee et al.* (2020) [23], a different approach is taken, assuming that the aircraft itself must change direction to avoid collision. In this way, the UAV adjusts its direction angle to minimize the deviation from its destination point while ensuring a minimum separation distance through the *Lyapunov* function.

Although indirectly, *Becker et al.* (2012) [24] consider the possibility of the UAV adjusting its trajectory or maintaining its course. Since it is a rotary-wing aircraft capable of hovering, the UAV can hold its position if the detection system identifies a tendency for the obstacle to move away.

1.2.2.4 Alternative Trajectory

The alternative trajectory is associated with considerable importance, as it is through this trajectory that the aircraft has the ability to either circumvent or avoid the obstacle. In *Shim* (2009) [17], the referred trajectory is obtained using an MPC algorithm that considers the dynamic limitations of the system and, based on minimizing a cost function, ensures that the trajectory effectively resolves the conflict while always keeping the new route within the physical limits of the system.

Similarly, in *Bousson* (2008) [18], a similar approach was used, also employing an MPC algorithm that minimizes the total distance to be traveled by all aircraft. The control is estimated at consecutive time intervals with the additional objective of predicting future collisions. Likewise, in *Kumar et al.* (2019) [11], this approach is used to obtain the alternative trajectory.

In another approach, *Chen et al.* (2020) [19], multiple aircraft with potential conflicts are considered. From each of them, various routes are determined with the objective of selecting, among all these routes, those that resolve the conflict. The resolution is ensured through one or more combinations, and the selection of the route to be adopted also considers the impact of the new routes on the final destinations of each aircraft.

1.2.3 Regulatory Standards

The aeronautical industry operates under precise and well-defined regulations to ensure compatibility. However, given that an in-flight issue can lead to severe consequences, the safety of both people and equipment must never be neglected.

CAS systems must comply with the ICAO Standards and Recommended Practices (SARPs) published in Annex 10, Volume IV, as well as the following reference documents:

- RTCA DO-185B and EUROCAE ED-143 (for TCAS II v7.1).
- RTCA DO-385 and EUROCAE ED-256 (for ACAS Xa/Xo).
- RTCA DO-386 and EUROCAE ED-275 (for ACAS Xu, used in UAVs).

In Europe, the mandatory use of ACAS II (TCAS II v7.1) has applied to civil aircraft over 5,700 kg or with more than 19 seats since 2015. Practical implementation guidelines

and operational aspects are also publicly available through the EUROCONTROL ACAS II Guide [25].

1.2.3.1 Minimum System Components

A valid CAS must include the following components:

- TCAS/ACAS Computer: Processes the avoidance logic and issues alerts.
- Mode S Transponder: Essential for communication between aircraft.
- Surveillance Antennas: Responsible for signal interrogation and reception.
- Control Panel: Allows pilots to configure the system.
- Traffic Display: Shows nearby aircraft and alerts.
- Audible Alerts: Notify the crew about traffic and recommended maneuvers.

These components must be adapted to the mission requirements, such as for an autonomous UAV.

The TCAS II/ACAS logic defines sensitivity levels that determine when an alert is issued. The system must identify conflicts between 15 and 35 seconds before the Closest Point of Approach (CPA).

1.2.3.2 Types of Alerts and Valid Maneuvers

A CAS must be capable of issuing:

- TA (Traffic Advisory): Notifies the pilot of nearby traffic.
- RA (Resolution Advisory): Recommends climbing or descending.
- Reversible RA: If the initial maneuver is ineffective.

Additionally, the system must coordinate maneuvers via Mode S to ensure that aircraft actions are complementary.

1.2.3.3 TCAS Thresholds

The activation of Traffic Advisories (TA) and Resolution Advisories (RA) in TCAS II depends on the aircraft's own altitude and the corresponding Sensitivity Level (SL). For each

SL, the system defines specific thresholds for horizontal time to closest approach (τ), minimum horizontal separation (DMOD), vertical separation (ZTHR and ALIM), and vertical closure time (TVTHR).

These thresholds are designed to ensure adequate separation and provide timely alerts as altitude increases. Table 1.1 compiles the values used in TCAS II, as defined in the EUROCONTROL Safety ACAS Guide (2022) [25].

Own Altitude	SL	$\tau(s)$	$\tau(s)$	TVTHR (s)	DMOD (NM)	DMOD (NM)	ZTHR (ft)	ZTHR (ft)	ALIM (ft)
		TA	RA	RA	TA	RA	TA	RA	RA
0–1000 ft AGL	2	20	no RA	no RA	0.3	no RA	850	no RA	no RA
1000–2350 ft AGL	3	25	15	15	0.33	0.2	850	600	300
2350 ft AGL – FL50	4	30	20	18	0.48	0.35	850	600	300
FL50 – FL100	5	40	25	20	0.75	0.55	600	600	350
FL100 – FL200	6	45	30	22	1	0.8	850	600	600
FL200 – FL420	7	48	35	25	1.3	1.1	850	700	600
Above FL420	7	48	35	25	1.3	1.1	1200	800	700

Table 1.1: TCAS II thresholds by own altitude and sensitivity level (SL).

1.3 Limitations

Based on the analysis of the reviewed articles, it was possible to identify several limitations, both in comparison with established standards and with the practical reality of the problem.

In general, a pattern can be observed in the approach to the analysis of potential conflicts: in most cases, this analysis is performed pairwise, which constitutes the most direct way of assessing the existence of collisions. However, the resolution of these conflicts tends to take into account only the pair of aircraft being analysed, without considering the impact on the overall traffic. Furthermore, it is common to impose predefined positions on the aircraft, which results in repetitive and inflexible resolution strategies.

Another aspect identified is that most systems rely on visual or ultrasonic sensors for conflict detection. Although these are valid and accurate methods, their effectiveness depends heavily on external conditions, such as visibility or environmental interference.

Finally, it was found that a large proportion of the studies do not establish any type of hierarchy or priority in conflict resolution. The responsibility for altering the trajectory is often systematically assigned to the user, which may prove ineffective in practical scenarios. For instance, if the obstacle is an aircraft of smaller dimensions and

1.4 Objectives of the Dissertation

Considering the aspects discussed in Section 1.3, it becomes possible to outline a development strategy for the algorithm to mitigate some of the limitations identified in previously

analysed models. Conflict detection remains based on a pairwise analysis, adopting the logic of TCAS detection, as this constitutes the most direct and consistent approach to identifying potential collisions. However, the evasive manoeuvre is not determined solely from the pair under consideration, but rather by taking into account the surrounding aircraft, up to a maximum of five, ensuring that the solution reflects more realistically the complexity of air traffic.

Regarding the acquisition of information, instead of relying on visual or ultrasonic methods, the approach is based directly on the aircraft's positions, which may be transmitted through transponders or equivalent systems. This choice eliminates dependence on external operating conditions, thereby ensuring greater robustness and reliability in the detection process.

1.5 Thesis Structure

The remainder of this dissertation is organized as follows. Chapter 2 introduces the fundamental concepts and models for air collision avoidance, providing the theoretical basis for subsequent developments. Chapter 3 describes the proposed algorithm in detail, integrating Model Predictive Control (MPC) within the air collision avoidance framework and outlining the adopted control strategies. Chapter 4 presents the simulation methodology and the obtained results, highlighting the performance of the developed approach under different conflict scenarios. Finally, Chapter 5 summarizes the main conclusions of the present work, discusses its limitations, and suggests directions for future research.

Chapter 2

Models for Air Collision Avoidance

2.1 Traffic Collision Avoidance System

The **Traffic Collision Avoidance System (TCAS)** is an onboard system widely used in commercial aviation with the goal of reducing the risk of mid-air collisions. It acts as a last line of defense, operating autonomously and independently from air traffic controllers. It is based on communication between aircraft transponders, allowing each aircraft to detect the presence and trajectory of nearby aircraft.

TCAS functions by continuously monitoring the relative position and closing rate between aircraft, issuing **Traffic Advisories (TA)** or **Resolution Advisories (RA)** when potential conflict situations are identified. These alerts aim to provide timely evasive instructions in order to ensure a minimum safe separation, both horizontally and vertically.

In its most advanced version (**TCAS II**), the system is capable of predicting the *Closest Point of Approach (CPA)* between two aircraft, based on the extrapolation of their current trajectories. From this prediction, the system evaluates the risk of breaching *minimum separation thresholds*, according to the following criteria:

- A defined **time horizon**, which allows past or distant future conflicts to be disregarded, focusing only on imminent threats;
- A **minimum horizontal distance**, represented by the CPA, whose violation indicates a dangerous proximity between aircraft;
- A **minimum vertical separation**, in accordance with the safety margins defined by aviation authorities.

The time until the closest point of approach, denoted by τ_{CPA}, corresponds to the moment when the distance between two aircraft will be at a minimum, assuming both maintain their current speed and direction. This value is obtained through the projection of the relative position vector onto the relative velocity vector. Equation (2.1) provides the calculation for τ_{CPA}.

$$\tau_{\text{CPA}} = \frac{(\mathbf{r}_1 - \mathbf{r}_2) \cdot (\mathbf{v}_1 - \mathbf{v}_2)}{\|\mathbf{v}_1 - \mathbf{v}_2\|^2} \tag{2.1}$$

Once the value of τ_{CPA} is known, the predicted position of each aircraft at that moment can be determined using *Equation* (2.2), allowing the calculation of the **minimum horizontal distance** between them, referred to as CPA_h. This metric considers only separation in the horizontal plane, disregarding the vertical component.

$$\text{CPA}_h = \left\| \left[(\mathbf{r}_1 + \mathbf{v}_1 \cdot \tau_{\text{CPA}}) - (\mathbf{r}_2 + \mathbf{v}_2 \cdot \tau_{\text{CPA}}) \right]_{xy} \right\| \tag{2.2}$$

As for the vertical separation, it can be obtained based on vertical position and the vertical component of velocity, as shown in *Equation* (2.3). This approach should be adopted considering a consistent reference frame; alternatively, if feasible, the altitudes may be directly compared, as seen in *Equation* (2.4).

$$\Delta h = \left\| (z_2 + v_{z2} \cdot \tau_{\text{CPA}}) - (z_1 + v_{z1} \cdot \tau_{\text{CPA}}) \right\| \tag{2.3}$$

$$\Delta h = \| h_2 - h_1 \| \tag{2.4}$$

In the vertical dimension, the *time to vertical convergence*, denoted by τ_{vertical}, must also be considered. This parameter represents the time interval required for two aircraft, assuming they maintain their current trends in altitude variation, to reach the same flight level/altitude. *Equation* (2.5) defines the expression adopted for the calculation of this parameter.

$$\tau_{\text{vertical}} = \frac{h_1 - h_2}{\dot{h}_2 - \dot{h}_1} \tag{2.5}$$

Where h_1 and h_2 represent the instantaneous altitudes of aircraft 1 and 2, respectively, and $\dot{h}_1$ and $\dot{h}_2$ are their respective vertical velocities.

The computation of the vertical velocity $\dot{h}_n$ depends on the reference frame adopted. It can either be obtained from a specific component of the velocity vector, or alternatively estimated through finite differences, depending on whether altitude values are available and how the coordinate system is defined.

2.1.1 Conflict Definition in TCAS

From a conflict detection perspective, the decision must be based on three key variables: the time to closest point of approach τ_{CPA}, the horizontal separation at CPA CPA_h, and the vertical separation Δh.

In general terms, a potential conflict is said to occur when all of the following conditions are simultaneously satisfied:

$$\begin{cases} \tau_{\text{CPA}} < \tau_{\text{CPA}_{\text{threshold}}} \vee \tau_{vertical} < \tau_{v_{\text{threshold}}} \\ \text{CPA}_h < \text{CPA}_{h_{\text{threshold}}} \\ \Delta h < \Delta h_{\text{threshold}} \end{cases} \tag{2.6}$$

These inequalities define the minimum safety thresholds in the temporal, horizontal, and vertical dimensions, respectively. The conflict condition is therefore triggered *if and only if* all three criteria in *Equation* (2.6) are simultaneously violated. This approach ensures that minor or distant encounters are filtered out, focusing the system's response on imminent and significant threats to flight safety.

2.1.2 Threshold Values Adopted for Conflict Detection

The threshold table presented earlier (Table 1.1) defines distinct values for parameters such as τ, DMOD, ZTHR, ALIM, and TVTHR depending on the aircraft's altitude, which determines its corresponding Sensitivity Level (SL). In real systems like TCAS II, each aircraft independently assesses the risk of conflict by applying the thresholds appropriate to its own flight level.

However, in the implementation developed in this dissertation, conflict detection is performed in a centralized manner by analyzing all pairs of aircraft simultaneously. This creates a practical limitation: before determining which aircraft should perform an avoidance manoeuvre, it is not possible to directly apply the altitude-specific thresholds, since the concept of "own aircraft" has not yet been defined.

To address this limitation, the adopted strategy relies on using general or conservative criteria for conflict detection, evaluating separation parameters symmetrically for each pair of aircraft. Once a potential conflict is identified, one of the two aircraft whose planned trajectories are known is selected to perform the avoidance manoeuvre.

To overcome this limitation, the adopted trajectory is based on a more conservative approach, using the highest altitude among all aircraft to determine a more critical Sensitivity Level (SL). Once a potential conflict is identified, one of the two aircraft whose planned trajectories are known is selected to perform the avoidance manoeuvre.

2.2 Flight Dynamics Model

The flight of an aircraft, although often associated with a sequence of *waypoints*, can also be described discretely through variables such as speed, heading angle (χ), and flight path angle (γ). When combined with other system modules, this modelling approach enables the analysis and potential modification of the aircraft's trajectory over time.

2.2.1 Aircraft Displacement in the ENU Reference Frame

Based on the motion of an aircraft and the previously introduced variables, one can define the equations that describe its displacement in space. These relationships are expressed in *Equation* (2.7), which represents the aircraft's displacement in a local ENU (East-North-Up) reference frame, constructed from a point defined in geodetic coordinates. Ensuring consistency among the reference frames used is essential to guarantee that all variables are expressed in a manner compatible with the expected results. This choice is justified by the fact that both the heading angle (χ) and the flight path angle (γ) are naturally defined with respect to this type of local frame.

$$\begin{aligned} \Delta x_{\text{ENU}} &= v \cdot \cos(\gamma) \cdot \sin(\chi) \cdot \Delta t \\ \Delta y_{\text{ENU}} &= v \cdot \cos(\gamma) \cdot \cos(\chi) \cdot \Delta t \\ \Delta z_{\text{ENU}} &= v \cdot \sin(\gamma) \cdot \Delta t \end{aligned} \tag{2.7}$$

It is important to note that, in this model, the velocity v refers to the magnitude of the velocity vector (i.e., the speed), as in *Equation* (2.7). Since this is a scalar quantity, its value remains the same regardless of whether it is computed in the ENU or ECEF reference frame, provided that the angular variables χ and γ are defined consistently with the chosen frame.

2.2.2 Transformation of Displacement from ENU to ECEF

Initially, in *Equation* (2.7), the aircraft's displacement is computed in the local ENU frame, to ensure consistency with the variables that describe the motion. However, position-related computations are typically performed in the Earth-Centered Earth-Fixed (ECEF) frame.

The ECEF frame is used for all position-related calculations due to its global consistency and fixed orientation relative to the Earth. Unlike the local ENU frame, which depends on a specific reference point and varies with location, the ECEF frame provides a continuous Cartesian coordinate system that is well-suited for large-scale simulations and interactions between multiple agents. For this reason, although displacements may be conve-

niently computed in ENU, they are immediately transformed to ECEF for integration and propagation.

This transformation is carried out using a rotation matrix R, which maps local ENU vectors to the corresponding directions in the ECEF frame. The matrix is defined based on the geodetic latitude ϕ and longitude λ of the ENU origin, as shown in *Equation* (2.8):

$$R = \begin{bmatrix} -\sin(\lambda) & -\sin(\phi)\cos(\lambda) & \cos(\phi)\cos(\lambda) \\ \cos(\lambda) & -\sin(\phi)\sin(\lambda) & \cos(\phi)\sin(\lambda) \\ 0 & \cos(\phi) & \sin(\phi) \end{bmatrix} \tag{2.8}$$

The displacement vector in the ECEF frame is then computed according to *Equation* (2.9):

$$\vec{v}_{\text{ECEF}} = R \cdot \Delta\vec{r}_{\text{ENU}} \tag{2.9}$$

Finally, the updated position of the aircraft in ECEF coordinates is obtained using *Equation* (2.10):

$$\vec{r}_{\text{ECEF}} = \vec{r}_{\text{ECEF,prev}} + \vec{v}_{\text{ECEF}} \tag{2.10}$$

2.3 Model Predictive Control (MPC)

Model Predictive Control (MPC) is an advanced control strategy widely recognised for its ability to handle multivariable dynamic systems subject to operational constraints. Its central principle is the use of a system model to predict future behaviour over a finite prediction horizon, denoted by p. Based on these predictions, MPC determines control actions by optimising a cost function that reflects system performance and constraint satisfaction.

MPC has shown great versatility across a wide range of engineering applications [26], making it particularly suitable for coordinating complex systems under operational constraints. In the context of UAV conflict resolution, it provides a systematic way of anticipating future states and enforcing safety requirements while optimising the control effort.

Figure 2.1, adapted from [27], presents the general MPC framework. The optimisation process relies on predictions from the system model and delivers the control actions that best achieve the desired objectives under the defined constraints. While standard implementations rely on real-time measurements, in this work the feedback corresponds to predicted UAV states generated by the dynamic simulation model.

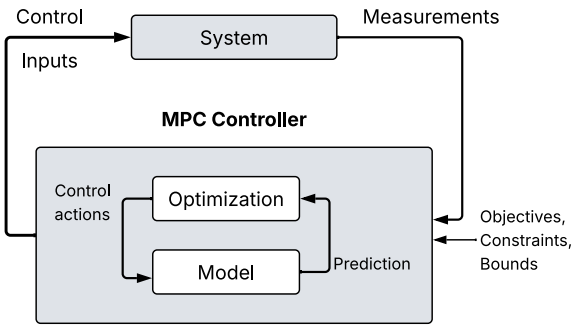

Figure 2.1: Illustration of model predictive control.

2.3.1 Fundamental Concepts

Model Predictive Control (MPC) operates by repeatedly solving an optimization problem at each time step, using the current state of the system as the starting point. The goal of this optimization is to determine the best possible sequence of control inputs over a finite prediction horizon with p steps, taking into account the system dynamics, imposed constraints, and a predefined cost function.

Importantly, the system dynamics are embedded within the cost function itself: the optimization seeks to minimize or maximize a cost that depends on the predicted evolution of the system state, which is computed through the dynamic model, as well as on other parameters associated with system performance or control objectives.

Typically, instead of applying the entire optimized sequence, only the first control input is implemented. At the next time step, the process is repeated: the new state of the system is measured, a new optimization is carried out, and a new control input is applied. This approach, known as the receding horizon strategy, forms the core of MPC. By continuously updating the control actions based on the most recent state information, the controller remains adaptive to disturbances, modelling errors, and environmental changes, providing greater robustness and flexibility compared to open-loop control strategies.

However, in certain situations, applying only one input per iteration may lead to undesirable behaviours, such as oscillations or excessive control effort, especially when control continuity is required. In such cases, it may be beneficial to apply a short sequence of control inputs from the optimized trajectory before re-optimizing the system. This strategy can improve system stability and smooth the control response, without significantly compromising the controller's adaptability.

For improved computational efficiency, it is good practice to reuse the control input vector obtained in one iteration as the initial guess for the next. This approach aims to reduce optimization time by taking advantage of the fact that the optimal solution typically changes little between consecutive time steps. Additionally, the maximum number of optimizer iterations can be adjusted to ensure response times are compatible with the system's real-time requirements.

Figure 2.2 presents a flowchart of the MPC optimization process, highlighting the reuse of the control vector from the previous iteration as the initial guess for the next one. This strategy contributes to the computational efficiency of the method.

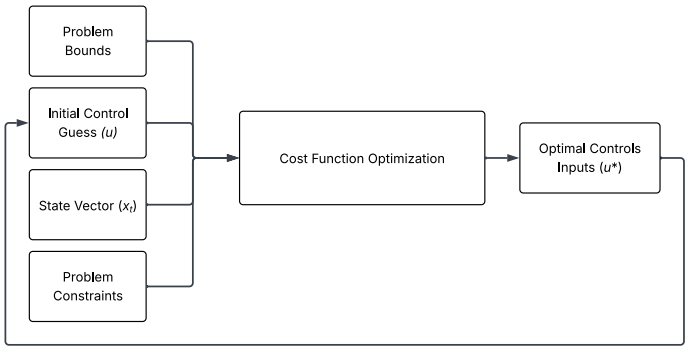

Figure 2.2: Flowchart of the MPC optimization process.

2.3.2 Bounds, Constraints and Optimization Algorithm

2.3.2.1 Bounds and Constraints

As shown in Figure 2.2, to achieve a proper optimization of the cost function—whether it involves minimization or maximization—and to realistically represent the system's behavior, it is essential to define problem-specific bounds and constraints. These elements ensure that the control inputs optimized by the MPC respect the physical and operational limitations of the system.

In general, Bounds refer to upper and/or lower limits applied directly to individual control or state variables. These restrictions are typically constant and enforced throughout the prediction horizon.

On the other hand, Constraints represent more general limitations, involving relationships between multiple variables or explicit dependencies on the system dynamics. They can be expressed as equality or inequality conditions and may take either linear or nonlinear forms.

Both bounds and constraints are always specific to the problem and system under consideration, and should be defined to replicate, as closely as possible, all the limitations that would be expected in a real-world application context. A proper definition of these restrictions contributes significantly to a more stable and reliable behavior of the optimization algorithm.

2.3.2.2 Optimization Algorithms

An optimization problem, and in this specific case a problem involving the use of Model Predictive Control (MPC), can be solved using different numerical algorithms. Two widely used methods are Sequential Least Squares Programming (SLSQP) and Broyden–Fletcher–Goldfarb–Shanno (BFGS). The choice among the available methods depends closely on the nature of the problem as well as the presence or absence of constraints and limitations.

The *Sequential Least Squares Programming* (SLSQP) is a gradient-based iterative optimization algorithm designed to solve nonlinear programming problems with constraints [28]. It is widely used in problems involving equality and/or inequality conditions, due to its ability to efficiently handle such constraints.

This approach combines elements of Newton's methods with Quadratic Programming techniques, enabling effective treatment of problems in which both the cost function (or objective function) and the constraints are differentiable.

As shown in Algorithm 1, at each iteration, the SLSQP constructs a Quadratic Programming (QP) subproblem that represents a local approximation of the original problem. This subproblem incorporates a linearization of the constraints and a quadratic approximation of the Lagrangian function, based on the available gradients and an estimate of the Hessian matrix. The solution to the subproblem provides a descent direction that guides the update of the solution, promoting the convergence of the method.

Algorithm 1 Iterative resolution process of the SLSQP optimization algorithm.

