SCHOOL OF ENGINEERING AND DESIGN
AEROSPACE AND GEODESY

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Semester Thesis in Mechatronics, Robotics and Biomechanical Engineering

Safe Corridor-Based Trajectory Optimization and Conflict Resolution for Multi-UAV Systems in Narrow Passages

Okan Arif Güvenkaya

SCHOOL OF ENGINEERING AND DESIGN
AEROSPACE AND GEODESY

TECHNISCHE UNIVERSITÄT MÜNCHEN

Master's Semester Thesis in Mechatronics, Robotics and Biomechanical Engineering

Safe Corridor-Based Trajectory Optimization and Conflict Resolution for Multi-UAV Systems in Narrow Passages

Author:	Okan Arif Güvenkaya
Supervisor:	Lukas Pries
Advisor:	Prof. Dr.-Ing. Markus Ryll
Submission Date:	21.03.2026

I confirm that this master's semester thesis is my own work and I have documented all sources and material used.

Munich, 21.03.2026 Okan Arif Güvenkaya

Acknowledgments

I would like to express my sincere gratitude to Lukas Pries for the direct supervision, continuous support, and valuable technical guidance throughout this thesis. Their feedback and encouragement were essential to the completion of this work.
I would also like to thank Prof. Dr. Markus Ryll for providing me with the opportunity to carry out this thesis at Autonomous Aerial Systems and for the academic guidance and support.

Abstract

This semester thesis compares Spatio-Temporal Optimization (STO) and Mixed-Integer Quadratic Programming (MIQP) for safe Unmanned Aerial Vehicle (UAV) trajectory planning in obstacle-rich environments, using safe flight corridors to define collision-free regions. The comparison focuses on solution quality, constraint satisfaction, and computational cost under a shared problem setup. The core planning pipeline first uses 3D A* for global pathfinding, followed by convex decomposition to generate Safe Flight Corridors (SFCs), which define guaranteed collision-free regions in constrained environments. Within these corridors, trajectories are optimized using 5th-order minimum control (MINCO) polynomials. In STO, both spatial waypoints and segment timings are optimized jointly ; in MIQP, time allocation is fixed and only waypoints are optimized. This achieves smoothness and dynamic feasibility.

The STO framework is extended to multi-UAV systems through a local temporal conflict-resolution strategy for swarm coordination. Instead of global replanning, the method detects space–time conflicts and resolves them by assigning a temporal delay to one UAV per conflict (the "loser") and re-optimizing only that UAV's trajectory. The injected delay is preserved by freezing the pre-conflict segment timings, while the path after the conflict is kept by freezing the corresponding waypoints; only the pre-conflict waypoints and post-conflict timings are re-optimized, so feasibility and the overall structure of the solution are maintained. This allows efficient coordination even in narrow-passage environments where multiple UAVs share limited free space.

Contents

Symbols

P	Number of safe corridors / convex polytopes
C_p	p-th safe corridor (convex polytope)
A_p	Half-space matrix defining corridor C_p
b_p	Half-space vector defining corridor C_p
m_p	Number of half-space inequalities of corridor C_p
N	Number of trajectory segments
T_i	Duration of trajectory segment i
T	Total trajectory duration
q_i	Waypoint i
$p(t)$	Position trajectory
$\dot{p}(t)$	Velocity trajectory
$\ddot{p}(t)$	Acceleration trajectory
$p^{(3)}(t)$	Jerk
$v_{\max}$	Maximum velocity bound
$a_{\max}$	Maximum acceleration bound
$\sigma(i)$	Corridor index assigned to segment i
J_{jerk}	Jerk-minimization cost
J	Total objective/cost function
λ_{time}	Weight of total flight-time penalty
λ_{jerk}	Weight of jerk cost in STO
λ_c	Weight of corridor penalty
λ_v	Weight of velocity penalty
λ_a	Weight of acceleration penalty
K	Number of sampled time instants used for penalties
t_k	Sampled time instant
J_v	Velocity penalty term
J_a	Acceleration penalty term
J_c	Corridor penalty term
J_c^{L1}	L1 corridor penalty
J_c^{L2}	L2 corridor penalty
$J_c^{\log}$	Logarithmic corridor penalty

d_k	Corridor violation vector at time sample t_k
τ_i	Unconstrained STO time variable for segment i
$z_{i,p}$	Binary variable indicating assignment of segment i to corridor p
$\bar{T}_i$	Prescribed/fixed segment duration in MIQP
$r_{i,k}$	Bézier position control point
$v_{i,k}$	Bézier velocity control point
$a_{i,k}$	Bézier acceleration control point
π_i	Trajectory of UAV i
r_i	Safety radius of UAV i
r_j	Safety radius of UAV j
d_{safe}	Additional safety margin
t_s	Conflict start time
t_e	Conflict end time
Δt	Sampling time resolution for conflict detection
L_i	Path length traveled by UAV i during the conflict interval
k^*	Segment index containing the conflict start time
t_{delay}	Injected temporal delay
α	Delay scaling factor
$t_{\text{min delay}}$	Minimum guard delay
$R_{\max}$	Maximum number of conflict-resolution rounds
Δt_{cd}	Sampling interval used for conflict detection
$\Delta t_{\min}$	Minimum delay in the iterative swarm-planning algorithm
M	Number of segments in the local replanning formulation

1 Introduction

1.1 Motivation and Problem Statement

The popularity and utility of Unmanned Aerial Vehicles (UAVs) have increased significantly in recent years due to their broad range of applications, including package delivery, environmental monitoring, disaster response, and search-and-rescue missions [Unk24]. For such autonomous systems to operate reliably in cluttered or dynamic environments, they must repeatedly generate trajectories that are not only collision-free, but also smooth and dynamically feasible [JCR+21; Wan+22]. This requirement is particularly critical in real-world deployments where the vehicle must react to obstacles, remain within free space, and satisfy physical limits on motion throughout the flight [Wan+22; RBR16].

For a single UAV, the trajectory planning problem can therefore be stated as follows: given an initial state, a target state, and an environment containing obstacles, compute a collision-free trajectory that remains inside a safe region while satisfying dynamic constraints such as bounds on velocity, acceleration, and higher-order motion derivatives [Wan+22; JCR+21]. Although single-UAV navigation is already a challenging problem, the approach is extended to multi-UAV operations by introducing a local temporal resolution strategy, which detects space-time conflicts and resolves them through localized re-optimization and delays [ZLZ22].

1.2 Safe Flight Representation

To ensure safety, the flight environment can be represented by safe flight corridors derived from a voxelized map of free and occupied space [JCR+21]. A global planner first computes a collision-free reference path, which serves as the basis for a convex decomposition of the surrounding free space. The resulting sequence of overlapping convex polyhedra defines a structured safe region for subsequent trajectory optimization [JCR+21; Wan+22].

1.3 Optimization Approaches for Safe Trajectory Planning

This thesis considers two methods for safe trajectory planning within safe flight corridors. The first is a spatio-temporal optimization (STO) approach, in which the spatial trajectory and its time allocation are optimized jointly. The trajectory is represented using the MINCO formulation, which provides parameterization while maintaining smoothness and dynamic feasibility [Wan+22]. The second method is based on mixed-integer quadratic programming (MIQP), where binary decision variables are used to enforce corridor allocation and other discrete planning decisions [JCR+21]. Unlike STO, MIQP-based formulations typically rely on a fixed time allocation in order to keep the problem tractable [JCR+21].

To enable a meaningful comparison, both methods are evaluated under the same corridor representation and a comparable waypoint-based trajectory structure. This makes it possible to highlight their main differences in constraint handling, optimization structure, and time allocation strategy.

1.4 Objective and Scope

The main objective of this thesis is to compare STO and MIQP for UAV safe trajectory planning in terms of trajectory quality, constraint satisfaction, and computational cost [JCR+21; RBR16; Wan+22]. To ensure a fair comparison, both methods are evaluated in the same environment, under the same safe-corridor representation and dynamic limits [JCR+21; Wan+22]. This allows the analysis to focus on the effect of the optimization formulation and solver strategy rather than differences in environmental modeling.

As a secondary contribution, the thesis extends the planning framework to multi-UAV operation in constrained environments, with a particular focus on swarm navigation through narrow passages [Mao+24]. In this setting, individual trajectories are planned within safe flight corridors, while inter-agent conflicts are resolved through a local temporal resolution strategy instead of computationally expensive global replanning [ZLZ22; Mao+24]. This enables safe and scalable coordination in bottleneck regions and other cluttered spaces.

1.5 Thesis Structure

The remainder of this thesis is organized as follows. Chapter 2 reviews the state of the art in UAV path planning and trajectory optimization, with particular emphasis on hierarchical frameworks, safe flight corridors, and multi-UAV coordination in constrained environments. Chapter 3 formulates the trajectory generation problem

and introduces the optimization objectives, constraints, and assumptions underlying the considered single- and multi-UAV scenarios. Chapter 4 presents the proposed methodology, including the STO and MIQP formulations, the hierarchical planning pipeline, and the local temporal conflict resolution strategy for multi-UAV coordination. Chapter 5 presents the experimental results and compares STO and MIQP in terms of trajectory quality, feasibility, and computational cost for both single- and multi-UAV cases, including swarm navigation through narrow passages. Finally, Chapter 6 concludes the thesis and outlines directions for future work.

2 State of Art

2.1 Trajectory Planning for UAVs

Trajectory planning for autonomous vehicles is commonly addressed using search-based, sampling-based, and optimization-based methods. In the UAV literature, sampling-based planners such as PRM, RRT, and RRT* are widely used due to their suitability for high-dimensional spaces and their probabilistic completeness [Wan+22; RBR16]. However, for systems with nontrivial dynamics, these methods can become computationally expensive, particularly when dynamically feasible connections between states must be generated [RBR16].

In optimization-based approaches, a collision-free geometric path is typically generated first and then refined into a smooth and dynamically feasible trajectory using polynomial or sparse parameterizations [RBR16]. A key concept in quadrotor planning is *differential flatness*, which enables trajectory generation from flat outputs and their derivatives, and has formed the basis of many minimum-snap and minimum-control methods [RBR16; Wan+22].

Within this context, the present thesis focuses on optimization-based trajectory generation inside explicitly constructed safe flight corridors, with emphasis on smoothness, dynamic feasibility, and corridor-constrained motion planning [JCR+21; Wan+22].

2.2 Safe Corridors and Convex Decomposition

In cluttered environments, collision-free navigation is often built on occupancy representations such as voxel grids or occupancy maps [JCR+21; RBR16]. Based on a collision-free reference path, the surrounding free space can be represented as a sequence of overlapping convex regions forming a safe flight corridor [JCR+21]. This representation is particularly attractive for trajectory optimization, since safety can be enforced through containment in convex polyhedra rather than direct collision checking against complex obstacles [JCR+21; Wan+22]. As a result, safe flight corridors provide a common geometric basis for both continuous and mixed-integer planning methods.

