Three-dimensional Many-objective Path Planning and Traffic Network Optimization for Urban Air Mobility Applications Under Social Considerations

vom Fachbereich
Elektrotechnik und Informationstechnik
der Technischen Universität Darmstadt

zur Erlangung des akademischen Grades
eines Doktor-Ingenieurs (Dr.-Ing.)

Dissertation
von Nikolas Hohmann, M. Sc.

Referent:	Prof. Dr.-Ing. J. Adamy
Korreferent:	Prof. Dr. rer. nat. B. Sendhoff
Tag der Einreichung:	09.09.2024
Tag der mündlichen Prüfung:	28.11.2024

Darmstadt 2024

Hohmann, Nikolas: Three-dimensional Many-objective Path Planning and Traffic Network Optimization for Urban Air Mobility Applications Under Social Considerations
Darmstadt, Technische Universität Darmstadt
Jahr der Veröffentlichung der Dissertation auf TUprints: 2025
URN: urn:nbn:de:tuda-tuprints-288399
Tag der mündlichen Prüfung: 28.11.2024
Veröffentlicht unter CC BY 4.0 International
https://creativecommons.org/licenses/

Contents

Abbreviations and Symbols

Abbreviations

A* A* algorithm.

ACO Ant Colony Optimization.

ADS Approximated Dijkstra Solution.

ANCP Adaptive Number of Control Points.

AWDS Adaptive Weight Determination Scheme.

CFO Central Force Optimization.

DA Dijkstra Algorithm.

DBS Divide By Sum.

EA Evolutionary Algorithm.

EAs Evolutionary Algorithms.

EDT Euclidean Distance Transform.

ES Evolution Strategy.

FMM Fast Marching Method.

GA Genetic Algorithm.

GD Generational Distance.

H. ES Hybrid Evolution Strategy.

H. L-BFGS-B Hybrid Limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm with Bounds.

H. MO-CMA-ES Hybrid Multi-objective Covariance Matrix Adaptation Evolution Strategy.

H. NSGA2 Hybrid Non-dominated Sorting Genetic Algorithm 2.

H. NSGA3 Hybrid Non-dominated Sorting Genetic Algorithm 3.

H. RVEA Hybrid Reference Vector Guided Evolutionary Algorithm.

H. SMS-EMOA Hybrid $\mathcal{S}$-Metric Selection Evolutionary Multiobjective Optimization Algorithm.

HV Hypervolume.

IGD Inverted Generational Distance.

L-BFGS Limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm.

L-BFGS-B Limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm with Bounds.

LS Least Square.

MILP Mixed-Integer Linear Programming.

MO-CMA-ES Multi-objective Covariance Matrix Adaptation Evolution Strategy.

MOEA Multi-objective Evolutionary Algorithm.

MOEAs Multi-objective Evolutionary Algorithms.

MONO Multi-objective Traffic Network Optimization.

MOO Multi-objective Optimization.

MOPP Multi-objective Path Planning.

MWSP Multiple Weighted Start Points.

NAMOA* New Approach to Multi-objective A*.

NSGA2 Non-dominated Sorting Genetic Algorithm 2.

NSGA3 Non-dominated Sorting Genetic Algorithm 3.

NURBS Non-uniform Rational B-Spline.

OCID OpenCelliD.

OSM OpenStreetMap.

PRM Probabilistic Road Map.

PSO Particle Swarm Optimization.

RRT Rapidly-exploring Random Tree.

RVEA Reference Vector Guided Evolutionary Algorithm.

SMS-EMOA $\mathcal{S}$-Metric Selection Evolutionary Multiobjective Optimization Algorithm.

SOO Single-objective Optimization.

UAM Urban Air Mobility.

UAV Unmanned Aerial Vehicle.

UAVs Unmanned Aerial Vehicles.

Symbols

Latin Uppercase Letters

$\mathcal{B}^{\mathrm{2D}}$	Two-dimensional binary grid map
$\mathcal{B}_{\mathrm{B}}^{\mathrm{2D}}$	Building grid map
$\mathcal{B}_{\mathrm{S}}^{\mathrm{2D}}$	Street grid map
C	Set of point-to-point connections
$\mathbf{C}$	Parametrized curve
D	Dimension of optimization vector
D_0	Best radio signal strength
$\mathbb{D}_{\mathrm{c}}^{\mathrm{2D}}$	Two-dimensional continuous operation space
$\mathbb{D}_{\mathrm{c}}^{\mathrm{3D}}$	Three-dimensional continuous operation space
$\mathbb{D}_{\mathrm{d}}^{\mathrm{2D}}$	Two-dimensional discrete operation space
$\mathbb{D}_{\mathrm{d}}^{\mathrm{3D}}$	Three-dimensional discrete operation space
$\mathbb{D}_{\mathrm{obs}}$	Obstacle region
$\mathcal{D}$	Decoding operator
E	Number of objective functions
$\mathcal{E}$	Encoding operator
$\mathring{E}$	Set of edges in a graph
$\mathring{E}_0$	Set of initial edges (prior network optimization)
$\mathring{E}_{\mathrm{C}}$	Set of edges in a linear chain
$\mathring{E}_{\mathrm{ST}}$	Set of edges in a Steiner tree
$\mathring{E}_{\mathrm{ST,L}}$	Set of edges in a Steiner tree regarding edge length
$\mathring{E}_{\mathrm{ST,S}}$	Set of edges in a Steiner tree regarding social cost
F	Number of constraints
F_{eq}	Number of equality constraints
F_{ineq}	Number of inequality constraints
G	(Grid) Graph
G^{3D}	Three-dimensional grid graph
$G_{\mathrm{E}}^{\mathrm{3D}}$	Three-dimensional energy grid graph
G_0	Initial graph (prior network optimization)

G_{C}	Linear chain
G_{ST}	Steiner tree
$G_{\mathrm{ST,L}}$	Steiner tree regarding edge length
$G_{\mathrm{ST,S}}$	Steiner tree regarding social cost
$\mathcal{G}$	Grid map
$\mathcal{G}^{\mathrm{2D}}$	Two-dimensional grid map
$\mathcal{G}^{\mathrm{3D}}$	Three-dimensional grid map
$\mathcal{G}_{\mathrm{D}}^{\mathrm{2D}}$	Two-dimensional radio disturbance grid map
$\mathcal{G}_{\mathrm{H}}^{\mathrm{2D}}$	Height grid map
$\mathcal{G}_{\mathrm{N}}^{\mathrm{2D}}$	Two-dimensional noise grid map
$\mathcal{G}_{\mathrm{O}}^{\mathrm{2D}}$	Obstacle grid map
$\mathcal{G}_{\mathrm{R}}^{\mathrm{2D}}$	Two-dimensional risk grid map
$\mathcal{G}_{\mathrm{D}}^{\mathrm{3D}}$	Three-dimensional radio disturbance grid map
$\mathcal{G}_{\mathrm{N}}^{\mathrm{3D}}$	Three-dimensional noise grid map
$\mathcal{G}_{\mathrm{R}}^{\mathrm{3D}}$	Three-dimensional risk grid map
H	Connected component
$\mathcal{H}$	Heaviside function
$\mathcal{I}$	Linear, regular grid interpolation function
L_0	Sum of edge lengths in initial graph G_0
$L_{\mathrm{ST,L}}$	Sum of edge lengths in Steiner tree
$\mathcal{L}$	Linear, one-dimensional interpolation function
M	Median
N	Basis function in NURBS calculation
$\mathring{N}$	Set of nodes in a graph
$\mathring{N}_0$	Set of initial nodes (prior network optimization)
$\mathring{N}_{\mathrm{C}}$	Set of nodes in a linear chain
$\mathring{N}_{\mathrm{ST}}$	Set of nodes in a Steiner tree
$\mathring{N}_{\mathrm{ST,L}}$	Set of nodes in a Steiner tree regarding edge length
$\mathring{N}_{\mathrm{ST,S}}$	Set of nodes in a Steiner tree regarding social cost
$\mathring{N}_{\mathrm{T}}$	Set of vertiport nodes in a transportation network

$\mathring{N}_\tau$	Set of terminal nodes in a Steiner tree
O	Set of solutions in $\mathbb{O}$
O_{NDS}	Set of non-dominated solutions in $\mathbb{O}$
O_{Pareto}	Pareto front
$\mathbb{O}$	Objective space of an optimization problem
$\mathcal{O}$	Complexity of an algorithm
P	Set of paths
$\mathbf{P}$	Control point of a NURBS curve
Q	Route factor
R	Sequence of edges that build a shortest path
$\mathcal{R}$	Shortest path operator (e.g., Dijkstra)
S_0	Sum of social cost weights in initial graph G_0
$\mathcal{T}$	Euclidean Distance Transform (EDT)
$\mathbf{U}$	Knot vector of a NURBS curve
V	Set of solutions in $\mathbb{V}$
V_{NDS}	Set of non-dominated solutions in $\mathbb{V}$
$\mathbb{V}$	Search space of an optimization problem
X	Grid map dimension in x-direction
Y	Grid map dimension in y-direction
Z	Grid map dimension in z-direction

Latin Lowercase Letters

a	Generation/iteration counter
b	Generic binary variable
c	Boolean flag indicating a node n as crossing
c_{E}	UAV-specific energy parameter
d	Degree (e.g., of a node n, or of a NURBS curve)
d_{P}	Air corridor diameter
e	Edge in a graph
f	Objective function
f_{D}	Radio disturbance objective function
f_{E}	Energy objective function
$f_{i,\mathrm{C}}$	Constrained objective function
f_{M}	Maintenance cost objective function
f_{N}	Noise objective function
f_{R}	Risk objective function
f_{S}	Social cost objective function
f_{T}	Travel cost objective function
g	Inequality constraint
g_{H}	Minimum flight height constraint
g_{ij}, g_{ijk}	Grid map entries
g_{P}	Collision constraint in the xy-plane
g_{Z}	Collision constraint in the z-axis
h	Equality constraint
h_{C}	Connectivity constraint
h_{T}	Vertiport inclusion constraint
l	(Arc) length of a NURBS curve or an edge e
m	UAV mass
m_{GD}	Generational Distance (GD) metric
m_{HV}	Hypervolume (HV) metric
m_{IGD}	Inverted Generational Distance (IGD) metric
n	Node in a graph
n_{CP}^{*}	Optimal number of control points

n_{CP}	Number of control points in a NURBS curve
n_{FE}	Number of function evaluations
n_{NDS}	Number of non-dominated solutions
n_{WS}	Number of weighted solutions
$\tilde{n}_{\mathrm{WS}}$	Number of distinct weighted solutions
$n_{\mathbb{D}}$	Dimension of the operation space
$\mathbf{o}$	Solution of optimization problem mapped to $\mathbb{O}$
$\mathbf{o}_{\mathrm{ex},i}$	Extreme point regarding i^{th} objective
$\mathbf{o}_{\mathrm{HV,ref}}$	Reference point for the Hypervolume calculation
$\mathbf{o}_{\mathrm{knee}}$	Knee point
$\mathbf{o}_{\mathrm{N}}$	Nadir point
$\mathbf{o}_{\mathrm{U}}$	Utopia point
p	Significance level in Mann-Whitney U-Test
p_{mut}	Mutation probability
$p_{\mathrm{mut,add}}$	Add mutation probability
$p_{\mathrm{mut,del}}$	Delete mutation probability
$p_{\mathrm{mut,ind}}$	Individual mutation probability
p_{x}	Crossover probability
$\mathbf{p}$	Spatial point
$\mathbf{p}_{\mathrm{R}}$	Position of a radio signal tower
$\mathbf{p}_{\mathrm{T}}$	Position of a vertiport
$r_{\lvert\mathrm{e}\rvert}$	Opt. *Euclidean distance* route between two nodes
r_{l}	Opt. *edge length* route between two nodes
r_{s}	Opt. *social cost* route between two nodes
s	Social cost weight of an edge e
s_{off}	Offspring population size
s_{par}	Parent population size
s_{pop}	Population size
t	Boolean flag indicating a node n as vertiport
u	NURBS parameter
v	Element of optimization vector

v_{xy}	Cruise velocity (in xy-direction)
$v_{\mathrm{z},\uparrow}$	Climbing velocity
$v_{\mathrm{z},\downarrow}$	Descending velocity
$\mathbf{v}$	Optimization vector, solution of optimization
$\mathbf{v}_{\mathrm{ex},i}$	Extreme solution regarding i^{th} objective
w	Control point's weight in the NURBS formulation
x	x-component of a spatial point
$x_{\max}$	Maximum x-boundary of operation space
$x_{\min}$	Minimum x-boundary of operation space
y	y-component of a spatial point
$y_{\max}$	Maximum y-boundary of operation space
$y_{\min}$	Minimum y-boundary of operation space
z	z-component of a spatial point
$z_{\mathrm{F,min}}$	Minimum flight height
z_{R}	Height of the radio signal origin
$z_{\max}$	Maximum z-boundary of operation space
$z_{\min}$	Minimum z-boundary of operation space

Greek Letters

γ	Grid normalization constant
δ_x	Grid resolution in x-direction in second stage
$\tilde{\delta}_x$	Grid resolution in x-direction in first stage
δ_y	Grid resolution in y-direction in second stage
$\tilde{\delta}_y$	Grid resolution in y-direction in first stage
δ_z	Grid resolution in y-direction in second stage
$\tilde{\delta}_z$	Grid resolution in z-direction in first stage
ε	Approximation error
η_{mut}	Mutation crowding degree
η_{x}	Crossover crowding degree
λ	Weight in a weighted aggregation of objectives
μ	Mean
ν_1, ν_2	Number of samples in Mann-Whitney U-Test
ξ	Constraint parameter
Π	Path, sequence of waypoints
$\Pi^{\times}$	Set of unsafe waypoints
Π_{d}	Dijkstra path
Π_{xy}	Two-dimensional path projection (in xy-plane)
$\Pi_{\mathrm{z},\uparrow}$	Upwards path projection (in z-direction)
$\Pi_{\mathrm{z},\downarrow}$	Downwards path projection (in $-z$-direction)
$\pi^{\times}_{i,\mathrm{x}}$	x-component of i^{th} unsafe waypoint
$\pi^{\times}_{i,\mathrm{y}}$	y-component of i^{th} unsafe waypoint
$\pi^{\times}_{i,\mathrm{z}}$	z-component of i^{th} unsafe waypoint
$\pi_{i,\mathrm{x}}$	x-component of i^{th} waypoint
$\pi_{i,\mathrm{y}}$	y-component of i^{th} waypoint
$\pi_{i,\mathrm{z}}$	z-component of i^{th} waypoint
$\boldsymbol{\pi}$	Waypoint
$\boldsymbol{\pi}^{\times}_i$	i^{th} unsafe waypoint
$\boldsymbol{\pi}_{\mathrm{d}}$	Waypoint in a Dijkstra path
$\boldsymbol{\pi}_{\mathrm{g}}$	Goal point of a path

$\boldsymbol{\pi}_i$	i^{th} waypoint
$\boldsymbol{\pi}_\text{s}$	Start point of a path
ρ	Radio signal scaling factor
σ	Standard deviation
$\boldsymbol{\sigma}$	Strategy parameter vector of an ES
τ_A	Approximation threshold
τ_G	Generation threshold
ω	Weight of an edge e in a graph G

Abstract

This dissertation proposes and investigates solution approaches to *two* problems in urban air mobility, considering the different perspectives of many stakeholders, including societal interests. The *many-objective path planning problem* seeks Pareto-optimal, three-dimensional, and smooth paths connecting two given locations in the city. The *multi-objective traffic network optimization problem* searches for Pareto-optimal and three-dimensional transportation networks that can be constructed from a given set of paths. Since this work also explicitly considers social objectives within these problems, it has both a societal and a practical relevance. This thesis analyzes both stated problems and proposes a new framework to solve the first problem efficiently. Then, it shows how the optimized paths can be combined into a three-dimensional traffic network. Afterward, this dissertation presents another new framework to optimize the obtained traffic network in terms of multiple objectives. It tests the influence of integrating social criteria on the economic costs of the networks obtained.

Using geospatial data from four different cities, paths and networks were optimized to evaluate the efficiency of the path planning framework against current methods and to compare the network solutions with conventional strategies. The developed path planning framework showed a significant advantage over comparable approaches. When traffic networks were optimized, including social criteria, their social acceptance increased much more than the monetary costs. An essential finding of this work is that the many-objective path planning problem can be solved efficiently in the three-dimensional operation space by an intelligent combination of existing algorithms and the inclusion of three new algorithmic features. Beyond that, it is beneficial to integrate social criteria into optimization problems when the solutions obtained are the basis for decisions in the area of conflict between the economy and human welfare.

Kurzfassung

In dieser Arbeit werden Lösungsansätze für *zwei* Probleme im Bereich der urbanen Luftverkehrsmobilität vorgeschlagen und untersucht, wobei die unterschiedlichen Perspektiven vieler Interessengruppen, einschließlich sozial-gesellschaftlicher Interessen, berücksichtigt werden. Das *multikriterielle Pfadplanungsproblem* sucht nach Pareto-optimalen, dreidimensionalen und glatten Pfaden zwischen zwei gegebenen Punkten in der Stadt. Das *multikriterielle Verkehrsnetzoptimierungsproblem* sucht nach Pareto-optimalen und dreidimensionalen Transportnetzen, die aus einer gegebenen Menge an Pfaden konstruiert werden können. Da diese Arbeit auch explizit soziale Kriterien in diesen Problemen berücksichtigt, besitzt sie neben der praktischen auch eine sozial-gesellschaftliche Relevanz. Diese Arbeit analysiert die beiden Probleme und stellt zunächst eine neue Methode vor, das erste effizient zu lösen. Es wird weiterhin aufgezeigt, wie die optimierten Pfade zu einem dreidimensionalen Verkehrsnetz kombiniert werden können. Anschließend wird ein zweites neues Framework präsentiert, um das entstandene Netzwerk hinsichtlich mehrerer Kriterien zu optimieren. Dann wird untersucht, welchen Einfluss die Integration sozialer Kriterien auf die ökonomischen Kosten der erhaltenen Netzwerke hat.

Anhand von Geodaten aus vier verschiedenen Städten wurden Pfade und Verkehrsnetze optimiert, um die Effizienz des Pfadplanungsframeworks im Vergleich zu aktuellen Methoden zu bewerten und im Weiteren die erhaltenen Verkehrsnetzwerke mit denen herkömmlicher Strategien zu vergleichen. Das entwickelte Pfadplanungsframework zeigte einen deutlichen Vorteil gegenüber vergleichbaren Ansätzen. In den unter Einbeziehung sozialer Kriterien optimierten Verkehrsnetzen stieg die soziale Akzeptanz deutlich stärker als die monetären Kosten. Eine wichtige Erkenntnis der Arbeit ist, dass das multikriterielle Pfadplanungsproblem im dreidimensionalen Planungsraum durch eine intelligente Kombination bestehender Algorithmen und unter Einbeziehung von drei neuen Algorithmen effizient gelöst werden kann. Darüber hinaus ist es vorteilhaft, immer dann soziale Kriterien in Optimierungsprobleme zu integrieren, wenn die erhaltenen Lösungen die Grundlage für Entscheidungen im Spannungsfeld zwischen Ökonomie und menschlichem Wohlergehen bilden.

1 Introduction

Path planning is the generic term for many sub-disciplines that deal with where an agent should go or can move. In a defined environment, the path planning problem generally searches for a path leading an agent from a given start point to a given endpoint while optimizing some cost function(s). Depending on the problem domain, the representations of a path can differ [1]. In the upper part of Fig. 1.1, two typical representations are visualized, which can be seen as the two ends in a spectrum of degree of abstraction.

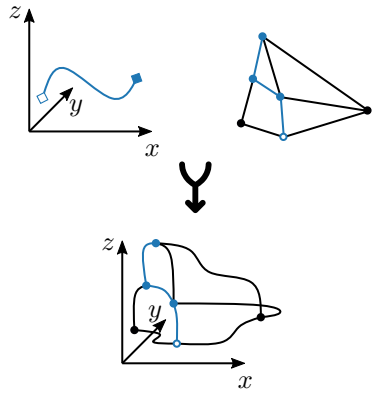


Figure 1.1: The two edge cases of path representations, which are 1) continuous paths in less-abstracted environments (left) and 2) discrete paths in abstracted environments (right), are fused to obtain a three-dimensional, and spatially continuous *aerial corridor network* (bottom) to approach path planning and routing problems in *unstructured* and *occupied* environments.

- In a less abstracted world, the agent's environment is a continuous space of possible spatial locations. A path is then a sequence of spatial points that could be fed to an agent's controller to let it move in its real environment. The advantage of this continuous path representation is that it allows agents to move smoothly in complex

and irregular, i.e., *unstructured* environments. The disadvantage is that the higher computational complexity of the continuous representation makes it difficult to operate multiple agents in a shared environment [2].

- In a more abstracted world, the agent's environment is a graph consisting of nodes, which do not necessarily represent spatial positions, and edges, which connect the nodes logically. A path is a sequence of nodes instructing the agent how to traverse the graph. Most often, this abstract path can not directly be used to move an agent in its real environment. The abstract view is instead used to find possible routes for an agent in a graph (i.e., routing). The advantage of the discrete path representation is that its abstracted structure provides an efficient approach to path planning in *occupied* environments, i.e., environments with many agents. However, the discretized, and thus less precise, path resolution makes navigation in complex environments difficult.

Depending on the structuredness (*structured* vs. *unstructured*) and occupancy (*occupied* vs. *unoccupied*) of the environment, many application-related path planning problems can be solved by choosing only one of the described representation schemes.

For example, consider a single autonomous Unmanned Aerial Vehicle (UAV) flying through a dense forest [3], [4]. It must avoid trees while maintaining a smooth and feasible path. Here, the environment is *unoccupied* (i.e., no other agents are around), but highly *unstructured*, so that a continuous path representation is used. In contrast, a group of warehouse robots must navigate through a *structured* grid of shelves to pick up and deliver items [5]. Here, the grid-like layout of the warehouse allows for a discrete path representation that facilitates path planning for multiple agents in the *occupied* environment [6].

When it comes to an *occupied and unstructured* environment, a combination of both representation schemes becomes reasonable. Such a combined representation may be obtained by assembling continuous smooth paths into a discrete *aerial corridor network*, as shown in lower Fig. 1.1. At its graph level, this network representation allows path planning for multiple agents in *occupied* environments. At the same time, at the edge level, it allows for the complex and smooth motions required to traverse *unstructured* environments.

For mobility applications on the ground, we find such a combined representation in the road network for cars and other traffic participants. Every

intersection can be seen as a node of the road network, whereas the courses of the streets build the edges. Since most road networks on the ground have grown historically, the question of how to design them from scratch has not played an important role so far [7]. In the new field of Urban Air Mobility (UAM), however, it is necessary to design infrastructure concepts from scratch. Moreover, in urban air traffic, just as in road traffic, one also faces the dual problem of many agents moving in an unstructured space. Therefore, it seems reasonable to set up *aerial corridor networks*, as indicated in the lower Fig. 1.1, in urban air spaces, similar to the road networks for cars.

This new need for the design of an aerial infrastructure that ideally combines locally optimized spatial paths (i.e., aerial corridors) into a complete transportation network ultimately leads to two main problems, which are:

Problem One

Given a start and an endpoint, the Multi-objective Path Planning (MOPP) problem asks to place a flight corridor in the urban airspace accommodating all parties involved. That includes the question of the criteria by which the path should be optimized. This problem deals with the individual design of the smooth paths that eventually form the edges of the new *aerial corridor network*.

Problem Two

Given a set of paths, the Multi-objective Traffic Network Optimization (MONO) problem asks how to merge the paths to form an *aerial corridor network*, and how this network can be further optimized concerning various criteria that again meet the needs of all stakeholders involved. This problem is about the intelligent combination of the individual paths obtained by solving *Problem One* multiple times into the final *aerial corridor network*.

This thesis deals with the solution to these two main problems and the sub-problems, always following the scheme of problem formulation, problem analysis, solution approach, solution evaluation, and evaluated solution improvement.

After a motivational introduction in the following Section 1.1, the contributions of this work are highlighted in Section 1.2, before the outline of the dissertation is presented in Section 1.3.