1. Define initial estimate $\mathbf{x}_0$, tolerances, and constraints.
2. Initialize the approximate Hessian matrix $\mathbf{B}_0$
3. Repeat until convergence:
   (a) Evaluate gradients: ∇f, ∇g_i, ∇h_j
   (b) Solve QP subproblem:

$$\min_{\Delta \mathbf{x}} \quad \nabla f(\mathbf{x}_k)^T \Delta \mathbf{x} + \frac{1}{2} \Delta \mathbf{x}^T \mathbf{B}_k \Delta \mathbf{x}$$

   with linearized constraints:

$$\nabla g_i(\mathbf{x}_k)^T \Delta \mathbf{x} + g_i(\mathbf{x}_k) = 0, \quad \nabla h_j(\mathbf{x}_k)^T \Delta \mathbf{x} + h_j(\mathbf{x}_k) \leq 0$$

   (c) Update $\mathbf{x}_{k+1}$ and $\mathbf{B}_{k+1}$
4. Return $\mathbf{x}^*$

The *Broyden–Fletcher–Goldfarb–Shanno* (BFGS) method is a gradient-based optimization algorithm, classified as a quasi-Newton method [29]. This approach is particularly effective in nonlinear programming problems without explicit constraints, as long as the

objective function is continuously differentiable over the discretization domain.

Unlike Newton's method, which explicitly requires the computation of the Hessian matrix, BFGS iteratively builds and updates an approximation of the inverse Hessian using only evaluations of the gradient of the cost function. This strategy significantly reduces computational cost while ensuring fast convergence to an optimal solution.

As illustrated in Algorithm 2, at each iteration the algorithm updates the inverse Hessian approximation based on the variation of the cost function gradient between consecutive points. This matrix is then used to compute the descent direction that guides the solution update.

Algorithm 2 Iterative resolution process of the BFGS optimization algorithm.

1. Define initial estimate $\mathbf{x}_0$, tolerance, and initial inverse Hessian approximation $\mathbf{H}_0 = \mathbf{I}$
2. Repeat until convergence:
   (a) Evaluate the gradient: $\nabla f(\mathbf{x}_k)$
   (b) Compute the search direction: $\mathbf{p}_k = -\mathbf{H}_k \nabla f(\mathbf{x}_k)$
   (c) Perform a line search to determine step size α_k
   (d) Update the solution: $\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{p}_k$
   (e) Compute:
   $$\mathbf{s}_k = \mathbf{x}_{k+1} - \mathbf{x}_k, \quad \mathbf{y}_k = \nabla f(\mathbf{x}_{k+1}) - \nabla f(\mathbf{x}_k)$$
   (f) Update the inverse Hessian approximation:
   $$\mathbf{H}_{k+1} = \left(\mathbf{I} - \rho_k \mathbf{s}_k \mathbf{y}_k^T\right) \mathbf{H}_k \left(\mathbf{I} - \rho_k \mathbf{y}_k \mathbf{s}_k^T\right) + \rho_k \mathbf{s}_k \mathbf{s}_k^T$$
   where $\rho_k = \frac{1}{\mathbf{y}_k^T \mathbf{s}_k}$
3. Return $\mathbf{x}^*$

Although the BFGS method is originally designed for unconstrained optimization problems, it is possible to adopt the L-BFGS-B variant, which allows the inclusion of Bound constraints in the resolution process. This extension enables the algorithm to handle problems where variables are restricted within specified lower and upper limits.

Chapter 3

Model Predictive Control Based Air Collision Avoidance

3.1 Algortithm Flowchart

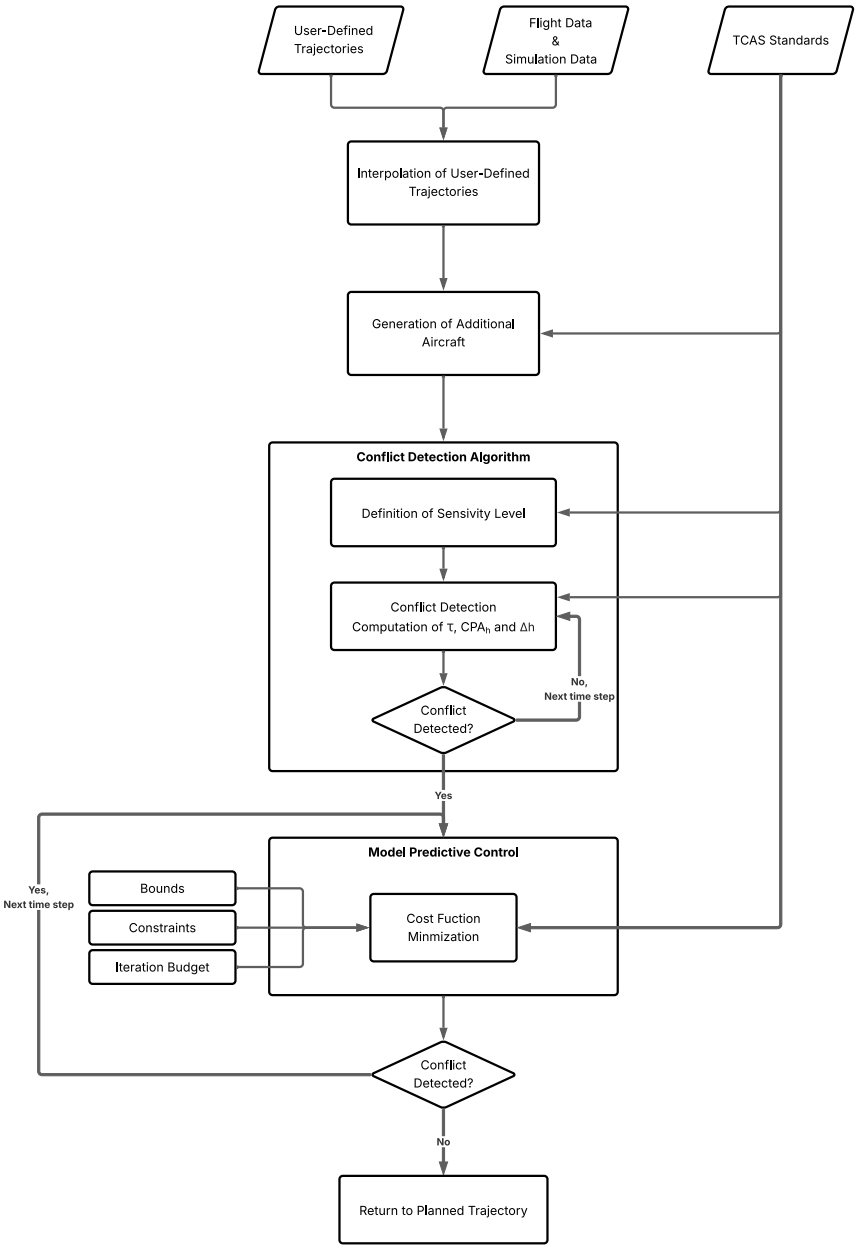

Figure 3.1: Algorithm flowchart illustrating the main simulation steps.

3.2 Simulation Input Data

In order to meet all the necessary conditions for the simulation to begin, a set of global input data must be defined. As mentioned in Section 1.4, this approach requires two predefined trajectories (specified by waypoints) to ensure that an imminent conflict will occur, making the conflict resolution action critical for safety. The choice of using only two trajectories is justified by the intention to generalize the algorithm's behaviour as much as possible. If five trajectories were defined via waypoints, the solution would always be essentially the same, reducing the relevance of evaluating the algorithm's decision-making. The predefined trajectories can be arbitrary and are left to the user's discretion, including whether or not a conflict occurs throughout the simulation time.

To run the simulation, two *EXCEL* input tables must be completed. As shown in Table 3.1, and as previously mentioned, two trajectories must be defined. For each waypoint, the following parameters are required: Latitude and Longitude coordinates (in degrees), the aircraft's altitude (in meters), and the corresponding timestamp for that waypoint (in seconds).

WP_0	ϕ_0 (°)	λ_0 (°)	h_0 (m)	t_0 (s)	WP_1	ϕ_1 (°)	λ_1 (°)	h_0 (m)	t_1 (s)
1	39.8745	-7.4565	600	0	1	39.8306	-7.4335	500	0
2	39.8602	-7.4499	600	70	2	39.8490	-7.4451	500	70
3	39.8484	-7.4447	600	80	3	39.8614	-7.4507	500	80
4	39.8388	-7.4383	600	90	4	39.8707	-7.4498	500	90
5	39.8359	-7.4244	600	100	5	39.8772	-7.4459	500	100

Table 3.1: Example of predefined trajectories for aircraft 0 and 1.

Table 3.2 specifies the parameters for each aircraft, including the velocity (in meters per second)—except for aircraft 0 and 1, whose velocity is implicitly defined by their trajectory—the minimum and maximum velocity (m/s), minimum and maximum acceleration (m/s^2), minimum and maximum roll rate (°/s), and minimum and maximum flight path angle rate (°/s). It should be noted that the "Category" field may correspond to a user-defined criterion or a specific regulatory classification.

Índice	$V(m/s)$	$V_{\min}$ (m/s)	$V_{\max}$ (m/s)	$a_{\min}$ (m/s^2)	$a_{\max}$ (m/s^2)	$\dot{\phi}_{\min}$ (°/s)	$\dot{\phi}_{\max}$ (°/s)	$\dot{\gamma}_{\min}$ (°/s)	$\dot{\gamma}_{\max}$ (°/s)	$b\,(m)$	Categoria
0	––	15	50	0	10	0	20	0	5	10	2
1	––	15	50	0	10	0	20	0	5	6	1
2	36	15	50	0	10	0	20	0	5	8	1
3	36	15	50	0	10	0	20	0	5	6	1
4	36	15	50	0	10	0	20	0	5	10	2

Table 3.2: Example of physical and dynamic limits for each aircraft.

Subsequently, the simulation time step (Δt) must also be defined. This time step determines how frequently the state of each aircraft is updated during the simulation. A smaller Δt increases the temporal resolution and allows for more precise tracking of aircraft positions and conflict detection, but it also leads to higher computational costs. Conversely, a larger Δt reduces computational load but may compromise accuracy, especially in fast-

changing scenarios. The choice of Δt is therefore a key parameter in balancing accuracy and performance in both the simulation and the conflict resolution strategy.

3.3 Interpolation of Predefined Trajectories

3.3.1 Continuous Trajectory Generation from Waypoints

Both the trajectory of aircraft 0 and aircraft 1 are known in advance and defined by a discrete set of waypoints. This definition, combined with an appropriate choice of time step Δt, allows for accurate temporal discretization of the trajectories.

Each defined waypoint contains the geodetic coordinates of the aircraft — latitude, longitude, and altitude — at a given time instant. However, to facilitate the computation process — particularly for conflict detection and trajectory management — the provided coordinates are converted into the Earth-Centered, Earth-Fixed (ECEF) reference frame, resulting in a set of (x, y, z) coordinates expressed in meters.

A cubic spline interpolation is then applied independently to each of the converted spatial components. This interpolation is performed using the `CubicSpline` function from the `SciPy` library, which constructs a third-degree polynomial while ensuring continuity of the first and second derivatives. By default, natural boundary conditions are assumed, meaning that the second derivatives at the endpoints of the interpolation interval are set to zero.

In a general form, *Equation* 3.1 shows the spline formulation adopted by the function over each interval $[t_i, t_{i+\Delta t}]$:

$$S_i(t) = a_i + b_i(t - t_i) + c_i(t - t_i)^2 + d_i(t - t_i)^3 \tag{3.1}$$

The coefficients a_i, b_i, c_i, and d_i are computed internally by the function to ensure both smoothness and continuity across the defined waypoints.

This approach enables the evaluation of the positions of aircraft 0 and 1 at any simulation instant $t_k = k \cdot \Delta t$, thereby significantly enhancing the system's ability to predict and prevent potential conflicts.

3.3.2 Finite Difference Approach for Velocity Estimation

After obtaining a continuous representation of the aircraft positions through cubic spline interpolation, it is necessary to compute the corresponding velocity magnitude at each simulation step. Since the position is interpolated independently in each spatial com-

ponent, the instantaneous velocity can be estimated using a first-order finite difference applied component-wise.

Specifically, the scalar velocity is obtained by computing the Euclidean distance between two consecutive position samples and dividing it by the simulation time step. This leads to the finite difference expression shown in *Equation* 3.2:

$$V(t_k) \approx \frac{1}{\Delta t}\sqrt{[x(t_k) - x(t_{k-1})]^2 + [y(t_k) - y(t_{k-1})]^2 + [z(t_k) - z(t_{k-1})]^2} \tag{3.2}$$

This expression provides an estimate of the instantaneous velocity magnitude based on the distance travelled between two consecutive time steps. The method ensures consistency with the simulation resolution defined by Δt, and allows for an accurate assessment of the aircraft's kinematic state at each iteration.

3.4 Random Generation of Aircraft for Conflict Scenario Simulation

In order to test the algorithm under constantly changing scenarios, and in line with the objectives defined in Section 1.4, it is necessary to establish a method to determine the initial positions of the three additional aircraft in a completely random manner. This approach arises from the need to avoid consecutively similar cases in terms of the solution, allowing for a broader assessment of the algorithm's generalisation capability.

The implementation of a stochastic logic enables consecutive simulations with distinct initial conditions in each run, regardless of the prediction horizon considered. In this way, the algorithm can be tested in terms of robustness, operational flexibility, and stability when facing more challenging scenarios.

In a collision avoidance system, scenario diversity is a crucial element to ensure that the algorithm is not tailored to solve a limited number of situations, but is instead capable of handling any possible configuration among the five aircraft involved. Therefore, the ability to dynamically generate the remaining aircraft randomly brings dynamism and unpredictability to the simulation, reflecting more realistic air traffic conditions.

3.4.1 Definition of Initial Positions for Random Aircraft

The definition of the initial positions of the remaining aircraft is based on an iterative and stochastic algorithm, designed to introduce spatial variability into the simulated scenario without ever compromising the minimum aircraft separation criteria. This approach aims to create realistic initial conditions, while also providing a general-purpose component to

the algorithm as a whole.

The procedure is applied individually to each of the three additional aircraft. Through an iterative process, it seeks to determine an admissible spatial configuration within a predefined maximum number of attempts. The methodology can be described as follows:

1. Reference Center Selection:

   For each additional aircraft, a reference center is randomly selected from the set of aircraft already positioned. From this set, an index is chosen at random (corresponding to one of the previously positioned aircraft). This reference center serves as the basis for defining the region in which the new aircraft may be placed.

2. Position Generation in the Local Frame (ENU):

   Considering the chosen reference point as the origin of a local East-North-Up (ENU) coordinate system, a circular geometry is defined between two limits: a minimum radius equal to the horizontal separation threshold CPA_h, and a maximum radius of 25 kilometres, representing an imposed bound that may reflect communication or sensing limitations.

   Within this ring, a random position is generated using polar coordinates, by selecting an angle $\theta \in [0, 2\pi]$ and a radius $r \in [\text{CPA}_h, r_{\text{range}}]$. The position in ENU coordinates is then computed using *Equations* (3.3) and (3.4).

   $$\Delta n = r \cdot \cos(\theta) \qquad (3.3) \qquad\qquad \Delta e = r \cdot \sin(\theta) \qquad (3.4)$$

3. Conversion to Geodetic Coordinates:

   Since ENU displacements are expressed in meters, it is necessary to convert them into degrees in order to ensure consistency between reference frames. To this end, the metric scale of a geodetic degree is used to convert the displacements obtained from *Equations* (3.3) and (3.4) so that the resulting latitude and longitude can be obtained by adding them to the original reference position. This conversion is carried out using *Equations* (3.5) and (3.6), where the factor 111320 corresponds to the approximate length in meters of one degree of latitude under the spherical Earth assumption.

   $$\Delta\phi = \frac{\Delta n}{111320} \qquad (3.5) \qquad\qquad \Delta\lambda = \frac{\Delta e}{111320 \cdot \cos(\phi_{\text{ref}})} \qquad (3.6)$$

   As for the altitude, a fully random value is selected between a minimum flight altitude fixed at 200 meters and a maximum flight altitude fixed at 5000 meters.

4. Determination of the New Position:

   Based on the computed variations $\Delta\phi$ and $\Delta\lambda$, and considering the reference latitude and longitude ϕ_{ref} and λ_{ref}, the coordinates of the new aircraft position are obtained using *Equations* (3.7) and (3.8).

$$\phi = \phi_{\text{ref}} + \Delta\phi \tag{3.7}$$

$$\lambda = \lambda_{\text{ref}} + \Delta\lambda \tag{3.8}$$

5. Position Validity:

   As outlined in Section 1.4, the objective is for the additional aircraft to be generated in such a way that they challenge the algorithm with respect to spatial constraints, without however creating an immediate conflict at the moment of their insertion. For this purpose, the separation distance is checked both in the horizontal and vertical planes, relative to all previously positioned aircraft — whether predefined by the user or randomly generated in earlier iterations.

   Given two points defined by their geodetic coordinates, (ϕ_1, λ_1) and (ϕ_2, λ_2), the horizontal distance d_h between them can be computed using the Haversine formula [**?**]. This calculation is based on an auxiliary term a, defined in *Equation* (3.9), which contributes to the numerical stability of the expression, the central angle c, which represents the angular distance between the two points on the spherical surface (*Equation* (3.10)), and finally the multiplication of this angle by the Earth's radius R, as shown in *Equation* (3.11). This method is widely used to estimate geodetic distances over the Earth's surface, considering the Earth as a sphere with a mean radius of $R = 6\,371\,000$ m.

$$a = \sin^2\left(\frac{\Delta\phi}{2}\right) + \cos(\phi_1)\cos(\phi_2)\sin^2\left(\frac{\Delta\lambda}{2}\right) \tag{3.9}$$

$$c = 2 \cdot \text{atan2}\left(\sqrt{a}, \sqrt{1-a}\right) \tag{3.10}$$

$$d_h = R \cdot c \tag{3.11}$$

   where $\Delta\phi = \phi_2 - \phi_1$ and $\Delta\lambda = \lambda_2 - \lambda_1$, with all angles expressed in radians.

   The difference in altitude is computed directly by comparing the altitude of the newly generated aircraft with the altitudes of the remaining aircraft, as shown in *Equation* (3.12).

$$d_v = |h_2 - h_1| \tag{3.12}$$

   With these two parameters — the horizontal distance d_h and the vertical difference d_v — defined, the validity of the newly generated aircraft's position is evaluated, as expressed in *Equation* (3.13). For each pair formed by the new aircraft and one of the existing aircraft, it is verified whether the minimum spatial separation criteria are satisfied. This verification ensures that the generated situation constitutes a meaningful scenario for the algorithm, placing it under decision-making pressure and promoting the activation of the conflict detection and resolution module. For

the separation criteria, sensitivity level 7 (Table 1.1) is considered in order to validate the position generated by the algorithm.

$$\begin{cases} d_h > \text{CPA}_{h_{\text{Limit}}} \\ d_v > \Delta h_{\text{Limit}} \end{cases} \tag{3.13}$$

If the position is confirmed to be valid within the predefined iteration limit — an arbitrarily chosen value to ensure the feasibility of generating all three aircraft —, the corresponding coordinates are then converted to the ECEF reference frame, in order to maintain consistency of reference frames throughout the algorithm.

6. Trajectory control variables:

   The value of χ, corresponding to the aircraft's heading, is generated completely at random, assuming a uniform distribution in the interval $[-60°, 60°]$. This choice aims to introduce diversity in the initial trajectory directions, promoting realistic and variable scenarios for evaluating the conflict detection and resolution algorithm.

   On the other hand, since the altitude of the aircraft is randomly generated within a defined range, the climb or descent angle γ cannot be assigned arbitrarily. Instead, it must be adapted to the aircraft's initial altitude to ensure that, throughout the total simulation time t_{sim}, the vertical trajectory remains within operational flight limits—namely, between a minimum altitude and a maximum admissible ceiling.

   To ensure this behavior, a simplified linear model was used, in which the minimum admissible value of γ is defined according to the aircraft's initial altitude, in proportion to a reference vertical range, the simulation time, and the maximum slope considered. The general form of the adopted equation is presented in *Equation* (3.14):

$$\gamma_{\text{min predicted}} = -\left(\frac{h}{h_{\text{ref}}}\right) \cdot \frac{t_{\text{ref}}}{t_{\text{sim}}} \cdot \gamma_{\text{ref}} \tag{3.14}$$

   It is important to note that the value of h_{ref} was not defined through analytical calculations, but rather based on preliminary simulations carried out with $\gamma = 5°$ and $t = 500$ s. These simulations showed that the vertical variation achieved by the aircraft was approximately 1290 meters. To ensure an additional safety margin that guarantees the aircraft reaches the operational minimum altitude (200 meters), this value was added to the reference, resulting in $h_{\text{ref}} = 1490$ m.

   With this empirical parameterization, *Equation* (3.14) was particularized, resulting in the concrete form shown in *Equation* (3.15).

$$\gamma_{\text{min predicted}} = -\left(\frac{h}{1490}\right) \cdot \frac{500}{t_{\text{sim}}} \cdot 5^{\circ} \tag{3.15}$$

Analogously, the maximum admissible value of γ, corresponding to the climb angle, is obtained from the same structure of *Equation* (3.14), adjusting only the vertical range considered. In this case, instead of descending to a minimum altitude, the aircraft must be able to climb to the operational ceiling h_{max}. The range is therefore given by $h_{\text{max}} - h$, with all other parameters remaining unchanged. The reference altitude in this expression remains fixed, based on the variation observed in simulations with $\gamma = 5^{\circ}$. The resulting equation for the predicted maximum value of γ is shown in *Equation* (3.16).

$$\gamma_{\text{max predicted}} = \left(\frac{h_{\text{max}} - h}{h_{\text{ref}}}\right) \cdot \frac{t_{\text{ref}}}{t_{\text{sim}}} \cdot \gamma_{\text{ref}} \tag{3.16}$$

Accordingly, *Equation* (3.16) can also be particularized using the empirical parameters obtained. In this case, the flight ceiling was fixed at 5000 m, resulting in the concrete expression shown in *Equation* (3.17).

$$\gamma_{\text{max predicted}} = -\left(\frac{5000 - h}{1290}\right) \cdot \frac{500}{t_{\text{sim}}} \cdot 5^{\circ} \tag{3.17}$$

Based on these limits, the value of γ assigned to each aircraft is randomly selected within the interval $[\gamma_{\text{min}}, \gamma_{\text{max}}]$, thus ensuring that the resulting trajectory is compatible with the initially generated altitude, without exceeding the imposed vertical boundaries.

This process prevents unrealistic situations, such as dropping below ground level or exceeding the operational ceiling, ensuring physical consistency and robustness in the simulation.

3.4.2 Random Aircraft Simulation

With the initial position parameters randomly defined, it is possible to simulate the trajectory of each additional aircraft over time, assuming constant values for the heading angle (χ) and climb/descent angle (γ), as described in Section 1.4. The simulation is performed using the Flight Dynamics Model presented in Section 2.2, which allows the aircraft position to be computed at each time step.

The total simulation time, time step, and separation limits are predefined by the user in order to standardise all simulations. As a result of this process, the x, y, and z coordinates

of each additional aircraft are obtained at every time instant.

3.4.3 Considerations Regarding the Simulation Approach

3.4.3.1 Local Reference Frame Selection

In order to generate the initial positions of additional aircraft, it was necessary to define a reference point from which the spatial dispersion would be applied. Although both ECEF (Earth-Centered, Earth-Fixed) and ENU (East-North-Up) coordinate systems are valid representations of spatial position, the ENU system was selected for this purpose due to its intuitive local interpretation.

Using ENU allows the dispersion of aircraft to be defined relative to a local origin point, with the axes aligned with the cardinal directions on the Earth's surface. This greatly simplifies the process of generating random positions within a controlled horizontal and vertical range, especially when using circular or ring-shaped patterns centered around a known aircraft.

In contrast, working directly in the ECEF system would require more complex geometric transformations to define local directions and ranges, due to the global, Earth-centered nature of the coordinate frame. Therefore, the ENU system was used for the generation of relative positions, which were later converted to geodetic and ECEF coordinates as needed for simulation and conflict detection.