2.3 Spatio-Temporal and Continuous Trajectory Optimization

A major branch of UAV trajectory planning is based on the joint optimization of spatial and temporal variables. In this thesis, this class of methods is referred to as *Spatio-Temporal Optimization* (STO). By optimizing both path geometry and time allocation together, STO aims to improve smoothness and dynamic feasibility [Wan+22; RBR16].

A representative framework in this category is MINCO (*Minimum Control*), which provides a sparse spatial-temporal trajectory parameterization that is well suited to gradient-based optimization [Wan+22]. Such formulations support smoothness-oriented objectives, including jerk- and snap-related costs, which are widely used in quadrotor trajectory generation [Wan+22; RBR16].

To account for corridor constraints and dynamic limits, continuous optimization methods often use differentiable penalty terms, resulting in an unconstrained or lightly constrained nonlinear program [Wan+22]. This enables efficient solution by gradient-based solvers such as L-BFGS, although strict feasibility is not enforced as directly as in hard-constrained formulations [Wan+22].

2.4 Mixed-Integer Trajectory Planning and MIQP

An alternative approach is to formulate safe trajectory planning as a Mixed-Integer Quadratic Program (MIQP), where binary variables are introduced to represent discrete decisions such as corridor or interval allocation [JCR+21]. When free space is described by overlapping convex polyhedra, MIQP can enforce hard geometric consistency by determining which trajectory intervals are assigned to which polyhedra while optimizing a quadratic motion objective [JCR+21].

A key advantage of MIQP is that safety-related constraints can be imposed explicitly rather than approximated through penalties [JCR+21]. However, this stronger feasibility handling usually comes with higher computational cost, and the problem complexity increases with the number of intervals and corridor primitives [JCR+21]. For this reason, practical MIQP-based planners often assume fixed time allocation and optimize only the discrete interval assignment, since joint optimization of allocation and timing is typically too expensive for real-time replanning [JCR+21].

2.5 Multi-UAV Coordination in Narrow-Passage Environments

Multi-UAV planning in narrow or constrained environments is particularly challenging because agents must not only avoid obstacles, but also coordinate their motion in space and time when traversing shared restricted regions [Mao+24; ZLZ22]. In such

settings, purely geometric planning is often insufficient, since collision-free paths may still produce inter-agent conflicts.

Zhang et al. [ZLZ22] address this problem through Optimal Path and Timetable Planning (OPTP), where near-shortest paths are generated first and then coordinated in path-time space to avoid conflicts. Mao et al. [Mao+24] propose a hierarchical cooperative planning framework for multi-UAV systems that combines global spatio-temporal guidance, local safe corridors, and localized collaborative optimization. Both works emphasize that safe multi-agent navigation in narrow environments requires temporal coordination in addition to geometric feasibility.

These studies show that state-of-the-art methods for multi-agent planning in constrained environments typically rely either on explicit path-time coordination or on hierarchical safe-corridor-based conflict resolution strategies [Mao+24; ZLZ22].

3 Problem Formulation

This chapter defines the mathematical framework used for the trajectory optimization problem and the high-level coordination problem considered in the multi-UAV application. The formulation builds on corridor-based trajectory planning and sparse polynomial trajectory representations commonly used in the UAV literature [JCR+21; Wan+22; RBR16].

3.1 Environment and Safe Flight Corridors

The UAV operates in a discrete three-dimensional environment represented as a voxel grid, where each voxel is classified as either free or occupied according to a chosen map resolution [JCR+21]. A global planner, such as A* or Jump Point Search (JPS), is first used to compute a collision-free reference path connecting the start and goal locations through free space [JCR+21].

To obtain a continuous representation that is suitable for trajectory optimization, convex decomposition is performed along this reference path. As a result, the traversable free space is represented as a sequence of P overlapping convex polytopes. Let the p-th safe corridor be denoted by $\mathcal{C}_p$, where

$$\mathcal{C}_p = \left\{ \mathbf{x} \in \mathbb{R}^3 \;\middle|\; A_p \mathbf{x} \leq b_p \right\}, \qquad p = 1, \ldots, P, \tag{3.1}$$

with $A_p \in \mathbb{R}^{m_p \times 3}$ and $b_p \in \mathbb{R}^{m_p}$. Here, m_p denotes the number of half-space inequalities defining the polytope. The generated trajectory must remain inside the assigned safe corridors in order to guarantee collision-free motion.

3.2 Trajectory Representation

The trajectory is divided into N segments. Each segment i is associated with a strictly positive duration

$$T_i > 0, \qquad i = 1, \ldots, N. \tag{3.2}$$

The total trajectory duration is therefore

$$T = \sum_{i=1}^{N} T_i. \tag{3.3}$$

The geometric path is parameterized by $N+1$ waypoints

$$q_0, q_1, \ldots, q_N \in \mathbb{R}^3, \tag{3.4}$$

where q_0 and q_N denote the start and goal positions, respectively, while $q_1, \ldots, q_{N-1}$ are intermediate waypoints to be optimized.

In addition to positional constraints, the trajectory must satisfy boundary conditions on position, velocity, and acceleration at the initial and terminal times. Let

$$p(0) = q_0, \qquad p(T) = q_N, \tag{3.5}$$

and let the corresponding boundary velocity and acceleration values be specified according to the mission requirements. In many cases, the UAV is required to start and end at rest, i.e.,

$$\dot{p}(0) = \dot{p}(T) = 0, \qquad \ddot{p}(0) = \ddot{p}(T) = 0. \tag{3.6}$$

In this work, the trajectory is represented using the MINCO (*Minimum Control*) framework [Wan+22]. For jerk minimization, each segment is modeled as a fifth-order polynomial. A key property of MINCO is that the polynomial coefficients can be expressed analytically as functions of the waypoint variables, segment durations, and boundary derivatives. Consequently, the optimization can be carried out directly over waypoints and segment times instead of raw polynomial coefficients. Furthermore, continuity of position, velocity, acceleration, jerk, and snap is enforced at all intermediate waypoints, yielding a smooth piecewise-polynomial trajectory

3.3 Dynamics and Constraints

To ensure that the generated motion is dynamically feasible, the trajectory must satisfy bounds on velocity and acceleration throughout the full time horizon:

$$\|\dot{p}(t)\| \le v_{\max}, \qquad t \in [0, T], \tag{3.7}$$

$$\|\ddot{p}(t)\| \le a_{\max}, \qquad t \in [0, T]. \tag{3.8}$$

In addition, the safe corridor requirement imposes geometric containment constraints. If trajectory segment i is assigned to corridor $\mathcal{C}_{\sigma(i)}$, where $\sigma(i)$ denotes the corridor index associated with that segment, then the position must satisfy

$$A_{\sigma(i)}\, p(t) \le b_{\sigma(i)}, \qquad t \in \left[\sum_{j=1}^{i-1} T_j,\ \sum_{j=1}^{i} T_j\right]. \tag{3.9}$$

Thus, each segment is required to remain within its assigned convex polyhedron for its entire duration. Together with the boundary and continuity conditions introduced previously, these constraints define the feasible trajectory set.

Although jerk is not imposed as an explicit hard bound in the present formulation, it is accounted for in the objective function in order to promote smooth motion.

3.4 Objective Function

The primary objective is to minimize the total jerk effort over the trajectory duration. Let $p^{(3)}(t)$ denote the third derivative of the position with respect to time. Then, the smoothness cost is defined as

$$J_{\text{jerk}} = \int_0^T \left\| p^{(3)}(t) \right\|^2 dt. \tag{3.10}$$

In the STO setting, an additional penalty on the total flight time is introduced in order to balance trajectory smoothness against mission speed. The complete objective is then written as

$$J = J_{\text{jerk}} + \lambda_{\text{time}} T, \tag{3.11}$$

where $\lambda_{\text{time}} > 0$ is a weighting parameter. In addition, for STO, this objective is extended with soft penalties for violations of velocity, acceleration, and corridor constraints; the detailed formulation is presented in the methodology chapter. Within the MINCO framework, this objective is expressed as a function of the waypoint variables and segment durations, which enables efficient gradient-based optimization [Wan+22].

3.5 Summary of the Trajectory Optimization Problem

At a high level, the trajectory optimization problem considered in this thesis seeks a smooth and dynamically feasible trajectory $p(t)$ that connects the given start and goal states while remaining inside the safe flight corridor. In compact form, the problem can be written as

$$\begin{aligned}
\min \quad & J \\
\text{s.t.} \quad & \|\dot{p}(t)\| \le v_{\max}, \\
& \|\ddot{p}(t)\| \le a_{\max}, \\
& p(t) \in \mathcal{P}(t), \\
& p(0),\ \dot{p}(0),\ \ddot{p}(0) \text{ satisfy the start conditions}, \\
& p(T),\ \dot{p}(T),\ \ddot{p}(T) \text{ satisfy the goal conditions}, \\
& p(t) \text{ satisfies the required continuity conditions}.
\end{aligned} \tag{3.12}$$

Here, J denotes the trajectory cost, which is primarily based on jerk minimization and, in the STO case, additionally includes penalties on flight time and on constraint violations (velocity, acceleration, and corridor). This common problem statement provides the basis for both the STO and MIQP approaches studied in this thesis. The detailed formulation of each method, including their respective decision variables and constraint handling mechanisms, is presented in the methodology chapter.

3.6 Multi-UAV Conflict Detection and Temporal Deconfliction

In the multi-UAV setting considered in this thesis, each UAV first receives an individually feasible corridor-guided trajectory generated by the single-UAV planning pipeline described in the previous sections. However, even if every trajectory is collision-free with respect to static obstacles, conflicts may still arise between UAVs when their safety volumes overlap in space and time. This issue is particularly critical in narrow-passage environments, where multiple UAVs may be forced to traverse the same corridor-like region [Mao+24; ZLZ22].

For this reason, the multi-UAV coordination problem is treated here as a *spatio-temporal conflict detection and resolution* problem. Rather than solving a fully centralized joint optimization over all UAV trajectories, the approach adopted in this thesis follows a decoupled strategy: trajectories are first planned independently, pairwise conflicts are then detected, and only the affected trajectories are locally re-optimized [Mao+24; ZLZ22].

3.6.1 Conflict Detection

Let the trajectory of UAV i be denoted by

$$\pi_i : [0, T_i] \to \mathbb{R}^3, \tag{3.13}$$

where T_i is the total duration of the trajectory. A pairwise conflict between UAV i and UAV j over a time interval $[t_s, t_e]$ is declared if

$$\|\pi_i(t) - \pi_j(t)\| < r_i + r_j + d_{\text{safe}}, \qquad \forall t \in [t_s, t_e], \tag{3.14}$$

where r_i and r_j denote the safety radii of the two UAVs, and $d_{\text{safe}} \geq 0$ is an additional configurable safety margin.

In practice, conflict detection is performed by sampling the trajectories at a fixed time resolution Δt and checking the pairwise Euclidean distance at each sample instant. For each UAV pair, only the earliest detected conflict event is retained for the subsequent resolution step. This design avoids redundant handling of multiple overlapping violations that belong to the same interaction episode.