1.1 Motivation

Why is studying urban air mobility, particularly air paths and transport networks over cities, worthwhile? The so-called last-mile delivery is a bottleneck in modern transportation systems [8]. Especially in congested urban areas, delivering packages or passengers from an intermediate stop to their final destination (i.e., last-mile problem) is resource-intensive and costly.

Therefore, the problem has spawned a variety of different optimization approaches [9] and ideas for new modes of transportation. Unmanned Aerial Vehicles (UAVs) have sparked the interest of companies in logistics [10] and mobility [11] as they open a whole new dimension of local transportation possibilities.

Start-ups like Wingcopter [12] already utilize UAVs to carry out important or urgent deliveries.

The market for UAV deliveries is expected to increase with a growth rate of approximately 20% in the next 12 years to a market value of 4 billion US$, and the number of delivery UAVs is rising rapidly from 7 thousand in 2020 to approximately 125 thousand in 2035 [11]. So, there is a market for urgent last-mile air deliveries. But what about the necessary transport infrastructure?

Today, the prevalent modes of operation for aircraft are *airspace-based operations* in controlled airspace and *free flight operations* within the visual line of sight in uncontrolled airspace [13]. However, with the anticipated densities of UAV flights in urban airspaces, both concepts will likely become inefficient, unsafe, or even infeasible [14].

Alternative approaches propose some form of structural traffic regulation for aerial near-ground operations in urban areas, e.g., flight corridors [15], [16].

So, there is a need for air transport infrastructure, which motivates the question about optimal flight path placement and optimal aerial corridor network design above the city. The first problem in answering this question arises as soon as it is formulated. What does *optimal* mean?

In the SESAR 3 Joint Undertaking [17], a partnership between the European Union and other private and public partners to advance aerial mobility in Europe, they state:

> *"Further to the two first principles on safety and economic growth, drone operators [...] should consider that the flight of drones at low altitudes can disturb the people and nature on*

> *the ground nearby. The aim [...] is to balance the commercial pressure for growth of drone use with the preservation of nature, people's health, personal privacy and European security. Consideration of social acceptance from the start of drone operations is likely to produce a better result in the long term than a brief boom in drone use followed by a public backlash."*

The term *optimality* is always relative. It depends on the specific perspective and context in which it is evaluated. Especially in cities, where many different interest groups meet, the design of a new transportation system has to satisfy many demands, e.g., the legal and safety requirements of the aviation authority, the economic interests of logistics companies, and the social factors among the city residents.

In this thesis, the term *social factors* describes all negative influences that the introduction of UAVs in the city entails. This could be 1) the hazard of air crashes that may harm people underneath, 2) the noise pollution generated by a propeller-driven aircraft, or 3) privacy concerns induced by camera-equipped aircraft.

Bauranov *et al.* [14] point out that the consideration of social factors like noise pollution has been neglected in many recent approaches to urban air mobility, even though it is one of the most crucial factors for scaling an urban air mobility application.

Quite often, the different demands on a transportation system conflict, calling for some trade-off solutions. Therefore, the solutions for optimizing air corridors and networks presented in this thesis are designed as *multi-criteria* optimization frameworks, and the conducted studies also consider a social perspective.

1.2 Contributions

The two main problems whose solution is the subject of this thesis have already been presented in the introduction to this chapter. The motivation section 1.1 emphasized that the solutions to these problems should be approached not only from an economic perspective but also from a human-centered one.

Two frameworks were developed in this area of tension that can together derive a continuous, three-dimensional aerial corridor network for UAM applications by 1) optimizing single paths regarding multiple objectives and 2) merging them into optimized corridor networks. In summary, the contributions that emerged from this work are the development of a

Multi-objective Path Planning (MOPP) framework

that is able to ...

- optimize three-dimensional smooth paths on large urban scenarios,
- integrate arbitrarily many constraints and objectives that may be functions or black-box-simulators,
- boost the optimization by a pre-processing step that adheres to the formulated objectives and constraints,
- work with real-world data from OpenStreetMap (OSM) and OpenCelliD (OCID),

and the development of a

Multi-objective Traffic Network Optimization (MONO) framework

that is capable of ...

- merging multiple optimized paths into a network,
- optimizing the three-dimensional spatial network structure,
- with various objectives and constraints, including social ones,
- showing the importance and economic costs of social perspectives.

1.3 Outline of this Dissertation

The structure of this work is closely aligned with the two frameworks described. From Chapter 2 until Chapter 5, we consider various aspects of the MOPP framework. First, Chapter 2 deals with essential preliminaries necessary for the upcoming sections and related work in the field of path planning. In the following chapters 3 to Chapter 5, the MOPP framework is then presented, evaluated and improved for more general problems.

In detail, the MOPP problem is initially approached in Chapter 3 by keeping its complexity as low as possible. Only two criteria are included,

and neither boundary conditions nor the third dimension of the operating space are considered, asking:

Q1: How can the Pareto-based bi-criteria path planning problem with continuous smooth path representation be efficiently solved in the two-dimensional planning space?

The complexity of the problem is then increased in stages. Chapter 4 deals with the questions:

Q2: What are the challenges that need to be considered in the three-dimensional planning space, and how can they be addressed?

and

Q3: How can constraints be integrated into the optimization problem effectively?

Then, more than two objectives are included into the MOPP problem in Chapter 5, which is answering the question:

Q4: Are there problems with integrating more than two objective functions into the problem, and how can they be solved?

Additionally, this chapter analyzes the resulting trade-off solutions (i.e., paths) in more detail - especially concerning social criteria.

Chapter 6 then deals with the MONO framework. First, related work is presented. Then, the framework for the solution of the problem is introduced:

Q5: How to get from a set of optimized paths to a Pareto set of optimized aerial corridor networks?

Finally, we analyze the obtained solutions (i.e., networks), focusing on their social compatibility. Here, the pursued research question is:

Q6: How much higher are the economic costs in a more socially acceptable network, and how much social acceptance is gained for these additional costs?

Finally, Chapter 7 summarizes the work and gives an outlook on potential future research questions.

The Appendix A contains on the one side a tabular categorized literature overview on multi-objective path planning approaches, and on the other side, visualizations of all scenarios that were considered as part of the studies carried out.

2 Path Planning Problem

This chapter introduces preliminaries and related work regarding the path planning problem. We will only briefly discuss long-established approaches by referring to relevant literature and otherwise limit ourselves to a detailed presentation of the basics needed for the following chapters.

Thereby, Section 2.1 deals with different possibilities for environment and path representations. Then, in Section 2.2, multi-objective optimization is introduced in general, particularly the path planning problem. Finally, we discuss related work on multi- and many-objective path planning in Section 2.3.

2.1 Representations

2.1.1 Environment

Grid Maps

As a two-dimensional grid map, we define a scalar function

$$\mathcal{G}^{\mathrm{2D}} : \underbrace{\{1, \dots, X\} \times \{1, \dots, Y\}}_{\mathbb{D}_{\mathrm{d}}^{\mathrm{2D}}} \to \mathbb{R}, \quad (i, j) \mapsto g_{ij}$$

that maps each cell of a two-dimensional discrete domain to a real value. Respectively, a three-dimensional grid map is a scalar function

$$\mathcal{G}^{\mathrm{3D}} : \underbrace{\{1, \dots, X\} \times \{1, \dots, Y\} \times \{1, \dots, Z\}}_{\mathbb{D}_{\mathrm{d}}^{\mathrm{3D}}} \to \mathbb{R}, \quad (i, j, k) \mapsto g_{ijk}$$

that assigns real values to a three-dimensional discrete domain. It is worth pointing out that only positive grid values are meaningful when optimizing paths on grids. Otherwise, the optimization would converge to infinite undesired path cycles over non-positive cells. Therefore, the grid maps $\mathcal{G}^{\mathrm{3D}}$ presented in this thesis are always mapped to positive values without the restriction of generality

$$\hat{\mathcal{G}}^{\mathrm{3D}} = \mathcal{G}^{\mathrm{3D}} + |\min(0, \min(\mathcal{G}^{\mathrm{3D}}) - 1)|. \tag{2.1}$$

Two-dimensional grid maps can be seen as a matrix, as visualized on the left side in Fig. 2.1, three-dimensional grid maps can be represented by tensors of order three.

Grid Graphs

A general graph $G = \{\mathring{N}, \mathring{E}\}$ consists of a set of nodes $\mathring{N}$, and a set of edges $\mathring{E}$. The number of nodes and edges in the respective sets is denoted by $|\mathring{N}|$ and $|\mathring{E}|$ in the following. An edge $e_{ij} = (n_i, n_j)$ connects two nodes n_i and n_j and may be undirected or directed, indicating some (information) flow direction. Moreover, an edge can be weighted, meaning it is assigned a vector of weights that could, for example, represent different costs to traverse the edge.

In this thesis, when discussing graphs in the context of path planning environment representations, usually *spatial grid graphs* are meant. *Spatial* means that the graph's nodes are located within a metric space, in our case, the Euclidean space. *Grid* means that the graph's nodes are arranged in a regular structure, in our case, a rectangular tiling.

Grid graphs are a generalization of grid maps. Every grid map environment representation can be transferred into a grid graph environment representation, as visualized in Fig. 2.1. Therefore, each cell in the grid map corresponds to a node in the grid graph. The cell's value is a weight for all directed edges that point to the node. The connectivity information, i.e., the information about which nodes are linked in the graph, is an additional piece of information that makes the graph representation more potent than the grid representation. Moreover, the grid graph allows edges leading to a specific node to have different weights, which can not be mapped into the grid map. Nevertheless, in the following, when connectivity information is unimportant and different edge weights are not necessarily needed, the two terms grid map $\mathcal{G}$ and grid graph G are used interchangeably.

2.1.2 Path

A path Π connects a start point $\boldsymbol{\pi}_\mathrm{s} \in \mathbb{R}^{n_\mathbb{D}}$ and an endpoint $\boldsymbol{\pi}_\mathrm{g} \in \mathbb{R}^{n_\mathbb{D}}$ with a sequence of $n_\mathbb{D}$-dimensional points $\boldsymbol{\pi}_i \in \mathbb{R}^{n_\mathbb{D}}$

$$\Pi = [\boldsymbol{\pi}_0 = \boldsymbol{\pi}_\mathrm{s}, \boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_{|\Pi|-1} = \boldsymbol{\pi}_\mathrm{g}],$$

where $|\Pi|$ is the path's number of waypoints. The path points can lie equidistantly at a distance $\Delta\boldsymbol{\pi}$ from each other, but generally, they are

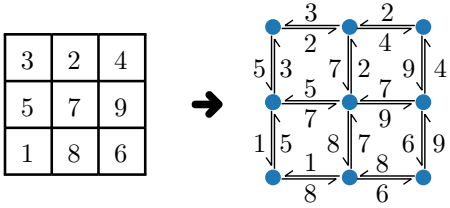


Figure 2.1: A 3×3 grid map visualization with exemplary cell entries and the corresponding generalized grid graph. Due to the connectivity information, the grid graph contains more information.

not.

Here, the notation distinguishes the generally continuous path representation Π from a discrete path representation Π_d, which, for example, consists of a sequence of neighboring cell positions in the grid map $\mathcal{G}$ or neighboring node positions in the grid graph G.

An advanced continuous path representation used throughout this thesis is presented in the following.

Non-uniform Rational B-Splines

A Non-uniform Rational B-Spline (NURBS) is a mathematical description for a parametrized curve (or surface). Adapting the notion of Piegl *et al.* [18], a NURBS curve of order $d+1$ is described as

$$\mathbf{C}(u) = \frac{\sum_{i=0}^{n_\mathrm{CP}-1} N_{i,d}(u) w_i \mathbf{P}_i}{\sum_{i=0}^{n_\mathrm{CP}-1} N_{i,d}(u) w_i} \qquad 0 \leq u \leq 1,$$

where

- d is the degree of the basis function $N_{i,d}$,
- $\mathbf{P}_i = \begin{bmatrix} x_i & y_i & z_i \end{bmatrix}$ is the i^th so-called control point whose position influences the shape of the curve (here assumed to be 3D),
- $w_i \in \mathbb{R}^+$ is the i^th control point's weight and
- n_CP is the number of control points with

$$n_\mathrm{CP} \geq d+1. \tag{2.2}$$

A knot vector $\mathbf{U}$ must be defined with the mentioned parameters. The knot vector is a monotone sequence of $m+1$ entries (i.e., knots), with $m = n_{\mathrm{CP}} + d$, and can be written as

$$\mathbf{U} = \begin{bmatrix} u_0 = 0 & u_1 & \dots & u_{m-1} & u_m = 1 \end{bmatrix}.$$

The basis functions $N_{i,d}(u)$ are dependent on the parameter u and can be calculated recursively following the De-Boor-Cox formulas [19], [20], [21]

$$N_{i,0}(u) = \begin{cases} 1 & \text{if } u_i \leq u \leq u_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

$$N_{i,d}(u) = \frac{u - u_i}{u_{i+d} - u_i} N_{i,d-1}(u) + \frac{u_{i+d+1} - u}{u_{i+d+1} - u_{i+1}} N_{i+1,d-1}(u).$$

For a particular $u = u_t$, the knot vector $\mathbf{U}$ defines the intervals (i.e., knot spans) $u_t \in [u_i, u_{i+1}[$, in which the zero-degree basis function $N_{i,0}(u_t)$ influences the recursive calculation of basis function $N_{i,d}(u_t)$ of degree d. Outside the interval $[u_i, u_{i+1}[$, the zero-degree basis function $N_{i,0}(u_t) = 0$ is zero.

If the first and the last knots in the knot vector $\mathbf{U}$ are repeated with multiplicity $d+1$

$$u_0 = u_1 = \ldots = u_d = 0 \qquad \text{and} \qquad u_{m-d} = \ldots = u_{m-1} = u_m = 1,$$

the NURBS curve $\mathbf{C}(u)$ is clamped, which means that it starts in the first and ends in the last control point

$$\mathbf{C}(0) = \mathbf{P}_0 \qquad \text{and} \qquad \mathbf{C}(1) = \mathbf{P}_{n_{\mathrm{CP}}-1}.$$

An exemplary two-dimensional clamped NURBS curve is visualized in Fig. 2.2. To finally obtain the path Π from the NURBS representation, the curve $\mathbf{C}(u)$ is evaluated at all points of a parameter vector $\mathbf{u} \subset [0, 1]$, whose elements can be freely selected between zero and one

$$\Pi = \mathbf{C}(\mathbf{u}).$$

For later calculations, it is beneficial if the resulting path points

$$\boldsymbol{\pi}_i = \begin{bmatrix} \pi_{i,\mathrm{x}} & \pi_{i,\mathrm{y}} & \pi_{i,\mathrm{z}} \end{bmatrix} \in \Pi \subset \mathbb{R}^3$$

are spatially equally spaced, i.e., $\Delta\boldsymbol{\pi} = \text{const}$. However, the standard uniform, i.e., equidistant, sampling in the curve's parameter space

$$\mathbf{u} = \mathbf{u}_{\mathrm{U}} = \begin{bmatrix} 0 & \cdots & \dfrac{1}{|\Pi| - 1} k & \cdots & 1 \end{bmatrix} \quad \text{with} \quad k = 1, \ldots, |\Pi| - 2$$

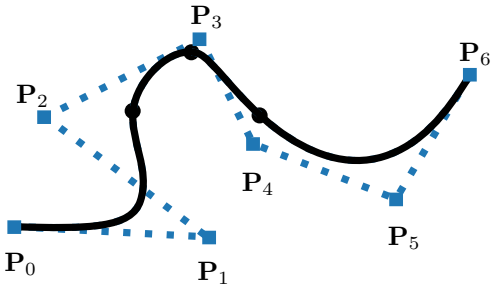


Figure 2.2: An exemplary (clamped) NURBS curve with degree $d = 3$, number of control points $n_{\mathrm{CP}} = 7$, weights $\begin{bmatrix} w_0 & \dots & w_6 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 3 & 1 & 1 & 1 \end{bmatrix}$, and the knot vector $\mathbf{U} = [0\ 0\ 0\ 0\ 0.25\ 0.5\ 0.75\ 1\ 1\ 1\ 1]$. The blue rectangles denote the control point positions. The black dots show the waypoints obtained for setting $u = 0.25$, $u = 0.5$, and $u = 0.75$ respectively. Due to the relatively greater weight w_3, the curve moves closer to the control point $\mathbf{P}_3$.

(i.e., uniform/equidistant parametrization) does not result in an equidistant sampling in the operation space. This can also be seen by looking at the black dots in Fig. 2.2, which are not equidistant, although the corresponding parameters u are equidistant.

To achieve a spatially equidistant sampling, a chordal parametrization $\mathbf{u}_{\mathrm{C}}$ is needed [22], where the curve parameter intervals $u_{i+1} - u_i$ are selected proportional to the Euclidean distance between consecutive path points $\Delta\boldsymbol{\pi} = |\boldsymbol{\pi}_{i+1} - \boldsymbol{\pi}_i|$. The path is then traversed approximately at constant velocity.

To calculate $\mathbf{u}_{\mathrm{C}}$, we start with an uniform NURBS parameter vector $\mathbf{u}_{\mathrm{U}}$ of fixed size $|\mathbf{u}_{\mathrm{U}}| = 1001$, yielding a path Π_{U} of non-equally spaced waypoints $\boldsymbol{\pi}_{\mathrm{U},i}$. Next, the cumulative sum of this path's waypoint distances is calculated, resulting in the cumulative chord length vector

$$\Pi_{\mathrm{U}}^{\mathrm{CL}} = \begin{bmatrix} 0 & \cdots & \displaystyle\sum_{i=0}^{k} |\boldsymbol{\pi}_{\mathrm{U},i} - \boldsymbol{\pi}_{\mathrm{U},i+1}| & \cdots & \tilde{l} \end{bmatrix} \quad \text{with} \quad k = 0, \dots, |\Pi_{\mathrm{U}}| - 3,$$

where the last element $\tilde{l}$ approximates the true arc length l of the curve $\mathbf{C}(u)$.

The goal is a spatially equidistant path Π_{C} and thus a (chordal) cumu-

lative chord length vector

$$\Pi_{\mathrm{C}}^{\mathrm{CL}} = \begin{bmatrix} 0 & \cdots & \Delta\boldsymbol{\pi} k & \cdots & \tilde{l} \end{bmatrix} \quad \text{with} \quad k = 1, \ldots, \left\lfloor \frac{\tilde{l}}{\Delta\boldsymbol{\pi}} \right\rfloor - 1,$$

with $\Delta\boldsymbol{\pi}$ being the desired constant distance between two waypoints, i.e., the path resolution. The corresponding chordal parameter vector $\mathbf{u}_{\mathrm{C}}$ can finally be calculated as linear interpolation $\mathcal{L}$ of $(\Pi_{\mathrm{U}}^{\mathrm{CL}}, \mathbf{u}_{\mathrm{U}})$ at the points $\Pi_{\mathrm{C}}^{\mathrm{CL}}$

$$\mathbf{u} = \mathbf{u}_{\mathrm{C}} = \mathcal{L}(\Pi_{\mathrm{U}}^{\mathrm{CL}}, \mathbf{u}_{\mathrm{U}}, \Pi_{\mathrm{C}}^{\mathrm{CL}}).$$

Piegl *et al.* [18] give a detailed look into the definition of non-uniform rational B-spline curves and their properties. This thesis will address three properties that will play an important role in the following.

- The *convex hull property* denotes that the NURBS curve $\mathbf{C}(u)$ lies within the convex hull spanned by the control points $\mathbf{P}_i$ of the curve.
- *Local approximation* means that a slight change in a control point's position $\mathbf{P}_i$ or its weight w_i will affect the curve's shape only locally around this control point in the knot span $[u_i, u_{i+d+1}[$.
- The curve $\mathbf{C}(u)$ is *infinitely differentiable* within its knot spans. It is $d - c$ times differentiable at its knots, with c being the knot's multiplicity.

2.2 Optimization

2.2.1 General Optimization Problem

In a general formulation, an optimization problem denotes the minimization of E objective functions

$$f_i(\mathbf{v}), \quad i = 1, \ldots, E, \tag{2.3}$$

that are subject to F_{ineq} inequality constraints

$$g_j(\mathbf{v}) \geq 0, \quad j = 1, \ldots, F_{\mathrm{ineq}}, \tag{2.4}$$

and to F_{eq} equality constraints

$$h_k(\mathbf{v}) = 0, \quad k = 1, \ldots, F_{\mathrm{eq}}. \tag{2.5}$$

We differentiate between a multi-objective ($E \leq 3$) and a many-objective ($E > 3$) optimization problem depending on the number of objectives E [23]. However, if it is irrelevant whether $E \leq 3$ or $E > 3$ applies, the more common formulation *multi-objective* is used in the following.

The D-dimensional optimization vector $\mathbf{v} = \begin{bmatrix} v_1 & \dots & v_D \end{bmatrix}$ is a solution to the problem. Each entry of $\mathbf{v}$ can be limited by a lower and an upper bound

$$v_l^{(L)} \leq v_l \leq v_l^{(U)}, \quad l = 1, \dots, D. \tag{2.6}$$

The totality of all solutions $\mathbf{v}$ forms the search space $\mathbb{V}$. A set of solutions in the search space is denoted $V \subset \mathbb{V}$. We call a solution $\mathbf{v}$ feasible if it satisfies all constraints (2.4), (2.5) and variable bounds (2.6). All feasible solutions make the so-called feasible region. The objective functions (2.3) map a solution $\mathbf{v} \in \mathbb{V}$ to a point $\mathbf{o}$ in the objective space $\mathbb{O}$. Here, $O \subset \mathbb{O}$ is a set of solutions in the objective space. The solutions

$$\begin{bmatrix} \mathbf{v}_{\mathrm{ex},1} = \arg\min_{\mathbf{v} \in \mathbb{V}} f_1 & \dots & \mathbf{v}_{\mathrm{ex},E} = \arg\min_{\mathbf{v} \in \mathbb{V}} f_E \end{bmatrix}$$

that minimize the objective functions (2.3) separately are called extreme solutions in the search space, and extreme points $\begin{bmatrix} \mathbf{o}_{\mathrm{ex},1} & \dots & \mathbf{o}_{\mathrm{ex},E} \end{bmatrix}$ in the objective space. A visualization can be found in Fig. 2.3 Generally, we try to find a solution that minimizes all E objective functions together. This desired point in the objective space is called *utopia point*

$$\mathbf{o}_{\mathrm{U}} = \begin{bmatrix} \mathbf{e}_1^{\top} \mathbf{o}_{\mathrm{ex},1} = \min f_1 & \dots & \mathbf{e}_E^{\top} \mathbf{o}_{\mathrm{ex},E} = \min f_E \end{bmatrix}$$

that can be composed of the extreme points and the unit vectors $\mathbf{e}_i$ of the objective space. The corresponding utopia solution usually does not exist or is not feasible for practical problems. Instead, a good solution regarding one objective function is often bad regarding another and vice versa. Therefore, the optimization community introduced the concept of Pareto dominance.

A solution $\mathbf{v}_1$ Pareto-dominates another solution $\mathbf{v}_2$ ($\mathbf{v}_1 \preccurlyeq \mathbf{v}_2$) if $\mathbf{v}_1$ is not worse than $\mathbf{v}_2$ for all objectives and $\mathbf{v}_1$ is strictly better than $\mathbf{v}_2$ in at least one objective. All solutions $\mathbf{v}$ not Pareto-dominated by another solution form the set of non-dominated solutions $V_{\mathrm{NDS}} \subset \mathbb{V}$. If we map the set of non-dominated solutions into the objective space, we obtain the Pareto set $O_{\mathrm{NDS}} \subset \mathbb{O}$. The ideal Pareto set containing all non-dominated solutions to a problem is called Pareto front O_{Pareto}.

The counterpart to the utopia point is the *nadir point* $\mathbf{o}_\mathrm{N}$ that consists of the worst separate objective function values

$$\mathbf{o}_\mathrm{N} = \begin{bmatrix} \max f_1 & \dots & \max f_E \end{bmatrix}.$$

The output of a multi-objective optimization algorithm is usually a Pareto set. It is then up to a decision maker to select a solution from this Pareto set that will ultimately be used for further purposes. This process is called *decision making* and can be done based on Pareto set metrics or preference-based algorithms that go beyond the scope of this work [24]. A simple decision making procedure, which will also be used in this thesis, is the selection of the so-called *knee point* as a solution from the Pareto set. There are several ways to calculate the knee point [24]. In this thesis, the method by Sun *et al.* [25] is adopted calculating the knee point

$$\mathbf{o}_\mathrm{knee} = \underset{\mathbf{o} \in O_\mathrm{NDS}}{\arg\min} ||\mathbf{o} - \mathbf{o}_\mathrm{U}||_2$$

as the solution $\mathbf{o}$ that has the smallest Euclidean distance to the utopia point $\mathbf{o}_\mathrm{U}$. It often realizes a suitable compromise between the different objective functions.