Figure 3.2 [30] provides a visual representation that highlights the differences between the ENU and ECEF reference frames.

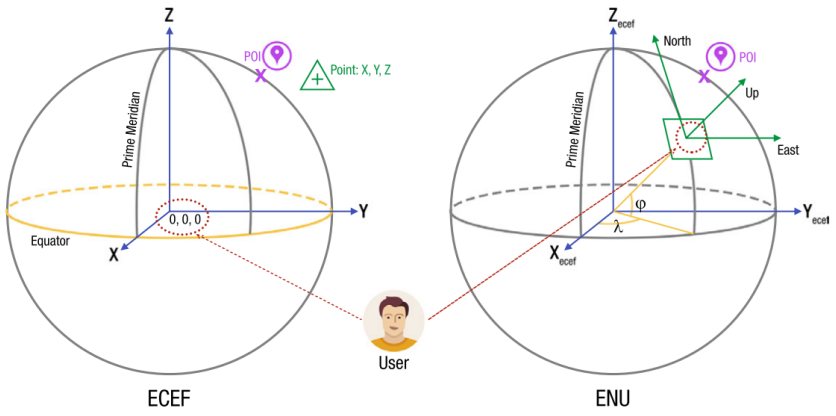

Figure 3.2: Comparison between the Earth-Centered Earth-Fixed (ECEF) and the East-North-Up (ENU) reference frames.

3.4.3.2 Constant Degree-to-Meter Factor Assumption

To convert distances in meters to approximate variations in latitude and longitude, a fixed scale factor of 111,320 meters per degree was used. This value corresponds to the average length of one degree of latitude and serves as a rough approximation for both latitude and longitude conversion.

In this context, the use of a simplified and constant conversion factor is fully justified, as the objective is not to obtain geodetically precise coordinates, but merely to generate random positions that satisfy predefined admissibility criteria. Since the goal is to ensure spatial separation and adequate distribution, exact positional accuracy is not required.

Therefore, the approximation introduces no relevant error and significantly simplifies the generation of random geographic coordinates within the intended spatial bounds.

3.4.4 Timing of Aircraft Generation

The generation of additional aircraft could, in principle, be triggered at any moment during the simulation. However, in this implementation, all aircraft are generated at the initial instant. This approach ensures that the maximum number of aircraft is present from the beginning, allowing for consistent and comprehensive conflict detection throughout the entire simulation horizon.

3.5 Traffic Collision Avoidance System Algorithm

The specific algorithm of the Traffic Collision Avoidance System (TCAS) is primarily responsible for the continuous monitoring of the movement of all aircraft involved in the simulation. In this context, and following the considerations presented in Section 1.4, this monitoring is implemented from an external and global perspective; that is, the system simultaneously analyses all aircraft in the simulated airspace, assessing potential collisions between pairs of aircraft at each time step.

Conflict detection is based on the analysis of the predicted trajectories for each aircraft, as defined in Sections 3.3 and 3.4, which respectively describe the interpolation of known trajectories and the random generation of new positions. Based on the criteria established in Section 2.1, the system performs real-time monitoring by iteratively calculating three key parameters: τ, CPA_h, and Δh.

The information regarding the existence of conflicts is stored in a three-dimensional Boolean matrix, where each layer represents a time step in the simulation. The rows and columns of this matrix correspond to the indices of the aircraft, allowing the identification, at each instant, of the pairs of aircraft in a conflict situation. This matrix-based representation

enables a clear visualization of the temporal evolution of conflicts and also provides essential data for the subsequent modules of the system. The matrix is dynamically updated throughout the simulation, being progressively filled as the iterations advance, following the total number of time steps defined by the user.

At each advancement of the simulation time step, the system checks whether, at that specific instant, the collision matrix contains any indication of a potential conflict between aircraft pairs, marked with the logical value `TRUE`. This check is crucial to promptly identify situations requiring evasive action. If a conflict is detected, the resolution procedure is immediately triggered through a call to the model predictive control (MPC) module, which is responsible for computing a corrective manoeuvre for the aircraft designated as the evader. This process ensures that separation decisions are made in a reactive yet informed manner, based on the predicted trajectories and the imposed operational constraints.

The designation of the aircraft that must execute the evasive manoeuvre is based on the category assigned to each aircraft, as exemplified in Table 3.2. This category can be defined according to various operational or physical criteria, such as maximum take-off weight (MTOW), wingspan, cruise speed, manoeuvrability, or any other parameter deemed relevant to the simulation. The underlying goal of this classification is to ensure that, whenever possible, the responsibility for the evasive manoeuvre falls on the aircraft with greater operational flexibility or lower strategic priority within the simulated air traffic scenario.

In more complex scenarios, in which multiple conflicts may occur simultaneously or where there are no significant differences in aircraft characteristics, the criteria for selecting the evader aircraft can be further refined. Additional factors may be considered, such as the number of simultaneous conflicts each aircraft is involved in, the distance to its destination, the urgency level of its mission, or even performance-related parameters such as its maximum climb rate or roll rate. This approach allows for a more informed decision-making process, dynamically adapting to the demands of the simulated scenario and providing greater robustness and realism to the collision avoidance system.

Nevertheless, in this particular approach, the use of aircraft categories proved sufficient to ensure the success of the simulations, allowing for a coherent selection of the evader aircraft without compromising the effectiveness of the conflict resolution process.

3.6 Model Predictive Control Algorithm

After the detection of a potential conflict that violates the predefined safety standards (Section 3.5), an effective and safe resolution becomes necessary. One possible approach consists of selecting, from among the aircraft involved, one that performs an evasive manoeuvre — that is, temporarily leaves its predefined trajectory with the intention of re-

turning to it once the conflict has been fully resolved.

This evasive action is implemented through a set of control inputs, which, based on a predefined cost function, allows for the computation of a locally optimal decision at the considered instant. This type of problem is naturally suited to the use of Model Predictive Control (MPC) algorithms, which can explicitly incorporate both physical and operational constraints of the aircraft. In this way, it is possible to generate a sequence of actions that result in a coherent, feasible, and physically realistic evasive trajectory that respects all imposed constraints while minimising the associated manoeuvre costs.

As discussed in Section 2.3, an MPC algorithm requires the definition of several parameters in order to ensure that the optimisation process is both fast and well-adapted to the problem at hand.

3.6.1 Control Vector

As previously mentioned, the evasive trajectory is determined based on a set of control inputs applied to the aircraft, resulting from the resolution of an optimisation problem that aims to temporarily adjust its route in order to avoid conflicts and ensure operational safety. This manoeuvre must be sufficiently flexible to respond to various types of conflicts, while also being constrained enough to guarantee that the actions taken are both physically and operationally feasible.

Considering the specific characteristics of the problem — namely the need for fast, realistic manoeuvres that comply with the dynamic limitations of the aircraft — three main control parameters were defined to serve as decision variables for the algorithm. These parameters provide the predictive controller with enough freedom to explore diverse solutions, allowing the aircraft to modify its trajectory effectively. The selected parameters are:

- Longitudinal acceleration (a): Allows the algorithm to directly adjust the aircraft's airspeed over time. This capability is crucial for increasing or decreasing longitudinal separation from other aircraft and can also help avoid conflicts by introducing timing differences when passing through potential intersection points.
- Roll rate ($\dot{\phi}$): Corresponds to the angular velocity at which the aircraft rolls about its longitudinal axis, being the primary control parameter responsible for initiating coordinated turns. By adjusting $\dot{\phi}$, the algorithm can change the aircraft's heading angle (χ), allowing lateral deviations from the original route in a smooth and controlled manner.
- Flight path angle rate ($\dot{\gamma}$): Represents the aircraft's ability to modify its climb or descent profile. By controlling $\dot{\gamma}$, the algorithm manages the vertical component of the trajectory, which is essential for avoiding altitude-related conflicts.

The implementation of the predictive control algorithm relies on a discrete prediction horizon composed of P steps. For each of these steps, the algorithm computes a set of three control variables — acceleration a_k, roll rate $\dot{\phi}_k$, and flight path angle rate $\dot{\gamma}_k$, with $k = 1, 2, ..., P$. These values are grouped into a single control vector with dimension $3 \times P$, which serves as the decision vector for the optimisation problem. The successive application of these controls allows the simulation of the aircraft's trajectory evolution over the considered horizon, and the optimal solution corresponds to the control sequence that minimises the cost function while simultaneously respecting all imposed constraints. *Equation* (3.18) shows the mathematical representation of the control vector.

$$\mathbf{u} = \begin{bmatrix} a_1 & \dot{\phi}_1 & \dot{\gamma}_1 & a_2 & \dot{\phi}_2 & \dot{\gamma}_2 & \cdots & a_P & \dot{\phi}_P & \dot{\gamma}_P \end{bmatrix} \tag{3.18}$$

3.6.2 Bounds

Therefore, the bounds represent the specific physical limitations defined for the evasive aircraft, ensuring that any manoeuvre generated by the MPC respects the constraints imposed by that particular vehicle. These limits guarantee that the control parameters do not exceed values that could compromise the aircraft's stability, structural integrity, or responsiveness.

Moreover, by incorporating the bounds directly into the optimisation process, it becomes unnecessary to perform post-processing checks to validate the feasibility of the generated trajectories. This approach also allows for the easy integration of aircraft-specific characteristics, making the algorithm flexible and adaptable to heterogeneous UAV fleets.

It is also worth noting that these constraints act in a preventive manner within the algorithm, reducing the search space for the optimal solution. This can have a positive impact on computational efficiency by eliminating a significant number of infeasible candidates during the optimiser's iteration process. *Equation* (3.19) presents the imposed limits in mathematical form.

$$\begin{bmatrix} a_{\min} \\ \dot{\phi}_{\min} \\ \dot{\gamma}_{\min} \end{bmatrix} \leq \begin{bmatrix} a_k \\ \dot{\phi}_k \\ \dot{\gamma}_k \end{bmatrix} \leq \begin{bmatrix} a_{\max} \\ \dot{\phi}_{\max} \\ \dot{\gamma}_{\max} \end{bmatrix} \tag{3.19}$$

3.6.3 Constraints

In addition to the limitations imposed directly on the control variables through the bounds, and with the aim of enhancing the optimisation process, a set of constraints was also considered, applied to the dynamic behaviour of the aircraft throughout the prediction hori-

zon. These constraints ensure that the predicted trajectory remains within the acceptable physical and operational limits, respecting not only the characteristics of the aircraft but also the requirements of the surrounding airspace.

The constraints are formulated as inequalities and are evaluated point by point, based on the simulation of the aircraft's dynamics under the sequence of candidate control inputs. This approach allows the effects of the system's behaviour to be integrated directly into the optimisation, ensuring that the resulting solutions are both physically feasible and operationally valid.

In the specific context of this work, the following dynamic constraints were implemented:

- Minimum and maximum altitude limits;
- A limit on the absolute value of γ;
- Minimum and maximum speed limits.

From the algorithm's perspective, since these constraints are evaluated before the cost function is optimised, it is necessary to simulate the aircraft's dynamics iteratively for each constraint along the prediction horizon, focusing exclusively on the parameters relevant to the variable being constrained.

This ensures that each constraint is applied based on values consistent with the system's realistic evolution, allowing the algorithm to verify, at each prediction step, whether the predefined limits are being respected.

Equation (3.20) presents the set of constraints imposed on the algorithm in mathematical form.

$$\begin{cases} h_{\text{min}} \leq h_k \leq h_{\text{max}} \\ |\gamma_k| \leq \gamma_{\text{limit}} \\ v_k \geq v_{\text{min}} \\ v_k \leq v_{\text{max}} \end{cases} \tag{3.20}$$

3.6.4 Optimization and Iterations

A Model Predictive Control (MPC) algorithm, due to its inherently iterative nature, requires the successive solution of an optimization problem, where a cost function is minimized subject to multiple constraints. As such, the choice of the optimizer is a central element in the implementation of this approach and must be guided by the specific characteristics of the problem, namely the presence of bounds, constraints, and computational performance requirements.

In Subsection 2.3.2.1, two optimization algorithms were presented, both available in the widely used `scipy.optimize` library for Python. After a comparative analysis—considering robustness, execution time, and the ability to handle nonlinear constraints—the chosen method was *Sequential Least Squares Programming* (SLSQP), whose operating logic is outlined in Algorithm 1. This algorithm allows for the explicit definition of variable bounds as well as equality and inequality constraints, which are essential features in the present problem, where it is necessary to ensure minimum separation between aircraft, respect physical control limits, and preserve trajectory smoothness.

The effectiveness of the algorithm depends not only on its logical structure and constraint handling but also on the number of iterations allowed per optimizer call. This number is particularly important, as it directly influences the algorithm's ability to reach a solution sufficiently close to the optimum. Based on the observed system behavior, a differentiated approach was adopted between the first optimization step and subsequent ones.

In the first call to the optimizer, the control parameters are randomly generated within the predefined bounds, and up to 100 iterations are allowed. This higher limit ensures adequate convergence from a non-optimized starting point. From that point onward, all subsequent optimizations use the previously optimized control vector as the initial guess, with the maximum number of iterations reduced to 50. This strategy provided a good balance between computational performance and solution quality, maintaining convergence while significantly reducing computational effort in the following iterations.

3.6.5 Application of Control Inputs

One of the fundamental principles of Model Predictive Control (MPC) is to apply only the first set of control inputs resulting from the optimization, then reassess the system at each time step and solve the control problem again based on the updated state. This approach ensures strong adaptability to disturbances or changes in the system over time.

However, given the nature of the present problem, and to ensure trajectory continuity and reduce abrupt fluctuations in the commands, it was decided to apply more than one control per iteration. Specifically, if the prediction horizon p is less than or equal to 10, all control vectors obtained from the optimization are applied. If the horizon is greater than 10, only the first 10 control vectors are applied, and the optimization is resumed from that point onward.

The maximum value of 10 steps was defined based on a heuristic process, where different configurations were tested, and a balance was sought between trajectory smoothness, solution robustness, and computational efficiency. This choice proved adequate for maintaining solution stability and avoiding oscillatory behaviour.

The application of the control inputs also determines the timing of the next MPC call, in case a conflict still exists. Thus, the optimization does not necessarily occur at the very

next time step. If $p \leq 10$, the next MPC iteration takes place $p \cdot \Delta t$ seconds later. If $p > 10$, the optimizer is called again after $10 \cdot \Delta t$ seconds. This strategy reduces the total number of optimization calls throughout the simulation while preserving the algorithm's ability to respond to the system's dynamics.

3.6.6 Cost Function

The cost function represents the logical core of the entire model predictive control (MPC) algorithm, acting as the "brain" of the decision-making process. It is through this function that the optimizer evaluates the quality of each possible control set and ultimately determines the most appropriate commands to apply.

In the context of the present work, the cost function was designed with the objective of ensuring safe and efficient avoidance of the predefined trajectories, while respecting the minimum separation criteria between aircraft and the established safety parameters.

3.6.6.1 Dynamic Simulation

Since the control vector is one of the main inputs to the cost function, it is essential to understand how this set of commands influences the predicted future positions along the prediction horizon. However, the simulation is not carried out directly from the controls, but rather from the state variables, which are updated iteratively based on those controls.

At each step of the prediction horizon, the control vector provides the variations in speed, heading (χ), and climb/descent angle (γ). From the control vector, it is then possible to obtain the speed (*Equation* (3.21)), χ (*Equation* (3.22)), and γ (*Equation* (3.23)).

$$v_{k+1} = v_k + a_k \cdot \Delta t \tag{3.21}$$

$$\chi_{k+1} = \chi_k + b_k \cdot \Delta t \tag{3.22}$$

$$\gamma_{k+1} = \gamma_k + c_k \cdot \Delta t \tag{3.23}$$

With these updated variables, the dynamic simulation of the aircraft's motion is performed, as described in Section 2.2. As a result, vectors containing the predicted coordinates x, y, and z for each instant of the prediction horizon are obtained. These coordinates are then used in the cost evaluation and in the detection of potential conflicts with other aircraft.

3.6.6.2 Smoothing of Control Angles

In order to promote smooth transitions in the aircraft's heading during the avoidance manoeuvre, a cost term was introduced to penalise successive variations of $\dot{\chi}$. The penalisation is expressed as the sum of the squared differences between consecutive values of $\dot{\chi}$, as shown in *Equation* (3.24). The term S_χ acts as a normalisation factor, ensuring that the magnitude of the variations is scaled consistently with respect to the other cost components.

$$\text{cost}_{\dot{\chi}} = \sum_{k=0}^{p} \text{weight}_{\dot{\chi}} \cdot \left(\frac{\dot{\chi}_{k+1} - \dot{\chi}_k}{S_\chi} \right)^2 \tag{3.24}$$

Similarly, to ensure smooth variations in the climb/descent angle during the avoidance manoeuvre, a cost term was added to penalise abrupt changes in γ. The penalisation is computed as the sum of squared differences between consecutive values of γ, as expressed in *Equation* (3.25). The term S_γ serves as a normalisation factor, scaling the variations consistently with the other cost components.

$$\text{cost}_{\dot{\gamma}} = \sum_{k=0}^{p} \text{weight}_{\dot{\gamma}} \cdot \left(\frac{\gamma_k - \gamma_{k-1}}{S_\gamma} \right)^2 \tag{3.25}$$

In addition to penalising increments of the commanded climb/descent angle, a second smoothness term was introduced to discourage oscillatory behaviour in the vertical trajectory. This term is based on the discrete second derivative of γ, ensuring that its evolution along the prediction horizon remains as smooth as possible. The penalisation is formulated as shown in *Equation* (3.26), where S_γ is the normalisation factor.

$$\text{cost}_{\Delta^2\gamma} = \text{weight}_{\Delta^2\gamma} \cdot \sum_{k=2}^{p} \left(\frac{\gamma_k - 2\gamma_{k-1} + \gamma_{k-2}}{S_\gamma} \right)^2 \tag{3.26}$$

3.6.6.3 TCAS Parameters

In order to ensure that the evasive trajectory maintains safe distances from the surrounding aircraft, a specific cost term was introduced, associated with the violation of the separation limits defined by the TCAS (Table 1.1). This term evaluates, at each prediction step, the minimum distance between the evading aircraft and all other aircraft in the simulation, both in the horizontal plane (CPA_h) and in the vertical component (Δh).

To quantify the degree of violation of the TCAS separation thresholds, normalized penalty factors were defined for the horizontal and vertical components. These factors are calculated according to *Equations* (3.27) and (3.28), respectively, with the maximum value across all conflicting aircraft being considered.

$$P_{CPA_h}(k,i) = \left(\frac{\text{CPA}_{h_{\text{threshold}}} - \text{CPA}_{h,k,i}}{\text{CPA}_{h_{\text{threshold}}}} \right)^2 \tag{3.27}$$

$$P_{\Delta h}(k,i) = \left(\frac{\Delta h_{\text{threshold}} - \Delta h_{k,i}}{\Delta h_{\text{threshold}}} \right)^2 \tag{3.28}$$

The quadratic formulation amplifies the penalization as a function of the severity of the violation, assigning higher costs to separations substantially below the defined thresholds. Conversely, when the predicted separation is greater than the regulatory limit, the penalty value is zero, reflecting the absence of effective risk.

With this formulation, only distances smaller than the respective TCAS thresholds contribute to the cost. The quadratic structure further ensures that more severe violations are penalized disproportionately, reinforcing the priority of maintaining minimum separation standards.

To avoid singularities when the predicted distances approach zero, lower bounds were imposed on CPA_h and Δh, ensuring that these variables never assume values smaller than 10^{-3}. This safeguard prevents excessively high penalties that could compromise the numerical stability of the optimizer.

The final cost is obtained by accumulating the weighted penalties over the entire prediction horizon and across all aircraft, as expressed in *Equations* (3.29) and (3.30).

$$\text{cost}_{\text{CPA}_h} = \sum_{k=0}^{p} \sum_{i \neq \text{evader}} \text{weight}_{\text{CPA}_h} \cdot P_{CPA_h} \cdot \log \left(1 + \left(\frac{\text{CPA}_{h_{\text{threshold}}}}{\text{CPA}_{h,k,i}} \right)^4 \right) \tag{3.29}$$

$$\text{cost}_{\Delta h} = \sum_{k=0}^{p} \sum_{i \neq \text{evader}} \text{weight}_{\Delta h} \cdot P_{\Delta h} \cdot \log \left(1 + \left(\frac{\Delta h_{\text{threshold}}}{\Delta h_{k,i}} \right)^4 \right) \tag{3.30}$$

3.6.6.4 Adaptive Heading Cost with Conflict-Based Gating

In order to ensure that the trajectory defined to resolve the conflict does not deviate excessively from the originally planned route, a cost term associated with the aircraft heading (χ) was introduced. This term ensures that the deviation from the reference trajectory is taken into account during the optimization process. To make this penalization more efficient and adaptive, an activation function (*gate*) was implemented to regulate its

influence according to the conflict level.

The activation function was designed based on three distinct indicators — time to collision (τ), horizontal separation (CPA_h) and vertical separation (Δh) — each representing an essential dimension of conflict risk. The mathematical formulation of each indicator follows operational safety principles, ensuring consistency with criteria commonly applied in aviation.

The time to collision (τ) constitutes a fundamental metric, as it reflects the temporal window available to perform a manoeuvre before an actual conflict occurs. Its modelling relies on a logistic function, presented in *Equation* (3.31):

$$g_\tau(s) = \frac{1}{1 + \exp\big(-(\tau_{\min}(s) - \tau_{\lim})\big)} \tag{3.31}$$

The logistic function is particularly suitable in this context, as it provides a smooth transition between scenarios considered safe ($\tau \gg \tau_{\lim}$) and critical scenarios ($\tau \leq \tau_{\lim}$), avoiding abrupt discontinuities in the cost activation. The inflection point around $\tau_{\lim}$ corresponds to the minimum acceptable time for system reaction, directly translating an operational safety criterion.

Complementarily, the horizontal separation (CPA_h) was modelled through an increasing exponential function, expressed in *Equation* (3.32):

$$g_h(s) = 1 - \exp\left(-\frac{CPA_{h,\min}(s)}{S_H}\right) \tag{3.32}$$

This formulation ensures that for small separations the function value approaches zero, reflecting a critical condition. As the horizontal distance increases, the function grows rapidly, approaching 1 and saturating for values much higher than the regulatory minimum. This behaviour reflects operational reality: an aircraft at a distance substantially greater than the minimum required separation is considered equally safe, regardless of further increases. The parameter S_H acts as a calibration factor, allowing the sensitivity of the function to horizontal distance scales to be adjusted.

Similarly, the vertical separation (Δh) is represented by the function in *Equation* (3.33):

$$g_v(s) = 1 - \exp\left(-\frac{\Delta_{h,\min}(s)}{S_V}\right) \tag{3.33}$$

The use of the same functional structure reflects the conceptual symmetry between hori-

zontal and vertical risk, even though vertical separation standards are typically more restrictive. In this case as well, small separations correspond to values close to zero (critical conditions), while separations significantly above the operational limit yield values close to 1 (safe conditions). The parameter S_V provides flexibility in calibration, according to the acceptable regulatory vertical separation thresholds.

It is then necessary to combine the three indicators into a single activation function, capable of regulating the degree of freedom given to the aircraft with respect to following the reference heading. In general terms, the combination adopts a weighted average, whose purpose is to produce a continuous and interpretable activation level between 0 and 1.