3.6.2 Winner–Loser Assignment

For every detected conflict event

$$(i, j, t_s, t_e), \tag{3.15}$$

a winner and a loser are assigned in order to determine which trajectory remains unchanged and which one is re-timed.

The decision is based on the path length traveled by each UAV during the conflict window. In practice, this quantity is approximated by sampling the trajectories at the detection time resolution Δt and summing the Euclidean distances between consecutive sample points. For UAV i, the resulting path length over $[t_s, t_e]$ is computed as

$$L_i = \sum_{k=0}^{K-1} \left\| p_i(t_s + (k+1)\Delta t) - p_i(t_s + k\Delta t) \right\|, \tag{3.16}$$

where K is chosen such that $t_s + K\Delta t \approx t_e$. The same definition applies to UAV j.

The UAV with the greater path length over the conflict interval is designated as the *winner*, while the other UAV is designated as the *loser*. Intuitively, this corresponds to assigning priority to the UAV that traverses the conflict zone more aggressively or efficiently.

Once a winner–loser assignment has been established for a given UAV pair, it is kept fixed in later resolution rounds. This prevents oscillatory behavior in which the same pair repeatedly exchanges priority across iterations.

3.6.3 Temporal Deconfliction by Delay Injection

Let the loser trajectory belong to UAV i, and let k^* denote the segment index that contains the conflict start time t_s. A temporal delay t_{delay} is introduced in order to postpone the arrival of the loser UAV at the shared conflict region. In constrained multi-agent environments, temporal separation provides an efficient alternative to complete spatial replanning when multiple agents must traverse shared regions [ZLZ22]. The delay is computed as

$$t_{\text{delay}} = \max\big(\alpha(t_e - t_s),\ t_{\text{mindelay}}\big), \tag{3.17}$$

where $\alpha \in (0, 1]$ is a user-defined scaling factor and $t_{\text{mindelay}} > 0$ is a minimum guard delay.

Instead of assigning the entire delay to a single segment, the delay is distributed proportionally across all segments from the beginning of the trajectory up to segment k^*. Let the original segment durations be $T_{i,0}, \ldots, T_{i,k^*}$. The updated durations are then

defined as

$$\tilde{T}_{i,k} = T_{i,k} + t_{\text{delay}} \frac{T_{i,k}}{\sum_{\ell=0}^{k^*} T_{i,\ell}}, \qquad k = 0, \ldots, k^*. \tag{3.18}$$

This proportional distribution avoids concentrating the full delay on a short interval, which could otherwise create excessive velocity or acceleration changes at segment junctions. As a result, the temporal perturbation remains smoother and better suited for subsequent trajectory re-optimization.

3.6.4 Frozen-Segment Re-optimization

A direct re-optimization after delay injection may undo the intended temporal separation, since the optimizer may reduce the added delay in order to improve the time-related objective. To prevent this effect, this thesis introduces a *frozen-segment re-optimization* strategy. Let the optimization variables of UAV i consist of waypoint variables c_i and time-allocation variables τ_i, where τ_i defines the duration of each segment.

- the time-allocation variables corresponding to the pre-conflict segments $0, \ldots, k^*$ are *frozen*,
- the time-allocation variables of the post-conflict segments remain *free*,
- the pre-conflict interior waypoints are allowed to move,
- the post-conflict waypoints are kept fixed in order to preserve the original route structure.

More precisely, the frozen pre-conflict durations are enforced as

$$\tau_{i,0:k^*} = \tilde{\tau}_{i,0:k^*}, \tag{3.19}$$

while the post-conflict time variables $\tau_{i,k^*+1:M}$ remain optimization variables. Similarly, the post-conflict waypoint variables are fixed, whereas the pre-conflict waypoint variables remain free to adapt spatially.

The resulting local re-optimization problem can therefore be written in the abstract form

$$\min_{c_{i,0:k^*-1},\, \tau_{i,k^*+1:M}} \lambda_j J_{\text{jerk}} + \lambda_T J_{\text{time}} + \lambda_v J_v + \lambda_a J_a + \lambda_c J_c, \tag{3.20}$$

subject to

$$\tau_{i,0:k^*} = \tilde{\tau}_{i,0:k^*}, \qquad c_{i,k^*:M-1} \text{ fixed}, \tag{3.21}$$

together with the same dynamic and corridor constraints introduced for the single-UAV STO formulation.

Here, J_{jerk} denotes the smoothness cost, J_{time} the time-related cost, J_v and J_a the penalties associated with velocity and acceleration violations, and J_c the corridor feasibility penalty. The weighting parameters λ_j, λ_T, λ_v, λ_a, and λ_c balance these terms.

This formulation has three important consequences. First, the injected delay is preserved exactly, because the corresponding pre-conflict time variables are no longer optimizable. Second, pre-conflict waypoints remain adjustable, allowing the optimizer to recover corridor compliance and kinematic feasibility after the temporal stretching. Third, the post-conflict route structure is preserved, so the UAV continues toward the same goal through the same overall spatial path, but at a later time.

3.6.5 Iterative Conflict Resolution Loop

In the multi-UAV setting, independently planned trajectories are not guaranteed to be mutually conflict-free. To address this, inter-UAV conflicts are resolved through an iterative detect–resolve strategy based on pairwise conflict detection and local temporal replanning.

At each resolution round, the current trajectories are re-evaluated for pairwise conflicts. The retained conflict events are then processed sequentially, and the affected trajectories are updated accordingly. Since resolving one conflict may create or expose another, this procedure is repeated until no pairwise conflicts remain or a prescribed maximum number of rounds $R_{\max}$ is reached.

The detailed algorithmic realization of this iterative coordination procedure is presented in the methodology chapter.

4 Methodology

This chapter describes the solution methodology used in this thesis. First, a common setup is established to ensure a fair comparison between the Spatio-Temporal Optimization (STO) and Mixed-Integer Quadratic Programming (MIQP) approaches. Then, the two formulations are presented in detail, followed by the multi-UAV pipeline used in the secondary application. The methodology is built on corridor-based planning, MINCO-based trajectory representation, and optimization strategies inspired by existing work on safe UAV trajectory generation [JCR+21; Wan+22; RBR16].

4.1 Common Setup for STO and MIQP

To enable a fair comparison, STO and MIQP are evaluated under the same problem setup defined in Chapter 3. Both methods use the same three-dimensional voxel map, the same collision-free reference path and corresponding safe flight corridor sequence, and the same MINCO-based trajectory representation [JCR+21; Wan+22]. In addition, the initial and terminal boundary conditions, as well as the dynamic limits on velocity and acceleration, are identical for both formulations.

The number of polynomial segments N (and thus the number of interior waypoints) is chosen to be the same for STO and MIQP, so that both optimizers operate on an equally refined piecewise trajectory. For MIQP, segment durations are fixed; to keep the temporal horizon comparable to STO, the total flight time used for MIQP is set to the total duration obtained from the STO solution, with a uniform time allocation per segment (unless stated otherwise in the experiments).

Therefore, the two approaches differ only in the optimization formulation itself, namely in how segment times are handled (optimized versus fixed), how constraints are enforced (soft penalties versus hard constraints), and which solver structure is used (gradient-based versus mixed-integer programming). This ensures that the comparison reflects methodological differences rather than differences in environment modeling, trajectory discretization, or boundary assumptions.

4.2 Spatio-Temporal Optimization (STO)

4.2.1 Decision Variables and Time Reparameterization

In the STO formulation, the optimization variables consist of the intermediate waypoint coordinates and the segment-time parameters. The waypoint variables are

$$q = \{q_1, \ldots, q_{N-1}\}, \qquad q_i \in \mathbb{R}^3, \tag{4.1}$$

while the start and goal waypoints are fixed.

Instead of optimizing the segment durations $T_i > 0$ directly, STO introduces unconstrained auxiliary variables $\tau_i \in \mathbb{R}$, $i = 1, \ldots, N$, and maps them to positive segment times by

$$T_i = \exp(\tau_i). \tag{4.2}$$

This guarantees $T_i > 0$ by construction and avoids explicit positivity constraints during optimization. If an initial time allocation is given, the corresponding auxiliary variables are obtained from

$$\tau_i = \log(T_i). \tag{4.3}$$

Thus, the STO solver optimizes the variable pair

$$(q, \tau), \tag{4.4}$$

whereas the actual segment durations used in trajectory generation are recovered through $T_i = \exp(\tau_i)$. This reparameterization is specific to STO; in the MIQP formulation, segment durations are fixed.

4.2.2 Objective and Soft Constraints

The STO formulation minimizes a composite objective consisting of a smoothness term, a total-time term, and soft penalty terms:

$$L = \lambda_{\text{jerk}} J + \lambda_{\text{time}} T_{\text{total}} + \lambda_c J_c + \lambda_v J_v + \lambda_a J_a, \tag{4.5}$$

where

$$J = \int_0^T \|p^{(3)}(t)\|^2 \, dt \tag{4.6}$$

is the jerk cost,

$$T_{\text{total}} = \sum_{i=1}^{N} T_i \tag{4.7}$$

is the total trajectory duration, and $\lambda_{\text{jerk}}, \lambda_{\text{time}}, \lambda_c, \lambda_v$, and λ_a are fixed weighting coefficients.

The penalty terms are evaluated at K sampled time instants $t_1, \ldots, t_K$ along the trajectory. For a scalar quantity x, the positive-part operator is defined as

$$[x]_+ = \max(0, x), \tag{4.8}$$

so that only actual constraint violations contribute to the penalty.

Velocity penalty. The velocity penalty is defined as

$$J_v = \frac{1}{K} \sum_{k=1}^{K} \left[\|\dot{p}(t_k)\| - v_{\max} \right]_+^2, \tag{4.9}$$

which penalizes violations of the maximum velocity bound.

Acceleration penalty. The acceleration penalty is defined as

$$J_a = \frac{1}{K} \sum_{k=1}^{K} \left[\|\ddot{p}(t_k)\| - a_{\max} \right]_+^2, \tag{4.10}$$

which penalizes violations of the maximum acceleration bound.

Corridor penalty. For corridor containment, let $\sigma(i)$ denote the safe corridor assigned to segment i, with half-space representation

$$\mathcal{C}_{\sigma(i)} = \{ x \in \mathbb{R}^3 \mid A_{\sigma(i)} x \leq b_{\sigma(i)} \}. \tag{4.11}$$

At sampled time t_k, let i_k denote the active segment index and define the corridor violation vector as

$$d_k = A_{\sigma(i_k)} p(t_k) - b_{\sigma(i_k)}. \tag{4.12}$$

Three corridor penalty types are implemented and compared in this thesis.

The L1 corridor penalty is

$$J_c^{\mathrm{L1}} = \frac{1}{K} \sum_{k=1}^{K} \mathbf{1}^\top [d_k]_+, \tag{4.13}$$

the L2 corridor penalty is

$$J_c^{\mathrm{L2}} = \frac{1}{K} \sum_{k=1}^{K} \| [d_k]_+ \|^2, \tag{4.14}$$

and the logarithmic corridor penalty is

$$J_c^{\log} = \frac{1}{K} \sum_{k=1}^{K} \sum_{j=1}^{m} \log\left(1 + [d_{k,j}]_+\right). \tag{4.15}$$

In each optimization run, one of these corridor penalty types is selected:

$$J_c \in \left\{ J_c^{\mathrm{L1}},\ J_c^{\mathrm{L2}},\ J_c^{\log} \right\}. \tag{4.16}$$

These alternatives are compared experimentally to study how different penalty growth behaviors affect optimization performance and feasibility.