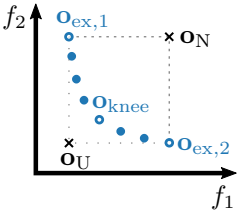


Figure 2.3: Exemplary two-dimensional Pareto set O_NDS (blue elements) with annotated knee point and extreme points, as well as the corresponding utopia and nadir point.

Metrics for Pareto Front Evaluation

There are various metrics to evaluate the quality of a multi-objective optimizer. Usually, the quality of the Pareto set generated by the solver is used for this purpose. Three common metrics are applied in this thesis:

Hypervolume (HV) The HV metric was first used by Zitzler *et al.* [26] under the name *"Size of the space covered"* [27] to quantify the quality of Pareto sets. It measures the E-dimensional volume between the E-dimensional solution set $O \subset \mathbb{O}$ and a user-defined reference point $\mathbf{o}_{\mathrm{HV,ref}} \in \mathbb{O}$ and can be defined as [28]

$$m_{\mathrm{HV}}(O, \mathbf{o}_{\mathrm{HV,ref}}) = \mathcal{A}\left(\bigcup_{\substack{\mathbf{o} \in O \\ \mathbf{o} \leq \mathbf{o}_{\mathrm{HV,ref}}}} [\mathbf{o}, \mathbf{o}_{\mathrm{HV,ref}}] \right),$$

the Lebesgue measure $\mathcal{A}$ of the union of all boxes $[\mathbf{o}, \mathbf{o}_{\mathrm{HV,ref}}] = \{\mathbf{p} \in \mathbb{O} | \, \mathbf{o} \leq \mathbf{p} \wedge \mathbf{p} \leq \mathbf{o}_{\mathrm{HV,ref}}\}$ spanned by $\mathbf{o}$ and $\mathbf{o}_{\mathrm{HV,ref}}$.

If a set of solutions in the objective space has a higher hypervolume than another for the same reference point, this indicates a better convergence towards the Pareto front O_{Pareto}.

Generational Distance (GD) The GD metric by Van Veldhuizen *et al.* [29] can be expressed as

$$m_{\mathrm{GD}}(O, O_{\mathrm{R}}) = \frac{\sqrt{\sum_{\mathbf{o} \in O} \min_{\mathbf{o}_{\mathrm{R}} \in O_{\mathrm{R}}} ||\mathbf{o} - \mathbf{o}_{\mathrm{R}}||_2^2}}{|O|},$$

and measures the distance of the elements $\mathbf{o}$ in a set O of examined solutions towards the nearest solutions $\mathbf{o}_{\mathrm{R}}$ of a reference set O_{R}. The lower the GD metric, the closer in the objective space are the solutions of the examined set of solutions to the reference set. Thus, a lower GD value indicates a better convergence to the reference set, ideally the Pareto front.

Inverted Generational Distance (IGD) As the name suggests, the IGD indicator is the counterpart to the GD metric. Introduced by Coello *et al.* [30] as

$$m_{\mathrm{IGD}}(O, O_{\mathrm{R}}) = \frac{\sum_{\mathbf{o}_{\mathrm{R}} \in O_{\mathrm{R}}} \min_{\mathbf{o} \in O} ||\mathbf{o} - \mathbf{o}_{\mathrm{R}}||_2}{|O_{\mathrm{R}}|},$$

the IGD metric sums up the distances of all reference solutions $\mathbf{o}_{\mathrm{R}}$ to the nearest examined solution $\mathbf{o}$ in the objective space. The lower the Inverted Generational Distance (IGD) indicator, the closer in the objective space are the solutions of a reference set into the examined set. Thus, a lower IGD value shows better diversity and convergence towards a reference set, ideally the Pareto front.

2.2.2 General Path Planning Problem

In a specific[1] formulation, a path planning problem is defined by a given operation space $\mathbb{D} \subset \mathbb{R}^{n_\mathbb{D}}$, an obstacle region $\mathbb{D}_{\text{obs}} \subset \mathbb{D}$, a start point $\boldsymbol{\pi}_{\text{s}} \in \mathbb{D}$, a goal point $\boldsymbol{\pi}_{\text{g}} \in \mathbb{D}$, E objective functions $f_i(\Pi)$, and F constraint functions $g_j(\Pi)$, which measure the quality and feasibility of a path. The path planning problem aims to find a path Π that optimizes the given objectives f_i and satisfies the given constraints g_j.

2.2.3 Path Planning Algorithms

As the name suggests, $E = 1$ applies to single-objective path planning problems. They have been well-studied for mobile robotic applications in the last decades [31]. Solution approaches often account for finding the shortest and collision-free path for a single robot known as the only agent in a cluttered environment. The problem's only objective function is a distance function in the Euclidean space (e.g., Euclidean distance or Manhattan distance). The only constraint of the problem is the collision avoidance. A selection of classic path finding algorithms for solving such problems is given by potential field approaches [32], the Probabilistic Road Map (PRM) [33], or the Rapidly-exploring Random Tree (RRT) [34]. The interested reader is referred to the comprehensive overview by LaValle [35].

A rising interest in Multi-objective Path Planning (MOPP) approaches has been developing with the increasing integration of autonomous agents into human living and working spaces, where different robots and humans act in the same environment. Such human-machine systems must not only minimize Euclidean distances and avoid obstacles. They lead to more complex path planning problems with additional criteria describing, for example, human interests in the robot's environment (e.g., safety).

This need to evaluate and optimize paths from the perspective of different stakeholders has given rise to the research field of multi-criteria path planning, to which the first part of this thesis also belongs. The following Section 2.3 gives an overview of various solution methods.

In the following, the principles of two different classes of algorithms are presented, which will play an essential role in this work.

[1] In the following, the path planning environment is notably more extensive than the agent size. Moreover, the paths represent corridors in which the agents can move freely. Therefore, the spatial extent and dynamics of the agents are neglected. Agents are assumed to be omnidirectional moving points. The obstacle region is expanded accordingly.

Evolutionary Algorithms

The first class of algorithms that play an important role in this thesis are Evolutionary Algorithms (EAs) [36]. They are not only tailored to path planning problems but are general optimizers. Particularly, EAs are randomized search heuristics, which makes them belong to the class of metaheuristic optimization algorithms. Two classes of EAs are the Genetic Algorithm (GA) and the Evolution Strategy (ES), which have historically been sharply differentiated but whose boundaries are becoming increasingly blurred today. Before looking at one main difference, their common basic functionality is first briefly outlined and visualized in the flowchart diagram in Fig. 2.4 For a detailed introduction, please refer to relevant literature [37]. As their name suggests, the GA and the ES both

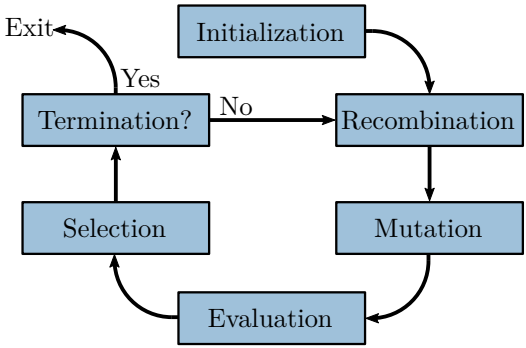


Figure 2.4: Flowchart diagram of a typical Genetic Algorithm (GA) or an Evolution Strategy (ES)

find inspiration in the concepts of evolutionary theory in order to speed up the iterative search process. The search starts (*initialization*) with a set of randomized candidate solutions (*population*). This set is updated in every iteration (*generation*). The update consists of three processes. First, solutions (*individuals*) from the old solution set are randomly modified (*reproduction*), either by combining two solutions to build a new one (*recombination*) or by slightly adapting a single solution (*mutation*). This process allows for a volatile change in the solution vector and an exploratory search behavior. Second, the objective function (*fitness*) value (for $E = 1$) or vector (for $E > 1$) for all newly created individuals is determined (*evaluation*) according to the optimization problem. Third, some solutions are chosen in a sampling process (*selection*) to form the new

solution set for the next iteration. During selection, solutions that have better fitness are preferably selected. This preference for better solutions allows for an exploitative search behavior.

One way to distinguish between a Genetic Algorithm and a Evolution Strategy is to look at the parameters to be optimized. In a GA, these usually only consist of the optimization vector $\mathbf{v}$ of the optimization problem to be solved. In an ES, hyperparameters of the algorithm, so-called strategy parameters $\boldsymbol{\sigma}$, are also optimized together with the optimization vector $\mathbf{v}$ and are, therefore, also subject to the evolutionary recombination and mutation process. This is illustrated by the following exemplary design of an ES for multi-objective optimization, which is also used and evaluated later in Chapter 3 and Chapter 4.

Exemplary Evolution Strategy (ES)

1. The ES starts in generation $a = 0$ by initializing a population consisting of s_{par} parent individuals. Each initial individual $\mathbf{v}_0$ is accompanied by an initial strategy parameter vector $\boldsymbol{\sigma}_0$ of the same size $\dim \mathbf{v}_0 = \dim \boldsymbol{\sigma}_0 = D$. Here, the strategy parameter vector $\boldsymbol{\sigma} = \begin{bmatrix}\sigma_1 & \dots & \sigma_D\end{bmatrix}$ contains the mutation step sizes of the respective entry in $\mathbf{v} = \begin{bmatrix}v_1 & \dots & v_D\end{bmatrix}$.

2. The following variation routine is applied until the resulting offspring population reaches the size s_{off}.

   a) A step size crossover averages the strategy parameter vectors $\boldsymbol{\sigma}_{a,i}$ and $\boldsymbol{\sigma}_{a,j}$ of two randomly selected individuals $\mathbf{v}_{a,i}$ and $\mathbf{v}_{a,j}$ at generation a yielding

   $$\boldsymbol{\sigma}_{a',i} = \boldsymbol{\sigma}_{a',j} = \frac{1}{2}\left(\boldsymbol{\sigma}_{a,i} + \boldsymbol{\sigma}_{a,j}\right).$$

   b) Then, a mutation operator is applied to both individuals based on the extended log-normal rule by Beyer *et al.* [38]. Each individual's strategy parameter vector is adapted first, following

   $$\boldsymbol{\sigma}_{a+1} = e^{\tau_0 \xi_0}\begin{bmatrix}\sigma_{a',1} e^{\tau \xi_1} & \dots & \sigma_{a',D} e^{\tau \xi_D}\end{bmatrix}.$$

Here, all $\xi_l \sim \mathcal{N}(0,1), \quad l = 0, \ldots, D$ are different random numbers drawn from the standard normal distribution. Furthermore, the default hyperparameters [38] $\tau_0 = 1/\sqrt{2D}$ and $\tau = 1/(2\sqrt{D})^{0.5}$ are used. Second, the optimization vector is adapted

$$\mathbf{v}_{a+1} = \left[v_{a,1} + \sigma_{a+1,1}\xi_1 \ \ldots \ v_{a,D} + \sigma_{a+1,D}\xi_D\right],$$

where $\xi_l \sim \mathcal{N}(0,1), \quad l = 1, \ldots, D$ holds again.

3. All newly created individuals are evaluated, i.e., their fitness vectors are determined.

4. The population of the $a+1$ generation's parent population is selected from the union of the current parent and offspring population. For this purpose, the non-dominated sorting selection scheme as introduced by Deb *et al.* [39] is applied, which is specifically tailored to select among individuals with multiple fitness values, i.e., to multi-objective optimization problems.

5. The steps 2-4 are repeated until a termination criterion is met.

The general, problem-detached formulation of EAs (as of metaheuristics in general) has advantages and drawbacks. The randomized nature of metaheuristics requires many objective function evaluations, which means high computational resources. Therefore, given an understanding of the optimization problem (e.g., gradients of the objective functions), dedicated optimizers that use this domain knowledge are always preferable. However, metaheuristics are often the only practical alternative to a brute force approach in all other cases - e.g., if only a calculation rule, a simulation output, or a black box model is available as an objective 'function'.

It is in the nature of any randomized search that, given sufficient calculation time, it finds the global optimum of a problem, which means that it can escape the local optima of multi-modal problems. However, it is never possible to say what *sufficient* means. In other words, metaheuristics can never guarantee that they have found the optimum[2].

In multi-objective problems, the desired solutions lie on the Pareto front.

[2] Unless they have covered the entire search space.

However, two neighboring solutions on the Pareto front (i.e., in the objective space) can be very far apart in the search space. In general, there is no knowledge to infer the position of one solution in the search space from that of the other. Therefore, evolutionary optimization methods are especially well-suited for multi-objective optimization problems. For more detailed information on evolutionary algorithms in multi-objective optimization, the reader is referred to the respective literature [40], [41].

Shortest Paths in Graphs

The operations research community developed the second type of algorithm that plays a vital role in this thesis, finding the shortest paths in graphs. Depending on the properties of the graph (e.g., (un)weighted, (un)directed, (a)cyclic), there is an algorithm specifically designed for the problem. For example, the Dijkstra Algorithm (DA) is well-suited to finding shortest paths in directed graphs with positive edge weights.

By being tailored to the specific problem of finding shortest paths in graphs, these specialized algorithms come with advantages and disadvantages. No better solution is known if the optimization problem meets the algorithm's structural assumptions. In addition, optimality guarantees can usually be made. However, applying the structural assumptions to real-world problems (i.e., breaking down the operational space to a graph structure) usually leads to errors due to the necessary simplifications made.

In addition, the complexity, and thus, the necessary computational resources of any shortest path on graph algorithm, always depend on the number of edges and nodes. In the context of real problems, this causes a conflict between the accuracy of the solution and computing time.

Dijkstra Algorithm The Dijkstra Algorithm (DA) [42] plays a decisive role in this work and is briefly introduced. Given a graph $G = \{\mathring{N}, \mathring{E}\}$ consisting of a set of nodes $\mathring{N}$ and positively weighted edges $\mathring{E}$, the Dijkstra algorithm guarantees to find the shortest path in G from a given start node in the graph $n_\mathrm{s} \in \mathring{N}$ to a given goal node $n_\mathrm{g} \in \mathring{N}$. Starting at node n_s, the Dijkstra algorithm iteratively 1) adds unvisited neighboring nodes to a search set, 2) calculates the sum of edge costs from every node in the search set to n_s, and 3) makes a step to the node n with the minimal sum of costs. The algorithm terminates when step 2) has been performed on n_g. For more information, please refer to the original publication by Di-

jkstra [42] or the illustrative introduction to the A* algorithm[3] by Russell and Norvig [44].

Multi-Criteria Shortest Paths The extension of the shortest path problem to multiple objectives is a long-standing challenge in the operations research community [45], which has produced standard algorithms such as the label-setting algorithm by Martins [46] or the label-correcting algorithms by Gurriero *et al.* [47]. However, due to their high computational complexity, the direct application of such optimal multi-objective path planning algorithms, or improved (i.e., accelerated) adaptations of these [48],[49], is not feasible for the problem dimensions, i.e., the operation space sizes tackled in this thesis[4]. So, we will not go into any more detail about them here.

2.3 Related Work

In the following section, we will review related literature in the field of multi-objective and many-objective path planning for Unmanned Aerial Vehicles (UAVs). Publications in this research field are classified into six criteria (**C1** - **C6**) introduced in the following.

1. First, there is the *UAV's environment* (**C1**) that can be rural or remote terrain with few to no people being present or an urban environment.

2. Then, the *spatial dimension* (**C2**) for the UAV path planning is either simplified to two dimensions or set to a complete three-dimensional path representation.

3. Third, the chosen *path representations* (**C3**) can be defined differently. Adjacent line segments placed in the continuous planning space and grid-based, i.e., discrete representations, are commonly used. Alternatively, researchers use spline or polynomial functions.

4. Besides, each approach presents different *formulated objectives and constraints* (**C4**) ranging from path length, energy consumption, and

[3] The A* algorithm [43] is a generalization of the Dijkstra algorithm that adds a heuristic to the cost function. The Dijkstra algorithm is equivalent to the A* algorithm with the zero heuristic function $h(n) = 0$.

[4] The graph sizes that are handled in Chapter 5 are on the order of 1 million nodes.

travel time to safety and risk-related objectives and turning angle or flight height optimization.

5. Furthermore, the *strategy to handle multiple objectives* (**C5**) can differ. Many approaches either propose a weighted aggregation of multiple objectives to obtain a single-objective optimization problem or choose a Pareto-based approach.

6. Finally, there is the type of algorithm (**C6**) utilized to tackle the formulated path planning problem.

In the following literature review, we discuss selected articles in more detail, based on which decisions are made to design the presented path planning framework in the following chapters. Moreover, a complete categorization of all reviewed papers is provided in the Tables A.1 and A.2 in the Appendix A.1.

One early approach for many-objective path planning was proposed by Nikolos *et al.* [50], who optimize UAV paths for flights over rough terrain (**C1**). They use a three-dimensional (**C2**) B-spline path representation (**C3**) and optimize the paths concerning four optimization goals (**C4**), which are collision avoidance, path length minimization, safety distance assurance, and compliance with a minimum curvature radius. However, the objective functions are aggregated to simplify the problem into a single-objective optimization (**C5**) problem, which is then solved by an adapted single-objective Genetic Algorithm (GA) (**C6**). Thus, the approach depends on the weighting of the objectives and, therefore, is limited to finding only one solution within the Pareto set of possible solutions. Moreover, only convex parts of the Pareto front can be identified with a linear aggregation of the objectives.

In a later approach, Rubio-Hervas *et al.* [51] adopt an urban setting in Singapore (**C1**) to plan three-dimensional (**C2**) paths for UAVs. They propose a special path representation (**C3**) composed of the distance and the angle between waypoints and a straight line connecting the path's start and goal point. To optimize a path regarding its length and risk (**C4**), they make us of the NSGA2 algorithm [39] (**C6**), which is a state-of-the-art evolutionary algorithm for multi-objective optimization capable of calculating a Pareto set (**C5**). On the downside, they need large computational resources and are unable to guarantee optimality.

For an urban setting (**C1**), Ghambari *et al.* [52] propose a 3D path planning (**C2**) approach that adopts a grid-based path representation (**C3**).

The objectives are to minimize the UAV's energy consumption and to maximize the distance to obstacles (**C4**). They propose a detailed energy model that assumes a larger energy consumption for higher flight altitudes due to the decreasing atmospheric density. To solve the path planning problem, Ghambari *et al.* utilize different multi-objective evolutionary algorithms (**C6**) to obtain a Pareto set of paths (**C5**). Using a grid-based path representation by design results in sharp ninety-degree or at least 45-degree turns and thus in possibly inefficient jagged paths, unsuitable for UAVs. This disadvantage can be remedied by increasing the resolution of the grid, but only at the expense of increasing the optimization problem's complexity.

In a recent study, Sadallah *et al.* [53] present an urban environment (**C1**) to do path planning of two-dimensional (**C2**) paths concerning travel time and obstacle avoidance (**C4**). In their approach, a cost distribution map is computed by applying a Fast Marching Method (FMM) twice on an obstacle map. Then, a gradient descent operation (**C6**) calculates the discrete grid-based path (**C3**) in an offline search. A graph-based A* algorithm is used in a subsequent online search to avoid dynamic obstacles. The variation of the saturation weight for the input map assures the calculation of different Pareto solutions (**C5**). Nevertheless, changing the weights of a weighted grid map does not necessarily result in non-dominated solutions in the objective space, as shown later in Chapter 3. Moreover, the utilized path representation is two-dimensional, which may lead to unsolvable problems in complex and highly occluded urban environments. However, the use of graph-based solvers, which are interesting due to their optimality guarantee, is acknowledged.

For the sake of completeness, the path planning framework developed in this thesis is already classified into the established categories here. The classification can also be traced using the blue highlighted features in Table A.1 and Table A.2 in the Appendix A.1. The conclusions drawn from all reviewed studies are incorporated into the design of the proposed framework for solving a many-objective path planning problem in a city environment (**C1**).

We begin with a two-dimensional path representation in Chapter 3. Then, the approach is extended to a three-dimensional (**C2**) path representation in Chapter 4. The extension compensates for the complex spatial restrictions of the city and enables more degrees of freedom to address demanding requirements (e.g., noise restrictions).

The utilized spline curve representation (**C3**) is smooth by definition

and thus provides paths suitable for UAVs without the need for a post-smoothing step.

The optimization framework is tested with four designed criteria (**C4**). In addition to minimizing energy consumption and the risk for residents, two additional criteria are proposed. Due to the urban setting, it makes sense to minimize radio link interference. Special attention is paid to social needs by introducing a noise minimization criterion. In addition, the flight path is restricted by a minimum flight altitude and collision avoidance with static obstacles.

In this work, all objectives should be fulfilled independently using a Pareto-based approach (**C5**).

For an adequate sampling (of also concave parts) of a Pareto set of paths, a hybrid approach (**C6**) is pursued in this thesis. A metaheuristic optimizer is adopted, often used for multi-objective path planning, and compensates for its disadvantages by coupling it with an exact graph-based search.

3 Multi-objective Two-dimensional Path Planning without Constraints

3.1 Introduction

In this chapter, we will first consider a simplified variant of the Multi-objective Path Planning (MOPP) problem addressed later in the thesis. The problem is simplified by planning only two-dimensional flight paths, which may be sufficient, for example, for operation spaces (i.e., cities) in which all buildings are lower than the specified minimum flight altitude. An Unmanned Aerial Vehicle (UAV) climbs to the minimum flight altitude, flies along a two-dimensional flight path, and lands again. A three-dimensional path planning algorithm would be too complex for this problem. Moreover, boundary conditions considered later, such as collision avoidance with static obstacles, are irrelevant and ignored in this chapter. To summarize, in this chapter we focus on the two-dimensional multi-objective path planning problem without constraints, which is extended to three dimensions and the inclusion of constraints in Chapter 4 and to more than two objectives in Chapter 5.

The literature overview on path planning in the last Chapter 2.3 showed that different classes of optimizers exist that solve the MOPP problem. Conventional path planning techniques rely on gradient-based or exact optimizers. They are fast and nearly or, under some assumptions [54], completely optimal in solving single-objective optimization problems. Nevertheless, they show drawbacks in optimizing multiple objectives or multimodal problems. In recent years, especially metaheuristic path planning approaches, like Evolutionary Algorithms (EAs), have spread [55]. EAs have been shown to perform very well on MOO problems, identifying a well-diversified Pareto set for multimodal problems. However, EAs have disadvantages as they usually need more computational resources and are not able to guarantee optimality. The following chapter proposes a new hybrid path planning framework that combines the benefits of both classes of optimizers.

Motivated by Tovey's criticism of the isolated research on metaheuris-

tics [56], in this thesis, the ambition is set to benchmark the approach not only exclusively within the research area of nature-inspired algorithms but also within the area of exact and gradient-based solvers. Therefore, in Section 3.4, the proposed Multi-objective Path Planning (MOPP) framework is compared to the state-of-the-art multi-objective Non-dominated Sorting Genetic Algorithm 2 (NSGA2) [39], as well as to the gradient-based optimizer Limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm with Bounds (L-BFGS-B) [57] and to the graph-based Dijkstra algorithm [42]. The different approaches' performance on the formulated MOPP problem is evaluated on a real-world UAV path planning scenario in the German city of Darmstadt.

In good conscience, this is the first time such a diverse set of benchmark solvers has been used to investigate the MOPP problem. This chapter demonstrates that the new hybrid method outperforms current EAs while circumventing the limitations associated with graph-based and gradient-based solvers.