Equation (3.34) defines the activation function $g(s)$ at each prediction step s as a weighted average of the temporal and spatial components:

$$g(s) \;=\; \Big(\alpha_\tau \, g_\tau(s) \;+\; \alpha_H \, g_h(s) \;+\; \alpha_V \, g_v(s) \,,\, 0,\, 1 \Big) \tag{3.34}$$

Where the weights satisfy the normalization condition

$$\alpha_\tau + \alpha_H + \alpha_V \;=\; 1, \tag{3.35}$$

In this implementation, the following values were considered:

$$\alpha_\tau = 0.25, \qquad \alpha_H = 0.375, \qquad \alpha_V = 0.375. \tag{3.36}$$

This choice favours spatial metrics (horizontal and vertical), which represent the immediate risk of loss of separation, while maintaining sensitivity to the temporal predictor g_τ.

Values $g(s) \approx 0$ correspond to critical situations (reduced separations and/or low τ), in which the heading cost is attenuated to allow greater freedom for evasive manoeuvres. Conversely, $g(s) \approx 1$ identifies safe conditions, where the penalization of heading deviation should prevail, promoting stability and convergence to the planned route.

With the inclusion of the activation function $g(s)$, it is then possible to apply the mechanism that quantifies the deviation of the aircraft relative to the reference heading χ_{ref}.

The reference heading, χ_{ref}, corresponds to the direction the aircraft would follow in the absence of any avoidance manoeuvre, being obtained from the predefined trajectory. The difference between the predicted heading and the reference heading is computed in a way that respects the cyclic nature of angular variables, using the `arctan2` function. In this

way, the angular error $e_\chi(s)$, represented in *Equation* (3.37), always remains within the interval $[-\pi, \pi]$, avoiding artificial discontinuities in its evaluation:

$$e_\chi(s) = \text{atan2}\Big(\sin\left(\chi(s) - \chi_{\text{ref}}\right),\ \cos\left(\chi(s) - \chi_{\text{ref}}\right)\Big)\,. \tag{3.37}$$

The heading cost is then formulated as a quadratic penalization of this error, simultaneously weighted by the activation level $g(s)$ and by a temporal discount factor, as presented in *Equation* (3.38). Thus, in critical situations, when $g(s) \approx 0$, the heading penalization is practically suppressed, allowing greater manoeuvring freedom for conflict resolution. Conversely, when $g(s) \approx 1$, the penalization prevails, promoting the convergence of the aircraft to the reference heading and encouraging a return to the nominal trajectory as soon as safety conditions allow.

$$\text{cost}_\chi = \sum_{k=0}^{p} \text{weight}_\chi \cdot g(k) \cdot \Big[\text{atan2}\left(\sin(\chi_k - \chi_{\text{ref}}), \cos(\chi_k - \chi_{\text{ref}})\right)\Big]^2 \tag{3.38}$$

3.6.6.5 Sign Inversion of the Climb/Descent Angle

To promote more coherent and realistic vertical manoeuvres, a cost term was introduced to penalise sign inversions in the climb/descent angle γ throughout the prediction horizon. These inversions, which correspond to abrupt transitions between climbing and descending (or vice versa), are generally undesirable from both operational and safety perspectives, particularly when they occur repeatedly or suddenly.

Sign inversion detection is performed at each prediction step. In the first step ($s = 0$), the current value of γ is compared with the value stored from the previous MPC iteration, i.e., the last γ effectively applied to the aircraft. If no avoidance manoeuvre has yet been initiated, this reference corresponds to the pre-evasion state. For subsequent steps ($s > 0$), the sign of γ_s is compared with that of γ_{s-1}.

Whenever a sign inversion is detected, a fixed penalty is applied to the total cost, weighted by a specific factor. This mechanism aims to discourage unnecessary oscillations and to encourage the maintenance of a consistent flight attitude, promoting smoother and more stable vertical trajectories. The mathematical formulation of this penalisation is given in *Equation* (3.39).

$$\text{cost}_{\text{inversion}} = \sum_{k=0}^{p} \text{weight}_{\text{inversion}} \cdot \text{indicator}_{\text{change}_k} \tag{3.39}$$

Where $\text{indicator}_{\text{change}_k}$ takes the value 1 whenever a sign inversion in γ is detected at step k, and 0 otherwise.

3.6.6.6 Continuity of Vertical State

To ensure a consistent transition between the previously applied climb/descent angle and the predicted trajectory, a continuity cost term was introduced. This penalization enforces a smooth linkage with past maneuvers, avoiding unrealistic discontinuities at the start of the prediction horizon. The formulation is given in *Equation* (3.40).

$$Cost_{\gamma}^{\text{state}} \;=\; \text{weight}_{\gamma}^{\text{state}} \cdot \sum_{k=0}^{p} \left(\frac{\gamma_k - \gamma_{k-1}}{S_{\gamma}} \right)^2, \tag{3.40}$$

Where the reference γ_{-1} corresponds to the last value applied to the aircraft prior to the avoidance manoeuvre, or to the initial condition if no control has yet been executed.

3.6.6.7 Weights

Weights play a central role in the functioning of the cost function, as they allow the relative relevance of each penalised term in the optimisation process to be adjusted. Each component of the cost function reflects a distinct operational priority — such as control smoothness, adherence to the nominal trajectory, or the assurance of safe separation between aircraft. Careful tuning of the weights, therefore, directly conditions the behaviour of the predictive controller, adapting the system's response to the specific objectives of the mission.

The definition of the weights was based on a heuristic approach, being adjusted iteratively until a satisfactory balance was reached between safety, stability, and the performance of the evasive manoeuvre. It is important to highlight that the magnitude of a weight does not necessarily reflect the importance of the associated parameter, as the impact of each term on the cost function also depends on the order of magnitude of the variables involved.

A careful selection of weights is therefore essential to ensure that the optimiser's behaviour adequately reflects the trade-offs between safety, efficiency, and fidelity to the planned flight path.

Parameter	Weight
Smoothing of Controls - χ	30
Smoothing of Controls - γ	30
Smoothing of Controls (second order) - γ	100
TCAS Parameter - CPA_h	500
TCAS Parameter - Δh	150
Adaptive Heading Cost with Conflict-Based Gating	5
Continuity of Vertical State	30
Sign Inversion of the Climb/Descent Angle	3000

Table 3.3: Weights associated with the cost function terms considered in the MPC-based conflict resolution algorithm.

3.6.7 Post-Processing and Criteria for MPC Re-run

After the optimization of the cost function by the MPC, it is important to assess whether the proposed solution resolves the problem as a whole, or if it is necessary to reapply the MPC to guarantee the separation values presented in Table 1.1. Since the TCAS parameters are necessarily computed during the optimization process, the results from the last iteration are extracted and stored for this analysis.

The resolution of a conflict is considered valid when at least one of the scenarios presented in *Equation* (3.41) is satisfied.

$$\text{Conflict Resolved} = \begin{cases} \tau < 0 \ \wedge \ CPA_h > CPA_{h_{threshold}} \\ \tau < 0 \ \wedge \ \Delta_h > \Delta_{h_{threshold}} \\ CPA_h > CPA_{h_{threshold}} \ \wedge \ \Delta_h > \Delta_{h_{threshold}} \end{cases} \tag{3.41}$$

Among the three scenarios presented, the third one is conceptually the most desirable, as it ensures that, at the evaluation instant, both the horizontal separation (CPA_h) and the vertical separation (Δ_h) exceed their respective limit values. This simultaneous condition represents the situation with the greatest safety margin, ensuring that there is no imminent risk of conflict in either of the two dimensions considered by the TCAS.

Nevertheless, the first two scenarios are also regarded as valid solutions, provided they are accompanied by a negative τ value. This temporal parameter, when assuming a value below zero, indicates that the point of closest approach between the aircraft has already occurred, meaning that they have already crossed paths in space. In these cases, even if only one of the metrics — horizontal separation (CPA_h) or vertical separation (Δ_h) — is above its respective limit, the conflict resolution is operationally acceptable.

At the moment of crossing between the aircraft, the presence of a separation — whether horizontal or vertical — greater than the minimum prescribed value is, by itself, a sufficient condition to eliminate the risk of collision, thereby ensuring full compliance with the

requirements and purpose of the prevention system.

3.6.8 Returning to the Planned Trajectory

With the evasive maneuver completed, and given that the mission is executed fully autonomously, it becomes imperative to gradually guide the evading aircraft back to the trajectory initially defined in the flight plan. This process aims to ensure mission continuity, avoiding prolonged deviations and ensuring that the operation remains within the previously established spatial and temporal limits.

The return to the planned trajectory is based on reorienting the aircraft toward the waypoint that allows the most efficient resumption of the flight plan. This re-entry point does not necessarily have to be the one immediately following the last reached before the evasive maneuver, and may instead be chosen to ensure a smooth and effective transition, according to two main criteria.

The first criterion consists of minimizing the three-dimensional distance between the aircraft and the waypoint, avoiding situations in which, by being too close to a specific point, the aircraft would need to perform maneuvers that could temporarily reduce its effective forward progression along the route.

This distance is calculated from the three-dimensional vector defined between the current position of the aircraft and the position of the waypoint. The norm of this vector provides the actual spatial distance, allowing the identification of the nearest waypoint and, consequently, reducing the deviation required to return to the originally planned trajectory.

Let $\vec{p}_{aicraft} = (x_a, y_a, z_a)$ be the current position of the aircraft and $\vec{p}_{wp} = (x_w, y_w, z_w)$ the position of the waypoint. The vector between the aircraft and the waypoint is obtained according to *Equation* (3.42).

$$\vec{v}_{aw} = (x_w - x_a,\ y_w - y_a,\ z_w - z_a) \tag{3.42}$$

The corresponding three-dimensional distance is calculated using *Equation* (3.43).

$$d_{aw} = \|\vec{v}_{aw}\| = \sqrt{(x_w - x_a)^2 + (y_w - y_a)^2 + (z_w - z_a)^2} \tag{3.43}$$

The second criterion considers the approach angle between the current flight direction and the direction to the waypoint. The aircraft's displacement vector, obtained between the current and the previous time step, is defined in *Equation* (3.44).

$$\vec{v}_{displacement} = \left(x_a^{(t)} - x_a^{(t-1)},\ y_a^{(t)} - y_a^{(t-1)},\ z_a^{(t)} - z_a^{(t-1)} \right) \tag{3.44}$$

The vector between the aircraft and the waypoint has already been defined in *Equation* (3.42). The angle θ between the two vectors is obtained according to *Equation* (3.45).

$$\theta = \arccos\left(\frac{\vec{v}_{displacement} \cdot \vec{v}_{aw}}{\|\vec{v}_{displacement}\| \, \|\vec{v}_{aw}\|} \right) \tag{3.45}$$

Using a heuristic approach, the distance calculated in *Equation* (3.43) was set to 200 m. Similarly, a reference value of 60 degrees was established for the angle between the vectors, obtained from *Equation* (3.45), serving as an additional selection constraint.

In selecting the re-entry waypoint, all points that have already been passed, whether during normal navigation or throughout the evasive trajectory, are disregarded. The selection process is then applied to all remaining waypoints that can still be reached within the planned route.

The choice of re-entry waypoint aims to simultaneously minimize the three-dimensional distance between the aircraft and the candidate point, and the approach angle relative to its current flight direction, with the selected waypoint being the one that offers the most favorable combination of shortest distance and smallest angle.

Chapter 4

Simulation Methodology and Results

This chapter presents the methodology adopted for the numerical simulations and discusses the results obtained. The main objective is to evaluate the performance of the collision avoidance algorithm in controlled yet representative scenarios. For this purpose, two encounter geometries were selected as reference cases: the head-on scenario, in which two aircraft approach each other on opposite trajectories, and the orthogonal scenario, where the conflict results from perpendicular paths. These scenarios were chosen as they are considered the most critical in terms of encounter geometry and, consequently, the most challenging for collision avoidance systems.

In both cases, the trajectories of the two conflicting aircraft were defined by the user, ensuring the systematic presence of a collision situation. In addition, three extra aircraft were included, with initial positions generated at random outside the conflict zone, ensuring situational awareness without introducing new conflicts. This configuration allowed assessing the algorithm's ability to detect, signal, and resolve conflicts between the reference trajectories while maintaining awareness of the surrounding traffic.

4.1 Constraints Definition

As discussed in Subsection 3.6.3, a set of constraints was imposed to ensure that the solution obtained by the algorithm has both physical and operational validity. In particular, the altitude was limited to the range between 150 m and 5000 m, in order to prevent situations of flight too close to the ground as well as conditions exceeding the maximum operational limits considered. The flight path angle (γ) was also restricted, with its maximum value set at $15°$, both in climb and descent, reflecting the typical performance limitations of this type of aircraft. Finally, the flight speed was constrained to the interval between 15 m/s and 50 m/s, according to the specifications presented in Table 3.2, ensuring that the model always remains within realistic operational margins.

4.2 Trajectory Definition

To assess the robustness of the algorithm in particularly challenging scenarios, two test trajectories were defined, which are considered critical for resolving the problem. Although these configurations have a low probability of occurrence in operational terms, they allow the algorithm to be subjected to limit conditions, ensuring an adequate response in the face of any other scenario.

4.2.1 Head-on Encounter

The first scenario analysed corresponds to a head-on encounter, in which two aircraft follow collinear and opposite trajectories at the same altitude. This type of configuration is considered critical, as it leads to the highest possible closure rate between aircraft. Since they are on directly opposing routes, the individual speeds add up, resulting in a high relative velocity. This factor implies a drastic reduction in the available time for conflict detection and for the execution of an avoidance manoeuvre, placing the algorithm under highly demanding conditions.

The relevance of this case lies precisely in its severity: the temporal margin for response is minimal, and any delay in detection or in the execution of the manoeuvre may result in an inevitable collision. At the same time, this scenario presents a symmetry that allows isolating the algorithm's behavior from external influences such as altitude differences or biased routes. Thus, it becomes a reference case to evaluate the system's ability to react immediately and efficiently in the most unfavourable situations.

Specifically, a route with five waypoints was defined in the vicinity of Castelo Branco, at an altitude of $1000\,\text{m}$, to be covered in $500\,\text{s}$. The waypoint passage times are not uniformly distributed, resulting in four intervals: three of $100\,\text{s}$ and one of $200\,\text{s}$. This choice was made to ensure greater stability in the speed profile, avoiding excessive variations between segments of different lengths. In this configuration, the average distance between consecutive waypoints is approximately $4.43\,\text{km}$, and the overall average speed along the route is about $35.4\,\text{m/s}$.

In the head-on scenario, the aircraft follow the same route in opposite directions, with synchronised departures from the ends. Thus, the average relative closure speed is approximately twice the overall average speed ($70.8\,\text{m/s}$), meaning that, starting from an initial separation equal to the total length of the route ($\approx 17.7\,\text{km}$), the expected encounter occurs roughly halfway through the simulation ($250\,\text{s}$). This combination of high relative velocity with a reduced time window makes the case particularly severe for the algorithm, allowing for a rigorous assessment of its ability to detect and generate stable avoidance manoeuvres in a timely manner.

As discussed in Section 3.2, it is necessary to define the trajectory by specifying a set of

waypoints, indicating for each one its coordinates, flight altitude, and crossing time. In this way, it is possible to include in the simulation the trajectory according to what was stipulated in Section 1.4, ensuring that the analysis takes place over pre-determined routes consistent with the established objectives.

For the determination of distance, the Euclidean distance in three-dimensional space was considered, always calculated with respect to the starting point. Thus, the distance between the initial point and waypoint i is given by *Equation* (4.1)

$$d_i = \sqrt{(x_i - x_0)^2 + (y_i - y_0)^2 + (z_i - z_0)^2} \tag{4.1}$$

The average speed was obtained from the cumulative distance travelled up to each waypoint, divided by the total elapsed time until the corresponding instant, as shown in *Equation* (4.2).

$$v_i = \frac{\sum_{k=1}^{i} d_k}{t_i} \tag{4.2}$$

Tables 4.1 and 4.2 present the trajectory data of aircraft 0 and 1, respectively, for the head-on scenario.

Waypoint	φ (°)	λ (°)	h(m)	t(s)	Distance Cumulative (m)	Average Speed (m/s)
1	39.8389	-7.4133	500	0	––	––
2	39.8665	-7.4139	500	100	3076.9804	30.7698
3	39.9049	-7.4146	500	200	7358.4873	36.7924
4	39.9464	-7.4156	500	300	11975.0769	39.9169
5	39.9979	-7.4163	500	500	17709.3261	35.4187

Table 4.1: Head-on Trajectory Data of Aircraft 0

Waypoint	φ (°)	λ (°)	h(m)	t(s)	Distance Cumulative (m)	Average Speed (m/s)
1	39.9979	-7.4163	500	0	––	––
2	39.9464	-7.4156	500	200	5734.3163	28.6716
3	39.9049	-7.4146	500	300	10350.9135	34.5030
4	39.8665	-7.4139	500	400	14632.4337	36.5811
5	39.8389	-7.4133	500	500	17709.4141	35.4188

Table 4.2: Head-on Trajectory Data of Aircraft 1

Figure 4.1 presents the trajectory defined for the head-on encounter scenario, simultaneously representing the paths of the two aircraft under study. The map is oriented to the north to facilitate the spatial interpretation of the movement. Aircraft 0 starts its flight from the south, heading north, while Aircraft 1 departs from the north, moving towards the south. This configuration clearly highlights the symmetry of the encounter and the imminent collision resulting from the opposing trajectories of the two aircraft.

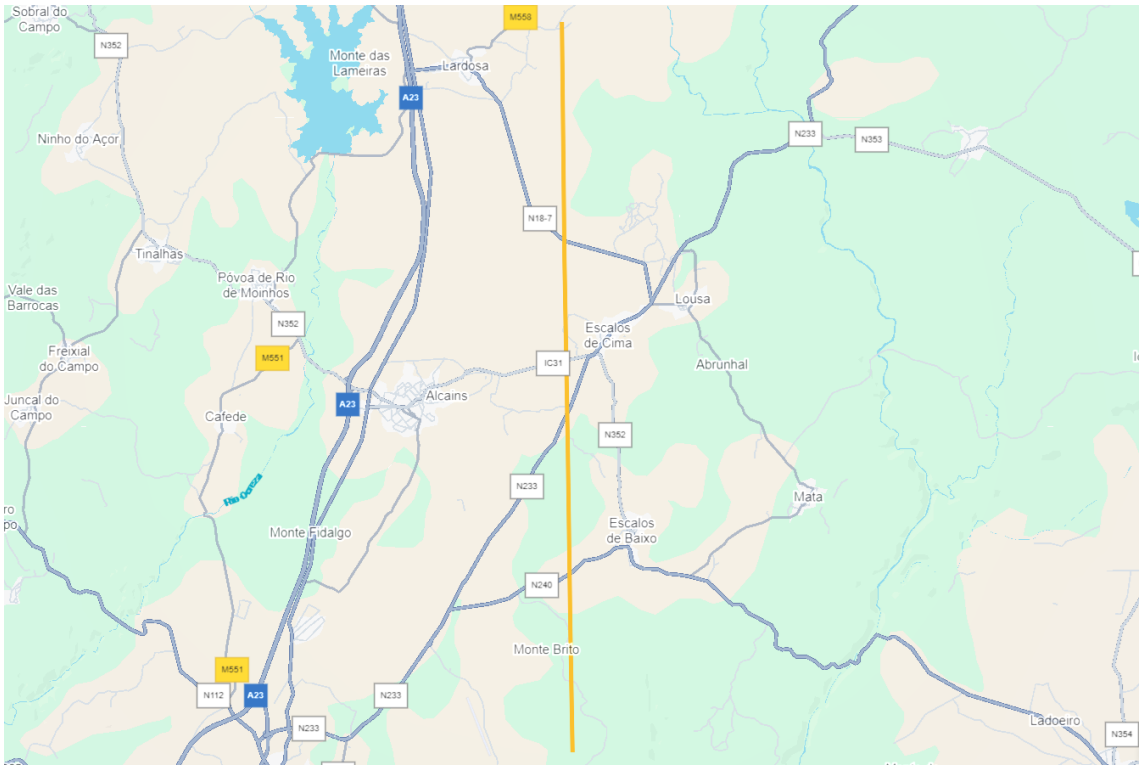

Figure 4.1: Trajectory defined for the head-on encounter scenario.

4.2.2 Orthogonal Encounter

The second scenario analysed corresponds to an orthogonal encounter, in which two aircraft follow perpendicular trajectories at the same altitude, converging towards a common intersection point. This configuration is particularly critical not only due to the high risk of collision, but also because of the complexity associated with defining the most appropriate avoidance manoeuvre.

Unlike the head-on encounter, in which the solution inevitably involves a lateral deviation from the route — relatively straightforwardly ensuring compliance with the minimum safety criteria — the orthogonal encounter presents an intrinsic ambiguity. Conflict resolution can be achieved either through anticipation of the crossing, ensuring that one aircraft reaches the intersection point before the other, or by delaying the passage, guaranteeing that the area is already vacated. This duality makes the problem more demanding, as it requires not only the geometric assessment of separation, but also a much stricter temporal coordination between the two trajectories.

Additionally, this scenario poses an increased difficulty regarding the simultaneous fulfilment of the minimum safety separations. While the vertical component can be ensured through an altitude change, the horizontal component is strongly constrained by the need to limit the deviation from the initially planned route. This restriction makes it more challenging to guarantee that both safety distances are respected simultaneously, reinforcing the complexity of the orthogonal encounter and its relevance as a critical test case.

Specifically, a route consisting of five waypoints was defined, in which one aircraft flies over the runway of Castelo Branco aerodrome, while the other crosses the same aerodrome along a perpendicular trajectory, both maintaining an altitude of $1000\,\text{m}$. Each of the routes has a total duration of $210\,\text{s}$. As in the previous scenario, the passage schedule was defined in a non-uniform way, considering four intervals between waypoints: three of $60\,\text{s}$ and the last one of $30\,\text{s}$. This configuration provided greater stability to the speed profile of both aircraft throughout the simulation.

The average distance between waypoints is approximately $1.85\,\text{km}$, which results in an overall average speed of about $34.6\,\text{m/s}$ for Aircraft 0 and $35.9\,\text{m/s}$ for Aircraft 1. The relative closure speed between two aircraft is obtained from the vector difference of their velocities, as expressed in *Equation* (4.3).

$$V_{\text{rel}} = \|v_0 - v_1\| = \sqrt{V_0^2 + V_1^2 - 2V_0V_1\cos\theta} \tag{4.3}$$

Where θ represents the angle between the trajectories. In the orthogonal case, where $\theta = 90^\circ$, the equation reduces to the form presented in *Equation* (4.4), resulting in a value of approximately $49.9\,\text{m/s}$. This result quantifies the high temporal sensitivity of the encounter: a mismatch of only $1\,\text{s}$ in crossing the intersection point corresponds to a horizontal separation of about $50\,\text{m}$. This demonstrates the increased severity of this scenario, in which the temporal safety margin is extremely reduced and a collision may occur due to small variations in the synchronisation of the trajectories.

$$V_{\text{rel}} = \sqrt{V_0^2 + V_1^2} \tag{4.4}$$

Repeating the process from Subsection 4.2.1, the coordinates, flight altitude and passage time at each waypoint were considered as input to the algorithm, as mentioned in Section 1.4. Similarly, in Tables 4.3 and 4.4, the distance shown — reflecting the total distance travelled up to the corresponding waypoint — and the average speed up to that point were calculated from *Equations* (4.1) and (4.2), respectively.

Waypoint	φ (°)	λ (°)	h(m)	t(s)	Distance Cumulative (m)	Speed (m/s)
1	39.8250	-7.4286	1000	0	––	––
2	39.8406	-7.4378	1000	60	1900.2162	31.6703
3	39.8523	-7.4440	1000	120	1415.7784	27.6279
4	39.8693	-7.4521	1000	180	2013.3283	29.5923
5	39.8857	-7.4602	1000	210	1947.7563	34.6373

Table 4.3: Orthogonal Trajectory Data of Aircraft 0.