Soft-constraint interpretation. Rather than enforcing corridor and dynamic constraints as hard requirements, STO incorporates them into the objective through penalty terms. As a result, feasibility is encouraged during optimization, but it is not guaranteed at every iteration.

Adaptive weights. An adaptive weighting strategy is used in which the penalty coefficients λ_v, λ_a, and λ_c are initialized with small values and increased over successive optimization phases. This progressively drives the solution toward feasibility while preserving the efficiency of unconstrained gradient-based optimization.

4.2.3 Solver and Implementation

The STO problem is solved using L-BFGS on the unconstrained variable pair (q, τ). Since both the MINCO mapping and the time reparameterization $T_i = \exp(\tau_i)$ are differentiable, the full objective can be optimized efficiently using gradient information [Wan+22]. The optimization output consists of the intermediate waypoints, the corresponding segment durations, and the resulting smooth polynomial trajectory.

Because corridor and dynamic constraints are handled through soft penalties, feasibility is encouraged during optimization but is not guaranteed a priori.

4.3 MIQP Formulation

4.3.1 Decision Variables

The MIQP formulation uses the same waypoint-based trajectory parameterization as STO, supplemented by binary variables for corridor assignment.

Continuous Variables The continuous optimization variables are the intermediate waypoints

$$q_1, \ldots, q_{N-1} \in \mathbb{R}^3. \tag{4.17}$$

Since the segment durations are fixed, the polynomial coefficients are not treated as independent decision variables; they are obtained analytically via the MINCO mapping from the boundary conditions and waypoint positions.

Binary Variables For corridor allocation, binary variables

$$z_{i,p} \in \{0, 1\} \tag{4.18}$$

are introduced, where $z_{i,p} = 1$ indicates that segment i is assigned to corridor p. For each segment, the corridor assignment is constrained by

$$\sum_{p=1}^{P} z_{i,p} \geq 1, \qquad i = 1, \ldots, N. \tag{4.19}$$

Time Handling Unlike STO, the MIQP formulation does not include segment times as optimization variables. Instead, they are prescribed in advance:

$$T_i = \bar{T}_i, \qquad i = 1, \ldots, N. \tag{4.20}$$

In the implementation considered here, a uniform duration is used for all segments, so no τ-based time reparameterization is required.

4.3.2 Objective Function

For fixed segment durations, the jerk-minimization objective can be written as a quadratic function of the waypoint variables through the MINCO mapping. Accordingly, the MIQP formulation minimizes the same smoothness cost as the STO formulation:

$$J = \int_0^T \|p^{(3)}(t)\|^2 \, dt. \tag{4.21}$$

Thus, both methods are evaluated under the same motion-quality criterion, differing only in their treatment of time allocation and constraint enforcement.

4.3.3 Hard Constraints via Control Points

A key feature of the MIQP formulation is that corridor and dynamic feasibility are enforced as hard constraints through control-point representations of each polynomial segment. Position feasibility is imposed using the Bezier control points of the segment, while velocity and acceleration feasibility are enforced through derivative control points derived from the same representation.

Corridor Constraints Each fifth-order polynomial segment is converted into an equivalent Bezier form with six position control points,

$$r_{i,0}, r_{i,1}, \ldots, r_{i,5} \in \mathbb{R}^3. \tag{4.22}$$

If segment i is assigned to corridor $\mathcal{C}_p$, all six control points must satisfy the half-space inequalities defining that corridor:

$$A_p r_{i,k} \leq b_p, \qquad k = 0, \ldots, 5. \tag{4.23}$$

These constraints are activated through the binary assignment variables $z_{i,p}$.

This construction relies on the convex hull property of Bezier curves: the entire polynomial segment lies inside the convex hull of its control points. Therefore, if all position control points lie inside the assigned convex polyhedron, the whole segment remains inside that corridor.

Velocity Constraints The velocity curve is represented by five derivative control points,

$$v_{i,0}, v_{i,1}, \ldots, v_{i,4} \in \mathbb{R}^3, \tag{4.24}$$

which are obtained from the Bezier control points as

$$v_{i,k} = \frac{5}{T_i} \left(r_{i,k+1} - r_{i,k} \right), \qquad k = 0, \ldots, 4. \tag{4.25}$$

The velocity bound is enforced at all derivative control points:

$$\| v_{i,k} \|^2 \leq v_{\max}^2, \qquad k = 0, \ldots, 4. \tag{4.26}$$

Acceleration Constraints Similarly, the acceleration curve is represented by four derivative control points,

$$a_{i,0}, a_{i,1}, \ldots, a_{i,3} \in \mathbb{R}^3, \tag{4.27}$$

with

$$a_{i,k} = \frac{4}{T_i} \left(v_{i,k+1} - v_{i,k} \right) = \frac{20}{T_i^2} \left(r_{i,k+2} - 2 r_{i,k+1} + r_{i,k} \right), \qquad k = 0, \ldots, 3. \tag{4.28}$$

The acceleration bound is imposed as

$$\| a_{i,k} \|^2 \leq a_{\max}^2, \qquad k = 0, \ldots, 3. \tag{4.29}$$

In the implementation, dynamic feasibility is enforced through quadratic Euclidean norm constraints rather than separate per-axis bounds. Thus, the MIQP formulation guarantees hard satisfaction of the corridor, velocity, and acceleration constraints, up to solver tolerance. Jerk is not imposed as an explicit hard constraint, but is instead minimized through the quadratic objective.

4.3.4 Solver and Implementation

The resulting optimization problem is solved with Gurobi as a mixed-integer quadratic program. It includes continuous waypoint variables, binary corridor assignment

variables, a quadratic jerk objective, indicator constraints for corridor membership, and quadratic norm constraints for velocity and acceleration.

To limit runtime, a solver time limit is defined. If a feasible solution is found, corridor and dynamic feasibility are guaranteed by construction, subject only to solver tolerances and the correctness of the control-point-based formulation.

The final output consists of the optimized intermediate waypoints, the corridor assignment variables, and the corresponding piecewise-polynomial trajectory reconstructed through the MINCO mapping.

4.4 Comparison Summary

The main methodological differences between STO and MIQP are summarized in Table 4.1.

Table 4.1: Methodological comparison between STO and MIQP.

Aspect	**STO**	**MIQP**
Decision variables	Waypoints q, time parameters τ	Waypoints q, binary assignment variables $z_{i,p}$; time fixed
Time handling	Optimized via $T_i = \exp(\tau_i)$	Fixed segment durations
Constraint treatment	Soft penalties for corridor, velocity, acceleration	Hard corridor constraints via position Bezier control points; hard velocity and acceleration constraints via derivative control points
Solver	L-BFGS (gradient-based)	Gurobi MIQP solver
Feasibility	No strict guarantee; depends on penalties	Guaranteed if a feasible solution is returned

4.5 Swarm Planning Methodology

As a secondary application, the STO framework is extended to a multi-UAV setting to generate coordinated trajectories through a shared constrained environment, including narrow passages. Instead of solving a single joint optimization problem over all UAVs, the proposed planner follows a sequential and modular strategy. Each UAV is first assigned an individually feasible trajectory using the single-UAV pipeline introduced earlier, after which inter-UAV conflicts are resolved through an iterative local replanning procedure.

4.5.1 Overview of the Pipeline

The swarm planner consists of two stages. First, an initial trajectory is generated independently for each UAV using the single-UAV planning procedure described in the previous sections, including path search, corridor construction, and STO-based trajectory refinement. Second, the resulting trajectories are checked for pairwise conflicts and, if necessary, locally adjusted through iterative coordination.

The coordination stage consists of

1. pairwise conflict detection,
2. winner–loser assignment for each detected conflict,
3. local replanning of the loser trajectory while previously accepted trajectory segments remain fixed.

These steps are repeated until no conflicts remain or a prescribed maximum number of coordination rounds is reached.

4.5.2 Initial Per-UAV Trajectories

For each UAV, an initial collision-free trajectory is generated using the single-UAV planning pipeline presented earlier. The environment is discretized as a three-dimensional voxel grid with inflated obstacles, a collision-free reference path is computed by A* and pruned, convex safe corridors are extracted along the path, and STO is then used to obtain a smooth dynamically feasible trajectory.

An initial segment-time allocation is computed from the segment lengths and the maximum allowable speed:

$$T_{i,k}^{(0)} = \frac{\|p_{i,k+1} - p_{i,k}\|}{v_{i,\max}}, \qquad k = 1, \ldots, M_i. \tag{4.30}$$

At the end of this stage, each UAV has a trajectory that is feasible with respect to the static environment, but not necessarily conflict-free with respect to the other UAVs.

4.5.3 Conflict Detection

After all individual STO runs are completed, the trajectories are checked pairwise for inter-UAV proximity violations. Let the trajectory of UAV i be denoted by

$$\pi_i : [0, T_i] \to \mathbb{R}^3. \tag{4.31}$$

For each pair (i, j), the trajectories are resampled at a fixed temporal resolution Δt_{cd}, and the pairwise distance is evaluated at every sample instant:

$$d_{ij}(t_k) = \|\pi_i(t_k) - \pi_j(t_k)\|. \tag{4.32}$$

A conflict event is declared when

$$d_{ij}(t) < r_i + r_j + d_{\text{safe}} \tag{4.33}$$

over a contiguous time interval $[t_s, t_e]$, where r_i and r_j denote the safety radii of the two UAVs and d_{safe} is an additional configurable safety margin. For each unordered UAV pair, only the earliest conflict event is retained. The retained events are then sorted by their start times so that the most imminent conflicts are handled first.

4.5.4 Winner–Loser Assignment

For each retained conflict (i, j, t_s, t_e), one UAV is designated as the winner and the other as the loser. The winner keeps its trajectory unchanged, whereas the loser is selected for local replanning.

The assignment is based on the arc length traveled during the conflict interval. For UAV i, this quantity is approximated by

$$L_i = \sum_{t_k \in [t_s, t_e]} \|\pi_i(t_{k+1}) - \pi_i(t_k)\|, \tag{4.34}$$

and analogously for UAV j. The UAV with the larger traveled distance is assigned as the winner, while the other UAV becomes the loser.

To avoid oscillatory behavior across successive coordination rounds, the winner–loser assignment is fixed after the first decision for each UAV pair and reused in subsequent iterations.

4.5.5 Temporal Delay Injection

Let UAV i be the loser for a given conflict event, and let k^* denote the segment containing the conflict start time t_s. To postpone the loser's arrival at the shared conflict region, a temporal delay is introduced:

$$\Delta t = \max(\alpha(t_e - t_s),\ \Delta t_{\min}), \tag{4.35}$$

where $\alpha \in (0, 1]$ is a scaling factor and $\Delta t_{\min} > 0$ is a minimum delay.