Parts of this chapter have already been published in:

- [N1] N. Hohmann, M. Bujny, J. Adamy, and M. Olhofer, „Hybrid evolutionary approach to multi-objective path planning for UAVs“, in *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*, IEEE, 2021, pp. 1–8. DOI: 10.1109/SSCI50451.2021.9660187

- [N2] N. Hohmann, S. Brulin, J. Adamy, and M. Olhofer, „Three-dimensional urban path planning for aerial vehicles regarding many objectives“, *IEEE Open Journal of Intelligent Transportation Systems*, vol. 4, pp. 639–652, 2023. DOI: 10.1109/OJITS.2023.3299496

3.2 Problem Formulation

In Section 2.2.2, the general path planning problem was already defined. Following this, the multi-objective two-dimensional path planning problem will be introduced. Section 3.2.1 presents the chosen environment and path representation. Section 3.2.2 presents all objectives used throughout this thesis. Finally, the formalized multi-objective two-dimensional path planning problem is presented in Section 3.2.3.

3.2.1 Representations

Environment

In this chapter, we assume a two-dimensional, rectangular operation space

$$\mathbb{D}_{\mathrm{c}}^{2\mathrm{D}} = \begin{bmatrix} x_{\min} & x_{\max} \end{bmatrix} \times \begin{bmatrix} y_{\min} & y_{\max} \end{bmatrix} \subset \mathbb{R}^2,$$

where $x_{\min} < x_{\max}$ and $y_{\min} < y_{\max}$ are the bounding coordinates of the operational space. Furthermore, a grid-based representation of the environment is used. Therefore, the operation space $\mathbb{D}_{\mathrm{c}}^{2\mathrm{D}}$ is discretized with resolution $\begin{bmatrix} \delta_x & \delta_y \end{bmatrix}$, yielding $\mathbb{D}_{\mathrm{d}}^{2\mathrm{D}} = \{1, \dots, X\} \times \{1, \dots, Y\}$, with

$$X = \left\lfloor \frac{|x_{\max} - x_{\min}|}{\delta_x} \right\rfloor, \quad \text{and} \quad Y = \left\lfloor \frac{|y_{\max} - y_{\min}|}{\delta_y} \right\rfloor$$

respectively.

Path

A parametrized curve called Non-uniform Rational B-Spline (NURBS) introduced in Section 2.1.2 is used throughout this thesis to represent a path Π in the continuous operation space $\mathbb{D}_{\mathrm{c}}$. A clamped and uniform [58] knot vector

$$\mathbf{U} = \left[\underbrace{0 \quad \cdots \quad 0}_{d} \quad \cdots \quad \frac{1}{n_{\mathrm{CP}} - p} k \quad \cdots \quad \underbrace{1 \quad \cdots \quad 1}_{d} \right],$$

where $k = 0, \dots, n_{\mathrm{CP}} - p$ is used, ensuring that the curve is clamped to the first and last control point, which equal the start point $\mathbf{P}_0 = \boldsymbol{\pi}_{\mathrm{s}}$ and the endpoint $\mathbf{P}_{n_{\mathrm{CP}}-1} = \boldsymbol{\pi}_{\mathrm{g}}$ of the path.

With a fixed degree d and a fixed knot vector $\mathbf{U}$, the curve's shape is solely affected by the position of the curve's control points $\mathbf{P}_i = \begin{bmatrix} x_i & y_i \end{bmatrix}$ with $i \in \{1, \dots, n_{\mathrm{CP}} - 2\}$ and the weights w_i with $i \in \{0, \dots, n_{\mathrm{CP}} - 1\}$. The number of control points n_{CP} is a hyperparameter that is either set by the user or automatically determined by an algorithm, which is later presented in Section 3.5.

Thus, the vector of optimization variables for the optimization problem that is given in Section 3.2.3, is given by

$$\mathbf{v} = \begin{bmatrix} w_0 & x_1 & y_1 & w_1 & \dots & x_{n_{\mathrm{CP}}-2} & y_{n_{\mathrm{CP}}-2} & w_{n_{\mathrm{CP}}-2} & w_{n_{\mathrm{CP}}-1} \end{bmatrix}. \quad (3.1)$$

When solving the path planning problem, the optimizer changes the positions of control points and, thus, the curve's shape. The NURBS curve

representation, in the following denoted $\Pi(\mathbf{v}) = \mathbf{C}(u)|_{\mathbf{v}}$, is advantageous as paths of different shapes and lengths can be represented without changing the size D of the optimization vector and thus the complexity of the optimization problem. Furthermore, especially three NURBS properties are useful that were introduced in Section 2.1.2:

- The *convex hull property* guarantees that the path lies within the operation space $\mathbb{D}_c$ if all $\mathbf{P}_i \in \mathbb{D}_c$. Therefore, the control point positions are bounded by the size of the design domain $x_{\min} \leq x_i \leq x_{\max}$, $y_{\min} \leq y_i \leq y_{\max}$, and later also $z_{\min} \leq z_i \leq z_{\max}$, with $i = 1, \ldots, n_{CP} - 2$.

- During the path optimization process, the *local approximation property* ensures that changing one element in the optimization vector $\mathbf{v}$ will only affect the path's shape locally at a specific section. Therefore, to guide a path around local maxima, the optimizer only needs to adjust specific elements in the vector $\mathbf{v}$.

- The *infinite differentiability property* ensures the smoothness of the path up to desired derivatives. Mellinger *et al.* [59] show that the dynamics of a UAV (quadrotor) can be modeled as differentially flat allowing an analytical calculation of the quadrotor's states and control inputs from the desired flight path (including the desired yaw angle) and its derivatives [60]. The motor commands of the designed flatness-based controller are proportional to the fourth derivative of the UAV's position (i.e., snap) [59]. To avoid discontinuous steps in the control inputs and thus to decrease the chances of the system running into actuator saturation, the desired flight trajectory should, therefore, be four times differentiable [3]. Apart from the first and last control point of a clamped NURBS curve, this desired forth-order differentiability can be achieved using NURBS curves of degree $d = 5$. Note that from this desired degree and equation (2.2), for the number of control points $n_{CP} \geq 6$ follows. Contrary to highly aggressive quadrotor control as in [59], [60], this thesis focuses on urban large-scale path planning. We, therefore, assume the resulting paths to be used by drones without aggressively-tuned controllers. Not concerning actuator saturations, more emphasis is placed on a minimal path representation with only $n_{CP} = 3$ control points (two fixed, one optimizable). Then, the degree of the curve must be set to $d = 2$, which will be the default value from now on.

3.2.2 Objectives

In path planning problems, the objectives for evaluating flight paths may differ depending on the use case, a stakeholder's specific requirement, characteristics of the environment, or the time (day or night, winter or summer). Therefore, the solution framework presented later in Section 3.3 is designed to incorporate arbitrary objectives. In the following, exemplary evaluation procedures for four objectives are specified to be able to test the framework. This chapter covers two-dimensional path planning. However, for completeness, all objectives are already defined in their generalized, three-dimensional formulation. These will be used later in chapters 4 and 5. Since the main focus in this part of the thesis is on solving the path planning problem and not on designing quality criteria, the objectives are designed to be realistic but relatively simple and easy to implement. If necessary, more advanced models can easily replace the proposed objectives. In the following, grid-based and non-grid-based objectives are distinguished. Grid-based objectives can be calculated on two-dimensional grid maps $\mathcal{G}^{2\mathrm{D}}$ or on three-dimensional grid maps $\mathcal{G}^{3\mathrm{D}}$.

In general, all grid-based objective functions $f(\mathbf{v})$ integrate over a grid map $\mathcal{G}^{3\mathrm{D}}$ along the path $\Pi(\mathbf{v})$. This is formalized as trapezoidal integral

$$f(\Pi(\mathbf{v})) = \sum_{i=1}^{|\Pi|-1} \frac{\mathcal{I}(\mathcal{G}^{3\mathrm{D}}, \boldsymbol{\pi}_{i-1}) + \mathcal{I}(\mathcal{G}^{3\mathrm{D}}, \boldsymbol{\pi}_i)}{2} |\boldsymbol{\pi}_i - \boldsymbol{\pi}_{i-1}|, \tag{3.2}$$

along the path Π, with $\mathcal{I}(\mathcal{G}^{3\mathrm{D}}, \boldsymbol{\pi})$ being the tri-linear regular grid interpolation [61] of the grid map $\mathcal{G}^{3\mathrm{D}}$ at the path point $\boldsymbol{\pi}$, and $|\boldsymbol{\pi}|$ being the Euclidean norm of $\boldsymbol{\pi}$.

Every objective that can not be described by (3.2) belongs to the category of non-grid-based objectives.

Definition We define the Euclidean Distance Transform (EDT) [62] $\mathcal{T}$ on a two-dimensional binary grid map $\mathcal{B}^{2\mathrm{D}} : \{1, \dots, X\} \times \{1, \dots, Y\} \to \{0, 1\}, \quad (i, j) \mapsto b_{ij}$ as

$$\begin{aligned} &\mathcal{T} : \{0,1\}^{X \times Y} \to \mathbb{R}^{X \times Y}, \quad b_{ij} \mapsto g_{ij} \\ &\text{with} \quad g_{ij} = \min_{k,l} \sqrt{(i-k)^2 + (j-l)^2}, \quad \text{s.t.} \quad b_{kl} = 1, \\ &\text{with} \quad (k, l) \in \{1, \dots, X\} \times \{1, \dots, Y\}. \end{aligned} \tag{3.3}$$

The EDT $\mathcal{T}$ calculates the distance from every cell in $\mathcal{B}^{2\mathrm{D}}$ to its nearest cell containing the value one.

In the following, four objective functions (three grid-based and one non-grid-based) are introduced, all of which are to be minimized. Visualizations for all grid maps are in the Appendix A.2.

Risk of Injury

Formally speaking, *risk* is defined as the product of damage and probability of occurrence. If the latter is assumed to be constant, we can model risk as the damage that a UAV can do to a human if it crashes. In case of a quadrotor's severe malfunction that leads to a crash, we assume a forced landing on a roof or over a water surface to be less risky than in other areas of a city. The three-dimensional continuous space $\mathbb{D}_{\mathrm{c}}^{3\mathrm{D}}$ where we want to plan paths is discretized to the discrete space $\mathbb{D}_{\mathrm{d}}^{3\mathrm{D}}$ containing cells with different risk values assigned. Consequently, spaces with little risk are defined as those over water surfaces and buildings (excluding special education buildings, hospitals, military and railway buildings, and places of worship). According to the risk objective function, the UAV is supposed to fly through low-risk cells.

The process of deriving the risk grid map starts with a two-dimensional, binary building grid map $\mathcal{B}_{\mathrm{B}}^{2\mathrm{D}}$ that consists of one-valued cells, where a building is located in the cell's spatial position, and of zero-valued cells, where no building is located. Then, the two-dimensional risk grid map $\mathcal{G}_{\mathrm{R}}^{2\mathrm{D}}$ is calculated by applying the Euclidean Distance Transform (EDT) (3.3) to the building grid map.

$$\mathcal{G}_{\mathrm{R}}^{2\mathrm{D}} = \mathcal{T}(\mathcal{B}_{\mathrm{B}}^{2\mathrm{D}}).$$

Thus, the risk map contains zero-valued cells where buildings are located and obtains a gradient towards those low-risk cells. Furthermore, cells of the risk grid map located in water areas are set to zero. In contrast, cells with special buildings like hospitals are replaced with values inclined towards the highest risk value at the center of special buildings. Then, the mapping (2.1) is applied to obtain a positive grid map. An exemplary two-dimensional risk grid map is visualized in Fig. A.16 in the Appendix A.2.

For the three-dimensional risk grid map $\mathcal{G}_{\mathrm{R}}^{3\mathrm{D}}$, the risk values for the z-dimension of the grid map need to be calculated. The uncertainty in determining the impact position of a crashing drone increases with the flight altitude of the UAV. Therefore, we derive the three-dimensional risk grid map $\mathcal{G}_{\mathrm{R}}^{3\mathrm{D}} : \mathbb{D}_{\mathrm{d}}^{3\mathrm{D}} \to \mathbb{R}^{+}$ by applying a two-dimensional maximum filter with a 3×3-window to $\mathcal{G}_{\mathrm{R}}^{2\mathrm{D}}$ layer by layer.

By calculating the line integral (3.2) over $\mathcal{G}_{\mathrm{R}}^{3\mathrm{D}}$ along the path $\Pi(\mathbf{v})$, we obtain the risk objective function $f_{\mathrm{R}}(\mathbf{v})$.

Noise Immission

Torija *et al.* [63] show that city residents perceive noise generated by aerial vehicles as less annoying if their flight paths lead over the city streets as the flight noise then vanishes in the traffic noise. Therefore, the noise violation objective function is modeled to favor paths that lead over streets. The starting point is a street grid map $\mathcal{B}_{\mathrm{S}}^{2\mathrm{D}}$ consisting of one-valued cells, where a street is located at the cell's spatial position, and zero-valued cells, where no street is located. Then, the Euclidean Distance Transform (EDT) (3.3) is applied to the street grid map

$$\mathcal{G}_{\mathrm{N}}^{2\mathrm{D}} = \mathcal{T}(\mathcal{B}_{\mathrm{S}}^{2\mathrm{D}}).$$

Thus, the resulting noise grid map contains zero-valued cells along the city streets and, apart from those, a gradient towards the city roads. Additionally, the noise grid map cells located in water areas or industrial regions are set to zero. In contrast, cells in the center of parks or residential areas are set to the highest noise value. From there, the noise values gradually decline towards the streets. Then, the mapping (2.1) is applied to obtain a positive grid map. A visualization of an exemplary noise grid map is given in Fig. A.17 in the Appendix A.2.

To calculate the three-dimensional noise grid map, we assume $\mathcal{G}_{\mathrm{N}}^{2\mathrm{D}}$ to be a layer of the three-dimensional noise grid $\mathcal{G}_{\mathrm{N}}^{3\mathrm{D}} : \mathbb{D}_{\mathrm{d}}^{3\mathrm{D}} \to \mathbb{R}^{+}$ at the UAV's minimum flight height $z_{\mathrm{F,min}}$. Perceived noise decreases quadratically with distance. To derive the remaining layers of $\mathcal{G}_{\mathrm{N}}^{3\mathrm{D}}$, we apply the inverse square law

$$\mathcal{G}_{\mathrm{N}}^{3\mathrm{D}}(z) = \begin{cases} \mathcal{G}_{\mathrm{N}}^{2\mathrm{D}}, & \text{if} \quad \Delta z \leq 0 \\ \dfrac{\mathcal{G}_{\mathrm{N}}^{2\mathrm{D}}}{(\Delta z + 1)^2}, & \text{else} \end{cases}$$

with $\Delta z = z - z_{\mathrm{F,min}}$. Finally, calculating the line integral (3.2) over the noise grid map $\mathcal{G}_{\mathrm{N}}^{3\mathrm{D}}$ along the path $\Pi(\mathbf{v})$ yields the noise objective function $f_{\mathrm{N}}(\mathbf{v})$.

Radio Signal Disturbance

The ability to continuously track the operating status (e.g., battery level) and the sensor measurements (e.g., pose, velocity) of the UAV, and the option to set a new flight trajectory, if necessary, enable safe flight operations. Therefore, we assume the need for a permanent radio connection between the aerial vehicle and a ground station, which makes a stable connection to a cell tower essential. We assume the radio signal disturbance at the position $\mathbf{p}_\mathrm{R}$ of a cell tower to attain an arbitrarily chosen best value of $D_0 = -100$. The signal disturbance increases following the inverse square law. Dependent on the position of a single radio cell tower $\mathbf{p}_\mathrm{R} \in \mathbb{D}_\mathrm{c}^{3\mathrm{D}}$, the radio signal disturbance at a point $\mathbf{p} \in \mathbb{D}_\mathrm{c}^{3\mathrm{D}}$ is qualitatively calculated according to

$$D(\mathbf{p}) = \frac{D_0}{(\rho r + 1)^2},$$

with $r = |\mathbf{p}_\mathrm{R} - \mathbf{p}|$ being the Euclidean distance between the point $\mathbf{p}$ and the radio cell tower position, and $\rho \in \mathbb{R}^+$ being a scaling factor that is arbitrarily set to $\rho = 0.01$ in this thesis. This results in a three-dimensional grid map for a single cell tower. Next, the grid maps for all cell towers in the area are merged by a minimum operator. Applying the mapping (2.1) yields a three-dimensional radio disturbance map $\mathcal{G}_\mathrm{D}^{3\mathrm{D}} : \mathbb{D}_\mathrm{d}^{3\mathrm{D}} \to \mathbb{R}^+$. An example of a radio disturbance grid map $\mathcal{G}_\mathrm{D}^{3\mathrm{D}}$ at the height of the radio tower cells $z = z_\mathrm{R}$, set to $z_\mathrm{R} = 80\,\mathrm{m}$ in this thesis, is provided in Fig. A.18 in the Appendix A.2.

When planning a path through the radio disturbance map, we prefer to fly through cells with lower signal disturbance. Thus, we compute the radio disturbance objective function $f_\mathrm{D}(\mathbf{v})$ by calculating the trapezoidal integral (3.2) over $\mathcal{G}_\mathrm{D}^{3\mathrm{D}}$ along the path $\Pi(\mathbf{v})$.

Energy Consumption

The energy consumption model used in this thesis for a UAV following a path Π is based on the physical energy consumption considerations for drones by Reid [64]. He considers energy models for the different flight states of a quadrotor, which are climbing, hovering, and forward motion. We assume that the UAV flies at a reduced speed $v_{\mathrm{z},\uparrow}$ during climb compared to its speed v_xy at level flight. Furthermore, the aerial vehicle must even fly at a slower speed $v_{\mathrm{z},\downarrow}$ during descent to avoid air turbulence underneath the rotors, which leads to an unstable flight.

Table 3.1: Vehicular and physical parameters for the energy consumption model

Parameter	Symbol	Value
Horizontal cruise velocity	v_{xy}	$14\,\mathrm{m/s}$
Ascending velocity	$v_{\mathrm{z},\uparrow}$	$2\,\mathrm{m/s}$
Descending velocity	$v_{\mathrm{z},\downarrow}$	$1\,\mathrm{m/s}$
UAV mass	m	$1.2\,\mathrm{kg}$
Drag coefficient	c_{d}	0.6
Drag area	A_{d}	$0.1\,\mathrm{m}^2$
Number of rotors	n_{r}	4
Rotor radius	r_{r}	$0.1\,\mathrm{m}$
Air Density	ρ_{air}	$1.225\,\mathrm{kg/m}^3$
Constant of gravitation	g	$9.81\,\mathrm{m/s}^2$

From Reid's [64] energy models and arbitrarily but realistically chosen remaining drone parameters, shown in Table 3.1, a simplified energy consumption model can be derived. For this purpose, the UAV's smooth flight path Π in the three-dimensional space is projected into the xy-plane resulting in a two-dimensional path Π_{xy}. Furthermore, the path's projection onto the z-axis is divided into the segments that point upwards $\Pi_{\mathrm{z},\uparrow}$ and those that point downwards $\Pi_{\mathrm{z},\downarrow}$. Assuming that a drone with the given parameters and velocities flies off from a standstill along these path projections results in the energy consumption model

$$f_{\mathrm{E}}(\mathbf{v}) = \frac{1}{2} m {v_{\mathrm{xy}}}^2 + c_{\mathrm{E}} \left(|\Pi_{\mathrm{xy}}| + 10|\Pi_{\mathrm{z},\uparrow}| + 15|\Pi_{\mathrm{z},\downarrow}| \right), \tag{3.4}$$

where $|\cdot|$ is the Euclidean length of the path projections, m is the UAV mass, and $c_{\mathrm{E}} = 9.12\,\mathrm{J/m}$ denotes a vehicle-specific energy parameter.

Grid Transformation Later in this thesis, it becomes apparent that a grid-based representation of an objective is useful for optional preprocessing steps. It makes sense to specify a grid transformation for non-grid-based objectives.

For an accurate grid transformation of the energy consumption objective (3.4), a three-dimensional energy grid map has to be created, whose cell entries should be dependent on the direction of movement of the UAV. In a grid map, this dependency on the direction of movement can not be modeled. Instead, a grid graph G is used to generalize the grid.

To derive an energy graph $G_{\mathrm{E}}^{3\mathrm{D}}$ that approximates the energy model (3.4) the directed edges $(n_{ijk}, n_{i'j'k'})$ of a three-dimensional grid graph $G^{3\mathrm{D}}$ with 18-neighborhood are weighted accordingly. All edges pointing in positive z-direction are weighted with

$$\omega_{ijki'j'k'} = 10, \quad \forall\, i' = i,\, j' = j,\, k' = k + 1.$$

Edges pointing in negative z-direction are weighted

$$\omega_{ijki'j'k'} = 15, \quad \forall\, i' = i,\, j' = j,\, k' = k - 1.$$

The weights of edges along x and y direction are set to

$$\omega_{ijki'j'k'} = 1, \quad \forall\, i' = i \pm 1,\, j' = j \pm 1,\, k' = k.$$

The weights of all remaining diagonal edges are calculated accordingly. An exemplary graph section is visualized in Fig. 3.1.

3.2.3 Optimization Problem

For the design of a multi-objective planning problem in two dimensions, it seems helpful to choose objective functions that favor conflicting path configurations in the two-dimensional planning space. The improvement of one objective should then lead to a high probability of the deterioration of the other objective, preventing the problem from degenerating into a pseudo-multi-objective problem as it can happen with correlated objective functions. Consequently, solving the problem should result in a broad Pareto set.

The *Risk of Injury* objective function $f_{\mathrm{R}}(\mathbf{v})$, introduced in Section 3.2.2, favors paths over buildings, while the *Noise Immission* objective function $f_{\mathrm{N}}(\mathbf{v})$ introduced in Section 3.2.2 rewards paths over roads (i.e., absence of buildings). Therefore, both are good candidates for satisfying the requirement of non-correlated objective functions. The vector $\mathbf{v}$ of optimization variables (3.1) has already been introduced in Section 3.2.1. The optimization problem discussed in this chapter is then modeled as

$$\min_{\mathbf{v}\in\mathbb{V}} \begin{cases} f_{\mathrm{R}}(\mathbf{v}), \\ f_{\mathrm{N}}(\mathbf{v}), \end{cases} \tag{3.5}$$

where the search space

$$\mathbb{V} = \underbrace{[\mathbb{D}_{\mathrm{c}}^{2\mathrm{D}}] \times \ldots \times [\mathbb{D}_{\mathrm{c}}^{2\mathrm{D}}]}_{n_{\mathrm{CP}}-2} \times \underbrace{[\mathbb{R}^{+}] \times \ldots \times [\mathbb{R}^{+}]}_{n_{\mathrm{CP}}} \tag{3.6}$$

has $\dim(\mathbb{V}) = 2(n_{\mathrm{CP}} - 2) + n_{\mathrm{CP}}$ dimensions.

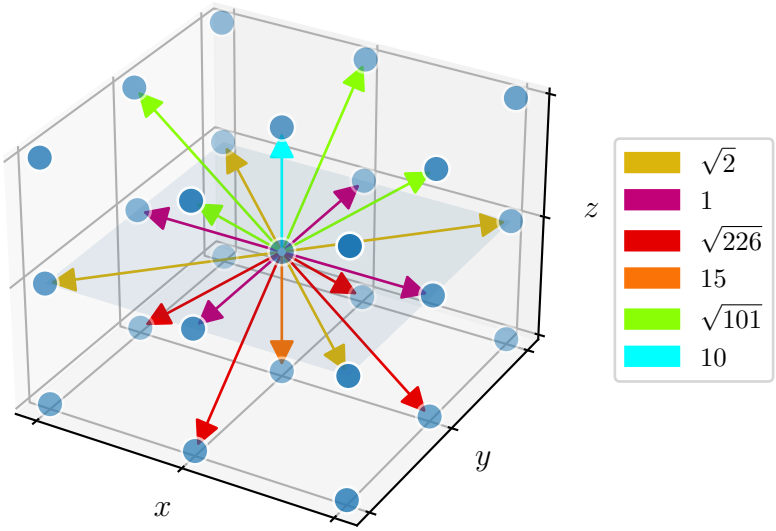


Figure 3.1: According to the energy model (3.4), the costs for traveling along an edge in the energy graph depend on the movement direction of the UAV.