Waypoint	φ (°)	λ (°)	h(m)	t(s)	Distance Cumulative (m)	Speed (m/s)
1	39.8446	-7.4885	1000	0	––	––
2	39.8492	-7.4679	1000	60	1828.0820	30.4680
3	39.8542	-7.4442	1000	120	2108.3405	32.8031
4	39.8578	-7.4273	1000	180	1496.5676	30.1830
5	39.8626	-7.4035	1000	210	2099.2801	35.8663

Table 4.4: Orthogonal Trajectory Data of Aircraft 1.

Figure 4.2 presents the trajectory defined for the orthogonal encounter scenario. The map is oriented to the north, with Aircraft 0 (yellow) following its route from southeast to northwest, while Aircraft 1 (red) moves from southwest to northeast. The intersection point is located near the threshold of runway 16 of Castelo Branco aerodrome, representing the critical region of potential collision.

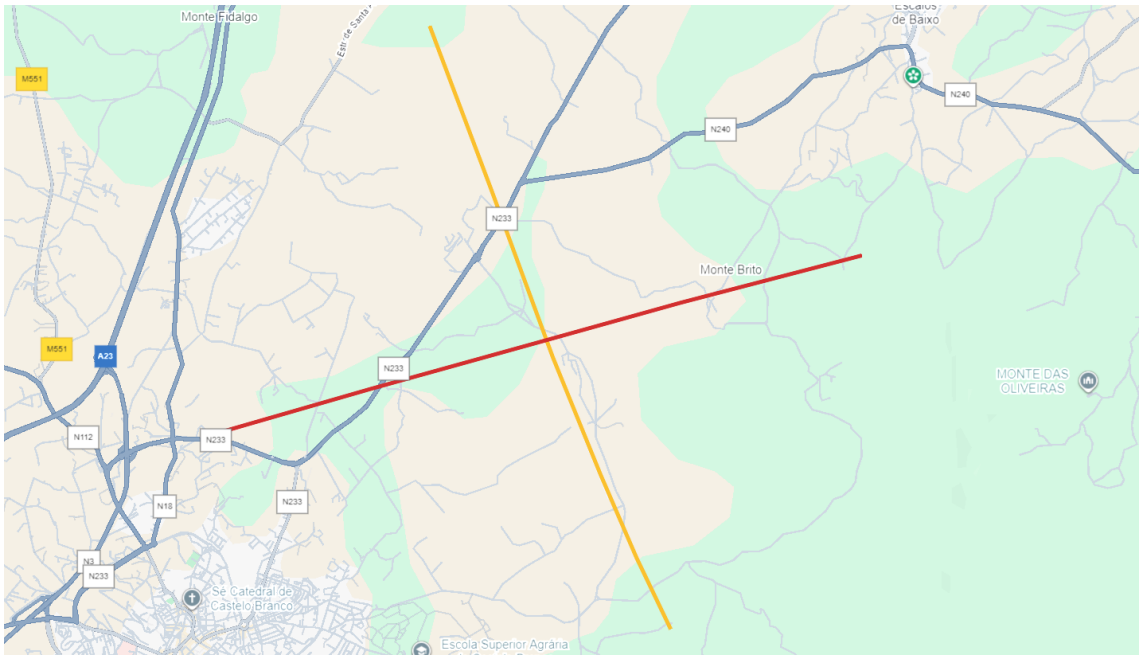

Figure 4.2: Trajectory defined for the orthogonal encounter scenario.

4.3 Results and Sensitivity to the Prediction Horizon

Based on the trajectories previously defined in Section 4.2, the respective development is simulated, integrating the application of the TCAS algorithm. The objective is to ensure the maintenance of safe trajectories, in accordance with the criteria established in the cost function described in Section 3.6.6.

4.3.1 Results

Since the simulation requires the definition of a fixed prediction horizon, this parameter assumes particular relevance in assessing the algorithm's performance. Its size conditions not only the system's ability to anticipate potential conflicts but also the overall stability of the obtained solutions.

The analysis begins with the establishment of a reference value for the prediction horizon, allowing the behavior of the algorithm to be evaluated under controlled conditions. All simulations also consider a constant time step of $\Delta t = 0.1\,\text{s}$, ensuring uniformity in the discretization of the trajectories.

As explored in Subsection 3.6.7, to consider a conflict resolved, it is sufficient to guarantee that at least one of the safety parameters lies within the desired range. Additionally, and given the importance of ensuring that the trajectory obtained by the MPC remains feasible throughout the entire analysis period, it is considered necessary that the value of τ be less than zero, meaning that the aircraft effectively cross, and that at least one of the other two parameters — horizontal separation CPA_h or vertical separation Δh — is above the defined minimum thresholds. In this way, not only is the immediate resolution of the conflict ensured, but the avoidance maneuver is also validated against the established safety criteria.

4.3.1.1 Head-on Case

As described in Subsection 4.2.1, the head-on case represents one of the critical scenarios defined to test the algorithm, assessing its ability to resolve conflicts while considering the operational particularities and the parameters established in the cost function presented in Subsection 3.6.6.

The trajectories defined in Tables 4.1 and 4.2 were adopted, together with the specifications in Table 3.2 and the remaining constraints presented in Section 4.1, where the physical limitations of each aircraft and the respective category are compiled. In this context, category 2 was assigned to aircraft with a wingspan greater than or equal to 10 meters, in order to maintain a generic and comprehensive classification. In association with the operational thresholds defined in Table 1.1, it thus becomes possible to apply the algorithm in a complete and consistent manner.

In the simulations, a reference prediction horizon of $P = 30$ steps was considered. With the defined time step of $\Delta t = 0.1\,\text{s}$, this value corresponds to a total horizon of 3 seconds. This choice results from a compromise between the need to adequately solve potential conflict situations and the limitation of the computational effort associated with the optimization process.

For this scenario, it is important to highlight that each simulation inevitably generates

a case distinct from the previous one, even if common characteristics may be observed. The algorithm assigned a *sensitivity level* equal to 5, which, according to Table 1.1, corresponds to reference values of $CPA_h = 1481.6\,\text{m}$ and $\Delta h = 182.9\,\text{m}$. This selection results from the fact that the maximum altitude recorded among the five aircraft reached $4215.9\,\text{m}$.

TCAS – CONFLICT ALERT

Conflict between aircraft **0** and **1** (t = 220.3s)

CPA_h = 0.5 m || Δh = 0.0 m || $\tau = 29.9\text{s}$ || $\chi = 359.03°$ || $\gamma = 0.00°$

Figure 4.3 shows the evolution of the closest point of approach in the horizontal plane (CPA_h), from the moment a potential conflict was detected until the moment it was considered resolved.

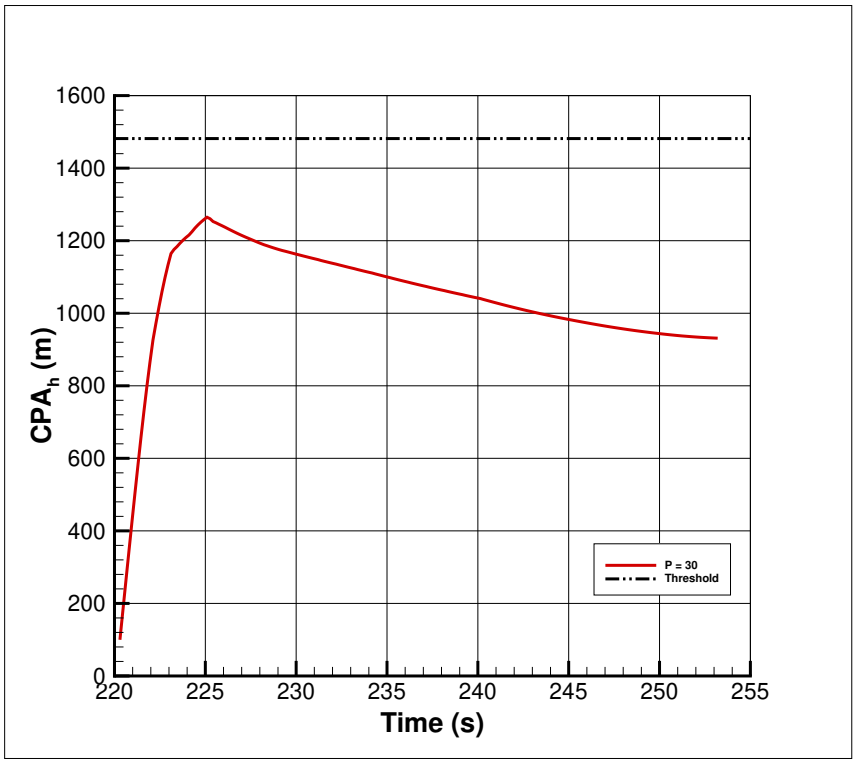

Figure 4.3: CPA_h resulting from applied controls across MPC iterations — Head-on scenario.

Figure 4.4 shows the temporal evolution of the vertical separation between the aircraft, reflecting the behavior of this parameter from the instant a potential conflict was detected until it was considered resolved.

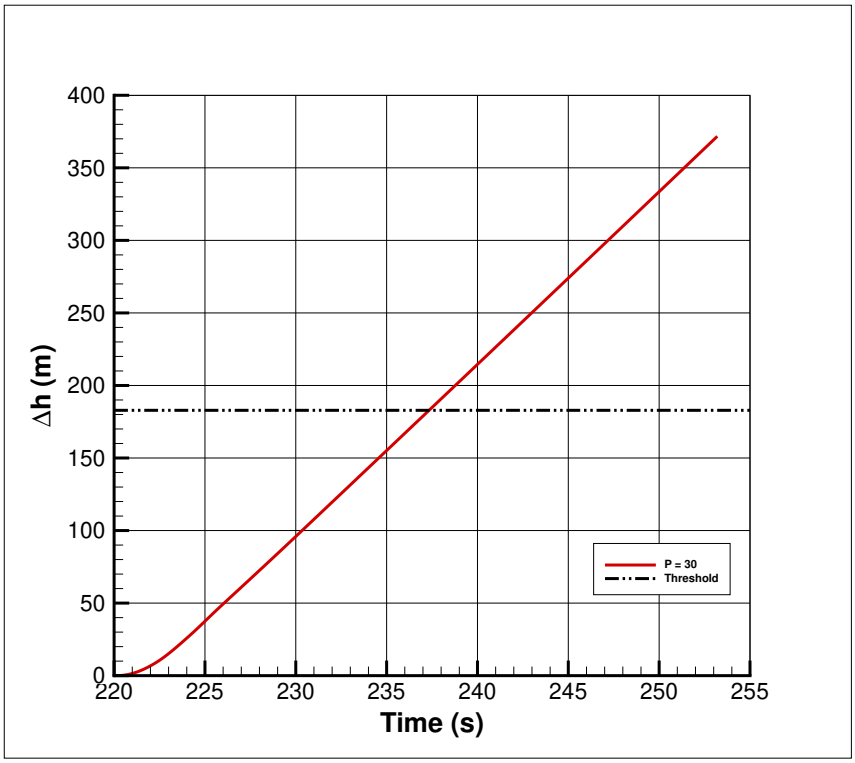

Figure 4.4: Vertical separation resulting from applied controls across MPC iterations — Head-on scenario.

Finally, Figure 4.5 shows the temporal evolution of the τ parameter, reflecting its behavior from the moment a potential conflict was detected until it was considered resolved.

The combined analysis of the considered parameters provides a comprehensive understanding of the avoidance strategy adopted by the algorithm. It becomes clear that the maneuver was characterized by a dynamic balance between horizontal and vertical separation components, each contributing at different moments of the resolution process. In the early stages, a sharp increase in horizontal distance is observed, directly reflecting the MPC's immediate action to reduce the short-term risk of collision. This behavior highlights the responsiveness of the controller, which initially prioritizes the most accessible degree of freedom to create an immediate buffer in the conflict geometry.

As the simulation progresses, however, the algorithm gradually shifts its focus toward reinforcing the vertical component of separation. This transition is strongly influenced by the cost function design, where penalties are imposed on excessive deviations from the predefined trajectory. Maintaining a predominantly horizontal avoidance would have required a larger lateral displacement, resulting in a significant deviation from the intended flight plan. By contrast, reinforcing the vertical separation allowed the algorithm to secure conflict resolution while simultaneously minimizing the deviation cost. As a result, the final configuration ensured a vertical clearance of 371.6 m, comfortably above the required safety limit, thereby confirming the robustness of the vertical maneuver.

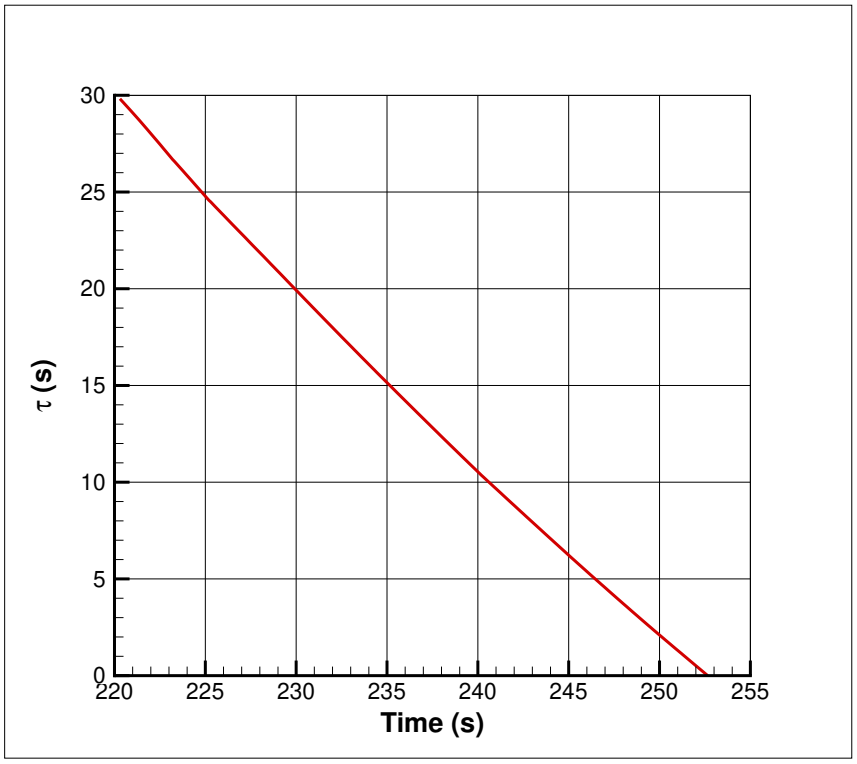

Figure 4.5: τ resulting from applied controls across MPC iterations — Head-on scenario.

This vertical dominance proved decisive at the critical instant when $\tau = 0$. At that moment, the vertical separation guaranteed compliance with the minimum threshold of 182.9 m established for sensitivity level 5. Although the horizontal criterion of 1481.6 m was not fully achieved, the algorithm maintained a horizontal CPA of 931 m. When combined with the vertical clearance, this configuration still validated the maneuver as safe, since the multi-dimensional separation ensured that no violation of safety margins occurred. These results highlight the algorithm's ability to adaptively manage the available degrees of freedom, exploiting spatial redundancy to secure conflict resolution within the prescribed thresholds. The capacity to articulate horizontal and vertical contributions according to cost function priorities reflects not only the flexibility of the optimization process but also its alignment with practical operational constraints.

The behavior of the parameter τ, shown in Figure 4.5, further supports this interpretation. Its evolution displays an almost linear decrease towards zero, reflecting the progressive shortening of the estimated time to collision. The occasional variations and local increases observed in τ correspond to trajectory adjustments introduced by the avoidance maneuver, which momentarily postponed the predicted instant of closest approach. This dynamic illustrates the iterative nature of the MPC: at each step, only the first ten control actions from the prediction horizon are applied, while the remaining twenty steps serve to ensure robustness by projecting the long-term consistency of the trajectory. Consequently, the continuous decrease of τ is not abrupt, but rather gradual and conservative, mirroring the adaptive decision-making process of the algorithm.

In Figure 4.6, the trajectory obtained after conflict resolution in the head-on scenario is presented.

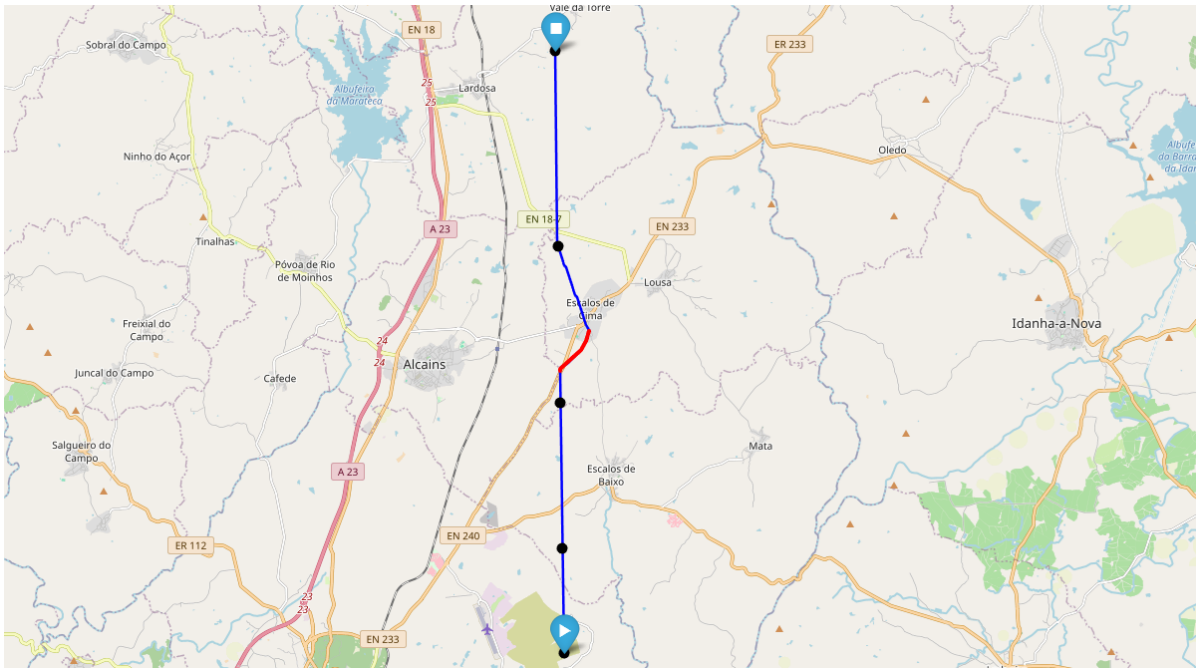

Figure 4.6: Resulting aircraft trajectory for the head-on scenario.

Figure 4.7 shows the evolution of the heading angle (χ) throughout the optimization process performed by the algorithm.

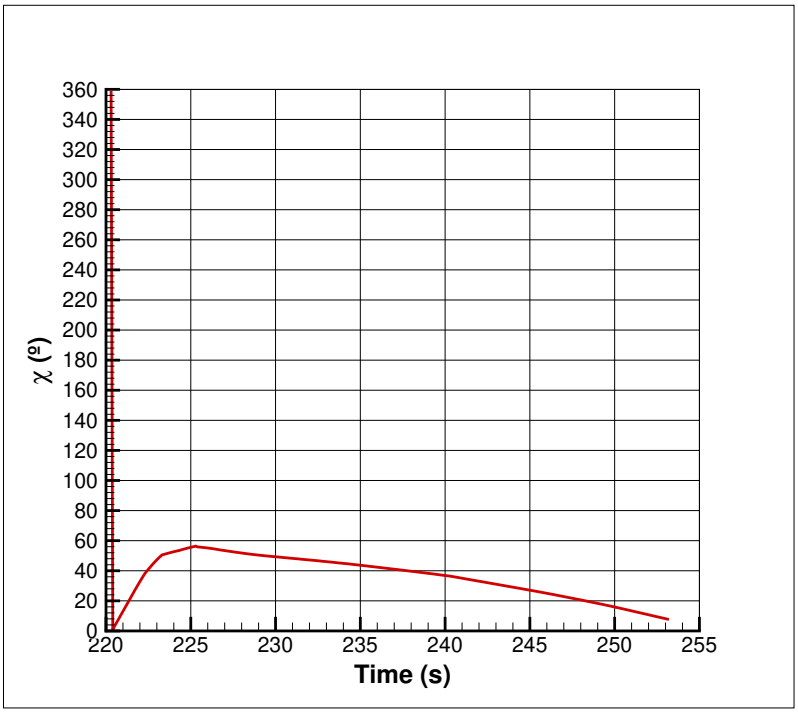

Figure 4.7: Evolution of heading angle χ across successive MPC iterations — Head-on scenario.

Analyzing Figure 4.7, and in line with the behavior of the CPA_h parameter, it can be observed that the algorithm promptly started addressing the problem by altering the aircraft's heading. The straight line seen in the graph arises from the cyclical nature of heading, where $0°$ and $360°$ denote the same direction. In the initial stages, a strategy of gradual deviation from the original trajectory is evident, followed by a correction driven by the cost function, which redirects the aircraft almost to its initial heading, but already shifted away from the trajectory that would have resulted in conflict.

Figure 4.8 shows the evolution of the flight path angle (γ) based on the controls applied by the MPC, while Figure 4.9 represents the evolution of the evasive aircraft's altitude based on the applied controls.

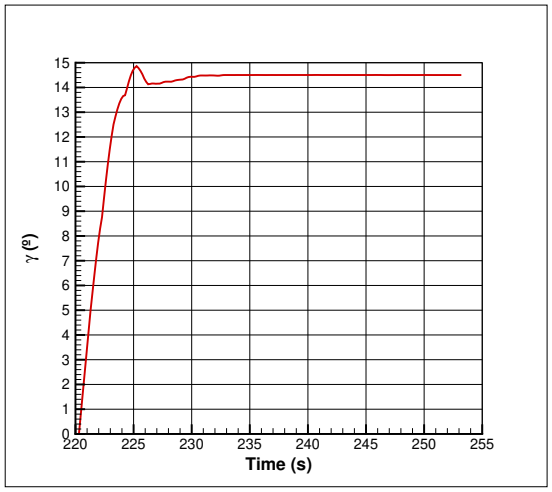

Figure 4.8: Evolution of flight path angle γ across successive MPC iterations - Head-on scenario.

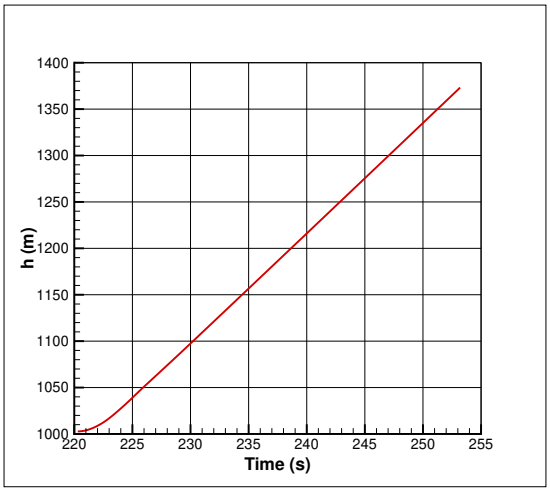

Figure 4.9: Evolution of altitude across successive MPC iterations - Head-on scenario.