Rather than assigning the full delay to a single segment, it is distributed proportionally over all segments from the start of the trajectory up to segment k^*. If the current segment durations are $T_{i,0}, \ldots, T_{i,k^*}$, the updated durations are

$$\tilde{T}_{i,k} = T_{i,k} \left(1 + \frac{\Delta t}{\sum_{\ell=0}^{k^*} T_{i,\ell}} \right), \qquad k = 0, \ldots, k^*. \tag{4.36}$$

This preserves the total added delay while avoiding excessive stretching of a single segment.

4.5.6 Frozen-Segment Re-optimization

After delay injection, the loser trajectory is re-optimized using a partially frozen STO formulation. The objective is to preserve the imposed temporal separation while allowing limited adaptation of the trajectory where necessary.

Let the STO decision variables for UAV i be the waypoint variables c_i and the unconstrained time variables τ_i. The re-optimization follows a partial freeze strategy:

- the pre-conflict time variables $\tau_{i,0:k^*}$ are fixed,
- the post-conflict time variables $\tau_{i,k^*+1:M}$ remain free,
- the pre-conflict waypoint variables $c_{i,0:k^*-1}$ remain free,
- the post-conflict waypoint variables $c_{i,k^*:M-1}$ are fixed.

Fixing the pre-conflict time variables ensures that the injected delay cannot be eliminated by the optimizer. At the same time, keeping the pre-conflict waypoints free allows local spatial adjustments if the modified timing affects corridor or dynamic feasibility. The post-conflict waypoint variables are fixed so that the overall downstream route toward the goal is preserved.

The resulting local replanning problem is

$$\min_{c_{i,0:k^*-1},\, \tau_{i,k^*+1:M}} \lambda_j J_{\text{jerk}} + \lambda'_T J_{\text{time}} + \lambda_v J_v + \lambda_a J_a + \lambda_c J_c, \tag{4.37}$$

subject to

$$\tau_{i,0:k^*} = \tilde{\tau}_{i,0:k^*}, \qquad c_{i,k^*:M-1} \text{ fixed}. \tag{4.38}$$

Here, λ'_T is chosen smaller than the nominal time weight used in the original STO problem. This discourages the optimizer from excessively shortening the remaining free segments to compensate for the added delay.

Algorithm 1 Iterative conflict resolution for multi-UAV coordination

Require: Initial trajectories $\{\pi_i\}_{i=1}^{n}$, coordination parameters $(\Delta t_{\text{cd}}, \alpha, \Delta t_{\min}, R_{\max})$
Ensure: Updated trajectories $\{\pi_i\}_{i=1}^{n}$

```
Detect pairwise conflicts on {π_i} using sampling interval Δt_cd
Retain the earliest conflict for each unordered UAV pair
r ← 0
while r < R_max do
    if no conflict events remain then
        return {π_i}
    end if
    Sort the retained conflict events by start time
    for each conflict (i, j, t_s, t_e) in order do
        Assign winner and loser
        Compute the delay Δt for the loser trajectory
        Determine the conflict segment k* and distribute the delay over the pre-conflict segments
        Re-optimize the loser trajectory using frozen-segment STO
        Update the corresponding trajectory in {π_i}
    end for
    Re-detect pairwise conflicts on the updated trajectories
    Retain the earliest conflict for each unordered UAV pair
    r ← r + 1
end while
return {π_i}
```

4.5.7 Iterative Detect–Resolve Loop

Since resolving one conflict may create or expose another, conflict detection and local replanning are performed iteratively. In each round, all current trajectories are checked again, the earliest conflict for each UAV pair is retained, and the resulting conflict events are processed sequentially in temporal order.

Let $R_{\max}$ denote the maximum number of coordination rounds. The procedure terminates when either no conflict events remain or the number of rounds reaches $R_{\max}$. The overall detect–resolve procedure is summarized in Algorithm 1.

This iterative detect–resolve loop provides a scalable alternative to centralized joint multi-UAV optimization. Only trajectories involved in detected conflicts are modified, while all other trajectories remain unchanged. The approach is therefore particularly suitable for narrow-passage scenarios, where conflicts are typically localized in space

but critical in time.

5 Results

5.1 Single-UAV Results

5.1.1 Environment, Global Path, and Safe Corridor Construction

Figure 5.1 shows the environment used in the first set of single-UAV experiments. The scenario is based on the `forest` map, whose obstacle layout creates a constrained flight space and thus provides a suitable testbed for corridor-based trajectory planning.

A collision-free reference path is first computed using A* on the voxelized environment and then simplified by pruning redundant intermediate nodes. The retained key waypoints are subsequently connected by straight-line segments, yielding a compact geometric path.

Based on this simplified path, a sequence of safe flight corridors is constructed. These corridors define locally convex regions inside free space that contain the corresponding path segments. As shown in the figure, the resulting pipeline consists of the environment, the A* reference path, the simplified waypoint-to-waypoint path, and the extracted safe corridors. The same corridor sequence is used as the geometric input for both STO and MIQP, ensuring a consistent basis for the subsequent comparison.

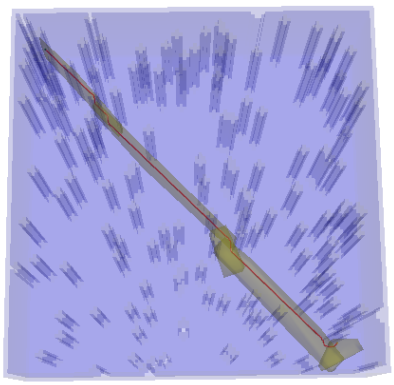

Figure 5.1: The `forest` environment, the A* reference path, the simplified waypoint-to-waypoint path, and the generated safe flight corridors used in the single-UAV experiments.

5.1.2 Comparison of Corridor Penalty Functions

To assess the sensitivity of the STO formulation to the choice of corridor penalty, three variants were evaluated: a quadratic L_2 penalty, a linear L_1 penalty, and a logarithmic `log1p`-type penalty. All experiments were conducted on the same environment with the same initial path, safe corridor structure, and dynamic limits, so that the observed differences are attributable only to the penalty design.

Table 5.1 summarizes the resulting runtime, jerk cost, total trajectory time, and maximum violations of the velocity, acceleration, and corridor constraints. Among the tested variants, the L_2 penalty achieved the lowest jerk cost and satisfied both the velocity and acceleration limits, while producing only a small residual corridor violation. The L_1 penalty reduced runtime, but led to both a larger corridor violation and a nonzero velocity violation. The logarithmic penalty eliminated corridor violation completely, but still produced a nonzero velocity violation and a higher jerk cost than the L_2 case.

Based on these results, the quadratic L_2 corridor penalty was selected for the subsequent STO–MIQP comparison. It provided the best trade-off between smoothness and feasibility, while maintaining runtime and total trajectory time at levels comparable to the other penalty choices.

Table 5.1: Comparison of different corridor penalty functions in the STO formulation.

Method	**Runtime (s)**	**Jerk Cost**	**Total T (s)**	**Vel. Viol.**	**Acc. Viol.**	**Corr. Viol.**
L2 (quadratic)	9.224	0.012197	84.220	0.000000	0.000000	0.045755
L1 (linear)	7.298	0.016305	83.507	0.053979	0.000000	0.311096
Log (log1p)	7.158	0.014189	83.660	0.023731	0.000000	0.000000

5.1.3 Comparison Between STO and MIQP

After selecting the quadratic L_2 corridor penalty for STO, the final single-UAV comparison between STO and MIQP was conducted on the same planning instance, using the same environment, safe corridor structure, boundary conditions, and dynamic limits. To ensure a fair comparison, both methods were evaluated with the same total trajectory time.

Table 5.2 summarizes the main performance metrics. Both methods produced trajectories with the same total duration of 77.56 s, confirming that the comparison is not affected by different mission-level time allocations. Both trajectories are smooth, as reflected by their low jerk costs, but STO achieved a substantially lower jerk cost than MIQP while requiring less than half of the runtime.

The constraint-violation results are reported in Table 5.3. As expected, MIQP satisfied the velocity, acceleration, and corridor constraints exactly in this experiment, consistent with its hard-constrained formulation. STO, in contrast, produced small residual violations in velocity and corridor containment while still satisfying the acceleration constraint. These violations remained very small relative to the scale of the environment and the overall trajectory.

The geometric and kinematic differences between the two solutions are illustrated in Figures 5.2, 5.3, and 5.4. The 3D and orthogonal views show that both methods generate very similar corridor-following trajectories between the same start and goal states. The kinematic comparison further confirms that the overall motion profiles are nearly identical, despite the different optimization formulations.

Overall, the results show the expected trade-off between the two methods. MIQP guarantees strict feasibility, but at the cost of significantly higher computation time. STO, in contrast, achieves comparable trajectory quality and the same total trajectory time with substantially lower runtime, at the expense of only minor residual violations. Since the multi-UAV swarm extension considered in this thesis requires repeated local replanning during conflict resolution, computational efficiency is especially important. For this reason, STO was selected as the trajectory generation method for the swarm experiments.

Table 5.2: Main comparison between STO and MIQP for the single-UAV trajectory planning problem.

Method	**Runtime (s)**	**Jerk Cost**	**Traj. Time (s)**
STO	13.977	0.011722	77.56
MIQP	32.221	0.038899	77.56

Table 5.3: Maximum constraint violations for STO and MIQP.

Method	**Velocity**	**Acceleration**	**Jerk**	**Corridor**
STO	0.012752	0.000000	0.000000	0.007118
MIQP	0.000000	0.000000	0.000000	0.000000

5.1.4 Swarm Environment, Initial Paths, and Safe Corridors

The multi-UAV experiments were conducted in a constrained environment designed to create a shared narrow-passage region between the start and goal sides of the workspace.

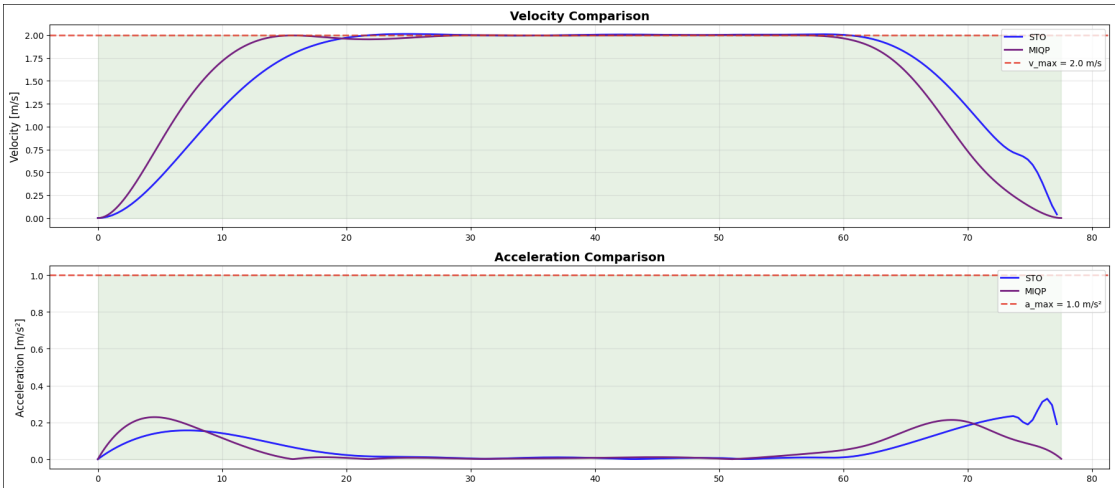

Figure 5.2: Velocity and acceleration profiles of the STO and MIQP trajectories. Both methods remain close in overall motion behavior, while MIQP strictly satisfies the imposed limits and STO exhibits only small residual violations.