3.3 Solution Approach

3.3.1 Problem Analysis

A simple example will illustrate the complexity of the optimization problem. For this purpose, the number of control points is set to $n_{\mathrm{CP}} = 3$, and all weights to $w_i = 1$, resulting in a two-dimensional optimization problem with $\mathbf{v} = \begin{bmatrix} x_1 & y_1 \end{bmatrix}$. We assume an exemplary two-dimensional grid map as shown in Fig. 3.2, where lower (i.e., better) objective function values are visualized in dark blue, and higher values in yellow. The first and last control points, i.e., the start and endpoints of the path, are fixed in two corners of the operation space, respectively. As shown in the figure, the path's shape changes depending on the middle control point's x- and y-coordinates. The paths are marked as solid lines, and dashed lines connect

the control points. If we push the variable control point over the entire

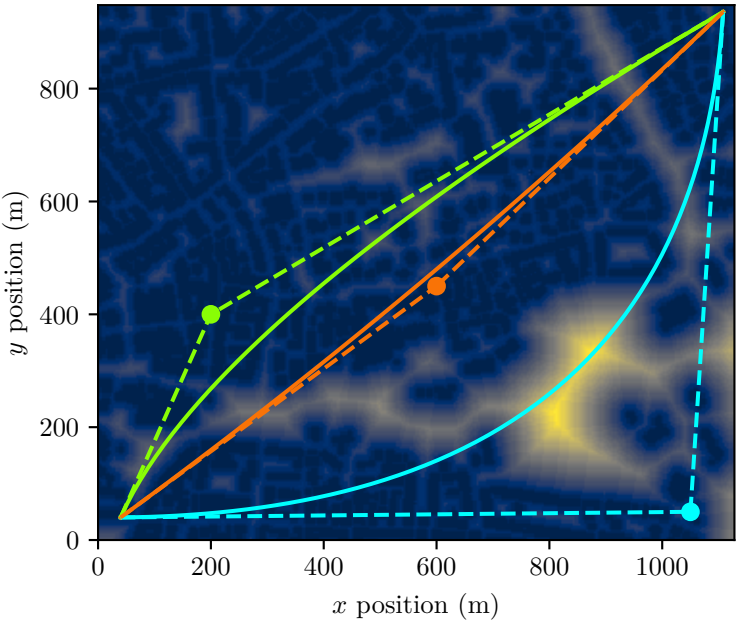


Figure 3.2: The shape of the three visualized paths (solid lines) depends on the position $\begin{bmatrix} x_1 & y_1 \end{bmatrix}$ of the variable middle control point (the dot that is connected by dashed lines with the start end endpoint of the path).

operation space and compute the objective function, in this case, the line integral (3.2) over the given grid map along the respective path, we obtain the optimization landscape shown in Fig. 3.3. The objective function values of the three example paths are visualized as colored dots accordingly. If we look at the blue path, it is longer than the others and leads through an area with high grid values (yellow), resulting in the maximum (blue dot) in the optimization landscape. The orange path does cross a yellow area at $\begin{bmatrix} 300 & 200 \end{bmatrix}$, but it is very short and forms a local minimum (orange dot) in the cost landscape. Although the green path bypasses the yellow region at $\begin{bmatrix} 300 & 200 \end{bmatrix}$, it is longer. It passes through another yellow region at $\begin{bmatrix} 590 & 580 \end{bmatrix}$, resulting in a higher line integral and a local maximum (green dot) in the cost landscape.

The cost landscape of the exemplary optimization problem already reveals local extrema. Accordingly, the tackled multi-dimensional optimization problem (3.5) is not to be considered multi-objective alone, but also multi-modal and thus non-convex.

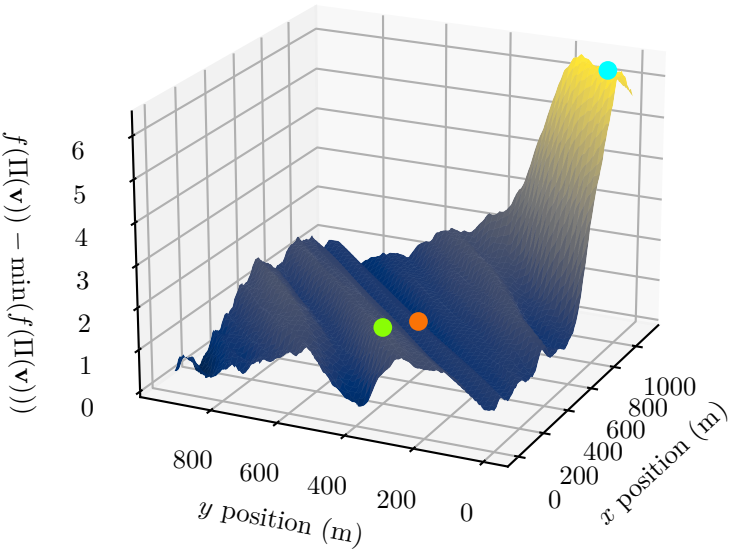


Figure 3.3: The optimization landscape for the described two-dimensional path planning problem is multi-modal and thus non-convex. The objective function values $f(\Pi(\mathbf{v})) - \min(f(\Pi(\mathbf{v})))$ of the three exemplary paths are visualized as colored dots and amount to $f_{\text{blue}} = 9.7$, $f_{\text{green}} = 5.1$, and $f_{\text{orange}} = 4.2$.

3.3.2 Solution

To solve the presented multi-objective and multi-modal path optimization problem, the Multi-objective Path Planning (MOPP) framework proposed in this thesis utilizes the advantages of two different solution methods, namely an Multi-objective Evolutionary Algorithm (MOEA) and the Di-

jkstra Algorithm (DA) [42]. Both approaches are combined in one hybrid[1] approach. The idea is that the two different solution methods complement each other and compensate for each other's disadvantages:

Evolutionary Algorithms (EAs) are particularly effective in solving non-convex multi-objective optimization problems since the randomized search is well suited to approximate Pareto fronts [65]. At the same time, however, this search method requires many computational resources. In order to simplify the search, the Dijkstra Algorithm (DA) will be built in front of the randomized search algorithm. The DA efficiently solves single-objective shortest path problems with guaranteed optimality. The disadvantage of the DA is its restriction to graph-based environments and thus to graph-based path representations.

In the proposed hybrid MOPP framework, the DA calculates n_{WS} weighted solutions to the multi-objective problem (3.5) that is beforehand transformed into a weighted and aggregated single-objective optimization problem

$$f_{\mathrm{A}}(\mathbf{v}) = \lambda_1 f_1(\mathbf{v}) + \ldots + \lambda_E f_E(\mathbf{v}), \tag{3.7}$$

where $\sum_{i=1}^{E} \lambda_i = 1$ holds. Assuming that the resulting solutions form good initial solutions for an MOEA despite a different representation, the graph-based solutions Π_{d} are converted into the desired path representation Π. Conversion errors are corrected by the MOEA[2], and at the same time, the pre-processing step facilitates the sampling of a good Pareto front.

The complete solution pipeline will be presented in the following.

Pipeline

The start point $\boldsymbol{\pi}_{\mathrm{s}}$ and goal point $\boldsymbol{\pi}_{\mathrm{g}}$ of the path and all grid maps used to calculate the E objectives are input into the framework. The Dijkstra Algorithm (DA) [42] and a Multi-objective Evolutionary Algorithm (MOEA), which the user can choose, constitute the two main steps (S1, S2) in the pipeline as it is visualized in Fig. 3.4. The pipeline consists of the following steps:

[1] Wherever the term *hybrid* is used in this thesis, it refers to the Multi-objective Path Planning (MOPP) framework proposed here.

[2] Within the proposed hybrid framework, any Multi-objective Evolutionary Algorithm (MOEA) can be used. In the following evaluation in Section 3.4, several state-of-the-art algorithms are tested either individually or within the hybrid framework. The latter will always be denoted as *hybrid* or with the prefix H.

1. In the *Grid Transfer* step, non-grid-based objectives are, if possible, converted into a grid or graph representation as it was done in Section 3.2.2 for the energy objective. Hence, the grid conversion step also makes the DA applicable for non-grid-based objectives. If the grid transfer is impossible, this step can be omitted for the respective objective. It is, therefore, not used in the Dijkstra step S1.

2. In the *Weighted Aggregation*, the original multi-objective optimization problem is converted into the single-objective optimization problem (3.7). Note that the subsequent DA is by design not suited to solve (3.7) directly. Instead, it is applied to the weighted aggregations of the grids (or grid transfers) $\mathcal{G}^{3D}$ belonging to the respective objective functions $f(\mathbf{v})$. Thus, the problem can be formulated as

$$G_{\mathrm{A}}^{3D} = \lambda_1 G_1^{3D} + \ldots + \lambda_E G_E^{3D}. \tag{3.8}$$

   Here, the generalized graph notation of the grid maps is used, as we have already seen in Section 3.2.2 that grid conversions exist that can only be expressed as graphs and not as grids anymore.

3. Then, in the first main step S1, the DA [42] is applied n_{WS} times, which provides n_{WS} optimal solutions (i.e., paths) for the weighted and aggregated single-objective problem (3.8). The calculated paths are discrete, meaning that the path can only run precisely along the edges of the underlying graphs G^{3D}.

4. In the *Smoothing & Approximation* step, these discrete paths are transformed into continuous NURBS curves. This step will be deepened in the following section.

5. The splines are the initial solutions in a MOEA, the second main step S2 of the pipeline, which solves the actual multi-objective problem (3.5) regarding the original (i.e., not grid-transferred) non-grid-based objectives. It results in a set of trade-off solutions that form the E-dimensional Pareto set. This set allows the user to select an optimal solution depending on preferences.

Smoothing & Approximation

At this point, we will briefly discuss the *Smoothing & Approximation* step of the pipeline. Its input is a three-dimensional discrete path Π_{d} that

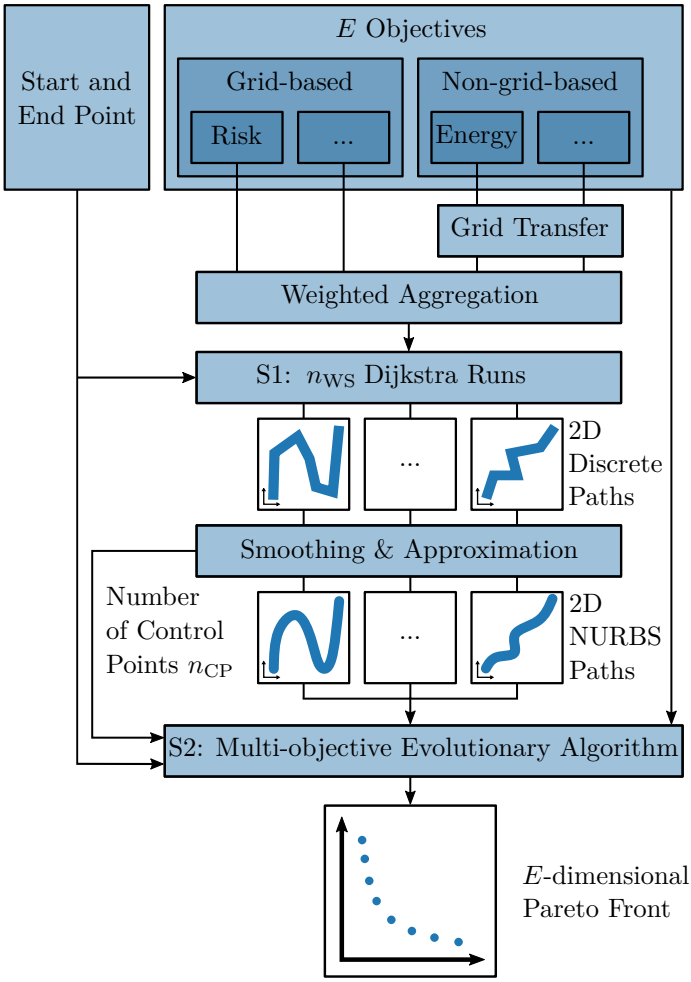


Figure 3.4: Diagram of the proposed hybrid optimization framework. Depending on the objective class (grid-based or non-grid-based), the corresponding grids are pre-processed before the Dijkstra Algorithm (DA) finds n_{WS} optimal solutions for the weighted aggregation of the multi-objective optimization problem. Then, the obtained solutions are smoothed and approximated by NURBS curves. Finally, the different smooth paths serve as initial solutions for a Multi-objective Evolutionary Algorithm (MOEA) set by the user, which generates a Pareto set of trade-off paths for the original multi-objective optimization problem.

should be represented, i.e., approximated by a NURBS curve, yielding a continuous path Π. For a more robust approximation, the discrete path is therefore smoothed by applying a Gaussian filter with kernel $\mathcal{K} = \begin{bmatrix} 1 & 2 & 1 \end{bmatrix}$ three times. Then, a NURBS curve approximates the smoothed path [66]. For this operation, the crucial parameter is the number of control points n_{CP} of the resulting NURBS curve. If n_{CP} is too small, the approximation error becomes too large. The resulting path has nothing in common with the original Dijkstra path, which makes the Dijkstra calculation obsolete. Whereas, if n_{CP} is too large, this will increase the size of the optimization vector $\mathbf{v}$ and, therefore, the search space in the multi-objective optimization step.

The number of control points n_{CP} is thus an important hyperparameter set by the user. Later in Section 3.5, a procedure is shown to determine the parameter adaptively.

3.4 Evaluation

This section evaluates the hybrid Multi-objective Path Planning (MOPP) framework proposed in this thesis for solving the multi-objective two-dimensional path planning problem. The following Section 3.4.1 presents the test scenario, algorithms for comparison, and their parameters before the evaluation results are presented and discussed in Section 3.4.2. Finally, Section 3.4.3 discusses an identified problem.

3.4.1 Setup

Scenario

The evaluation is based on OpenStreetMap (OSM) [67] data of a map section of the city of Darmstadt, which is visualized in Fig. A.1. The OSM data is used to sample different low-level grid maps, representing, e.g., the city's road network, semantic information as displayed in Fig. A.2, or the height of buildings shown in Fig. A.3. The derived high-level grid maps for risk $\mathcal{G}_{\mathrm{R}}^{\mathrm{2D}}$ and noise $\mathcal{G}_{\mathrm{N}}^{\mathrm{2D}}$ with the resolution $\begin{bmatrix} \delta_x & \delta_y \end{bmatrix}$ are shown in figures A.4 and A.5 and form the basis for the calculation of the two objective functions $f_{\mathrm{R}}(\mathbf{v})$ and $f_{\mathrm{N}}(\mathbf{v})$ as presented in Section 3.2.2. A summary of all parameters concerning the scenario setup and the representation of the environment and the paths can be found in Table 3.2.

For the evaluation, the proposed pipeline is tested on a set of 30 different start $\boldsymbol{\pi}_{\mathrm{s}}$ and goal $\boldsymbol{\pi}_{\mathrm{g}}$ positions that are randomly distributed over the op-

eration space $\mathbb{D}_c^{2D}$ such that the start/endpoint pair are evenly distributed in space as well as in distance. Their positions can be taken from Fig. A.1, where a straight line connects every start and goal point.

Table 3.2: Scenario Parameters

	Parameter	Symbol	Value
Scenario	Origin latitude	Lat	49.8705°N
	Origin longitude	Lon	8.6585°E
	Scenario dimensions	$x_{\min}$, $x_{\max}$	0 m, 1124 m
		$y_{\min}$, $y_{\max}$	0 m, 948 m
Environment	Discretization res.	δ_x, δ_y	4 m, 4 m
Path	Num. of control points	n_{CP}	15
	Basis function degree	d	2
	Parametrization		uniform
	Number of path points	$\lvert\Pi\rvert$	150
	Initial weights	w_i	1

Solvers

In the following, the performance of different standard Multi-objective Evolutionary Algorithms (MOEAs) solving the multi-objective two-dimensional path planning problem (3.5) is compared to their performance when embedded in the hybrid Multi-objective Path Planning (MOPP) framework proposed in this thesis. The first utilized MOEA is a Genetic Algorithm (GA) called Non-dominated Sorting Genetic Algorithm 2 (NSGA2) [39] that is specially designed for multi-objective optimization problems. When this algorithm is embedded in the proposed hybrid MOPP framework, it is called Hybrid Non-dominated Sorting Genetic Algorithm 2 (H. NSGA2) in the following. The Evolution Strategy (ES) that has been introduced in Section 2.2.3 is considered as second MOEA in the evaluation. Its hybrid embedding in the proposed framework is named Hybrid Evolution Strategy (H. ES). For the two hybrid methods, in stage S1 of the framework, the number of solution runs n_{WS} of the aggregated objective function (3.7) was arbitrarily set to $n_{\text{WS}} = 5$. The weights were set to

$$\begin{bmatrix}\lambda_1 & \lambda_2\end{bmatrix} \in \left\{ \begin{bmatrix}0 & 1\end{bmatrix}, \begin{bmatrix}0.05 & 0.95\end{bmatrix}, \begin{bmatrix}0.5 & 0.5\end{bmatrix}, \begin{bmatrix}0.95 & 0.05\end{bmatrix}, \begin{bmatrix}1 & 0\end{bmatrix} \right\}.$$

A more detailed exploration of how the determination of the weights λ_i affects the framework's performance follows in Section 5.4 of this thesis.

Furthermore, the problem is also solved with the gradient-based optimizer L-BFGS-B [57], a variant of the widely-used L-BFGS algorithm [68], which is capable of incorporating bound constraints. Moreover, the developed hybrid framework is compared with the Dijkstra Algorithm (DA) [42]. As the discrete path solution of the DA does not allow a meaningful comparison to the continuous NURBS path solutions of the other approaches, another algorithm is included in the evaluation that combines the standard DA [42] with an a posteriori NURBS curve approximation[3] step. This approach will be called Approximated Dijkstra Solution (ADS) in the following.

It should be noted that the DA and the L-BFGS-B algorithm are not capable of handling more than one objective at once. Therefore, to maintain comparability, they solve the weighted and aggregated single-objective optimization problem (3.7) or (3.8) instead. A Pareto set can then be obtained by running n_{WS} optimizations for different values of the aggregation weights $\begin{bmatrix}\lambda_1 & \lambda_2\end{bmatrix}$. The n_{WS} weights are chosen equally distributed on a two-dimensional straight line intersecting the axes of the coordinate system at 1.

The hyperparameters to all described solution approaches are given in Table 3.3, with all algorithms developed in this work marked in blue. Parameters that are not listed are set to standard values from the literature. Next, we want to discuss the initialization of all utilized algorithms. Without prior knowledge, the optimization vector for any problem is usually initialized with uniformly distributed random values drawn from the search space $\mathbb{V}$. With this method, the control points would be scattered in the operation space due to the control-point-based path representation (3.1). Thus, the obtained curves would be potentially meaningless and highly sub-optimal.

This is why, for initializing all non-hybrid optimizers, the control points are sampled equidistantly on a straight line segment between start point $\boldsymbol{\pi}_{\mathrm{s}}$ and goal point $\boldsymbol{\pi}_{\mathrm{g}}$. The control points' weights are initialized with $w_i = 1$. Then, the optimization vectors of all but one initial solution are imposed componentwise with Gaussian noise $\mathcal{N}(0, \sigma_i)$, with standard deviations that were set to $\sigma_x = \sigma_y = 5\,\mathrm{m}$ for x- and y-coordinates of the control points and to $\sigma_w = 0$ for the control points' weights.

[3] The NURBS approximation runs with the same number of control points n_{CP} as in the other approaches.

Table 3.3: The parameters for all evaluated solvers are listed. The algorithms developed in this work are highlighted in blue. The abbreviations stand for Evolution Strategy (ES), Hybrid Evolution Strategy (H. ES), Non-dominated Sorting Genetic Algorithm 2 (NSGA2), Hybrid Non-dominated Sorting Genetic Algorithm 2 (H. NSGA2), Limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm with Bounds (L-BFGS-B), Dijkstra Algorithm (DA), and Approximated Dijkstra Solution (ADS).

Method	Parameter	Symbol	Value
ES	Initial step size	$\boldsymbol{\sigma}_0$	$[1\,\mathrm{m}\ \ 1\,\mathrm{m}\ \ 0\ \ \ldots]$
	Parent population size	s_{par}	15
	Offspring population size	s_{off}	100
H. ES	Num. of weighted sols.	n_{WS}	5
NSGA2	Crossover crowding deg.	η_{x}	20
	Crossover probability	p_{x}	0.9
	Mutation crowding deg.	η_{mut}	20
	Mutation probability	p_{mut}	1
	Individual mut. prob.	$p_{\mathrm{mut,ind}}$	$1/D$
	Population size	s_{pop}	100
H. NSGA2	Num. of weighted sols.	n_{WS}	5
L-BFGS-B	Num. of weighted sols.	n_{WS}	65
DA			
ADS			

For the initialization of all hybrid solvers, n_{WS} of all initial solutions come from the Dijkstra [42] pre-processing step of the framework. Another initial solution is a straight-line path. To obtain the remaining initial solutions, these $n_{\mathrm{WS}}+1$ solutions are also applied with Gaussian noise, as described before.

The solvers' different performances are compared based on the Multi-objective Optimization (MOO) performance metrics introduced in Section 2.2.1 that are Hypervolume (HV), Generational Distance (GD), and Inverted Generational Distance (IGD), as well as the number of non-dominated solutions n_{NDS}. All four metrics measure the quality of the Pareto set that each solver generates. Since the optimal Pareto set (i.e., Pareto front) is not known, the Pareto set with the highest hypervolume is used as the reference set in the calculation of the two metrics GD, and IGD. The calculated Hypervolume (HV) of a Pareto set obtained by a

solver is normalized to the best and worst hypervolumes of all solvers for a particular scenario, yielding the normalized HV.

The time-sensitive part of the evolutionary and the gradient-based solvers is the objective function evaluation for all generated solutions. Accordingly, for a fair basis of comparison, the number of objective function evaluations is fixed to $n_{\mathrm{FE}} = 18800 \pm 3\%$ for the ES, H. ES, NSGA2, and H. NSGA2 approach. DA and ADS do not use function evaluations during the optimization phase. Therefore, the number of optimizations for DA and ADS is set to $n_{\mathrm{WS}} = 65$ so that their measured calculation time lies within the same range of $t_{\mathrm{C}} = 92\,\mathrm{s} \pm 11\%$ compared to the other solvers.

3.4.2 Results

Observation

The results of the statistical analysis are presented in Table 3.4. It should be noted that the scores generated by DA are highlighted in gray, as its solutions only serve as a reference set. They are optimal in the sense of a discrete path representation in a discrete environment but do not meet the requirement of a smooth continuous path.

Looking at the number of non-dominated solutions n_{NDS} in Table 3.4, it is interesting to see that the DA produces only $n_{\mathrm{NDS}} = 28$ non-dominated solutions from $n_{\mathrm{WS}} = 65$ optimization runs with differently weighted objectives. Furthermore, by the smoothing step in the ADS approach, the hypervolume drops from $m_{\mathrm{NHV}}(\mathrm{DA}) = 98\%$ to $m_{\mathrm{NHV}}(\mathrm{ADS}) = 72\%$, losing another 11 non-dominated solutions. All evolutionary approaches find substantially more non-dominated solutions.

Looking at the other metrics, the hybrid methods H. ES and H. NSGA2 in particular perform well. They achieve normalized hypervolumes, which are significantly[4] better than the normalized hypervolume results generated by the ADS approach in the time. For further analysis, the boxplot for the examined hypervolume metric can be seen in Fig. 3.5. When comparing the H. ES and the H. NSGA2 approaches, the first achieves a median of $M(m_{\mathrm{NHV}}(\mathrm{H.\,ES})) = 88\%$, the latter a median of $M(m_{\mathrm{NHV}}(\mathrm{H.\,NSGA2})) = 82\%$. Both solvers outperform their non-hybrid counterparts and the gradient-based solver by far. Among themselves, by

[4] Applied Mann-Whitney U-Test with $\nu_1 = \nu_2 = 30$, $p < 0.05$ and medians of $M(m_{\mathrm{NHV}}(\mathrm{H.\,ES})) = 88\%$, $M(m_{\mathrm{NHV}}(\mathrm{ADS})) = 75\%$ and $M(m_{\mathrm{NHV}}(\mathrm{H.\,NSGA2})) = 82\%$ yielding $U(m_{\mathrm{NHV}}(\mathrm{H.\,ES}), m_{\mathrm{NHV}}(\mathrm{ADS})) = 732$ and $U(m_{\mathrm{NHV}}(\mathrm{H.\,NSGA2}), m_{\mathrm{NHV}}(\mathrm{ADS})) = 609$.