From Figure 4.9, it can be observed that there is an almost constant increase in altitude throughout the entire conflict resolution interval. This result is corroborated by Figure 4.8, where the flight path angle γ initially rises and remains close to its upper limit, later stabilizing at a value slightly below this threshold, which was deemed sufficient to resolve the problem. This behavior, combined with the evolution of the heading angle χ, ensures the resolution of the initially identified conflict, which, in this case, is confirmed by compliance with the vertical separation stipulated by the standards.

Additionally, Figure 4.10 depicts the behavior of the aircraft's speed during the conflict resolution period.

The straight line observed reflects zero acceleration, indicating that the solution found did not involve changing the aircraft's speed, but rather modifying its flight condition. It is important to emphasize that, in principle, the algorithm may adopt different decision strategies, and simultaneous changes in all control variables may occur, since this implementation was conceived for application in fully autonomous missions.

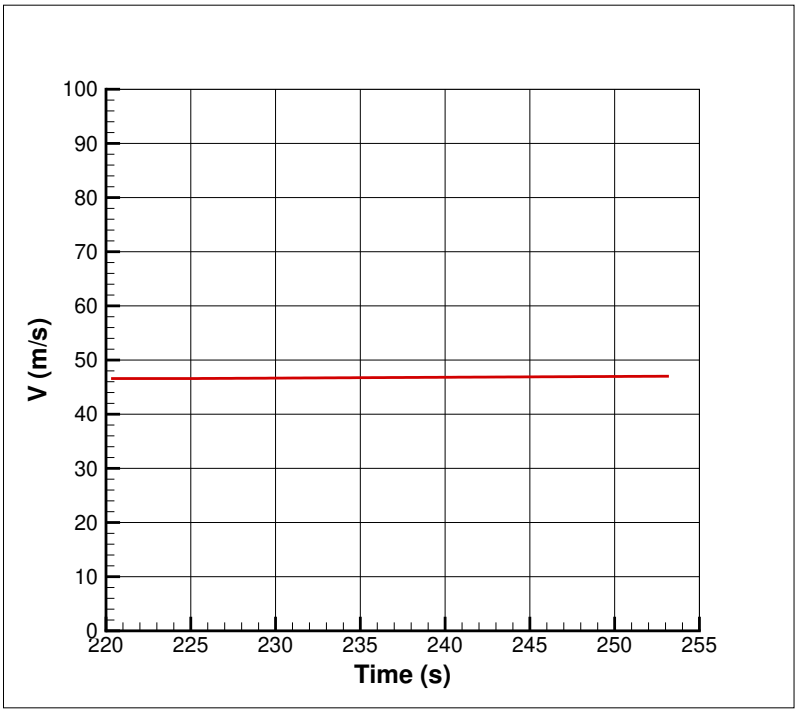

Figure 4.10: Evolution of speed across successive MPC iterations — Head-on scenario.

Thus, it is safe to state that the initially detected conflict was effectively resolved. In the specific case presented here, the resolution occurred through the guarantee of vertical separation between the aircraft. This result confirms that the algorithm was able to identify and execute an adequate maneuver, guiding the trajectory of the evasive aircraft to a condition in which the minimum vertical distance stipulated by the standards is respected. This outcome highlights not only the effectiveness of the implemented control strategy but also the system's ability to ensure compliance with regulatory safety criteria, thereby fulfilling the central objective of conflict resolution.

4.3.1.2 Effect of Prediction Horizon on the head-on Scenario

Since the prediction horizon directly influences the robustness of predictive control and, consequently, the development of the aircraft trajectory, it is essential to evaluate how different numbers of prediction steps affect the resolution of the problem. A longer horizon not only allows for greater anticipation of potential conflicts but also contributes to validating the consistency of the trajectory throughout the optimization process, even beyond the controls effectively applied in each iteration.

In general, longer prediction horizons provide a broader perspective of the future trajectory, enabling the algorithm to devise more stable and safer solutions. This forecasting capability reduces the likelihood of shortsighted decisions, ensuring that the obtained trajectory remains valid over the entire time interval considered, and not only at the instants when the controls are implemented. Furthermore, the fact that the trajectory is assessed

over extended time horizons reinforces confidence in the robustness of the solution found.

However, this benefit comes at an additional cost, since longer horizons imply a significant increase in computational effort and, consequently, in the time required for solution convergence. This limitation is particularly relevant in scenarios that demand real-time decisions, where the balance between speed and reliability is critical for the practical applicability of the system.

Thus, even though only a subset of controls is effectively applied at each iteration, as mentioned in Subsection 3.6.5, the definition of a longer prediction horizon proves essential to ensure that the optimization process has sufficient information. Therefore, the choice of horizon should be understood as a compromise between two opposing factors: on the one hand, the robustness and validity of the predicted trajectory; on the other, the computational feasibility of its implementation in operational scenarios.

To compare the stability of the parameters throughout the MPC action after the optimization process, simulations were carried out with prediction horizons of $p = 5, 15, 30, 60$. In all these simulations, the algorithm considered a sensitivity level equal to 5.

It should be noted that, although they share the same sensitivity level, the simulations are not identical. This results from the random factor associated with the initial position of the three additional aircraft, which makes it difficult to predict the exact behavior of the algorithm in each execution. Therefore, the analysis should focus primarily on the behavior of the parameters in terms of overall stability, rather than seeking absolute repeatability between scenarios.

Figure 4.11 presents the comparison of the results obtained for the different simulations, showing the effect of varying the prediction horizon on the parameter CPA_h.

In Figure 4.11, it can be observed that the different simulations followed the same resolution pattern. The algorithm prioritized proximity to the originally defined route rather than maximizing horizontal separation. This tendency is especially evident for longer prediction horizons, in which the formulation of the cost function strongly penalizes deviations from the initially planned trajectory.

The difference in stability between the various horizons considered is also evident. While smaller values of P result in more unstable trajectories subject to significant oscillations, the simulations with $P = 30$ and $P = 60$ steps stand out for the greater smoothness and consistency of their response. In both cases, horizontal separation evolves progressively and in a controlled manner, without presenting the peaks typically associated with the lack of prediction horizon.

Similarly, Figure 4.12 shows the results of the simulations obtained for the same set of prediction horizons and under the same conditions, allowing the effect of the horizon on vertical separation (Δh) to be studied.

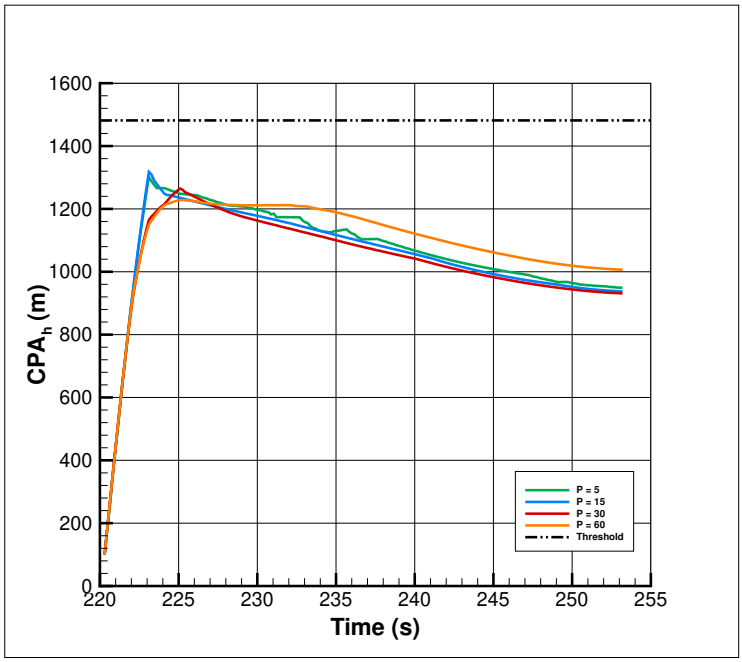

Figure 4.11: Comparison of horizontal separation parameter CPA_h for different numbers of prediction steps P — Head-on scenario.

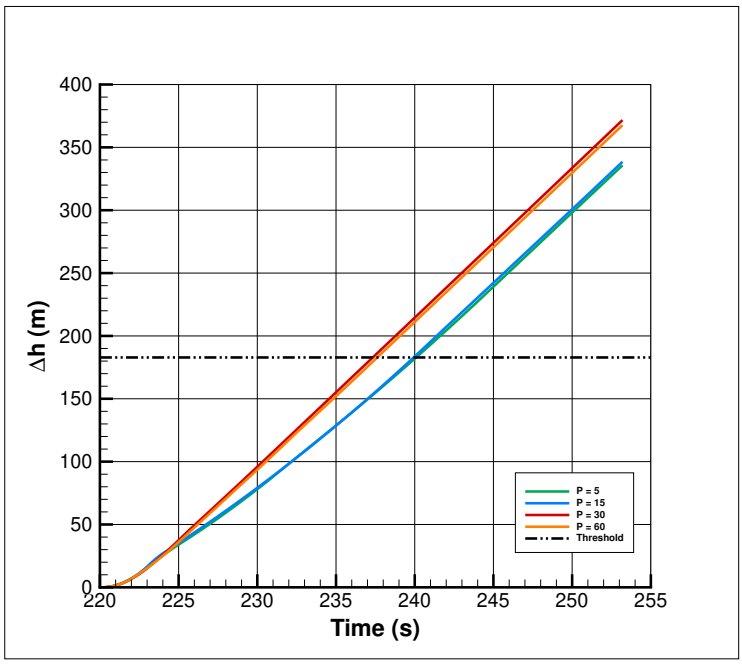

Figure 4.12: Comparison of vertical separation Δh for different numbers of prediction steps P — Head-on scenario.

It can be seen that there is an approximately constant upward trend for all trajectories, with vertical separation always ensured according to the safety level adopted. This behavior demonstrates that, regardless of the number of prediction steps considered, the algorithm is able to effectively decide how to resolve the conflict presented to it, leading to consistent solutions that ensure compliance with safety criteria and the stability of the

maneuver throughout the entire conflict resolution process.

Finally, Figure 4.13 depicts the results of the simulations, under the same guidelines as the results in Figures 4.11 and 4.12, showing the behavior of τ during the conflict resolution for various prediction horizons.

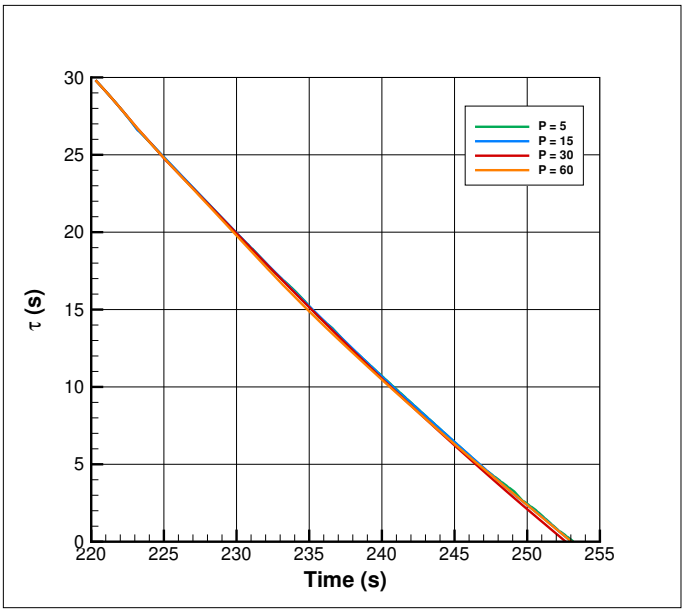

Figure 4.13: Comparison of parameter τ for different numbers of prediction steps P — Head-on scenario.

The decreasing trend confirms that an effective convergence between the aircraft occurred. Combined with the fact that the parameter Δh remained within the established safety limits, it is possible to conclude that the conflict was effectively resolved. Thus, regardless of the number of prediction steps considered, the algorithm was able to determine a valid solution, which became smoother as the value associated with P increased.

Among these aspects, one particularly relevant point concerns the execution time of the collision avoidance algorithm and the way it guarantees conflict resolution. It is important to highlight that the results obtained are intrinsically dependent not only on the mathematical formulation adopted but also on the optimization of the code and the programming language used in its development.

In this context, the *Python* language, as would be expected, proved to be limited in solving optimization problems, leading to execution times significantly longer than what would be compatible with an operational application. Therefore, the results in Table 4.5 should be interpreted only as a relative reference between different simulation configurations, not reflecting the actual order of magnitude that could be achieved in an optimized implementation using a more efficient language for the processes involved.

Prediction Steps	Execution Time (s)
5	219.91
15	690.06
30	2327.35
60	9735.39

Table 4.5: Execution time of the algorithm for different prediction steps - Head-on scenario.

4.3.1.3 Orthogonal Case

As in the procedure described in Subsection 4.2.1, the orthogonal case was now considered, representing a second approach to evaluating the conflict resolution capability of the algorithm under study, again using the cost function defined in Subsection 3.6.6.

The trajectories defined in Tables 4.3 and 4.4 were used, together with the specifications in Table 3.2 and the additional constraints in Section 4.1. As in the head-on case, category 2 was assigned to all aircraft with wingspan greater than or equal to $10\,m$, with the operational minima defined in Table 1.1.

The adopted prediction horizon was $P = 30$ steps, with a fixed time step of $\Delta t = 0.1\,s$, corresponding to a total horizon of $3\,s$. This choice reflects the compromise between ensuring effective and continuous conflict resolution and the computational effort required by the MPC optimization process.

Each simulation has a random character: while the defined trajectories remain constant, the position of the three additional aircraft varies in each run. In this scenario, the maximum altitude recorded was $3776.2\,m$, for which a sensitivity level equal to 5 was considered, according to Table 1.1, corresponding to $CPA_h = 1481.6\,m$ and $\Delta h = 182.9\,m$.

TCAS – CONFLICT ALERT

Conflict between aircraft **0** and **1** (t = 87.7s)

CPA_h = **200.2 m** || Δh = **0.0 m** || $\tau = 30$s || $\chi = 337.75°$ || $\gamma = 0.00°$

Figure 4.14 shows the evolution of the closest point of approach in the horizontal plane (CPA_h) in the orthogonal scenario, during the conflict resolution interval.

Figure 4.15 shows the evolution of the altitude difference (Δh) between the aircraft involved, considering the orthogonal scenario during the conflict analysis interval.

Finally, Figure 4.16 depicts the variation of the parameter τ throughout the simulation carried out under the orthogonal scenario.

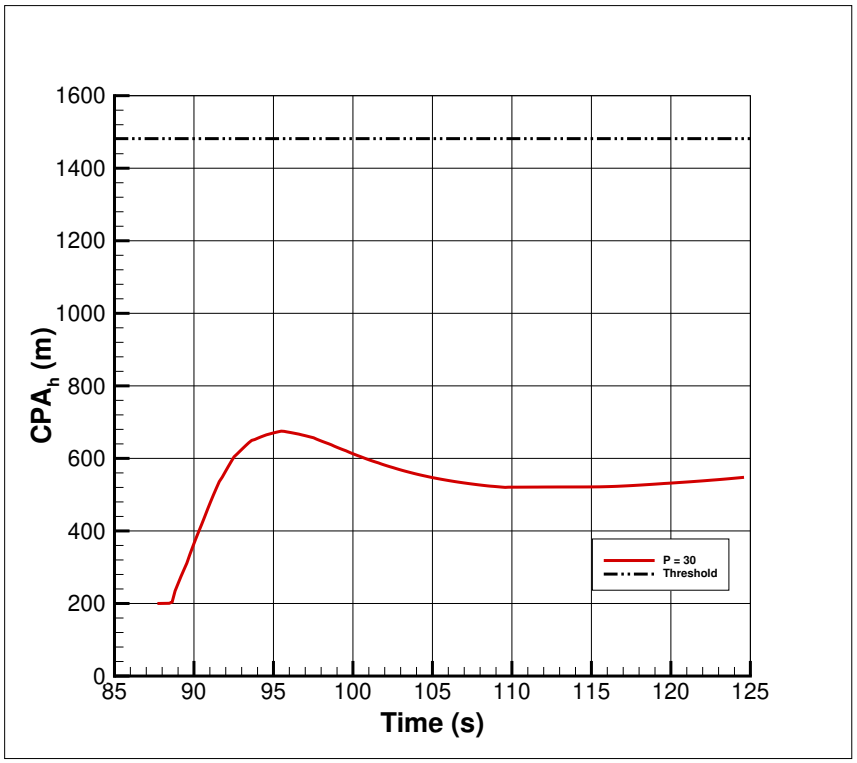

Figure 4.14: CPA_h resulting from applied controls across MPC iterations — Orthogonal scenario.

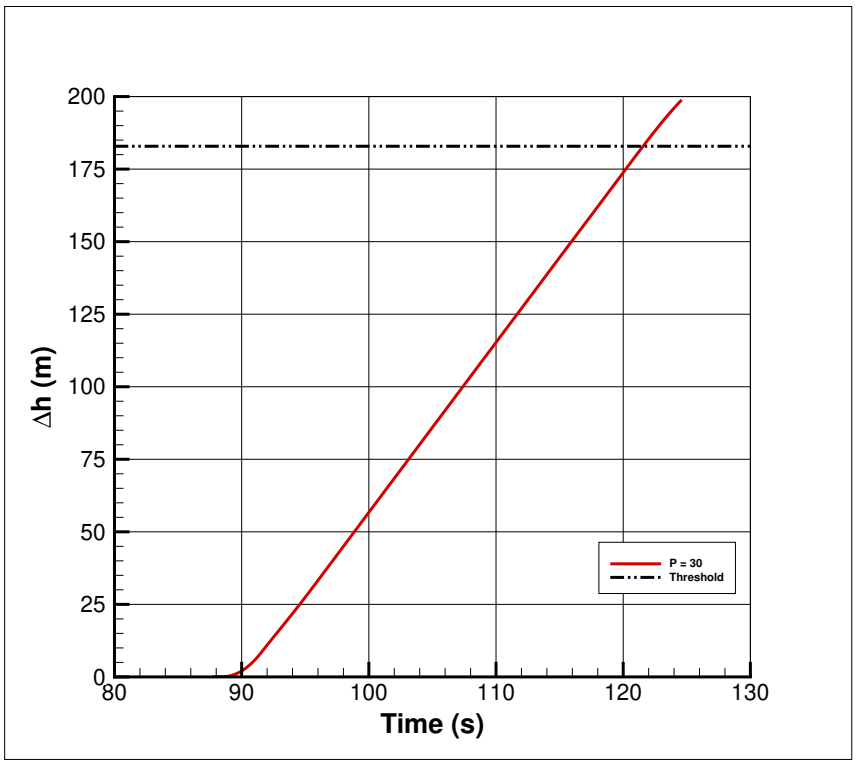

Figure 4.15: Δh resulting from applied controls across MPC iterations — Orthogonal scenario.

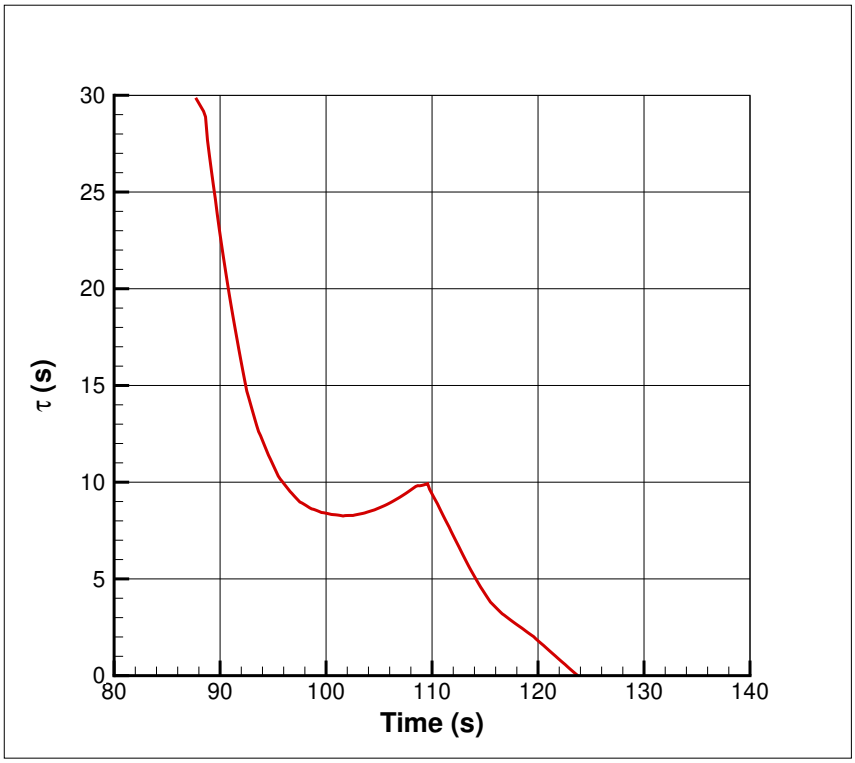

Figure 4.16: τ resulting from applied controls across MPC iterations — Orthogonal scenario.

The joint analysis of Figures 4.14, 4.15 and 4.16 shows that the algorithm once again adopted a hybrid approach to conflict resolution. Initially, an increase in CPA_h is observed (Figure 4.14), indicating an attempt to reinforce horizontal separation, in parallel with an increasing vertical separation Δh (Figure 4.15), showing that both components contributed to mitigating the collision risk.

However, as verified in Subsection 4.2.1, the algorithm ultimately favored vertical separation, progressively abandoning the maximization of horizontal distance and ensuring conflict resolution through Δh. This behavior results directly from the cost function, which penalizes deviation from the originally planned route and, whenever effectiveness is achieved through another parameter, redirects the aircraft towards the original trajectory, albeit not absolutely.

In this specific simulation, a vertical separation of $198.6\,m$ was achieved, higher than the minimum threshold of $182.9\,m$ required for a sensitivity level 5, ensuring an additional safety margin. At the same time, horizontal separation stabilized at $547.8\,m$, confirming the robustness of the solution, although it was not the determining parameter in conflict resolution.

Figure 4.16 further illustrates the evolution of τ over time. Initially, a rapid and almost linear decrease is recorded, reflecting the natural progression towards the point of conflict. This is followed by an inflection phase, in which τ stabilizes and even shows a slight

increase, a direct consequence of the sudden growth in CPA_h, which temporarily postpones the predicted instant of closest approach. Finally, τ decreases again towards zero, indicating that, despite the temporary effect of the evasive maneuver, the critical instant was inevitably reached, already with both horizontal and vertical separation ensured for conflict resolution.

Figure 4.17 presents the trajectory obtained after conflict resolution in the orthogonal scenario.

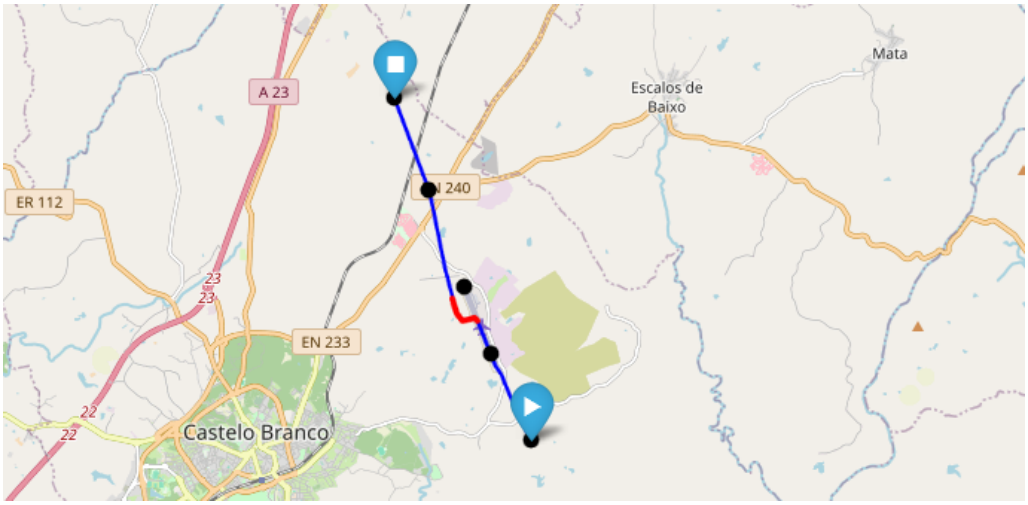

Figure 4.17: Resulting aircraft trajectory for the orthogonal scenario.