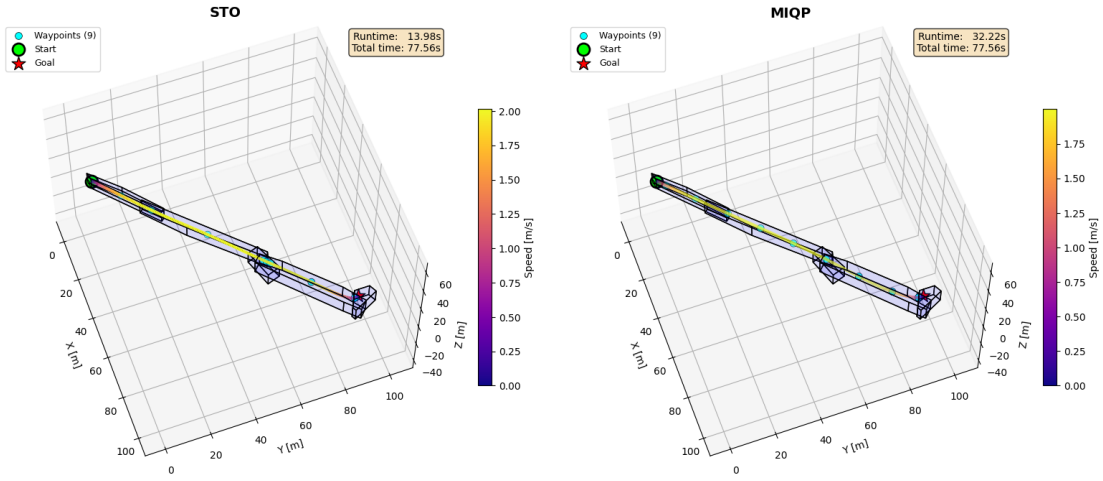

Figure 5.3: Three-dimensional comparison of the STO and MIQP trajectories in the same corridor-based environment, shown in an isometric view.

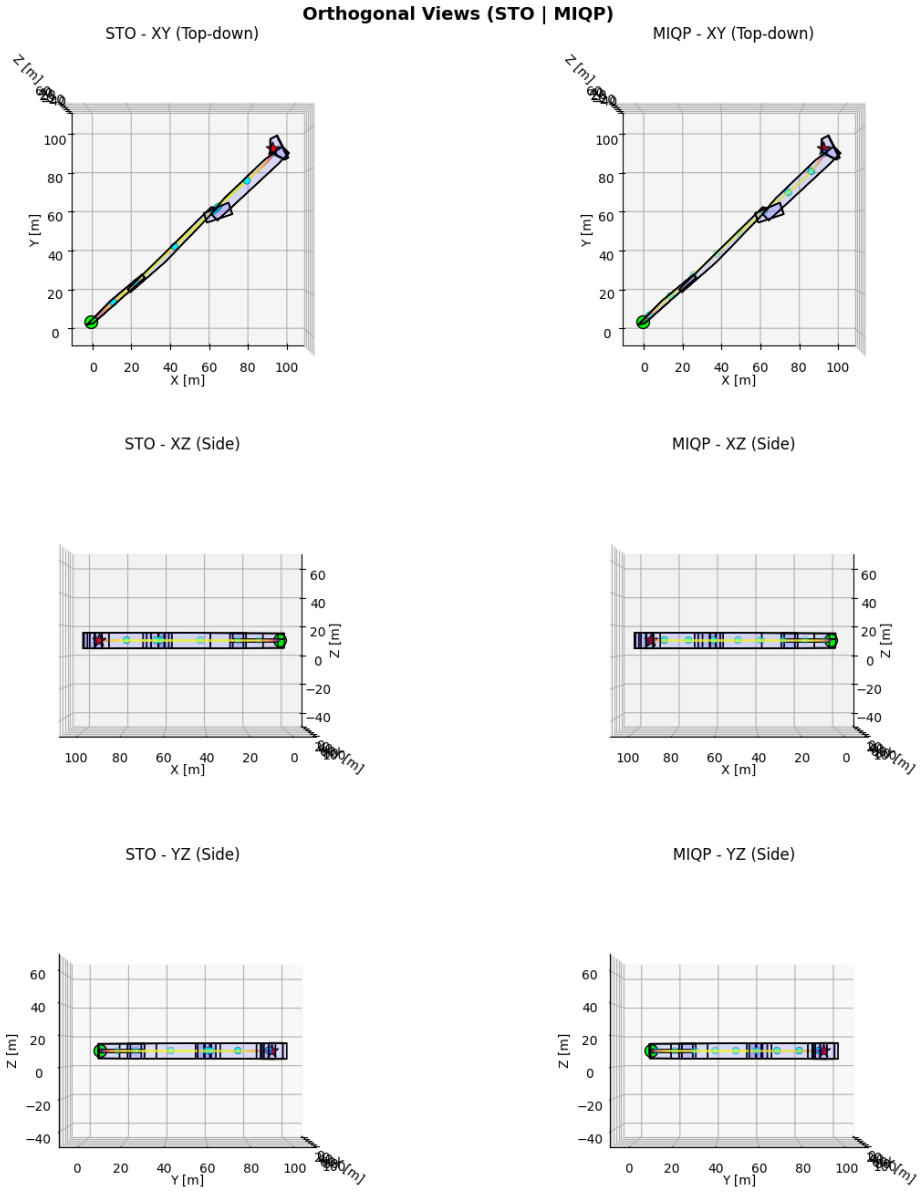

Figure 5.4: Orthogonal trajectory views for STO and MIQP, showing the XY, XZ, and YZ projections of the generated trajectories and the surrounding safe corridors.

Figure 5.5 shows the overall scenario, including the static obstacles, workspace boundaries, and the start–goal configuration of the three UAVs. Since all vehicles must pass through the same restricted region, the setup provides a suitable testbed for evaluating spatio-temporal coordination and local conflict resolution.

For each UAV, an initial collision-free geometric path was computed using A* on the voxelized map. As shown in Figure 5.6, these paths are individually feasible with respect to the static environment, but intersect in the shared narrow-passage region and therefore create a high likelihood of inter-UAV conflicts during simultaneous execution.

Based on these paths, safe flight corridors were constructed for each UAV path segment. The resulting corridor structure is shown in Figure 5.7. These corridor sequences define the geometric feasibility regions used by the STO solver for independent trajectory generation and also make the potential conflict region near the center of the map clearly visible.

Using these corridors as input, an initial STO trajectory was generated independently for each UAV. Figure 5.8 shows the resulting trajectories together with the safe corridors and reference paths. Although the generated trajectories are smooth and feasible with respect to the static environment, they are still planned independently and therefore do not account for inter-UAV interactions in the shared narrow passage. As a result, conflicts remain near the central bottleneck. The corresponding trajectory times, jerk costs, runtimes, and corridor violations are summarized in Table 5.4.

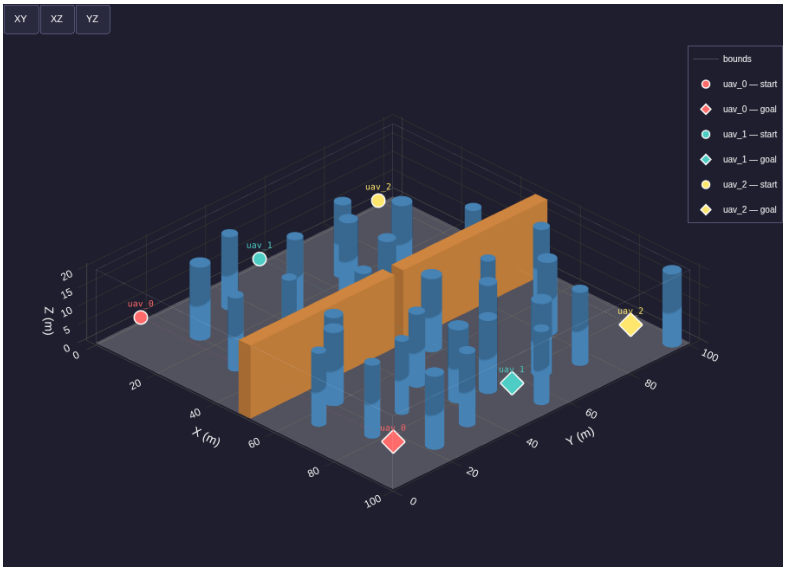

Figure 5.5: Multi-UAV swarm environment used in the narrow-passage experiments. The figure shows the workspace, static obstacles, and the start–goal positions of the three UAVs.

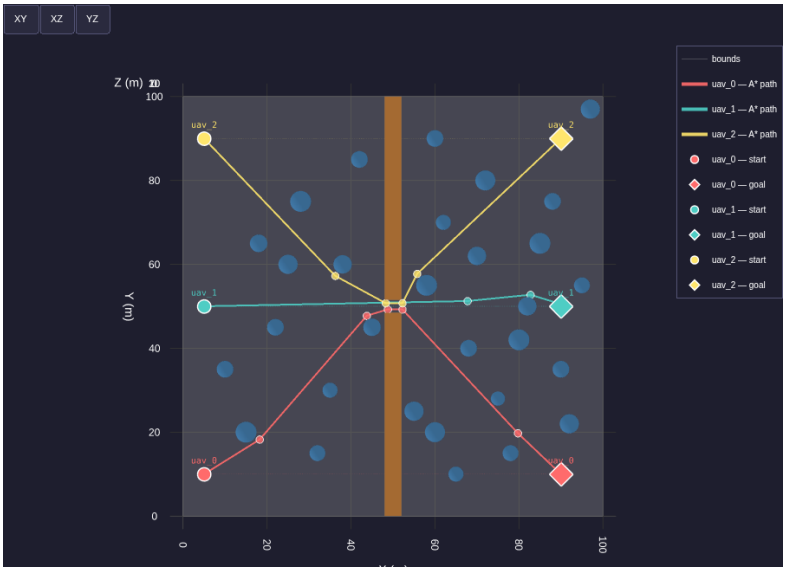

Figure 5.6: Initial A*-based paths for the three UAVs in the swarm scenario. The paths are individually collision-free with respect to the static obstacles, but intersect in the shared narrow-passage region.