Table 3.4: The mean results for evaluating 30 scenarios are given. The algorithms developed in this work are highlighted in blue. The results of the Dijkstra Algorithm (DA) highlighted in gray cannot be compared with the others because the Dijkstra algorithm works on a discrete instead of a continuous path representation. Each column's best value is bold. The abbreviations stand for Evolution Strategy (ES), Hybrid Evolution Strategy (H. ES), Non-dominated Sorting Genetic Algorithm 2 (NSGA2), Hybrid Non-dominated Sorting Genetic Algorithm 2 (H. NSGA2), Limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm with Bounds (L-BFGS-B), Approximated Dijkstra Solution (ADS), and Dijkstra Algorithm (DA).

	$\mu(\mathbf{n}_{\mathrm{NDS}})$	$\mu(\mathbf{m}_{\mathrm{NHV}})$	$\mu(\mathbf{m}_{\mathrm{GD}})$	$\mu(\mathbf{m}_{\mathrm{IGD}})$
ES	64	49%	353	461
H. ES	79	**87%**	**191**	**196**
NSGA2	81	3%	489	719
H. NSGA2	**170**	80%	206	245
L-BFGS-B	18	29%	547	606
ADS	17	72%	217	284
DA	28	98%	9	14

applying the Mann-Whitney U-Test[5], it can be found out that the normalized hypervolume distributions differ significantly for both solvers.

Meanwhile, the gradient-based L-BFGS-B solver performs poorly generating only $n_{\mathrm{NDS}} = 18$ non-dominated solutions. At the same time, these are mostly dominated by the solutions of the other solvers, which is reflected in the low normalized hypervolume $m_{\mathrm{NHV}}(\text{L-BFGS-B}) = 29\%$.

Looking at Table 3.4 again and comparing the Approximated Dijkstra Solution (ADS) with the hybrid approaches, the convergence GD improves by 12% for the H. ES and by 5% for the H. NSGA2 approach as well as the IGD measure by 31% and 15%, respectively. This result indicates a better convergence and diversity for the hybrid solvers compared to the ADS approach.

In Fig. A.1, two of the examined 30 scenarios are colored red (left: Scenario A; right: Scenario B). For these two scenarios, the Pareto set plots for all solvers are provided in Fig. 3.6 and Fig. 3.8 respectively. The Pareto set plots visualize and confirm the observations made. The Pareto set of the DA solver used as reference is shown in gray. Next, the solutions of H. ES

[5] $\nu_1 = \nu_2 = 30$, $p < 0.05$, two-tailed, $U(m_{\mathrm{NHV}}(\text{H. ES}), m_{\mathrm{NHV}}(\text{H. NSGA2})) = 609$

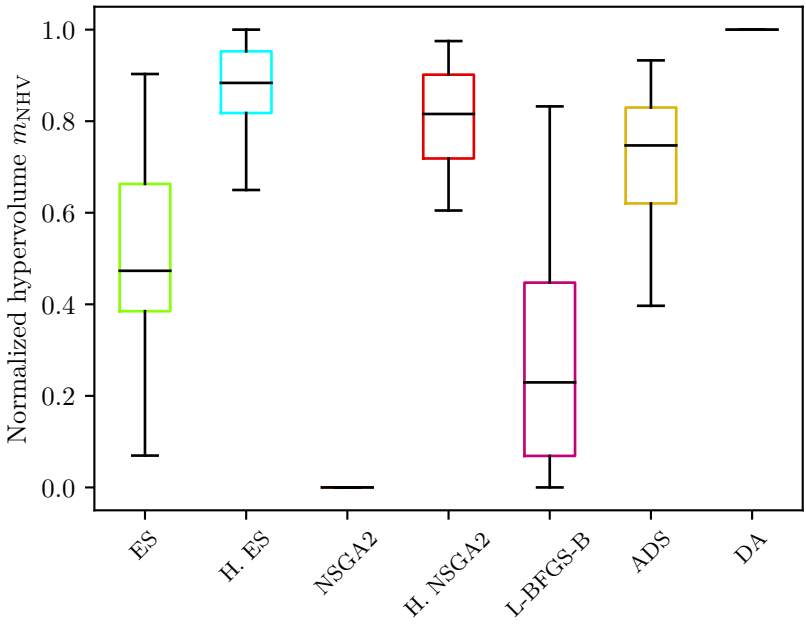


Figure 3.5: The boxplots for the different solvers regarding the normalized hypervolume metric are visualized in different colors. One box spans from the data's first quartile q_1 to the data's third quartile q_3. The black line indicates the median. The whiskers enclose all data points between the lower bound $2.5q_1 - 1.5q_3$ and the upper bound $2.5q_3 - 1.5q_1$. The abbreviations stand for Evolution Strategy (ES), Hybrid Evolution Strategy (H. ES), Non-dominated Sorting Genetic Algorithm 2 (NSGA2), Hybrid Non-dominated Sorting Genetic Algorithm 2 (H. NSGA2), Limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm with Bounds (L-BFGS-B), Approximated Dijkstra Solution (ADS), and Dijkstra Algorithm (DA).

in blue and H. NSGA2 in red come close to it. Then follows the Pareto set of Approximated Dijkstra Solutions (yellow), which is most notable for its low number in non-dominated solutions. The solutions of ES (green), NSGA2 (orange), and L-BFGS-B (violet) are also visually dominated by the solutions of the other solvers in Fig. 3.6.

In Fig. 3.7, the path representations of the Pareto set's risk extreme

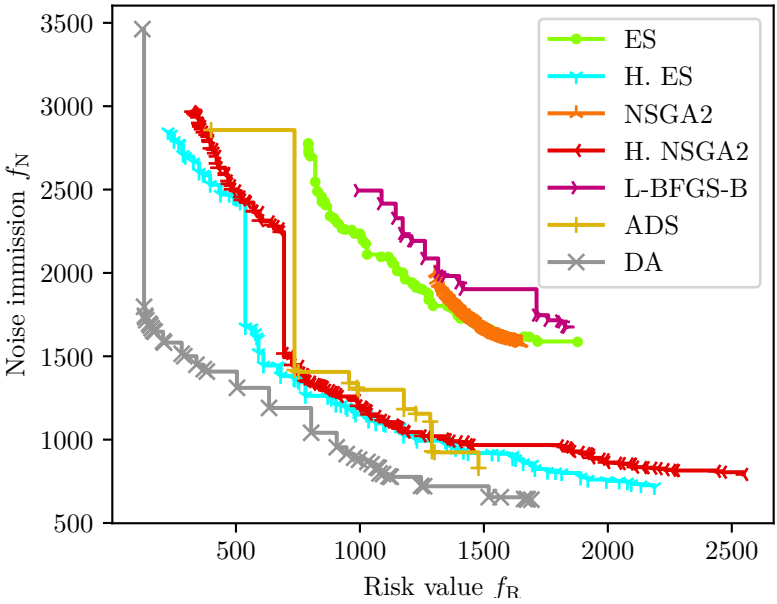


Figure 3.6: Pareto sets obtained by different solvers for one of the test set scenarios (Scenario A). The abbreviations stand for Evolution Strategy (ES), Hybrid Evolution Strategy (H. ES), Non-dominated Sorting Genetic Algorithm 2 (NSGA2), Hybrid Non-dominated Sorting Genetic Algorithm 2 (H. NSGA2), Limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm with Bounds (L-BFGS-B), Approximated Dijkstra Solution (ADS), and Dijkstra Algorithm (DA).

point of Scenario A are visualized for all solvers in the corresponding colors. It should be emphasized that the paths of L-BFGS-B (violet), as well as the non-hybrid solvers ES (green), and NSGA2 (orange) deviate little from the straight path between start and goal point that was used to initialize the search. The Dijkstra Algorithm (DA) finds another path (gray) that has a very small risk value but uses a discrete path representation. When approximating a continuous path in the ADS approach, the good risk value (yellow) deteriorates, which can be seen in Fig. 3.6. The hybrid approaches H. ES and H. NSGA2 succeed in improving the minimum risk value again.

As shown in Fig. 3.7, the corresponding path representations 'follow' the path of the ADS approach for a long time but then take different turns towards the end of the path.

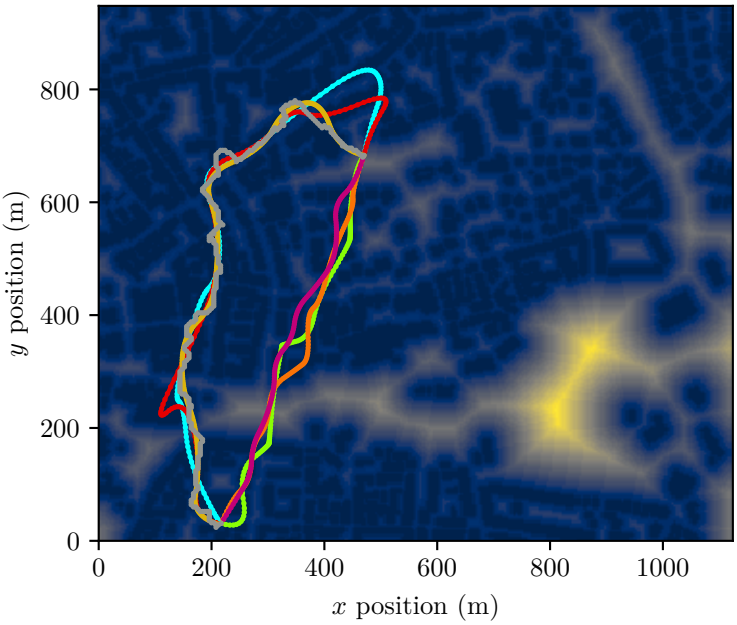


Figure 3.7: The best paths regarding the risk objective function f_R calculated by all optimizers for Scenario A are visualized in the respective same colors as in Fig. 3.6.

Accordingly, the minimum noise paths in the Pareto sets (Fig. 3.8) obtained for Scenario B are visualized in Fig. 3.9. The hybrid solvers find a new path alternative (blue and red) to the Approximated Dijkstra Solution (ADS) path in yellow. A look at the Pareto set plots in Fig. 3.8 reveals that these found alternative paths are considerably less noisy than the paths obtained by ADS and even almost as good as the reference solution of the DA.

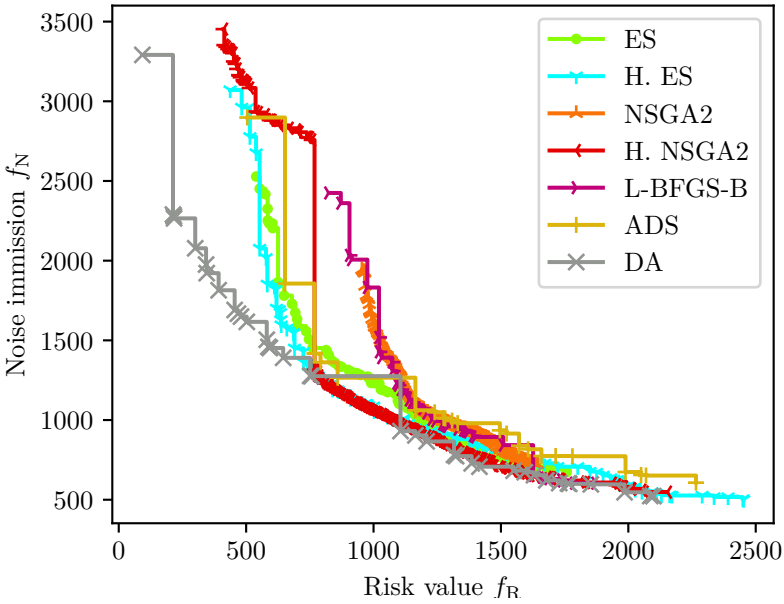


Figure 3.8: Pareto sets obtained by different solvers for one of the test set scenarios (Scenario B). The abbreviations stand for Evolution Strategy (ES), Hybrid Evolution Strategy (H. ES), Non-dominated Sorting Genetic Algorithm 2 (NSGA2), Hybrid Non-dominated Sorting Genetic Algorithm 2 (H. NSGA2), Limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm with Bounds (L-BFGS-B), Approximated Dijkstra Solution (ADS), and Dijkstra Algorithm (DA).

Discussion

The first important finding is that single-objective optimizers are of limited use when solving the problem. Aggregating weights and applying a single-objective solver multiple times generates dominated solutions in more than 50% of all cases. Consequently, the weighted aggregation (3.7) is not sufficient to efficiently sample a diversified Pareto set for the non-convex problem (3.5).

A second important insight is that for the multi-modal problem, both

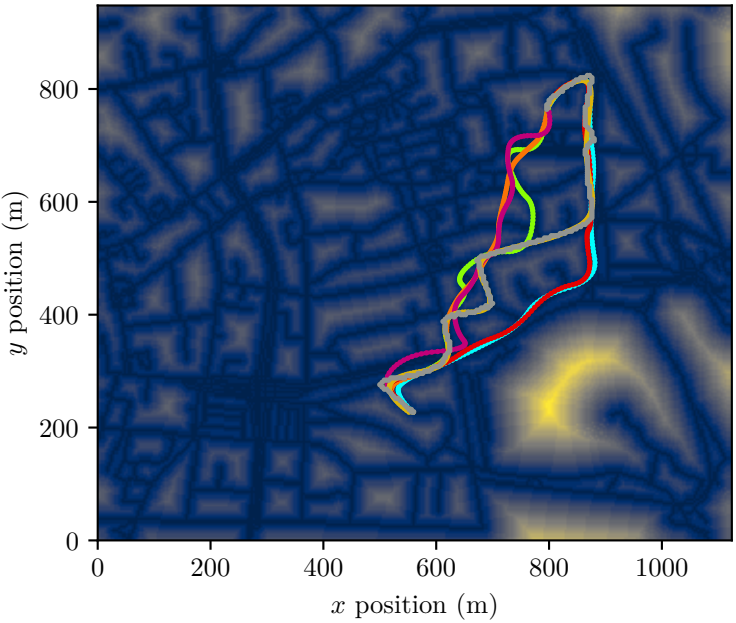


Figure 3.9: The best paths regarding the noise objective function f_N calculated by all optimizers in Scenario B are visualized in the respective same colors as in Fig. 3.8.

classical gradient-based optimizers and standard evolutionary methods reach their limits. As seen in Fig. 3.3, the problem of dimensionality $\dim(\mathbb{V}) = 2$ already comprises lots of local minima. The experiments have shown that the gradient-based solver L-BFGS-B, as well as ES and NSGA2, quickly get stuck in local optima that are not far away from the initial solutions in the search space $\mathbb{V}$.

A third insight is that the subsequent smoothing of an optimized discrete path has non-negligible effects on its quality. Consequently, smoothing as a post-processing step after a discrete path optimization is not desirable when aiming for a Pareto set of smooth continuous paths.

Finally, the proposed hybrid approach combining evolutionary and Dijkstra algorithms seems to be a reasonable answer to the addressed multi-objective and multi-modal problem. Compared to weighted aggregation

approaches, the evolutionary search scheme allows the computation of a diversified Pareto set. At the same time, the Dijkstra optimization stage, unaffected by locality minima, provides good convergence towards better non-dominated solutions.

On the downside, the hybrid approach is tailored to grid-based or at least grid-transferable cost functions. For objectives that do not fulfill this restriction, the Dijkstra stage S1 must be skipped.

3.4.3 Problem in the MOPP Framework

As indicated in Section 3.3.2, the choice of the number of NURBS control points n_{CP} plays a crucial role. It sets the degrees of freedom of the path and, therefore, should not be chosen too small depending on the scenario. In one scenario, a straight path, e.g., across an industrial area, realized by a few control points might be sufficient. In another scenario, a residential area must be crossed by a curvy path, requiring more control points to be realized.

At the same time, n_{CP} should not be chosen too large as it is directly proportional to the complexity of the optimization problem (3.6). Hence, there is a scenario-dependent optimal number of control points. The scenario dependence is problematic as correctly setting n_{CP} requires a priori knowledge of the planning space or a preceding hyperparameter optimization for each new scenario.

So far in this chapter, no thought has been given to the specification of n_{CP}. It was set to $n_{\mathrm{CP}} = 15$ in Table 3.2, which, on average, was a good fit empirically obtained for the scenarios treated. In the following, such pre-processing steps or the assumption of prior domain knowledge are eliminated by determining the number of control points n_{CP} adaptively, i.e., scenario-dependent within the framework. In the further course of the work, this feature is referred to as Adaptive Number of Control Points (ANCP).

3.5 Improvement (ANCP)

3.5.1 Explanation

The Adaptive Number of Control Points (ANCP) feature developed in this work is involved in the *Smoothing & Approximation* step during the

transition from the Dijkstra Algorithm (DA) in step S1 to the Multi-objective Evolutionary Algorithm (MOEA) in step S2 of the framework shown in Fig. 3.4. Here, the path representation changes from a discrete path Π_d to a continuous path Π. The applied NURBS approximation [66] takes the path Π_d and a number of control points n_CP as input.

The optimal number of control points n^*_CP is determined in an iterative Adaptive Number of Control Points (ANCP) process. It starts with the minimum possible value of $\tilde{n}_\mathrm{CP} = 3$. In every iteration, the discrete path Π_d is approximated by a NURBS curve with $\tilde{n}_\mathrm{CP}$ control points. Then, the approximation error ε between the discrete path Π_d and its NURBS approximation Π is determined. As error term ε, an adjusted Chamfer distance metric

$$\varepsilon(\Pi, \Pi_\mathrm{d}) = \max\left(\sum_{\boldsymbol{\pi}\in\Pi} \min_{\boldsymbol{\pi}_\mathrm{d}\in\Pi_\mathrm{d}} |\boldsymbol{\pi} - \boldsymbol{\pi}_\mathrm{d}|^2, \sum_{\boldsymbol{\pi}_\mathrm{d}\in\Pi_\mathrm{d}} \min_{\boldsymbol{\pi}\in\Pi} |\boldsymbol{\pi} - \boldsymbol{\pi}_\mathrm{d}|^2\right)$$

can be used. As long as the error ε exceeds a given threshold $\tau_\mathrm{A} = 7\,\mathrm{m}$, the current number of control points $\tilde{n}_\mathrm{CP}$ is increased by one, and the process is repeated. When the error falls below the threshold, the found continuous path Π approximates Π_d well enough, and we have found a suitable number of control points. Note that this process is carried out for each of the n_WS pre-processed solutions $\Pi_{\mathrm{d},i}$ separately producing potentially different optimal numbers of control points $n^\mathrm{A}_{\mathrm{CP},i}$. We finally obtain the optimal number of control points with $n^*_\mathrm{CP} = \max_{i\in\{1,\ldots,n_\mathrm{WS}\}}(n^\mathrm{A}_{\mathrm{CP},i})$.

3.5.2 Evaluation

Observation

The proposed ANCP feature is evaluated in the following. For 100 scenarios separately, a path optimization runs with an optimal parameter n^*_CP. Then, the optimization runs again for 100 scenarios separately with different numbers of control points reaching from $\tilde{n}_\mathrm{CP} = 5$ to $\tilde{n}_\mathrm{CP} = 30$ control points. This makes 27 optimization runs per scenario. For each scenario, the obtained hypervolumes are normalized to the best and worst. Figure 3.10 shows the mean and the standard deviation (as symmetric errorbar) of the normalized hypervolumes plotted over the difference $\tilde{n}_\mathrm{CP} - n^*_\mathrm{CP}$. If $\tilde{n}_\mathrm{CP}$ lies in the range around the optimal parameter (i.e., $\tilde{n}_\mathrm{CP} - n^*_\mathrm{CP} = 0$ on the x-axis), the mean normalized hypervolume reaches its maximum

value of $\mu(m_{\mathrm{NHV}}) = 0.96$. Apart from this, the mean normalized hypervolume values decrease visibly while their standard deviations increase. If ten more control points are used for the approximation, $\mu(m_{\mathrm{NHV}})$ decreases by 4.2%. If $\tilde{n}_{\mathrm{CP}}$ is reduced by ten control points, $\mu(m_{\mathrm{NHV}})$ even decreases by 10.4%.

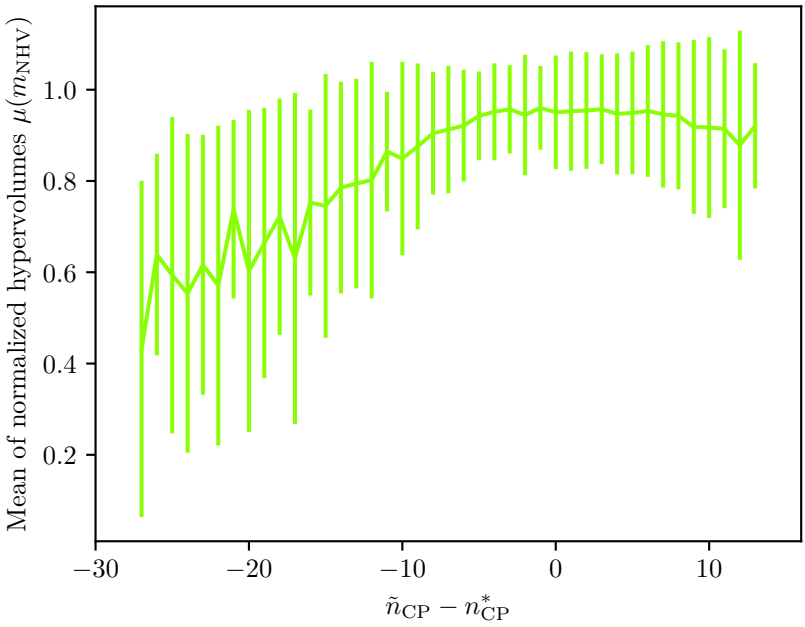


Figure 3.10: Evaluation results for the Adaptive Number of Control Points (ANCP) feature tested on 100 scenarios. The difference between the number of control points evaluated for this test $\tilde{n}_{\mathrm{CP}}$ and the previously calculated optimal number of control points n^*_{CP} is plotted on the x-axis.

Discussion

The test of the ANCP feature results in two insights. First, the number of control points n_{CP} has a strong influence on the hypervolumes and, therefore, the quality of the obtained Pareto sets, as we can deduce from the curve's incline in Fig. 3.10. If n_{CP} is too small, the NURBS curve no

longer approximates the pre-processed Dijkstra path well enough, and the resulting path is very likely to produce worse objective function values, resulting in a smaller hypervolume. Thus, the advantage of pre-processing is lost. On the contrary, if n_{CP} is increased further, the approximation stagnates, i.e., the approximation error no longer decreases. There is no quality improvement for the continuous path anymore. At the same time, the optimization vector becomes unnecessarily large, which increases the complexity of the optimization problem and, thus, the chances of a stagnating optimizer.

Second, using the optimal parameter n_{CP}^{*} consistently leads to high hypervolumes independently of the chosen scenario. The n_{WS} discrete path solutions computed in the pre-processing step S1 can already give a reasonable estimate for the final path's characteristic (i.e., curviness) on the scenario. The curviness of the path determines the number of control points n_{CP} needed to adequately represent it as a NURBS curve. A good NURBS approximation results in better Pareto sets as described in the first insight. This explains the better solution quality achieved with the ANCP feature.

In summary, the advantage of the ANCP feature lies in the applicability of the hybrid path planning framework to unknown scenarios without hyperparameter tuning as the crucial parameter n_{CP} is adaptively determined. However, this generalization is achieved by making the size of the optimization vector D and thus the optimization problem's complexity dependent on the curviness of the path.

3.6 Summary & New Results

This chapter covered the formulation, evaluation, and improvement of a method to solve the Multi-objective Path Planning (MOPP) problem for continuous two-dimensional paths. Grid-based and non-grid-based objectives were introduced as two general classes of objective functions. For both classes, exemplary but realistic objective functions were formulated. They include minimizing the risk and noise for people underneath the planned paths and minimizing the UAV's energy consumption and radio disturbance on its flight.

Including at least two objective functions, the problem is not only multi-objective but was also identified to be multi-modal and thus non-convex.

In the further course of the chapter, a hybrid Multi-objective Path Planning (MOPP) framework was proposed, which was designed to combine the

advantages of 1) an efficient and optimal, but 'discrete-represented' and single-objective shortest path algorithm, with 2) a multi-objective and continuously represented but computation-intensive metaheuristic optimizer.