Figure 4.18 represents the evolution of the heading angle (χ) during the optimization performed by the algorithm.

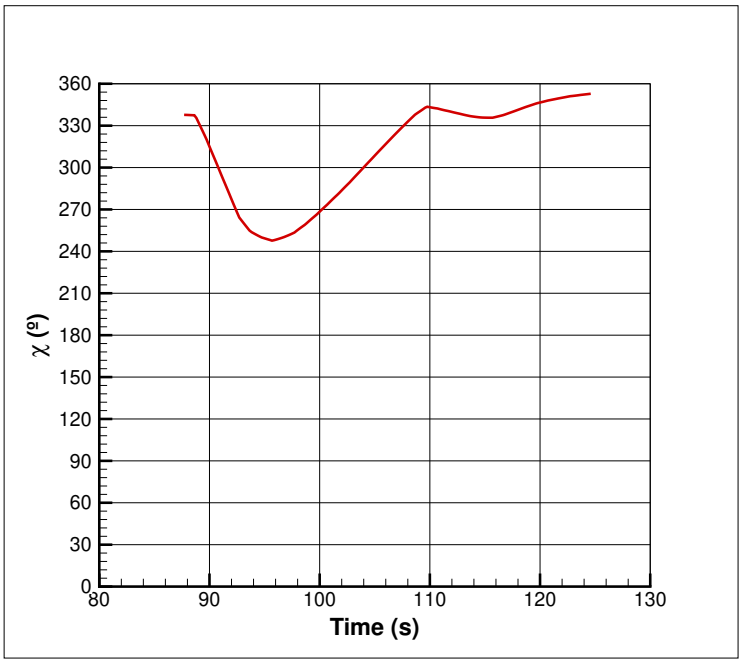

Figure 4.18: Evolution of heading angle χ across successive MPC iterations — orthogonal scenario.

Figure 4.18 shows that, immediately after the algorithm's intervention, the aircraft starts a right deviation from its original trajectory, which is later canceled in order to minimize the offset relative to the initial route. This behavior is consistent with Figure 4.14, which illustrates the evolution of CPA_h during conflict resolution. It can be seen that, whenever the deviation proves more advantageous in terms of cost, the heading is smoothly adjusted, gradually guiding the aircraft back towards the initial trajectory, but already displaced from the collision path.

Figures 4.19 and 4.20 represent the evolution of γ and altitude during conflict resolution, respectively.

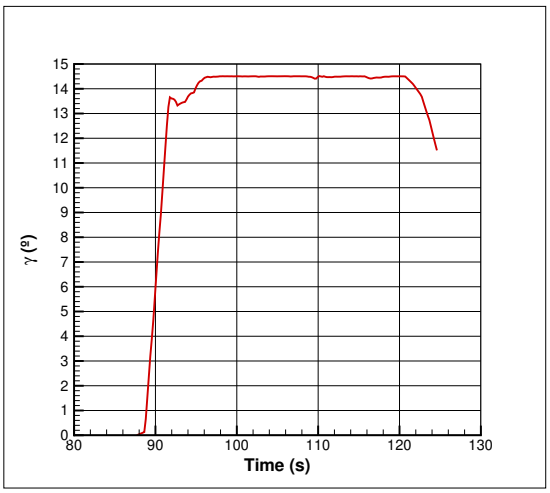

Figure 4.19: Evolution of flight path angle γ across successive MPC iterations - Orthogonal scenario.

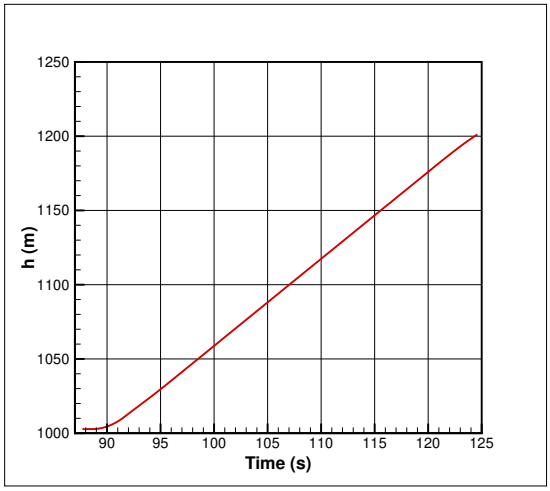

Figure 4.20: Evolution of altitude across successive MPC iterations - Orthogonal scenario.

Figure 4.19 shows that the algorithm opted from the outset to modify the value of γ in order to resolve the conflict quickly and efficiently. Shortly after the maneuver began, the value of γ rose rapidly to near its limit, stabilizing at that point before recording a gradual decrease, with the aim of stabilizing the route and eventually returning to the initial altitude. This behavior is confirmed by the evolution of altitude in Figure 4.20, which shows an almost linear increase. From a dynamic perspective, and although this case proved to be the most complex in terms of decision-making regarding γ, the parameter remained globally stable, with only small momentary variations, still ensuring the necessary vertical separation for conflict resolution.

Figure 4.21 illustrates the aircraft's speed during conflict resolution, showing that throughout the entire simulation no acceleration was imposed to modify its speed, which is reflected in the straight line observed in the graph.

Thus, it can be stated that the initially detected conflict was effectively resolved. In the specific case presented here, the resolution occurred through the guarantee of vertical separation between the aircraft, ensuring compliance with the minimum threshold stipulated by the standards. It should be emphasized, however, that each simulation produces

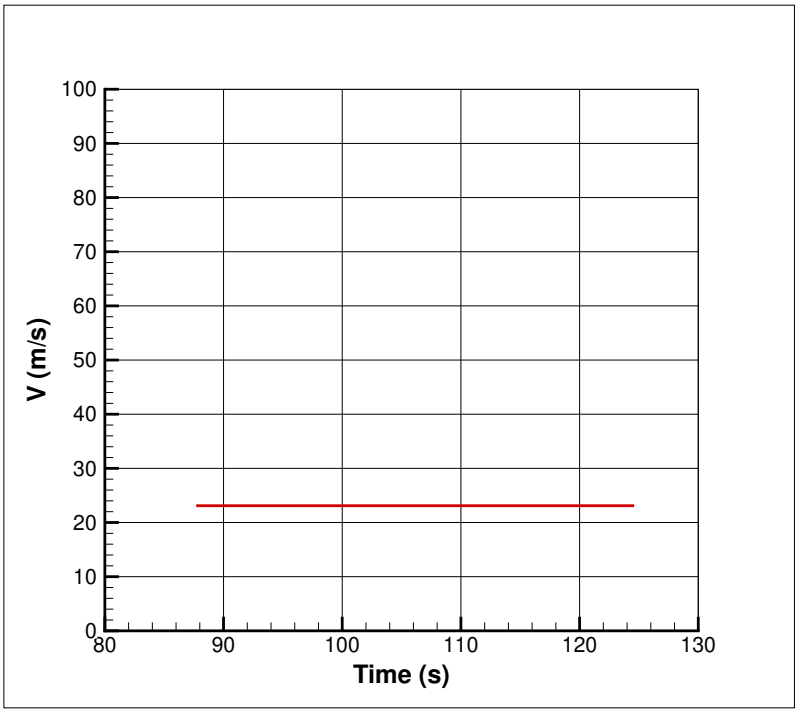

Figure 4.21: Evolution of speed across successive MPC iterations — Orthogonal scenario.

different results, due to the random character associated with the positioning of the additional aircraft, with the exception of the two with predefined trajectories. This fact reinforces not only the robustness of the implemented control strategy, but also the system's ability to ensure adequate safety levels in variable scenarios, thereby fulfilling the central objective of conflict resolution.

4.3.1.4 Effect of Prediction Horizon on the orthogonal Scenario

Since the prediction horizon directly influences the robustness of predictive control and, consequently, the evolution of the aircraft trajectory, it is essential to evaluate the impact of different values of p on the resolution of the orthogonal case. As discussed in Subsection 4.3.1.2, a longer horizon ensures that the predicted trajectory remains consistent during the optimization process, even beyond the controls effectively applied in each iteration.

This benefit, however, comes at a cost: longer horizons imply greater computational effort and a significant increase in the solution's convergence time. This limitation is particularly relevant in scenarios requiring real-time decisions, where the balance between speed and reliability is critical for the practical applicability of the system.

Therefore, the same testing protocol as in Subsection 4.3.1.2 was applied, with simulations conducted for horizons of $p = 5, 15, 30, 60$, all with a sensitivity level equal to 5. It should be noted again that, although they share this same level, the simulations are not identical, since the initial position of the three additional aircraft is generated randomly. Thus,

the analysis should focus on the overall stability of the resulting parameters, rather than seeking absolute repeatability between different runs.

Figure 4.11 presents the comparison of the results obtained for the different simulations, showing the effect of varying the prediction horizon on the parameter CPA_h.

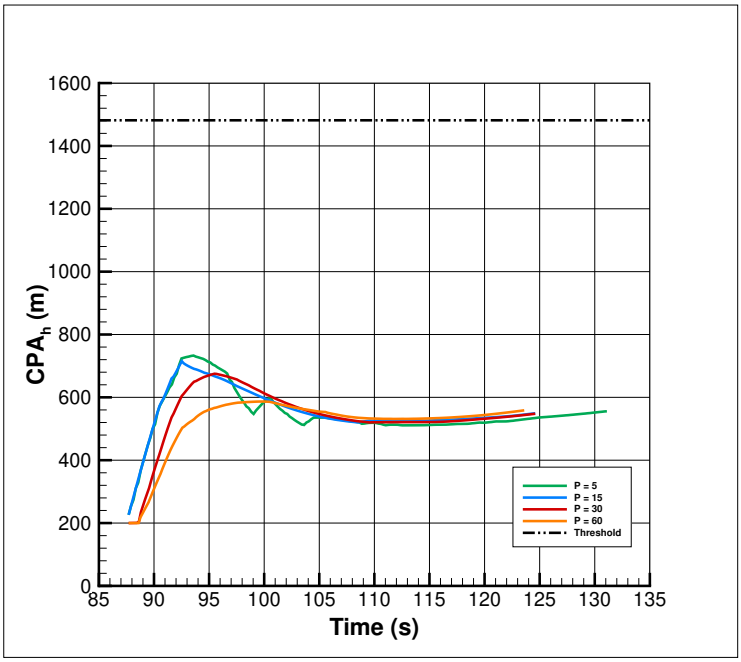

Figure 4.22: Comparison of horizontal separation parameter CPA_h for different numbers of prediction steps P — Orthogonal scenario.

Figure 4.22 shows again that the different simulations followed the same resolution pattern. The algorithm demonstrated a clear preference for keeping the trajectory as close as possible to the originally defined route, rather than maximizing horizontal separation. This tendency becomes particularly evident for longer prediction horizons, in which the cost function penalizes more significantly deviations from the planned trajectory.

The tendency to abandon the maximization of horizontal separation in favor of maintaining proximity to the initial route was repeated. This is more evident for larger prediction horizons, where the maximum horizontal separation is smaller, although at the instant of theoretical crossing, the horizontal separation tends to be slightly larger with longer horizons.

The comparison between the different horizons also reveals relevant differences in terms of stability. For small values of P, the trajectories show greater instability, with more pronounced oscillations during the maneuver. On the other hand, in the simulations with $P = 30$ and $P = 60$, greater smoothness and consistency of the response is observed, with a progressive and controlled evolution of horizontal separation, without the peaks that characterize the lack of an adequate prediction horizon.

Similarly, it is possible to compare the effect of the prediction horizon on vertical separation (Δh) during conflict resolution, as shown in Figure 4.23.

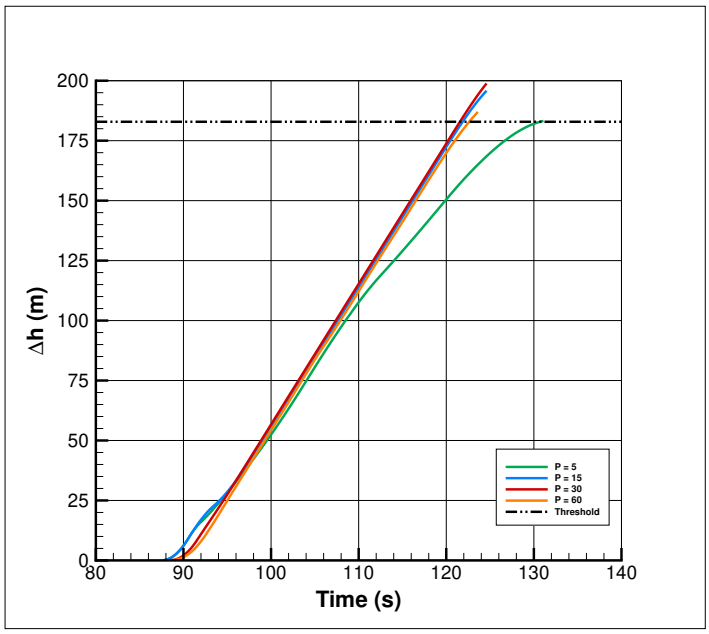

Figure 4.23: Comparison of vertical separation parameter Δh for different numbers of prediction steps P — Orthogonal scenario.

The behavior of this parameter across the simulations with different prediction horizons was practically identical, presenting an almost linear evolution in all cases, with the exception of horizon $P = 5$. In the latter, a tendency to stabilize earlier than the others was observed, although in all situations the minimum vertical separation required to consider the conflict resolved was ensured.

Figure 4.24 shows the evolution of parameter τ for different prediction horizons. Initially, in all cases, a rapid and almost linear decrease is observed, reflecting the natural progression towards the conflict point. This is followed by different behavior depending on the value of P: for shorter horizons, such as $P = 5$ and $P = 15$, the parameter shows more pronounced oscillations, reflecting less stable decisions by the algorithm, whereas for $P = 30$ and $P = 60$, the evolution is smoother and more consistent, with only small inflections resulting from the evasive maneuver. Overall, in all simulations, τ converges to zero.

As already discussed in Subsection 4.3.1.2, Python proved relatively limited in terms of efficient optimization capacity, which led to longer simulation times and, consequently, would result in delayed decision-making. In general, however, the algorithm proved capable of correctly resolving the orthogonal scenario, ensuring conflict mitigation for all prediction horizons tested.

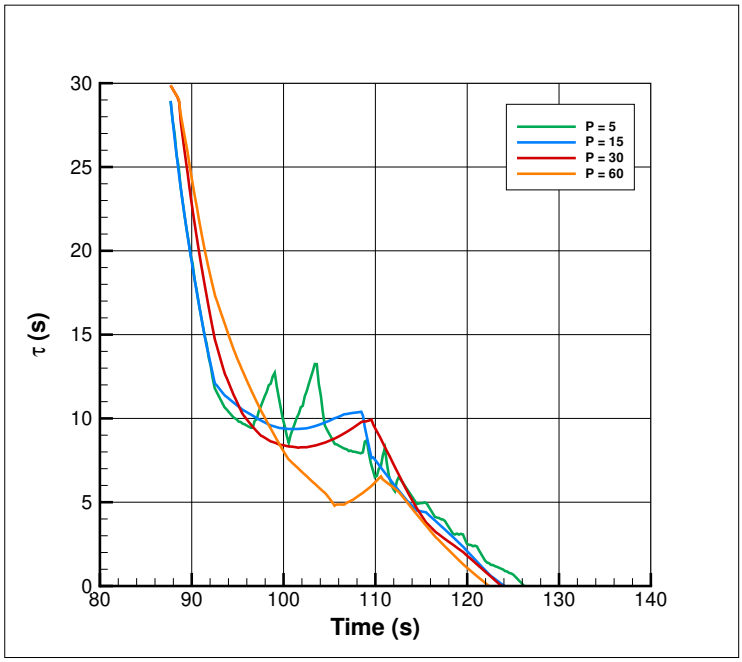

Figure 4.24: Comparison of τ parameter for different numbers of prediction steps P — Orthogonal scenario.

Table 4.6 presents the simulation times for the four prediction horizons adopted to test the algorithm.

Prediction Steps	Execution Time (s)
5	240.02
15	652.38
30	2832.55
60	8748.35

Table 4.6: Execution time of the algorithm for different prediction steps - Orthogonal scenario.

Chapter 5

Conclusions and Future Work

This dissertation set out to develop and evaluate a collision avoidance algorithm tailored to UAV operations, with the primary objective of ensuring conflict detection and resolution in three-dimensional scenarios involving multiple aircraft. The work aimed not only to design an algorithm capable of preventing collisions but also to guarantee that the resulting manoeuvres respected physical and operational constraints, maintained smoothness in control actions, and allowed a controlled return to the nominal trajectory. By doing so, the study sought to contribute to the development of reliable conflict management tools suitable for small UAVs operating in shared airspace.

The adopted methodology combined deterministic and stochastic approaches for trajectory generation, ensuring both reproducibility and robustness of the validation process. In the deterministic reference approach, two predefined trajectories were constructed from sequences of waypoints in latitude, longitude, and altitude, complemented by temporal and spatial interpolation to obtain a complete discretization of the flight paths. This setup guaranteed a systematic conflict scenario, suitable for baseline analysis. In parallel, the stochastic approach introduced three additional aircraft with randomized initial conditions, generated outside the conflict zone while respecting minimum separation margins. This ensured that conflicts would still emerge between the reference trajectories, but under more varied and realistic conditions. Importantly, this multi-aircraft configuration marked a departure from the pairwise logic of traditional TCAS: while the triggering of conflicts originated from two reference aircraft, the presence of five aircraft guaranteed continuous awareness of the surrounding traffic, ensuring that resolution manoeuvres did not inadvertently generate new conflicts.

At the core of the developed algorithm lies the TCAS module, responsible for conflict detection and alerting. The strategy of adopting the most restrictive sensitivity level among all aircraft led to conservative monitoring, ensuring that conflicts were consistently detected and signalled in advance. Detection was based on the continuous evaluation of three key metrics: the horizontal closest point of approach (CPA_h), the vertical separation (Δh), and the time to closest approach (τ). These quantities proved effective as triggering criteria, activating the conflict resolution logic whenever separation was threatened, in both structured (head-on and orthogonal) and stochastic encounters.

Conflict resolution was handled through a Model Predictive Control (MPC) framework, which incorporated explicit bounds and constraints to enforce operational feasibility. The use of the SLSQP method allowed these constraints to be directly respected while main-

taining numerical stability. To balance computational effort with solution accuracy, a variable iteration budget of 100–50 iterations per optimization cycle was employed, which proved sufficient to achieve convergence within the available time window. The cost function was carefully designed to reflect the objectives of conflict resolution: penalizations on χ and γ promoted smooth heading and altitude changes, second-difference terms avoided oscillatory control actions, and an additional penalty on sign inversions of γ prevented unrealistic switching between climb and descent. The trajectory-following component discouraged excessive lateral deviations, which led the controller to prioritize gains in Δh whenever vertical manoeuvres were more efficient. This balance ensured that the MPC consistently produced feasible, stable, and context-appropriate avoidance trajectories.

The obtained results confirmed the effectiveness of the proposed approach across all tested scenarios. In every simulation, the MPC respected the imposed physical and operational limits, while successfully recovering safe separation. Specifically, τ was consistently driven to positive and safe values, while both CPA_h and Δh increased during the avoidance phase. In head-on encounters, vertical manoeuvres dominated, as the cost structure made them the most efficient resolution path, whereas in orthogonal encounters, a balance of vertical and lateral manoeuvres was observed. In stochastic scenarios with five aircraft, the algorithm successfully managed the imposed conflicts while retaining awareness of the surrounding traffic, preventing the emergence of additional conflicts. The receding-horizon implementation with truncated application of control moves (applying the first 10 steps when $p \geq 10$) improved robustness to prediction uncertainties, preventing over-anticipation and ensuring stable responses. Furthermore, the return-to-path logic consistently guided the aircraft back to the nominal trajectory with smooth χ and γ profiles, selecting convergence waypoints that minimized deviations and guaranteeing re-entry without abrupt or unsafe manoeuvres.

In summary, the algorithm achieved its objectives: it demonstrated the capability to modify the evasive aircraft trajectory to prevent conflicts while maintaining situational awareness of the entire traffic set. Unlike conventional TCAS implementations that operate in a pairwise fashion, the proposed framework accounted for five aircraft simultaneously, ensuring that the selected manoeuvre was not only effective in resolving the immediate conflict but also safe with respect to the broader traffic context. The TCAS module provided timely and conservative detection, the MPC ensured feasible and efficient resolution manoeuvres, and the return-to-path strategy secured reintegration into the planned trajectory. Taken together, these elements formed a coherent and robust conflict avoidance framework. From a critical perspective, the solutions achieved were operationally feasible, conservative enough to guarantee safety, and robust to different encounter geometries. Nevertheless, limitations remain: the algorithm was validated under simulated conditions with five aircraft, and further testing under higher traffic densities, real-time constraints, and sensor uncertainties would be required to fully assess scalability and operational deployment. These aspects define natural directions for future research and refinement.

5.1 Limitations

Despite the results achieved, the work developed presents a set of limitations that should be acknowledged.

Although the primary objective was to create an algorithm capable of resolving potential conflicts, the execution time of the code remains a significant constraint. The repetitive resolution of the optimisation problem entails a considerable computational effort. This characteristic limits the practical applicability of the algorithm in real-time scenarios, where conflict detection and resolution require almost immediate responses. Thus, while the method is suitable for analysis and validation in a simulation environment, its direct implementation in embedded systems or operational control would require substantial performance improvements and/or the adoption of a more efficient programming language.

The problem formulation also presents restrictions, as it is limited to conflict situations involving only two aircraft, whose trajectories are predefined by the user. This simplification allows the study to focus on the performance of the conflict resolution algorithm but reduces its ability to generalise to more realistic scenarios. In operational contexts, air traffic typically involves multiple aircraft with dynamic trajectories subject to frequent changes, aspects that are not considered in the present approach.

This formulation further ensures the existence of an actual conflict scenario—assuming that the trajectories provided by the user inevitably lead to such a situation—while enabling the evaluation of the algorithm's resolution performance. The definition of categories for the selection of the evasive aircraft provides a generic approach to determine which aircraft should alter its trajectory. In a real implementation, it would be sufficient to define a simplified cost function to select, among the five simulated aircraft, the one that should perform the evasive manoeuvre.

Uncertainty is also associated with the simulation logic of randomly generated aircraft. The prediction of the future trajectory relies on maintaining the flight conditions observed at the immediately preceding instant, which introduces limitations to the reliability of the estimated path. This situation is particularly evident when information exchange between aircraft relies solely on transponders, since these transmit flight parameters (such as speed, altitude, or heading) but not the actual route that each aircraft intends to follow. As a result, only the aircraft itself has complete knowledge of its planned trajectory, while each surrounding unit must infer possible paths based on the transmitted data, which may not accurately reflect the real course.