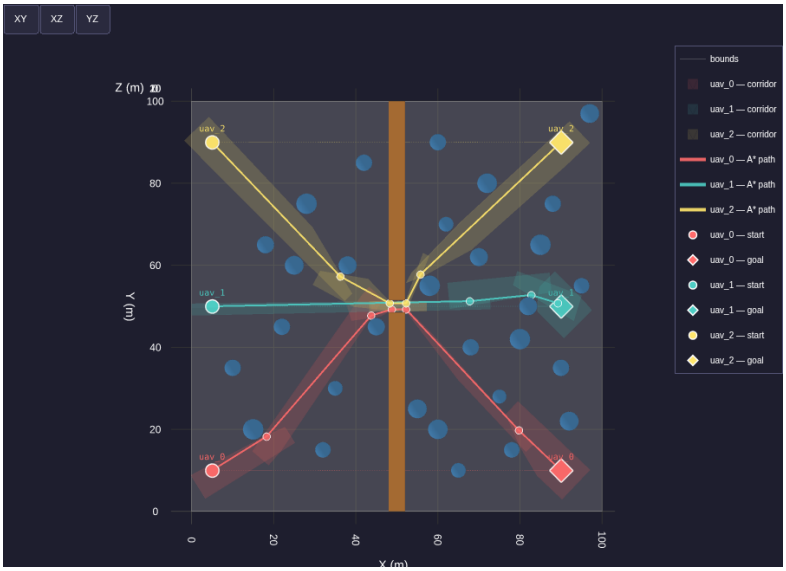

Figure 5.7: Safe flight corridors generated around the initial A* paths for each UAV. These corridor sequences define the geometric feasibility regions used by the STO solver for independent trajectory generation.

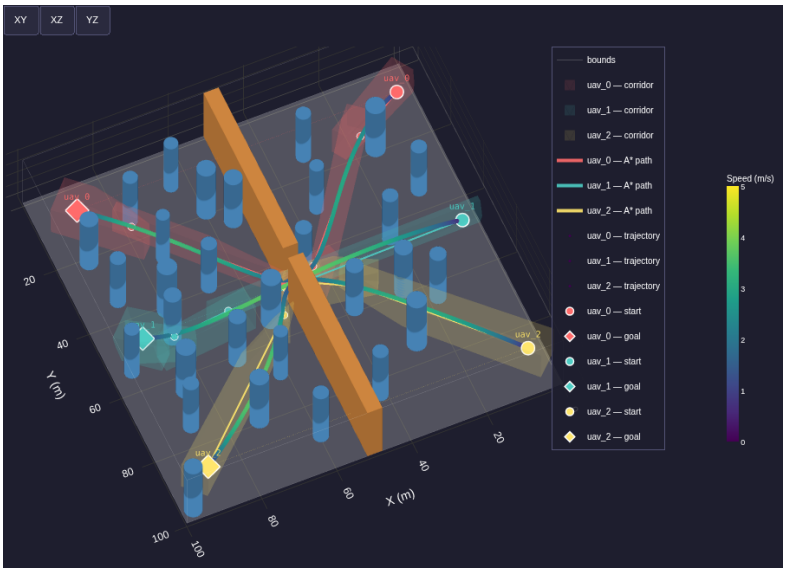

Figure 5.8: Independently optimized STO trajectories for the three UAVs, shown together with the safe corridors and the corresponding A*-based reference paths. Although each trajectory is feasible with respect to the static environment, simultaneous execution still leads to conflicts in the shared narrow-passage region.

Table 5.4: Initial independently optimized STO trajectories for the three UAVs in the swarm scenario.

UAV	Traj. Time (s)	Jerk Cost	Runtime (s)	Corridor Violation
uav_0	56.84	0.0608	7.06	0.000163
uav_1	39.61	0.0500	7.57	0.005402
uav_2	56.15	0.0599	7.07	0.000200

5.1.5 Conflict Resolution Results

After the independently optimized STO trajectories had been generated for all three UAVs, the resulting swarm motion was analyzed for pairwise space-time conflicts. Figure 5.9 shows the initial separation analysis before conflict resolution. One relevant safety-threshold violation was detected between UAV 0 and UAV 2 in the shared narrow-passage region, confirming that independent single-UAV trajectory generation is not sufficient to guarantee collision-free swarm execution.

To illustrate this conflict geometrically, Figures 5.10 and 5.11 show snapshot views of the swarm motion in the narrow-passage region before and after replanning. In Figure 5.10, UAV 0 and UAV 2 are highlighted in red to indicate the conflicting pair. Figure 5.11 shows the swarm configuration at the same simulation time after replanning, where sufficient temporal separation has been established and the conflict is no longer present.

The detected conflict was then processed using the proposed local temporal deconfliction strategy. In each round, the earliest conflict was selected, a winner–loser assignment was performed, and the loser trajectory was modified through delay injection followed by frozen-segment STO replanning. In all three rounds, UAV 0 remained the winner, while UAV 2 was selected as the loser. The corresponding conflict windows, injected delays, and updated final trajectory times are summarized in Table 5.5.

Three replanning rounds were required to remove the conflict completely. After the third round, no further pairwise separation violations were detected. Figure 5.12 shows the final separation analysis after conflict resolution, where all pairwise separations remain above the prescribed safety threshold over the full trajectory duration.

An important observation is that only the losing UAV trajectory was modified during the conflict-resolution process, while the winner trajectory and the remaining conflict-free UAV trajectory were preserved. This demonstrates the intended advantage of the proposed decoupled framework: instead of jointly re-optimizing the full swarm, only the trajectory directly involved in the active conflict is locally adjusted.

Table 5.5: Summary of the iterative conflict-resolution process for the swarm scenario.

Round	Conflict Pair	Winner	Loser	Conflict Window (s)	Injected Delay (s)	Loser Final Time (s)
1	UAV 0 vs UAV 2	UAV 0	UAV 2	[25.70, 36.15]	3.13	64.36
2	UAV 0 vs UAV 2	UAV 0	UAV 2	[27.35, 38.00]	3.19	66.81
3	UAV 0 vs UAV 2	UAV 0	UAV 2	[29.75, 37.10]	2.21	69.01

Overall, the results validate the frozen-segment STO replanning strategy for swarm coordination. The conflict was resolved in three rounds, the required temporal separation was successfully introduced, and the final solution remained smooth while

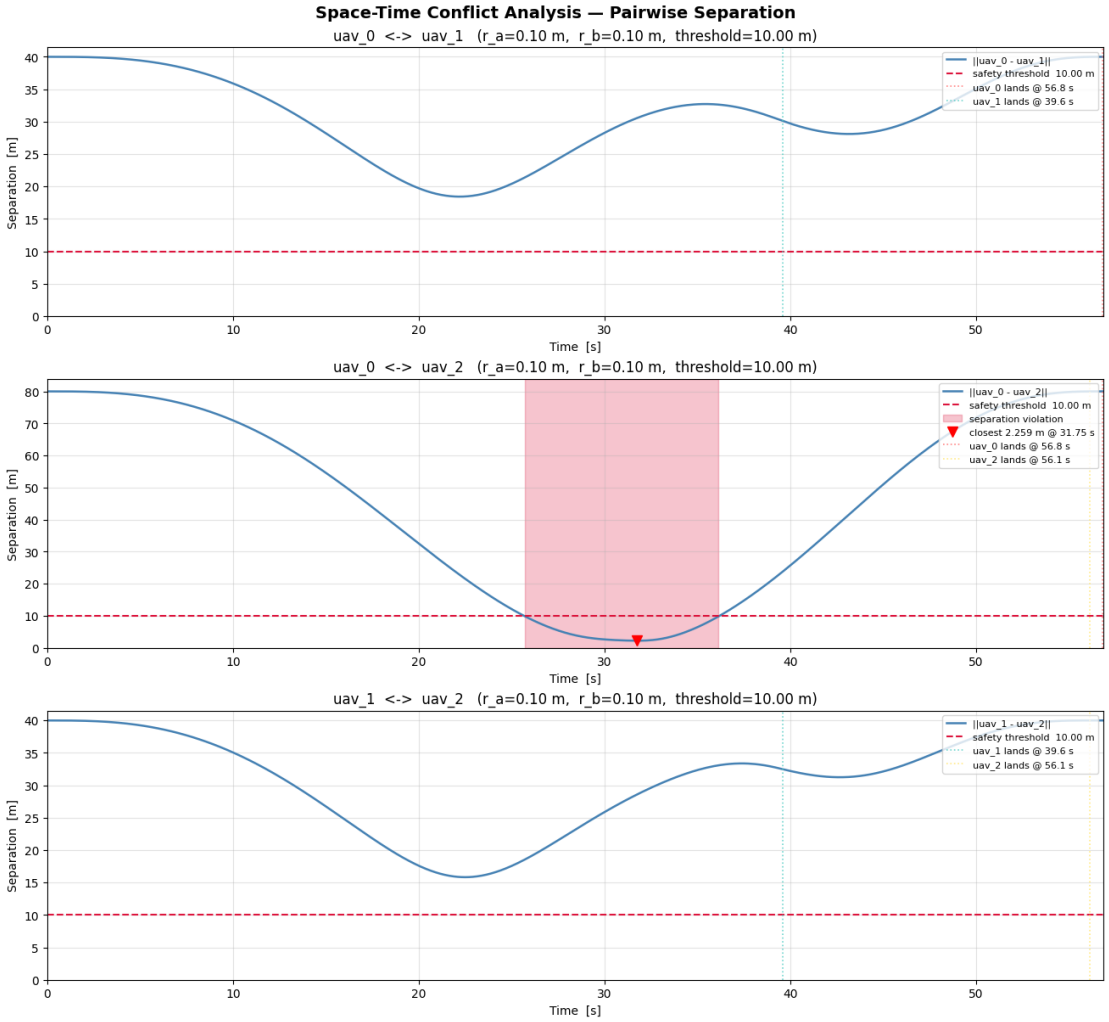

Figure 5.9: Pairwise separation analysis before conflict resolution. A safety-threshold violation is observed between UAV 0 and UAV 2, indicating a space-time conflict in the shared narrow-passage region.

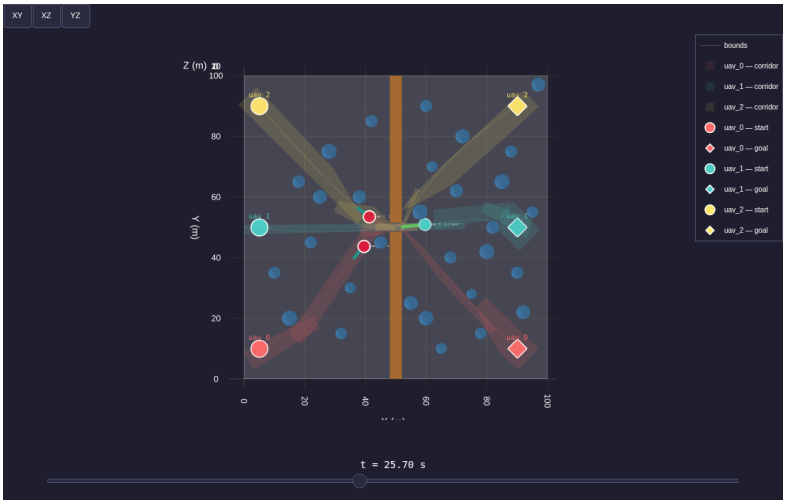

Figure 5.10: Snapshot of the swarm motion before conflict resolution at the critical instant. UAV 0 and UAV 2 are simultaneously located in the narrow-passage region and violate the required safety separation.