The proposed framework was benchmarked against state-of-the-art evolutionary, gradient-based, and graph-based (path planning) algorithms on several realistic path planning scenarios using geospatial data from OpenStreetMap (OSM).

Summarizing, the statistical results of the comparison revealed the advantages of the proposed hybrid frameworks. The Dijkstra Algorithm (DA) alone efficiently calculates optimal solutions for single-objective and discrete path problems. However, applying it in a weighted and aggregated formulation to the multi-objective problem is inefficient as it leads to comparably sparse solutions unequally distributed in the objective space. Multi-objective evolutionary algorithms are known to overcome those drawbacks but require more computational resources. Combining both fuses the strengths of exact (i.e., Dijkstra) and metaheuristic algorithms. The hybrid framework is faster and converges better towards a high-quality Pareto set than randomly initialized evolutionary approaches. Compared to the developed approach (previously called ADS) that a posteriori approximates Dijkstra solutions with continuous NURBS curves, the hybrid framework gains efficiency in generating non-dominated solutions and thus diversity.

Last, the number of control points influencing the complexity of the path's shape was identified as a crucial hyperparameter of the framework. This parameter is usually pointless to set without a hyperparameter tuning or domain knowledge gathered from the UAV's environment. However, the use of the Dijkstra step enables efficient hyperparameter tuning. This was exploited in the design of an Adaptive Number of Control Points (ANCP) determination scheme, which was presented, evaluated, and found to work well at the end of this chapter.

The contributions presented in this chapter are

- the idea of combining methods from two different optimization domains that have so far been kept separate in the scientific community,
- the proposal of a new Multi-objective Path Planning (MOPP) framework that is applicable to arbitrary geographic areas and allows the integration of any new objectives, and
- the large-scale statistical analysis of the framework in a cross-domain comparison with existing methods.

4 Multi-objective Three-dimensional Path Planning with Constraints

4.1 Introduction

In the previous chapter, the multi-objective path planning problem for two-dimensional paths was solved using a new hybrid Multi-objective Path Planning (MOPP) framework. However, UAVs move in the three-dimensional space. Also, including a third path dimension in the optimization problem can be helpful, for example, in collision avoidance maneuvers. Moreover, climbing and descending flights consume lots of energy, which is another reason for integrating movement in height into path optimization. In addition, no constraints have yet been considered in the optimization problem.

These two shortcomings will be addressed in the following chapter. For this purpose, two constraint formulations are introduced into the path optimization problem, and the path representation is extended to three dimensions. Approaching this extended problem with the proposed MOPP framework reveals new challenges, e.g., the increase in dimensionality. In order to solve these new problems, the framework requires extensions that are presented and tested in a statistical analysis on an example scenario in New York City.

Parts of this chapter have already been published in:

- [N3] N. Hohmann, M. Bujny, J. Adamy, and M. Olhofer, „Multi-objective 3D path planning for UAVs in large-scale urban scenarios“, in *2022 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, 2022, pp. 1–8. DOI: 10.1109/CEC55065.2022.9870265

- [N2] N. Hohmann, S. Brulin, J. Adamy, and M. Olhofer, „Three-dimensional urban path planning for aerial vehicles regarding many objectives“, *IEEE Open Journal of Intelligent Transportation Systems*, vol. 4, pp. 639–652, 2023. DOI: 10.1109/OJITS.2023.3299496

4.2 Problem Formulation

In the following, the multi-objective path planning problem introduced in Section 3.2 is extended to a three-dimensional path representation. Accordingly, in Section 4.2.1, the changes in the environment and the path representation are shown.

Furthermore, the problem is now extended with constraints, which are formulated in Section 4.2.2. Then, the multi-objective three-dimensional path planning formulation is presented in Section 4.2.3.

4.2.1 Representations

Environment

We extend the operation space to three dimensions

$$\mathbb{D}_{\mathrm{c}}^{3\mathrm{D}} = \begin{bmatrix} x_{\min} & x_{\max} \end{bmatrix} \times \begin{bmatrix} y_{\min} & y_{\max} \end{bmatrix} \times \begin{bmatrix} z_{\min} & z_{\max} \end{bmatrix} \subset \mathbb{R}^3,$$

with the bounding coordinates $x_{\min} < x_{\max}$, $y_{\min} < y_{\max}$, and $z_{\min} < z_{\max}$. Discretizing the continuous operation space with resolution $\begin{bmatrix} \delta_x & \delta_y & \delta_z \end{bmatrix}$, yields the three-dimensional discrete operation space $\mathbb{D}_{\mathrm{d}}^{3\mathrm{D}}$.

Path

For the path representation, the same Non-uniform Rational B-Spline (NURBS) curve formulation $\mathbf{C}(u)$ like in the previous chapter can be used, with all curve's control points $\mathbf{P}_i = \begin{bmatrix} x_i & y_i & z_i \end{bmatrix}$ now having dimension three.

Changing the position of a control point and changing the weight of a control point have a similar effect on the shape of the curve and, thus, the quality of the path represented. In order to not increase the complexity of the optimization problem unnecessarily, the weights w_i of the control points are removed from the optimization vector and set to fixed values $w_i = 1$ with an exception described together with the constraint formulation in the following Section 4.2.2.

The path's start point $\boldsymbol{\pi}_{\mathrm{s}} \in \mathbb{D}_{\mathrm{c}}^{3\mathrm{D}}$ being identical to the first control point $\mathbf{P}_0$, and the path's goal point $\boldsymbol{\pi}_{\mathrm{g}} \in \mathbb{D}_{\mathrm{c}}^{3\mathrm{D}}$ being identical to the last control point $\mathbf{P}_{n_{\mathrm{CP}}-1}$ are input into the path planning problem. The remaining $n_{\mathrm{CP}} - 2$ non-fixed control points $\mathbf{P}_i$ with $i \in \{1, \ldots, n_{\mathrm{CP}} - 2\}$ are the variables

$$\mathbf{v} = \begin{bmatrix} x_1 & y_1 & z_1 & \ldots & x_{n_{\mathrm{CP}}-2} & y_{n_{\mathrm{CP}}-2} & z_{n_{\mathrm{CP}}-2} \end{bmatrix} \tag{4.1}$$

of the optimization problem that is stated later in Section 4.2.3.

Evaluating the NURBS curve for respective parameters $\mathbf{u}_\mathrm{U}$ like described in Section 2.1.2 leads to the path

$$\Pi(\mathbf{v}) = \mathbf{C}(u)|_\mathbf{v} = [\boldsymbol{\pi}_0 = \boldsymbol{\pi}_\mathrm{s}, \boldsymbol{\pi}_1, \ldots, \boldsymbol{\pi}_{|\Pi|-1} = \boldsymbol{\pi}_\mathrm{g}]$$

with waypoints $\boldsymbol{\pi}_i = \begin{bmatrix}\pi_{i,\mathrm{x}} & \pi_{i,\mathrm{y}} & \pi_{i,\mathrm{z}}\end{bmatrix} \subset \mathbb{D}_\mathrm{c}^{3\mathrm{D}}$.

4.2.2 Constraints

Inequality constraints $g_j(\mathbf{v})$ and equality constraints $h_k(\mathbf{v})$, which were generally introduced in Section 2.2.1, can be included as so-called soft constraints into a multi-objective optimization problem. To that, they are added as penalty terms to every objective function $f_i(\mathbf{v})$ as similarly done by Coello [69] and Hoffmeister *et al.* [70]. This results in the constrained objective functions

$$f_{i,\mathrm{C}}(\mathbf{v}) = f_i(\mathbf{v}) + \sum_{j=1}^{F_\mathrm{ineq}} \xi_j \sqrt{\mathcal{H}(-g_j(\mathbf{v}))g_j(\mathbf{v})^2} + \sum_{k=1}^{F_\mathrm{eq}} \xi_k |h_k(\mathbf{v})|, \quad (4.2)$$

with the Heaviside function $\mathcal{H}$ and the penalty weights ξ that are chosen arbitrarily large to ensure that $f_{i,\mathrm{C}}(\mathbf{v}) \gg f_i(\mathbf{v})$ if the constraints are violated.

In the following, two inequality constraints are introduced. They are designed to be realistic for UAV applications but principally are exemplary to evaluate the path planning framework's performance concerning constraint handling.

Minimum Flight Height

Except for the take-off and landing phase, the UAV's path is supposed to lie above a minimum flight height $z_\mathrm{F,min}$. Introducing the height inequality constraint

$$g_{\mathrm{H},i}(\mathbf{v}) = z_i - z_\mathrm{F,min} \geq 0, \quad \forall i \in \{1, \ldots, n_\mathrm{CP} - 2\} \quad (4.3)$$

we make sure that the z-components z_i of the curve's control points will not decrease below $z_\mathrm{F,min}$. Note that the convex hull property of NURBS curves, which was described in 2.1.2, guarantees that no path point $\boldsymbol{\pi}$ will fall below $z_\mathrm{F,min}$ if there is no control point $\mathbf{P}_i = \begin{bmatrix}x_i & y_i & z_i\end{bmatrix}$ below $z_\mathrm{F,min}$.

If the path's fixed start control point $\mathbf{P}_0 = \begin{bmatrix} x_0 & y_0 & z_0 \end{bmatrix}$ or endpoint $\mathbf{P}_{n_{\mathrm{CP}}-1} = \begin{bmatrix} x_{n_{\mathrm{CP}}-1} & y_{n_{\mathrm{CP}}-1} & z_{n_{\mathrm{CP}}-1} \end{bmatrix}$ lie below $z_{\mathrm{F,min}}$, the second and second last control points are additionally locked in position. Their components are set to $\mathbf{P}_1 = \begin{bmatrix} x_0 & y_0 & z_{\mathrm{F,min}} \end{bmatrix}$ with weight $w_1 = 100$, as well as $\mathbf{P}_{n_{\mathrm{CP}}-2} = \begin{bmatrix} x_{n_{\mathrm{CP}}-1} & y_{n_{\mathrm{CP}}-1} & z_{\mathrm{F,min}} \end{bmatrix}$ with weight $w_{n_{\mathrm{CP}}-2} = 100$. During the take-off and landing phase of the aerial vehicle, this guarantees a completely vertical flight movement as long as the aircraft is below the minimum flight altitude.

Obstacle Collision Avoidance

The UAV's path must not go through static obstacles like buildings. A twofold approach enables a three-dimensional path to be pushed out of static obstacles by using two two-dimensional grid maps instead of a three-dimensional grid map.

A first inequality constraint formulation is designed to influence the z-component of waypoints within an obstacle. Therefore, a two-dimensional height grid map $\mathcal{G}_{\mathrm{H}}^{2\mathrm{D}}$ is introduced. Each cell of the height grid map contains the height value of the tallest building in the respective cell. An exemplary height grid map is visualized in Fig. A.15 in the Appendix A.2. Suppose a waypoint $\boldsymbol{\pi}_i = \begin{bmatrix} \pi_{i,\mathrm{x}} & \pi_{i,\mathrm{y}} & \pi_{i,\mathrm{z}} \end{bmatrix}$ of the path Π lies below the height of the respective height map value. In that case, it is stored in a set of unsafe waypoints $\Pi^{\times}$. All objective functions are subject to a penalty term according to (4.2) and the following zenith collision inequality constraint for all $i \in \{0, \ldots, |\Pi| - 1\}$

$$g_{\mathrm{Z},i}(\mathbf{v}) = \pi_{i,\mathrm{z}} - \mathcal{I}(\mathcal{G}_{\mathrm{H}}^{2\mathrm{D}}, (\pi_{i,\mathrm{x}}, \pi_{i,\mathrm{y}})) \geq 0, \tag{4.4}$$

with $\mathcal{I}(\mathcal{G}_{\mathrm{H}}^{2\mathrm{D}}, (\pi_{i,\mathrm{x}}, \pi_{i,\mathrm{y}}))$ being the bi-linear regular grid interpolation of the height grid map at point $(\pi_{i,\mathrm{x}}, \pi_{i,\mathrm{y}})$. Consequently, the unsafe waypoints are pushed out of static obstacles in the direction of the z-axis.

A second inequality constraint formulation is designed to create a drift that moves unsafe waypoints out of static obstacles in the xy-plane. Therefore, the building grid map $\mathcal{B}_{\mathrm{B}}^{2\mathrm{D}}$, already introduced in Section 3.2.2, is used. By inverting the building grid map, and applying the Euclidean Distance Transform (EDT) (3.3), an obstacle grid map is obtained

$$\mathcal{G}_{\mathrm{O}}^{2\mathrm{D}} = \mathcal{T}(\bar{\mathcal{B}}_{\mathrm{B}}^{2\mathrm{D}})$$

with gradients that point away from the centers of buildings and toward the nearest free-space cells. This obstacle grid map is used to calculate a

plain collision inequality constraint for all $i \in \{0, \ldots, |\Pi^{\times}| - 1\}$

$$g_{\mathrm{P},i}(\mathbf{v}) = -\mathcal{I}(\mathcal{G}_{\mathrm{O}}^{\mathrm{2D}}, (\pi_{i,\mathrm{x}}^{\times}, \pi_{i,\mathrm{y}}^{\times})) > 0, \tag{4.5}$$

with $\mathcal{I}$ again being the grid interpolation function. This constraint function ensures that during optimization unsafe waypoints $\boldsymbol{\pi}_i^{\times} = \begin{bmatrix} \pi_{i,\mathrm{x}}^{\times} & \pi_{i,\mathrm{y}}^{\times} & \pi_{i,\mathrm{z}}^{\times} \end{bmatrix} \in \Pi^{\times}$ are pushed in the direction of cells in the obstacle grid map that equal zero, i.e., are no obstacles.

4.2.3 Optimization Problem

As it has already been described in Section 3.2.3, we again want to satisfy the requirement of non-correlated objective functions for the design of the multi-objective three-dimensional path planning problem. When planning paths in the third dimension, especially the *Noise Immission* objective function $f_{\mathrm{N}}(\mathbf{v})$ presented in Section 3.2.2, which prefers high-altitude flight paths, as well as the *Energy Consumption* objective function $f_{\mathrm{E}}(\mathbf{v})$ presented in Section 3.2.2, which favors low-altitude flight paths, take opposite optimization goal roles and thus appear suitable.

The vector $\mathbf{v}$ of optimization variables (4.1) has already been given in Section 4.2.1. Eventually, the optimization problem with constraints considered in this chapter can be described as

$$\min_{\mathbf{v} \in \mathbb{V}} \begin{cases} f_{\mathrm{N}}(\mathbf{v}), \\ f_{\mathrm{E}}(\mathbf{v}), \end{cases}$$

$$\text{s.t.} \begin{cases} g_{\mathrm{H},i}(\mathbf{v}) \geq 0, & \forall i \in \{1, \ldots, n_{\mathrm{CP}} - 2\} \\ g_{\mathrm{Z},i}(\mathbf{v}) \geq 0, & \forall i \in \{0, \ldots, |\Pi| - 1\} \\ g_{\mathrm{P},i}(\mathbf{v}) > 0, & \forall i \in \{0, \ldots, |\Pi^{\times}| - 1\} \end{cases}$$

where n_{CP} is the number of control points in the NURBS path representation, $|\Pi|$ is the number of waypoints of the path Π, and $|\Pi^{\times}|$ is the number of waypoints of the path Π that lie within a static obstacle. Here, the search space

$$\mathbb{V} = \underbrace{[\mathbb{D}_{\mathrm{c}}^{\mathrm{3D}}] \times \ldots \times [\mathbb{D}_{\mathrm{c}}^{\mathrm{3D}}]}_{n_{\mathrm{CP}}-2}$$

has $\dim(\mathbb{V}) = 3(n_{\mathrm{CP}} - 2)$ dimensions.

4.3 Solution Approach

4.3.1 Problem Analysis

The runtime of the Dijkstra algorithm depends on the number of edges $|\mathring{E}|$ and nodes $|\mathring{N}|$ in the underlying graph G. More precisely, the time complexity is bounded from above by $\mathcal{O}(|\mathring{N}|log(|\mathring{N}|) + |\mathring{E}|)$ [71]. As it can be seen from Fig. 4.1, the transition from a two-dimensional to a three-dimensional environment representation and thus the change from 2D to 3D graphs comes at the cost of a massively increasing number of edges and nodes in the used objective functions' grid graph representations G^{3D}. Consequently, the runtime of the Dijkstra algorithm in step S1 of the presented Multi-objective Path Planning (MOPP) framework quickly becomes computationally infeasible for growing operation spaces $\mathbb{D}_{\mathrm{c}}^{\mathrm{3D}}$.

4.3.2 Solution

To face the presented curse of dimensionality, it makes sense to *not* use the same resolution $\begin{bmatrix}\delta_x & \delta_y & \delta_z\end{bmatrix}$ for the graphs in the Dijkstra step S1 as for the grid maps in the grid-based objective function evaluations (3.2) in the evolutionary step S2. Instead, the original graphs are resized according to a new resolution $\begin{bmatrix}\tilde{\delta}_x & \tilde{\delta}_y & \tilde{\delta}_z\end{bmatrix}$. Details about the extensions of the pipeline are described in the following section.

Pipeline

The MOPP framework's extensions are visualized blue in Fig. 4.2. Next to the path's start and end point and the E objectives, now $F = F_{\mathrm{eq}} + F_{\mathrm{ineq}}$ constraints are also input into the framework. Moreover, the framework has been extended by the following steps:

1. In the *Grid Resizing* step, the grids $\mathcal{G}^{\mathrm{3D}}$ of all grid-based objectives are scaled from their original resolution $\begin{bmatrix}\delta_x & \delta_y & \delta_z\end{bmatrix}$ to a coarser resolution $\begin{bmatrix}\tilde{\delta}_x & \tilde{\delta}_y & \tilde{\delta}_z\end{bmatrix}$ for the Dijkstra algorithm. These scaled grids are denoted

$$\tilde{\mathcal{G}}^{\mathrm{3D}} : \underbrace{\{1,\dots,\tilde{X}\} \times \{1,\dots,\tilde{Y}\} \times \{1,\dots,\tilde{Z}\}}_{\tilde{\mathbb{D}}_{\mathrm{d}}^{\mathrm{3D}}} \to \mathbb{R}^+, \quad (i,j,k) \mapsto g_{ijk},$$

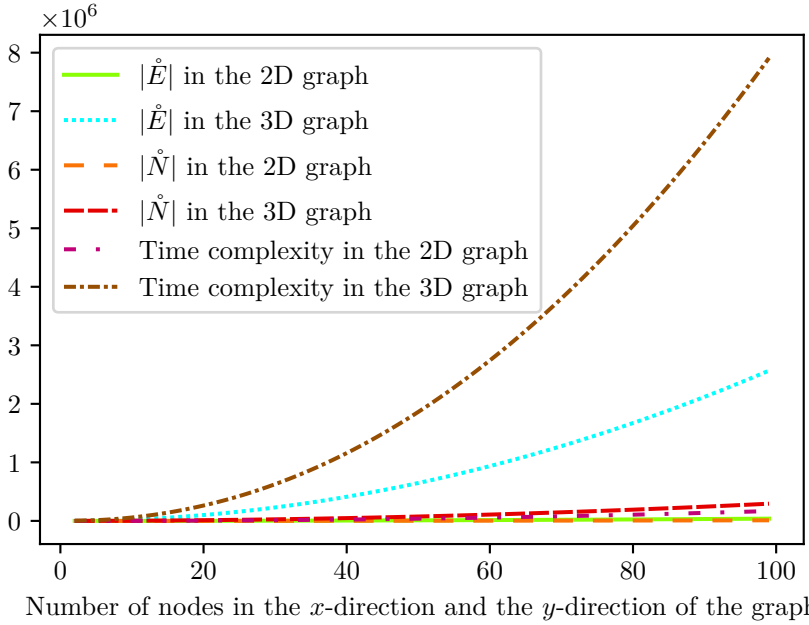


Figure 4.1: Comparison between a 2D (8-neighborhood) and a 3D (18-neighborhood) square grid graph with 30 z-layers regarding the number of edges $|\mathring{E}|$, the number of nodes $|\mathring{N}|$, and the upper time complexity bound $\mathcal{O}(|\mathring{N}|log(|\mathring{N}|) + |\mathring{E}|)$ of the Dijkstra algorithm finding a shortest path in the respective graphs.

with

$$\tilde{X} = \left\lfloor \frac{|x_{\max} - x_{\min}|}{\tilde{\delta}_x} \right\rfloor$$

and accordingly for the other dimensions. The generalized graph representations are designated by $\tilde{G}^{3D}$. The choice of a suitable resolution depends on the scenario size and is a trade-off. A less coarse resolution would increase the solution quality of the following Dijkstra algorithm but could be too slow or consume too much computational resources. A coarser resolution would make the Dijkstra algorithm need less computational time but could lead to poor

solutions, which could not serve as adequate initial solutions in step S2.

2. Domain knowledge from the constraints formulations can be used in the *Grid Resizing* and *Grid Transfer* steps to reduce the size of the resized graphs $\tilde{G}^{3D}$ further. More details on that follow in the next section.

3. Including constraints as soft constraints (4.2) into the framework opens a new problem, which requires an adjustment in the multi-objective optimization step S2. This is described at the end of the chapter in Section 4.5.

Integration of Constraints

Minimum Flight Height The minimum flight height constraint can be perfectly incorporated into the Dijkstra stage S1 of the solution framework. Waypoints $\boldsymbol{\pi}_i$ should generally be above the minimum flight height $\pi_{i,\mathrm{z}} > z_{\mathrm{F,min}}$, except for takeoff and landing phases where a vertical flight occurs. Thus, before the DA in step S1 repeatedly solves the weighted, aggregated, and transformed problem (3.8), the start and endpoints for the Dijkstra searches can be projected to the minimum flight heights $\boldsymbol{\pi}_{\mathrm{s}} = \begin{bmatrix} x_0 & y_0 & z_{\mathrm{F,min}} \end{bmatrix}$ and $\boldsymbol{\pi}_{\mathrm{g}} = \begin{bmatrix} x_{n_{\mathrm{CP}}-1} & y_{n_{\mathrm{CP}}-1} & z_{\mathrm{F,min}} \end{bmatrix}$. Eventually, in the aggregated grid map $\tilde{\mathcal{G}}_{\mathrm{A}}^{3D}$, where the Dijkstra algorithm looks for solutions, all cells $(i,j,k) \in \tilde{\mathbb{D}}_{\mathrm{d}}^{3D}$ with $k\tilde{\delta}_z < z_{\mathrm{F,min}}$ can be marked as occupied so that the DA does not look for solutions here at all. The search space of the Dijkstra algorithm is thus reduced. Further advantages will be discussed in the next section.

Obstacle Collision Avoidance Also, the two described obstacle collision avoidance constraints can already be considered in the Dijkstra algorithm stage S1. Waypoints are supposed to lie outside of static obstacles. In the aggregated grid map $\tilde{\mathcal{G}}_{\mathrm{A}}^{3D}$, all cells $(i,j,k) \in \tilde{\mathbb{D}}_{\mathrm{d}}^{3D}$ with $k\tilde{\delta}_z < \tilde{\mathcal{G}}_{\mathrm{H}}^{2D}(i,j)$ that are below the height value of the corresponding cell in the scaled height grid map $\tilde{\mathcal{G}}_{\mathrm{H}}^{2D}$ can be set to an occupied status, being ignored by the Dijkstra Algorithm (DA). Consequently, the search space for the Dijkstra algorithm becomes smaller again.