The impact of the simplifications adopted in the dynamic model of the aircraft and the simulation environment must also be noted. External factors such as wind, turbulence, or atmospheric variations were not considered, nor were uncertainties arising from sensor measurement errors. These simplifications made it possible to isolate the performance of

the algorithm and assess its robustness under controlled conditions, but they depart from the inherent complexity of real operations. The absence of such factors can significantly affect the accuracy of the predicted trajectories and, consequently, the effectiveness of conflict resolution manoeuvres in operational scenarios.

Finally, it should be highlighted that the entire validation of the algorithm was carried out in a simulation environment. While this approach allows variables to be isolated and performance to be assessed under controlled conditions, the absence of tests in real scenarios limits the demonstration of the method's robustness against factors such as sensor noise, communication failures, variable atmospheric conditions, or more complex aircraft dynamics. Nevertheless, simulation represents a fundamental first step, providing a solid basis for future work towards experimental implementation and real flight testing.

5.2 Future Work

Based on the results obtained in this dissertation, several directions for future work can be identified.

First, the computational efficiency of the algorithm should be improved to strengthen its applicability in real-time contexts. Possible approaches include implementing the method in faster programming languages such as C++ or Fortran, exploring parallelisation strategies, or making use of GPU-based computation. These developments would help reduce execution time without compromising the robustness of the solution.

Second, the fidelity of the simulation model could be increased by incorporating more realistic elements of aircraft dynamics and the operational environment. External factors such as wind, turbulence, or atmospheric variability, as well as sensor noise and actuator delays, were not considered in the present study but may have a significant impact on conflict resolution under real-world conditions.

Another promising line of research concerns the adoption of adaptive prediction horizons to improve the robustness of the manoeuvres. Instead of relying on a fixed and finite horizon, the algorithm could use horizons that vary according to the criticality of the conflict (for example, based on the values of τ or relative distance), or employ non-uniform temporal discretisation to balance accuracy and computational cost.

Finally, the cost function could be further refined. The weighting factors defined in this work should be re-evaluated to ensure their adequacy across different operational scenarios, while continuing to reflect safety and performance priorities as the application context becomes more complex.

Bibliography

[1] R. A. Paielli and H. Erzberger, "Conflict probability estimation for free flight," NASA Ames Research Center, Moffett Field, California, USA, NASA Technical Memorandum 110411, 1996. 1

[2] A. Batinovic, J. Goricanec, L. Markovic, and S. Bogdan, "Path planning with potential field-based obstacle avoidance in a 3d environment by an unmanned aerial vehicle," *arXiv preprint*, 2023. 2

[3] J. Choi and Y. Choi, "Path planning for unmanned aerial vehicle: A-star-guided potential field method," *Drones*, vol. 9, no. 8, p. 545, 2025. 2, 3

[4] S. A. Fadzli, S. I. Abdulkadir, M. Makhtar, and A. A. Jamal, "Robotic indoor path planning using dijkstra's algorithm with multi-layer dictionaries," in *IEEE International Conference on Computer, Communications, and Control Technology (I4CT)*. IEEE, 2015, pp. 190–195. 3

[5] G. Lavezzi, K. Guye, V. Cichella, and M. Ciarcià, "Comparative analysis of nonlinear programming solvers: Performance evaluation, benchmarking, and multi-uav optimal path planning," *Drones*, vol. 7, no. 8, p. 487, 2023. [Online]. Available: https://doi.org/10.3390/drones7080487 3, 4

[6] J. Xin, J. Zhong, F. Yang, Y. Cui, and J. Sheng, "An improved genetic algorithm for path-planning of unmanned surface vehicle," *Sensors*, vol. 19, no. 11, p. 2640, 2019. 3, 4

[7] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *IEEE International Conference on Neural Networks*. IEEE, 1995, pp. 1942–1948. 3, 4

[8] S. M. LaValle, "Rapidly-exploring random trees: A new tool for path planning," *Technical Report*, 1998. 4, 5

[9] S. M. LaValle and J. J. Kuffner, "Randomized kinodynamic planning," in *IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 1999, pp. 473–479. 4, 5

[10] S. Hrabar, "3d path planning and stereo-based obstacle avoidance for rotorcraft uavs," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Nice, France: IEEE, 2008, pp. 807–814. 6

[11] Y. Kumar, A. Manoharan, and P. B. Sujit, "Right of way rules based collision avoidance approach using model predictive control," *Preprint*, Dec. 2019, available at ResearchGate. [Online]. Available: https://www.researchgate.net/publication/337886698 6, 11

[12] S. Á. de Toledo, J. M. Barreiro, J. L. Fuertes, Á. L. González, and J. A. Lara, "An approach to fully automatic aircraft collision avoidance and navigation," in *Proceedings of the 7th WSEAS International Conference on Applied Computer Science*. Venice, Italy: WSEAS, 2007, pp. 259–265. [Online]. Available: https://www.researchgate.net/publication/239855312 7, 8

[13] E. Aldao, L. M. González-Desantos, H. Michinel, and H. González-Jorge, "Uav obstacle avoidance algorithm to navigate in dynamic building environments," *Drones*, vol. 6, 1 2022. 7, 8, 9

[14] Z. Lin, L. Castano, E. Mortimer, and H. Xu, "Fast 3d collision avoidance algorithm for fixed wing uas," *Journal of Intelligent & Robotic Systems*, vol. 97, pp. 577–604, 2019. [Online]. Available: https://doi.org/10.1007/s10846-019-01037-7 8

[15] J. Guo, C. Liang, K. Wang, B. Sang, and Y. Wu, "Three-dimensional autonomous obstacle avoidance algorithm for uav based on circular arc trajectory," *International Journal of Aerospace Engineering*, vol. 2021, 2021. 8

[16] W. Ouyang, X. Gan, and L. Tong, "Research on uavs collision avoidance with multiple dynamic obstacles," in *2022 IEEE International Conference on Advances in Electrical Engineering and Computer Applications, AEECA 2022*. Institute of Electrical and Electronics Engineers Inc., 2022, pp. 594–598. 9

[17] D. H. Shim, "An evasive maneuvering algorithm for uavs in sense-and-avoid situations," in *Aerial Vehicles*. InTech, 2009, pp. 622–636. [Online]. Available: https://www.researchgate.net/publication/221787571 9, 10, 11

[18] K. Bousson, "Model predictive control approach to global air collision avoidance," *Aircraft Engineering and Aerospace Technology*, vol. 80, pp. 605–612, 2008. 9, 10, 11

[19] X. Chen, Y. Wan, and S. Lao, "Graphical modeling and simulation for a multi-aircraft collision avoidance algorithm based on collaborative decisions," *Symmetry*, vol. 12, 6 2020. 9, 11

[20] M. Džunda, P. Dzurovčin, and L. Melníková, "Anti-collision system for small civil aircraft," *Applied Sciences*, vol. 12, no. 3, p. 1648, 2022. [Online]. Available: https://doi.org/10.3390/app12031648 9

[21] S. Yu and J. Park, "Collision avoidance algorithm for a fixed-wing uav utilizing geometric estimation considering multiple dynamic obstacles," *International Journal of Aeronautical and Space Sciences*, vol. 25, pp. 1446–1463, 10 2024. 10

[22] S. Roelofsen, A. Martinoli, and D. Gillet, "3d collision avoidance algorithm for unmanned aerial vehicles with limited field of view constraints," in *Proceedings of the 55th IEEE Conference on Decision and Control (CDC)*. Las Vegas, USA: IEEE, 2016, pp. 2555–2562. 10

[23] H. I. Lee, H. S. Shin, and A. Tsourdos, "Uav collision avoidance considering no-fly-zones," in *IFAC-PapersOnLine*, vol. 53. Elsevier B.V., 2020, pp. 14 748–14 753. 10

[24] M. Becker, R. C. B. Sampaio, S. Bouabdallah, V. de Perrot, and R. Siegwart, "In-flight collision avoidance for a mini-uav robot based on onboard sensors," in *Proceedings of the ABCM Symposium Series in Mechatronics*, vol. 4. Brazil: ABCM, 2012, pp. 961–970. [Online]. Available: https://www.researchgate.net/publication/3332569_Real-Time_Stabilization_and_Tracking_of_a_Four-Rotor_Mini_Rotorcraft 11

[25] EUROCONTROL, "Acas ii guide: Airborne collision avoidance systems," EUROCONTROL, Tech. Rep., Mar. 2022, supporting European Aviation. [Online]. Available: https://www.eurocontrol.int/system/acas 12, 13

[26] M. Schwenzer, M. Ay, T. Bergs, and D. Abel, "Review on model predictive control: an engineering perspective," *The International Journal of Advanced Manufacturing Technology*, vol. 114, no. 5-6, pp. 1327–1349, 2021. 19

[27] M. Arnold, R. R. Negenborn, G. Andersson, and B. De Schutter, "Multi-area predictive control for combined electricity and natural gas systems," *European Control Conference (ECC)*, pp. 3534–3541, 2015. 19

[28] D. Kraft, "A software package for sequential quadratic programming," *DFVLR Obersfaffeuhofen, Germany*, 1988, technical Report DFVLR-FB 88-28. 22

[29] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Mathematical Programming*, vol. 45, no. 1-3, pp. 503–528, 1989. 22

[30] R. Pryss, M. Schickler, J. Schöbel, M. Reichert *et al.*, "Enabling tracks in location-based smart mobile augmented reality applications," *Procedia Computer Science*, vol. 110, pp. 207–214, Dec. 2017, cC BY-NC-ND 4.0. 33

A Multi-aircraft Collision Avoidance System for Small Fixed-wing UAVs

Diogo Ferreira, Kouamana Bousson.

Abstract:

Unmanned Aerial Vehicles (UAVs) require reliable mid-air conflict management to enable safe flight mostly in dense traffic flow areas. The present paper investigates a conflict detection and resolution algorithm inspired by the principles of the Traffic Collision Avoidance System (TCAS), specifically its detection logic and separation threshold, and it adapts the underlying concepts to UAV scenarios. The method uses Model Predictive Control (MPC) to synthesize evasive, dynamically feasible trajectories that preserve safety and efficiency in three-dimensional flight. This design enforces diverse scenarios and tests whether resolutions for the primary pair remain safe with respect to nearby vehicles. Results show that the algorithm consistently accounts for the surrounding traffic flow so that new conflicts are not induced when separating conflicting aerial vehicles. Computational cost associated with on-line optimization is identified as one of the issues to be dealt with in future research work. Overall, the findings support the feasibility of MPC-based conflict resolution for UAVs and motivate to set a promising framework on multi-aircraft encounters, more realistic environmental effects, and computational efficiency.

Keywords: UAV conflict resolution, Collision avoidance, Traffic Collision Avoidance System, Model Predictive Control; Air traffic simulation, Evasive maneuvers.

Nomenclature

MPC	: Model Predictive Control
TCAS	: Traffic Collision Avoidance System
$CPAh$	: Horizontal Closest Point of Approach
Δh	: Differential Altitude
τ	: Time to Conflict

D. Ferreira[1], K. Bousson[1] (✉)
[1]University of Beira Interior, Faculty of Engineering, Department of Aerospace Sciences, Covilhã, Portugal
e-mail: diogo.santos.ferreira@ubi.pt; bousson@ubi.pt

T.H. Karakoc et al. (eds.), ISATECH'25 Proceedings of International Symposium on Aviation Technology, MRO and Operations 2025
DOI 10.1007/…….

1. Introduction and Simulation Considerations

The accelerated growth in the use of Unmanned Aerial Vehicles (UAVs) across multiple domains—ranging from surveillance and monitoring missions to small cargo transport and urban operations—has clearly highlighted the need for air traffic management systems capable of ensuring safety in shared airspace. The massive introduction of these vehicles into air corridors that, until now, were reserved for conventional aviation poses additional challenges: high traffic density, trajectory uncertainty, platform heterogeneity, and limitations in UAVs' sensing and reaction capabilities. In this context, there is a clear demand for robust collision avoidance solutions to integrate UAVs safely into complex airspace environments.

Conflict assessment is performed based on the parameters commonly associated with TCAS systems: $CPAh$, Δh, and τ (Eurocontrol, 2022). The trajectory of the evasive aircraft is then obtained using a Model Predictive Control (MPC) framework, with an associated cost function that balances all relevant variables to ensure the safe and efficient development of the evasive trajectory (Bousson, 2008; Su and Park, 2024), ultimately guaranteeing complete conflict resolution.

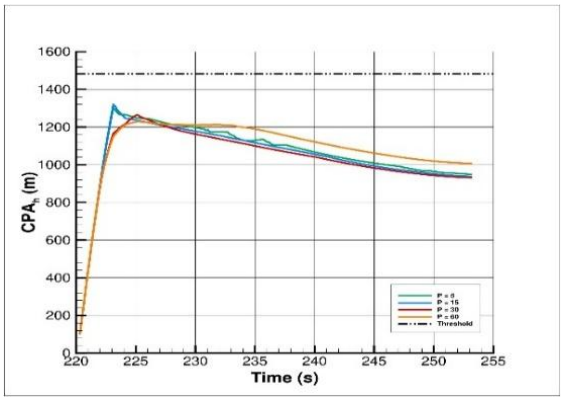

Fig.1. Evolution of CPA_h during MPC optimization.

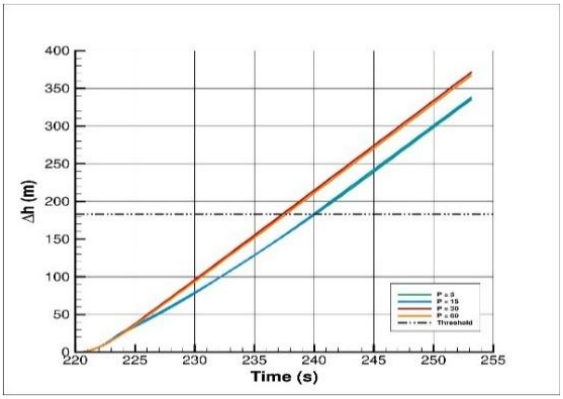

Fig.2. Evolution of Δh during MPC optimization.

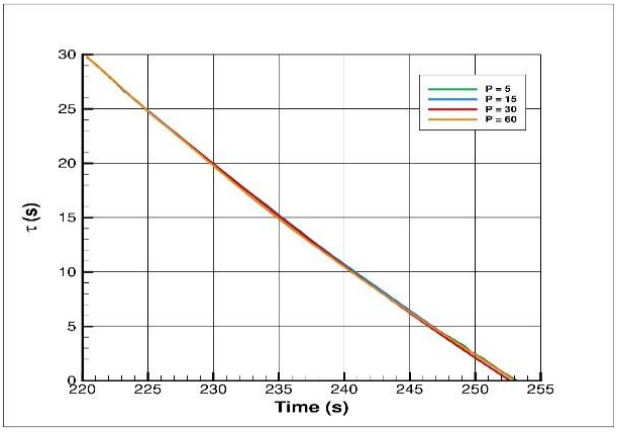

Fig.3. Evolution of τ during MPC optimization.

2. Head-on Encounter

The head-on configuration reflects a typical operational scenario of TCAS systems implemented in civil aviation, where two aircraft follow the same route but in opposite directions. For this scenario, the trajectories were defined using five waypoints located in the Castelo Branco region (in Portugal). To ensure mirrored route development, the first waypoint of aircraft 0 corresponds to the last waypoint of aircraft 1, and the same applies to the entire set of waypoints.

To test the algorithm, a prediction horizon of 30 steps was adopted, with a time increment of 0.1 seconds. This configuration allows the MPC to project its analysis 3 seconds ahead of the initial conflict detection instant. Additional simulations were later conducted with horizons of 5, 15, and 60 steps in order to assess the effect of the prediction horizon on the MPC's behaviour.

Thus, through a combined evaluation of the three parameters used to signal a conflict, it is possible to understand the impact of the MPC's actions on the trajectory development. Figures 1, 2, and 3 show the temporal evolution of $CPAh$, Δh, and τ, respectively.

Based on the results presented, it is possible to highlight the behaviour of the MPC in the conflict resolution process. At all prediction instants, the controller exhibited a tendency to mitigate risk immediately, initially maximizing horizontal separation. Subsequently, a shift in focus was observed toward maximizing vertical separation, at the expense of horizontal distance.

This behaviour directly reflects the properties of the cost function, in which a specific weight is assigned to the deviation of the evasive aircraft from its originally planned route. Thus, with the natural evolution of τ towards zero, it becomes evident that the MPC was able to determine an effective evasive trajectory capable of resolving the initially detected conflict. At the end of the avoidance maneuver, the

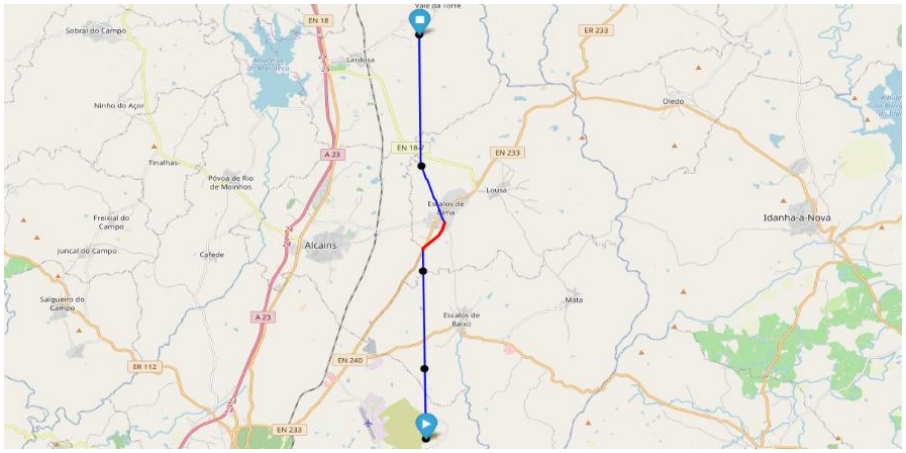

Fig.4. Resulting aircraft trajectory after conflict resolution in the head-on scenario.

conflict was completely mitigated through the increase in vertical separation. Figure 4 shows the trajectory of the evasive aircraft resulting from the MPC's action.

3. Orthogonal Encounter

The orthogonal encounter was introduced to complement the head-on scenario and broaden the evaluation of the conflict-resolution algorithm. In this case, two aircraft follow perpendicular trajectories converging toward a common intersection point, a geometry that increases control sensitivity and makes the optimal avoidance direction less straightforward than in the head-on configuration.

The trajectories follow the same 2–3 setup used previously: two predefined trajectories, defined by waypoints, crossing times, and altitude, contain the conflict, while three other trajectories are generated randomly to preserve situational awareness within the scenario. We are dealing here with a total of five aerial vehicles.

A prediction horizon of 30 steps with a 0.1-second increment was used, allowing the MPC to look 3 seconds ahead of the conflict-detection instant. Additional horizons (5, 15, and 60 steps) were also tested to examine how the controller's behavior varies with prediction length.

From the analysis of Figures 5–7, it becomes clear that the MPC initially commands increase in both horizontal and vertical separation, similarly to the head-on case. As the simulation progresses, however, the horizontal-separation component becomes less relevant, and the controller focuses on increasing vertical separation—the factor that ultimately resolves the conflict. The behavior of τalso differs from the previous scenario due to the distinct geometry of the encounter.

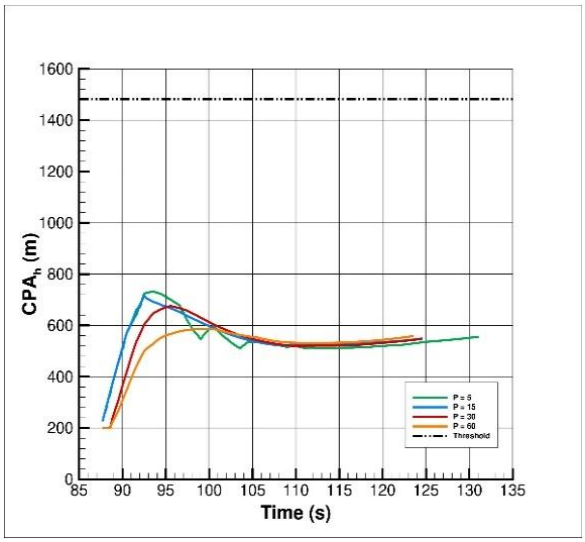

Fig.5. Evolution of CPA_h during MPC optimization.

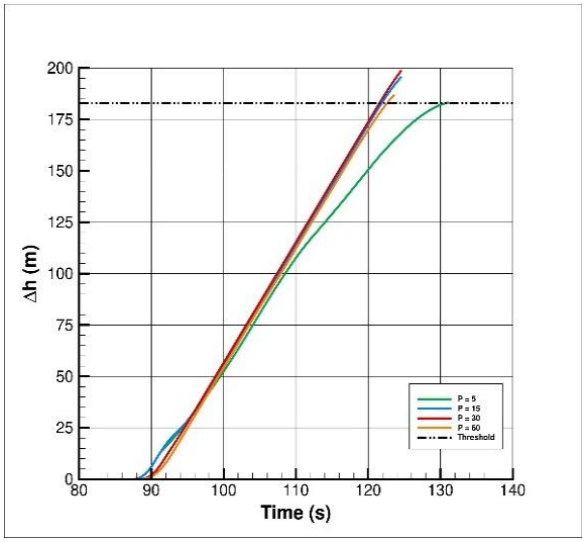

Fig.6. Evolution of Δh during MPC optimization.

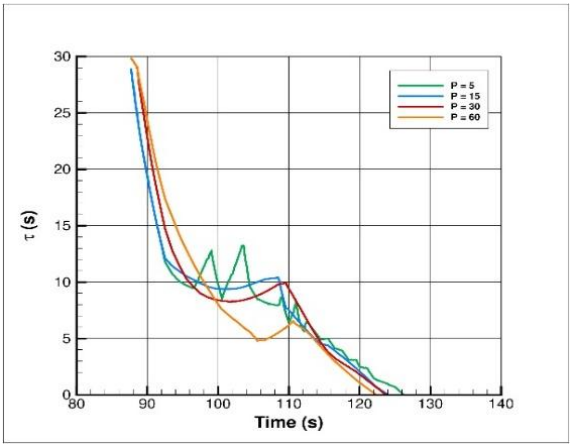

Fig.7 Evolution of τ during MPC optimization.

4. Conclusion

The proposed method achieved its objectives: it demonstrates the capability to alter the evasive aircraft trajectory to prevent conflicts while maintaining situational awareness of the entire traffic flow. Unlike conventional TCAS implementations that operate in a pairwise fashion, the proposed framework accounts for five aircraft simultaneously, ensuring that the selected maneuver was not only effective in resolving the immediate conflict but also safe with respect to the broader traffic flow context. Beyond the computational simulation results described in the paper, future work will focus on real scenarios to demonstrate the robustness of the method against factors such as sensor noise, communication failures, variable atmospheric conditions, or more complex aircraft dynamics. Nevertheless, the simulation done represents a fundamental first step, providing a solid basis for future work towards experimental implementation and real flight testing.

References

Eurocontrol (2022) ACAS II Guide: Airborne Collision Avoidance Systems. Technical report, Eurocontrol, 2022. Supporting 741 European Aviation.

Bousson K (2008) Model Predictive Control Approach to Global Air Collision Avoidance. Aircraft Engineering and Aerospace Technology, 80 (6): 605–612. https://doi.org/10.1108/00022660810911545.

Yu S, Park J (2024) Collision Avoidance Algorithm for a Fixed-wing UAV Utilizing Geometric Estimation Considering Multiple Dynamic Obstacles. International Journal of Aeronautical and Space Sciences, 25: 1446–1463. https://doi.org/10.1007/s42405-024-00741-5.