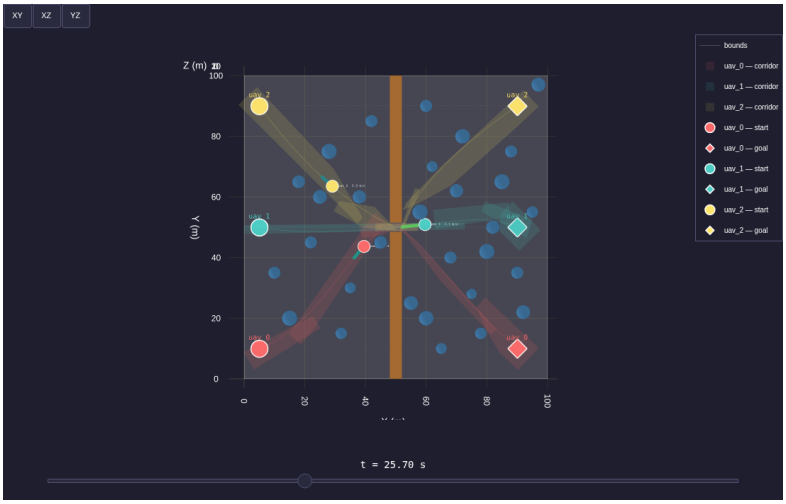

Figure 5.11: Snapshot of the swarm motion after conflict resolution. The same narrow-passage region is traversed with sufficient temporal separation, and the conflict between UAV 0 and UAV 2 is removed.

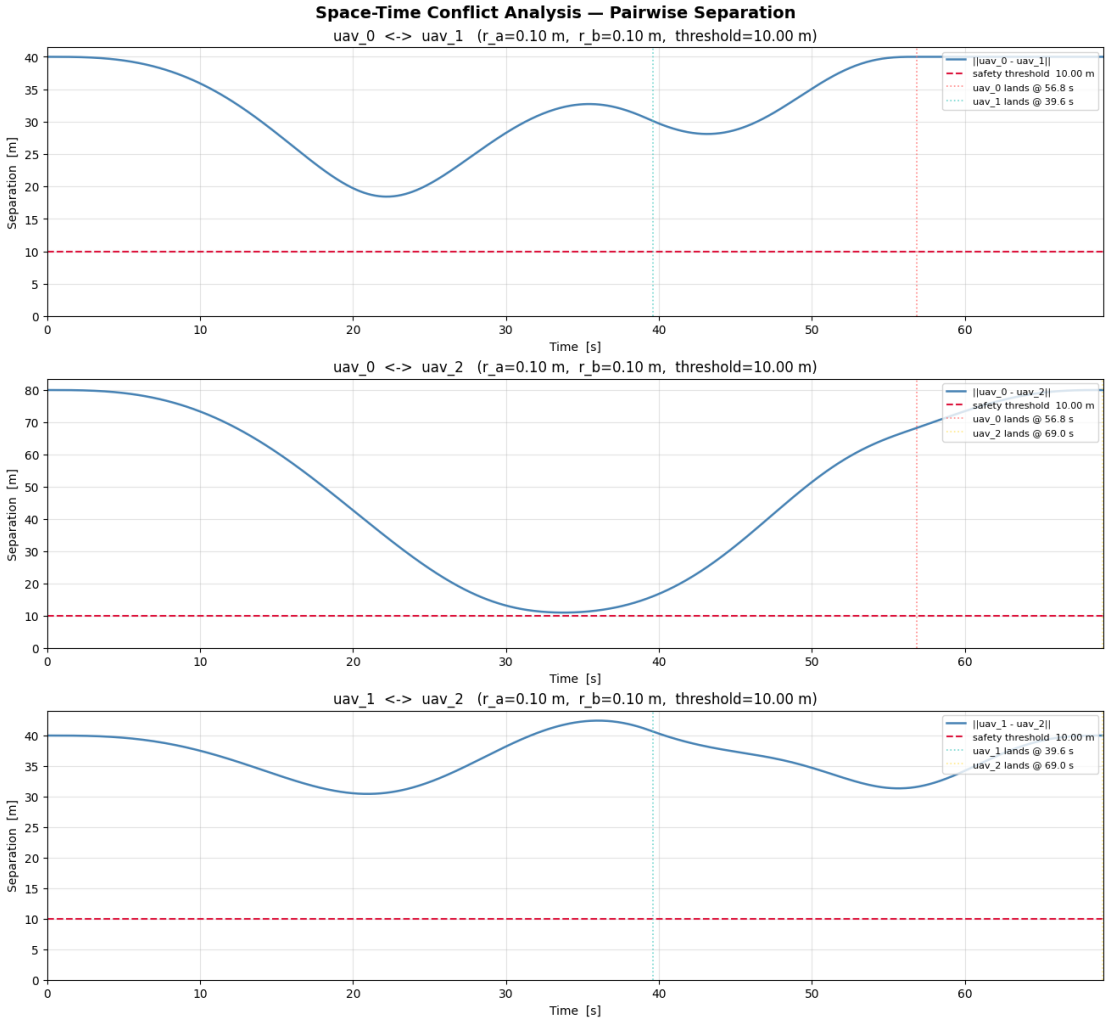

Figure 5.12: Pairwise separation analysis after iterative conflict resolution. The previous violation between UAV 0 and UAV 2 is removed, and all pairwise separations remain above the safety threshold.

avoiding the computational burden of centralized multi-UAV optimization.

6 Conclusion

This thesis investigated safe UAV trajectory generation under corridor constraints, with a particular focus on the comparison between STO and MIQP within a common planning framework. To enable a fair comparison, both methods were evaluated using the same voxel-based environment representation, the same A*-based global path initialization, the same safe-corridor structure, and a comparable MINCO-based trajectory parameterization. This made it possible to isolate the main formulation-level differences between the two approaches, namely soft versus hard constraint handling, variable versus fixed time allocation, and gradient-based versus mixed-integer optimization.

The single-UAV results showed that both STO and MIQP are capable of generating smooth corridor-guided trajectories. MIQP achieved strict feasibility in the tested scenario, with zero reported velocity, acceleration, and corridor violations, consistent with its hard-constrained formulation. STO, in contrast, produced a trajectory with the same total flight time and similarly high trajectory quality while requiring significantly less computation time. Although small residual violations remained in the STO solution, these were limited in magnitude. The comparison therefore revealed the expected trade-off: MIQP offers stronger feasibility guarantees, whereas STO provides substantially better computational efficiency and greater flexibility in time allocation.

As a second contribution, this thesis extended the STO framework to a multi-UAV narrow-passage setting. In this extension, each UAV first received an individually feasible corridor-based trajectory, after which pairwise conflicts were resolved through local temporal delay injection and frozen-segment re-optimization. Instead of globally replanning the entire swarm, only the trajectory directly involved in the active conflict was modified. The results showed that this decoupled strategy successfully resolved the detected narrow-passage conflict within a small number of replanning rounds while preserving smoothness and corridor compliance.

Taken together, these results indicate that STO is the more suitable method for the multi-UAV extension considered in this thesis. While MIQP remains attractive when strict feasibility is the primary objective, its higher computational cost makes repeated local replanning less practical in swarm settings. STO, on the other hand, offers a more favorable balance between smoothness, runtime, and near-feasible performance, which is especially important when conflicts must be resolved repeatedly in shared

constrained environments.

Several directions for future work remain open. The first is the application of the proposed framework to larger UAV fleets operating in more densely populated and structurally complex environments. As the number of vehicles increases, conflict patterns may become more frequent and interdependent, especially in scenarios with multiple narrow passages or simultaneously active bottlenecks. Evaluating how the proposed local temporal conflict-resolution strategy scales with increasing fleet size therefore represents an important next step. A second direction is the transition toward onboard and real-time deployment. In practical applications, trajectory generation and conflict resolution must be executed under limited computational resources and strict timing requirements. This represents another important challenge to be addressed in future studies.

In summary, the main contribution of this thesis is a fair comparative study of STO and MIQP for safe corridor-based UAV trajectory planning, together with a practical extension of STO to multi-UAV conflict resolution in narrow-passage environments. The results show that STO provides an effective and computationally efficient basis for smooth trajectory generation and local spatio-temporal coordination, making it a promising approach for scalable multi-UAV planning in constrained spaces.

Abbreviations

TUM Technical University of Munich
UAV Unmanned Aerial Vehicle
STO Spatio-Temporal Optimization
MIQP Mixed-Integer Quadratic Programming
SFC Safe Flight Corridor
MINCO Minimum Control
A* A-star graph search algorithm
JPS Jump Point Search
PRM Probabilistic Roadmap
RRT Rapidly-exploring Random Tree
RRT* Optimal Rapidly-exploring Random Tree
OPTP Optimal Path and Timetable Planning
L-BFGS Limited-memory Broyden–Fletcher–Goldfarb–Shanno
LOS Line of Sight
CDR Conflict Detection and Resolution
RVO Reciprocal Velocity Obstacles
ORCA Optimal Reciprocal Collision Avoidance

List of Figures

List of Tables

Bibliography

[JCR+21] V. M. Jorge, J. Capitán, A. Ribeiro, et al. "FASTER: Fast and Safe Trajectory Planner for Navigation in Unknown Environments." In: *IEEE Transactions on Robotics* (2021).

[Mao+24] Z. Mao, M. Hou, H. Li, Y. Yang, and W. Song. "Multi-UAV Cooperative Motion Planning Under Global Spatio-Temporal Path Inspiration in Constraint-Rich Dynamic Environments." In: *IEEE Transactions on Intelligent Vehicles* (2024). doi: 10.1109/TIV.2024.3422172.

[RBR16] C. Richter, A. Bry, and N. Roy. "Polynomial Trajectory Planning for Aggressive Quadrotor Flight in Dense Indoor Environments." In: *Robotics Research*. Springer, 2016.

[Unk24] Unknown. "Existing Path Planning Techniques in Unmanned Aerial Vehicles (UAVs): A Systematic Review." In: *Review Article* (2024).

[Wan+22] Z. Wang et al. "Geometrically Constrained Trajectory Optimization for Multicopters." In: *arXiv preprint arXiv:2103.00190* (2022).

[ZLZ22] C. Zhang, Y. Li, and L. Zhou. "Optimal Path and Timetable Planning Method for Multi-Robot Optimal Trajectory." In: *IEEE Robotics and Automation Letters* 7.4 (2022). doi: 10.1109/LRA.2022.3187529, pp. 10031–10038.