However, this integration of constraints into the step S1 is only an approximation of the real constraint formulations (4.3), (4.4), and (4.5). Due

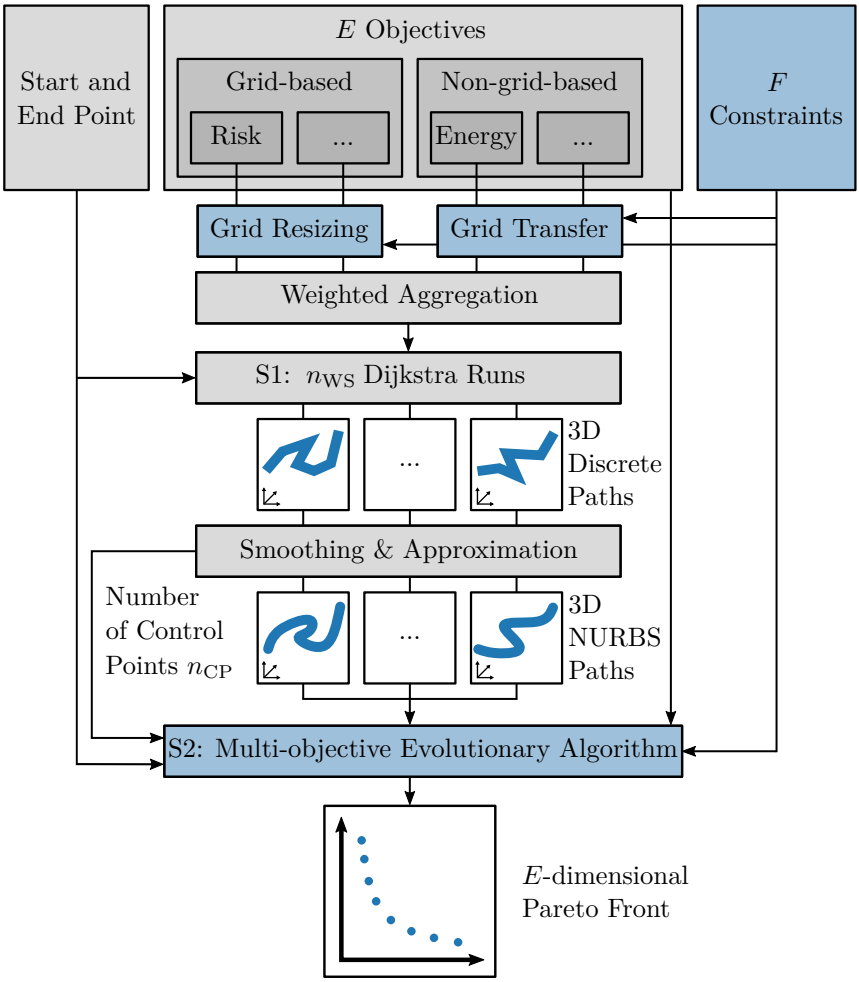


Figure 4.2: Diagram showing the changes in the hybrid optimization framework that allow for the incorporation of constraints and a three-dimensional path representation. The grids of grid-based objectives are resized, obtaining a coarser resolution suitable for the Dijkstra algorithm. Moreover, if possible, the constraint formulations can also be considered in the grids that are input into the weighted aggregation. Also, the multi-objective evolutionary algorithm must be adapted due to constraints in the hybrid framework. This is explained at the end of this chapter.

to the change to a continuous path representation in the *Smoothing & Approximation* step of the framework and the usage of the original grid graphs $\mathcal{G}^{3D}$, the derived initial solutions in step S2 do not necessarily satisfy the constraints anymore. Nevertheless, the described approximation errors are expected to be acceptably small. Hence, the solutions derived after S1 should already be close to the feasible region of the original optimization problem in step S2.

4.4 Evaluation

The introduced MOPP framework extension for solving the multi-objective three-dimensional path planning problem with constraints is evaluated in the following. Therefore, the following Section 4.4.1 presents the test scenario, comparative algorithms, and their parametrization used for the evaluation. The evaluation's results and discussion follow in Section 4.4.2. Lastly, a new problem that emerged by the framework extension is discussed in Section 4.4.3.

4.4.1 Setup

Scenario

The evaluation is based on OpenStreetMap (OSM) [67] data of a map section from New York City that is visualized in Fig. A.7. The corresponding semantic data is visualized in Fig. A.8 in the Appendix A.2. Visualizations of the derived height grid map $\mathcal{G}_{\mathrm{H}}^{2D}$, and a cross-section through the noise grid map $\mathcal{G}_{\mathrm{N}}^{3D}$ at height $z = 0\,\mathrm{m}$ can be seen in Fig. A.9, and Fig. A.11, respectively. These grid maps are used to calculate the noise objective function $f_{\mathrm{N}}(\mathbf{v})$, the height constraint $g_{\mathrm{H},i}$, the zenith collision constraint $g_{\mathrm{Z},i}$, and the plain collision constraint $g_{\mathrm{P},i}$ introduced in Section 4.2.2. A summary of all parameters concerning the scenario setup and the representation of the environment and paths can be found in Table 4.1.

The tests were conducted on 30 different start $\boldsymbol{\pi}_{\mathrm{s}}$ and goal $\boldsymbol{\pi}_{\mathrm{g}}$ pairs that are uniformly distributed over the operation space $\mathbb{D}_{\mathrm{c}}^{3D}$ and the distance space. In order to be more realistic, the z-components of all start and endpoints were projected onto the height grid map $\mathcal{G}_{\mathrm{H}}^{2D}$. The x-y-positions of all start and goal point pairs are visualized in Fig. A.7.

Table 4.1: Scenario Parameters

	Parameter	Symbol	Value
Scenario	Origin latitude	Lat	40.7417°N
	Origin longitude	Lon	-73.9953°E
	Scenario dim.	$x_{\min}$, $x_{\max}$	0 m, 2280 m
		$y_{\min}$, $y_{\max}$	0 m, 1500 m
		$z_{\min}$, $z_{\max}$	0 m, 300 m
	Min. flight height	$z_{\mathrm{F,min}}$	50 m
Environment	Discr. resolution	δ_x, δ_y, δ_z	4 m, 4 m, 10 m
	Resizing resolution	$\tilde{\delta}_x$, $\tilde{\delta}_y$, $\tilde{\delta}_z$	15 m, 15 m, 10 m
Path	Num. control pts.	n_{CP}	20[1]
	Basis func. degree	d	2
	Parametrization		uniform
	Num. of path pts.	$\lvert\Pi\rvert$	150

[1] Although the ANCP feature has already been introduced, it was not yet implemented at the time of this experiment setup, which is why the number of control points is still static here.

Solvers

In order to evaluate the extended Multi-objective Path Planning (MOPP) framework for the multi-objective three-dimensional path planning problem with constraints, different Multi-objective Evolutionary Algorithms (MOEAs) are chosen as optimizers in step S2 of the hybrid framework proposed in this thesis. Like in the previous chapter, the resulting approaches are again denoted with the H. prefix. The results are compared to the solvers' performances outside the developed hybrid framework. The Evolution Strategy (ES) and the Non-dominated Sorting Genetic Algorithm 2 (NSGA2), which were already used in Chapter 3, are again employed as solution methods. In addition, another standard solver for multi-objective optimization problems, Multi-objective Covariance Matrix Adaptation Evolution Strategy (MO-CMA-ES) [72], is evaluated.

Furthermore, beyond these metaheuristic approaches, the single-objective gradient-based optimizer L-BFGS-B is now also tested as a solver in step S2 within the framework (H. L-BFGS-B). Since this algorithm is not capable of handling more than one objective at once, the Pareto set generated by the L-BFGS-B approach is derived by a weighted aggregation of the objectives (3.7). This time, the necessary weights λ_i are not chosen

uniformly distributed, like described in Section 3.4.1, but are derived by an Adaptive Weight Determination Scheme (AWDS) [73].

For all hybrid solution methods, the number of weighted solutions generated in step S1 of the framework was set to the number of objectives[1] $n_{\mathrm{WS}} = E = 2$ with the weights being

$$\begin{bmatrix}\lambda_1 & \lambda_2\end{bmatrix} \in \left\{ \begin{bmatrix}0 & 1\end{bmatrix}, \begin{bmatrix}1 & 0\end{bmatrix} \right\}.$$

In Table 4.2, an overview of the hyperparameters for each approach is given, with all approaches developed in this work marked in blue. Parameters not introduced there are set to standard values from literature [39], [72]. Similar to the method in Chapter 3, all non-hybrid approaches are initialized with straight line paths, whereby the control points are applied with Gaussian noise $\mathcal{N}(0, \sigma_i)$, with standard deviations that were set to $\sigma_x = \sigma_y = 3\,\mathrm{m}$ for x- and y-coordinates and $\sigma_z = 5\,\mathrm{m}$ for the z-component of the control points. Additionally to the n_{WS} pre-processed solutions, all hybrid approaches are initialized with a straight line path. If more than $n_{\mathrm{WS}} + 1$ initial solutions are needed, these are applied with Gaussian noise.

The metrics that are used to evaluate the different approaches are the same as those used in Chapter 3.

During the optimization, using evolutionary and gradient-based solvers, the cost function evaluations need the most computational resources in each iteration. Accordingly, for a fair basis of comparison, the number of objective function evaluations is limited to $n_{\mathrm{FE}} = 40000$ for all optimizers, corresponding to an evolution of $a = 400$ generations for the evolutionary approaches.

4.4.2 Results

Observation

The results obtained for evaluating 30 scenarios can be seen in Table 4.3. The mean values for the optimizers' normalized HV, GD, and IGD metrics are given after $n_{\mathrm{FE}} = 1000$ and $n_{\mathrm{FE}} = 40000$ function evaluations, respectively. Additionally, Fig. 4.3 gives a more detailed look at the development of the normalized HV, which is averaged over all runs and plotted over the number of function evaluations.

[1] The benefit of a larger number of weighted solutions n_{WS} in step S1 is discussed under the name Multiple Weighted Start Points (MWSP) in the following Chapter 5.4 of the thesis.

Table 4.2: The parameters for all evaluated solvers are listed. The algorithms developed in this work are highlighted in blue. The abbreviations stand for Evolution Strategy (ES), Hybrid Evolution Strategy (H. ES), Non-dominated Sorting Genetic Algorithm 2 (NSGA2), Hybrid Non-dominated Sorting Genetic Algorithm 2 (H. NSGA2), Multi-objective Covariance Matrix Adaptation Evolution Strategy (MO-CMA-ES), Hybrid Multi-objective Covariance Matrix Adaptation Evolution Strategy (H. MO-CMA-ES), and Hybrid Limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm with Bounds (H. L-BFGS-B).

Method	Parameter	Symbol	Value
ES	Initial step size	$\boldsymbol{\sigma}_0$	$[3\,\mathrm{m}\ 3\,\mathrm{m}\ 5\,\mathrm{m}\ \ldots]$
	Parent pop. size	s_{par}	10
	Offspring pop. size	s_{off}	100
H. ES	Num. weighted sols.	n_{WS}	2
NSGA2	Crsvr. crowding deg.	η_{x}	20
	Crossover probability	p_{x}	0.9
	Mut. crowding deg.	η_{mut}	20
	Mutation probability	p_{mut}	1
	Individual mut. prob.	$p_{\mathrm{mut,ind}}$	$1/D$
	Population size	s_{pop}	100
H. NSGA2	Num. weighted sols.	n_{WS}	2
MO-CMA-ES	Initial step size	σ	1
	Parent pop. size	s_{par}	100
	Offspring pop. size	s_{off}	100
H. MO-CMA-ES	Num. weighted sols.	n_{WS}	2
H. L-BFGS-B	Num. weighted sols.	n_{WS}	2

The low final normalized HV $\mu(m_{\mathrm{NHV}}) = 31\%$ of the L-BFGS-B solver is particularly noticeable. By applying the Mann-Whitney U-Test[2], it can be shown that by embedding the gradient-based solver into the hybrid framework (i.e., H. L-BFGS-B), the performance of the L-BFGS-B algorithm can be increased significantly by 129%.

The same effect can be observed for the other optimizers: by utiliz-

[2]The parameters of the test are $\nu_1 = \nu_2 = 30$, $p < 0.05$, two-tailed, medians $M(m_{\mathrm{NHV}}(\text{L-BFGS-B})) = 0\%$, $M((m_{\mathrm{NHV}}(\text{H. L-BFGS-B}))) = 75\%$ yielding $U(m_{\mathrm{NHV}}(\text{L-BFGS-B}), m_{\mathrm{NHV}}(\text{H. L-BFGS-B})) = 822$.

ing the hybrid approach, the means of the solvers' achieved final normalized HVs can be increased by 41% (MO-CMA-ES), 26% (NSGA2), and 17% (ES), respectively. The overall best results can be achieved by the H. NSGA2 algorithm. Looking at the development of its mean normalized HV in Fig. 4.3 (orange curve) in comparison to NSGA2 without hybrid framework (red curve), it can be observed that the hybridization has the biggest impact in the beginning of the optimization, leading to a huge difference in the initial means of normalized HV of both solvers. In order to further examine characteristics of the solvers after only a few function evaluations, the boxplot for the normalized HV after $n_{\mathrm{FE}} = 1000$ function evaluations is shown in Fig. 4.4. Regarding the normalized HV, the NSGA2 and H. NSGA2 algorithms differ significantly[3].

All hybrid evolutionary approaches show the same beneficial characteristic regarding the GD metric. Looking in Table 4.3, we observe that after $n_{\mathrm{FE}} = 1000$ evaluations, the hybrid evolutionary optimizers achieve a smaller and, therefore, better generational distance measure than their standard counterparts. Again, the best performing algorithm is H. NSGA2 that performs significantly[4] better than the standard NSGA2 algorithm. Finally, a look at the performances regarding the IGD metric will be taken. From Table 4.3, it can be observed that after $n_{\mathrm{FE}} = 1000$ evaluations the H. ES approach achieves the best mean IGD value, differing significantly[5] from the standard ES.

Next, we look at paths that were calculated after $n_{\mathrm{FE}} = 1000$ function evaluations by each optimizer for the scenario that is depicted by the red line in Fig. A.7 in the Appendix A.2. In Fig. 4.5, the paths with the minimal noise values determined by each solver are visualized. Whereas the non-hybrid optimizers (blue, red, brown and gray paths) only find solutions at lower altitude, the hybrid optimizers found paths (green, orange, violet and gold) with lower noise values at higher flight altitudes. Similarly, the same approaches benefit from the hybrid framework regarding the calculated paths that achieved the lowest energy consumption values. They are visualized in Fig. 4.6. It can be seen that the paths that were

[3] Applied Mann-Whitney U-Test with $\nu_1 = \nu_2 = 30$, $p < 0.05$, and medians $M(m_{\mathrm{NHV}}(\mathrm{NSGA2})) = 37\%$, $M(m_{\mathrm{NHV}}(\mathrm{H.\,NSGA2})) = 95\%$ yielding $U(m_{\mathrm{NHV}}(\mathrm{NSGA2}), m_{\mathrm{NHV}}(\mathrm{H.\,NSGA2})) = 843$

[4] Applied Mann-Whitney U-Test with $\nu_1 = \nu_2 = 30$, $p < 0.05$, $M(m_{\mathrm{GD}}(\mathrm{NSGA2})) = 0.009$, $M(m_{\mathrm{GD}}(\mathrm{H.\,NSGA2})) = 0.003$ yielding $U(m_{\mathrm{GD}}(\mathrm{NSGA2}), m_{\mathrm{GD}}(\mathrm{H.\,NSGA2})) = 155$

[5] Applied Mann-Whitney U-Test with $\nu_1 = \nu_2 = 30$, $p < 0.05$, medians $M(m_{\mathrm{IGD}}(\mathrm{ES})) = 0.047$, $M(m_{\mathrm{IGD}}(\mathrm{H.\,ES})) = 0.0$ yielding $U(m_{\mathrm{IGD}}(\mathrm{ES}), m_{\mathrm{IGD}}(\mathrm{H.\,ES})) = 7.5$

Table 4.3: The mean results for evaluating 30 scenarios are given. The algorithms developed in this work are highlighted in blue. Each column's best value is bold. The abbreviations stand for Hybrid Multi-objective Covariance Matrix Adaptation Evolution Strategy (H. MO-CMA-ES), Multi-objective Covariance Matrix Adaptation Evolution Strategy (MO-CMA-ES), Hybrid Non-dominated Sorting Genetic Algorithm 2 (H. NSGA2), Non-dominated Sorting Genetic Algorithm 2 (NSGA2), Hybrid Limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm with Bounds (H. L-BFGS-B), Limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm with Bounds (L-BFGS-B), Hybrid Evolution Strategy (H. ES), and Evolution Strategy (ES).

	$\mu(\mathbf{m}_{\text{NHV}})$		$\mu(\mathbf{m}_{\text{GD}})\,(\cdot 10^2)$		$\mu(\mathbf{m}_{\text{IGD}})\,(\cdot 10^2)$	
$\mathbf{n}_{\text{FE}}$	1000	40000	1000	40000	1000	40000
H. MO-CMA-ES	75%	96%	1.4	1.6	0.7	**1.0**
MO-CMA-ES	23%	68%	4.2	2.5	4.3	2.7
H. NSGA2	75%	**98%**	**0.3**	**0.3**	2.5	1.9
NSGA2	37%	78%	1.0	0.6	5.6	3.9
H. L-BFGS-B	58%	71%	1.3	3.0	3.5	3.8
L-BFGS-B	13%	31%	0.6	2.0	6.5	5.9
H. ES	**80%**	90%	0.9	1.3	**0.2**	1.1
ES	36%	77%	4.1	1.3	4.3	2.3

generated by the standard optimizers still lead through static obstacles. In contrast, the paths that were generated by the hybrid algorithms are already nearly collision-free.

Discussion

Again, the results indicate the weakness of the standard gradient-based L-BFGS-B approach. The reason for that could be the multimodal character of the objective functions. The L-BFGS-B solver pushes the paths into local optima and then terminates. This time, the L-BFGS-B algorithm was plugged into the hybrid framework, showing that the Dijkstra preprocessing improved the obtained Pareto set's quality significantly. Nevertheless, both L-BFGS-B and H. L-BFGS-B lag behind the hybrid metaheuristic approaches. Those, i.e., H. ES, H. MO-CMA-ES, and H. NSGA2, show by far the best performances.

The focus of this chapter lies in extending the hybrid framework to

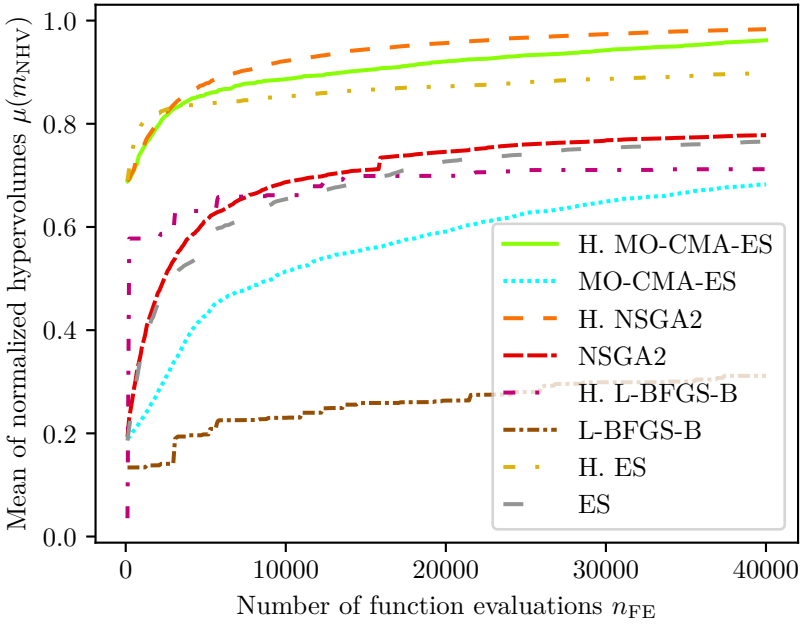


Figure 4.3: For each scenario, the optimizers' achieved hypervolumes are normalized to the best and worst achieved hypervolumes. The normalized hypervolumes are averaged over all 30 scenarios and shown dependent on the number of cost function evaluations. The abbreviations stand for Hybrid Multi-objective Covariance Matrix Adaptation Evolution Strategy (H. MO-CMA-ES), Multi-objective Covariance Matrix Adaptation Evolution Strategy (MO-CMA-ES), Hybrid Non-dominated Sorting Genetic Algorithm 2 (H. NSGA2), Non-dominated Sorting Genetic Algorithm 2 (NSGA2), Hybrid Limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm with Bounds (H. L-BFGS-B), Limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm with Bounds (L-BFGS-B), Hybrid Evolution Strategy (H. ES), and Evolution Strategy (ES).

three-dimensional operation spaces and including constraints. Thereby, the questions should be answered on how to deal with the dimension increase of the search space and the constraints, especially in step S1 of the framework. With the *Grid Resizing* step, it was possible to ensure that

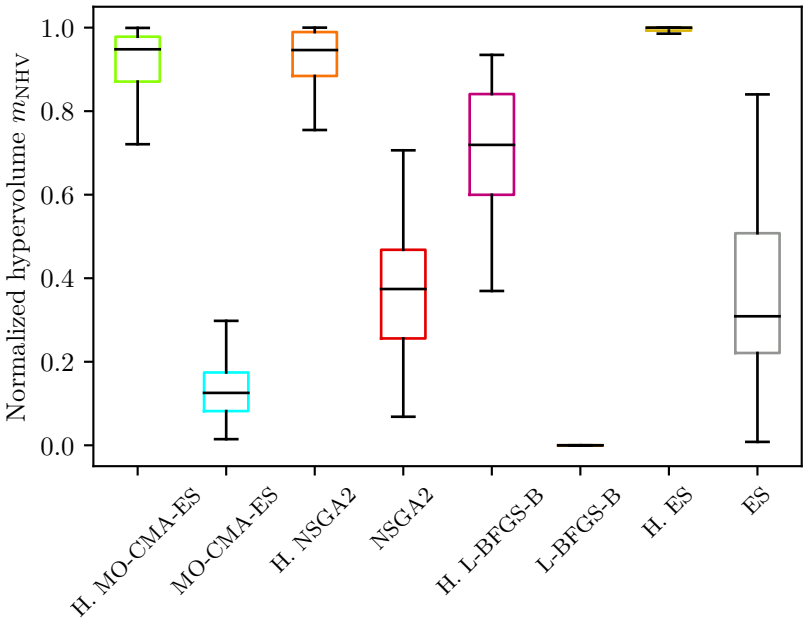


Figure 4.4: The boxplots for the normalized hypervolumes after $a = 1000$ iterations for 8 optimizers on 30 scenarios are visualized in different colors. One box spans from the data's first quartile q_1 to the data's third quartile q_3. The black line indicates the median. The whiskers enclose all data points between the lower bound $2.5q_1 - 1.5q_3$ and the upper bound $2.5q_3 - 1.5q_1$. The abbreviations stand for Hybrid Multi-objective Covariance Matrix Adaptation Evolution Strategy (H. MO-CMA-ES), Multi-objective Covariance Matrix Adaptation Evolution Strategy (MO-CMA-ES), Hybrid Non-dominated Sorting Genetic Algorithm 2 (H. NSGA2), Non-dominated Sorting Genetic Algorithm 2 (NSGA2), Hybrid Limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm with Bounds (H. L-BFGS-B), Limited-memory Broyden-Fletcher-Goldfarb-Shanno algorithm with Bounds (L-BFGS-B), Hybrid Evolution Strategy (H. ES), and Evolution Strategy (ES).

the Dijkstra algorithm remains applicable to the three-dimensional search spaces.

On the downside, the efficient pre-computation of initial solutions with

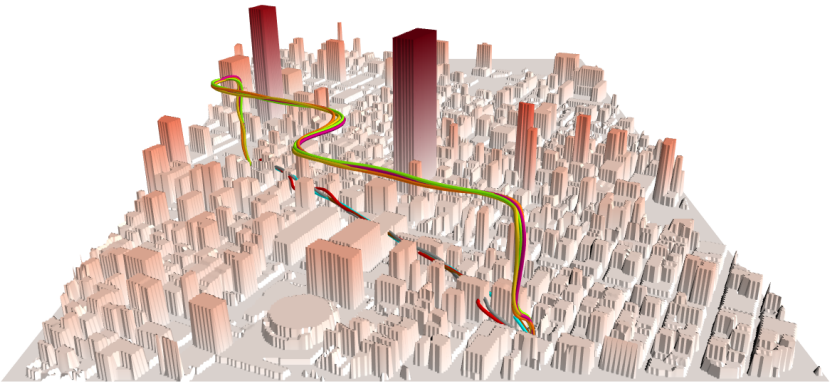


Figure 4.5: The optimal paths calculated by all solvers concerning the minimum noise criterion are shown in the colors corresponding to the previous figures. While the optima found by the non-hybrid solvers are not far from the straight-line initializations, the paths of the hybrid methods are more noise-optimized at higher flight altitudes.

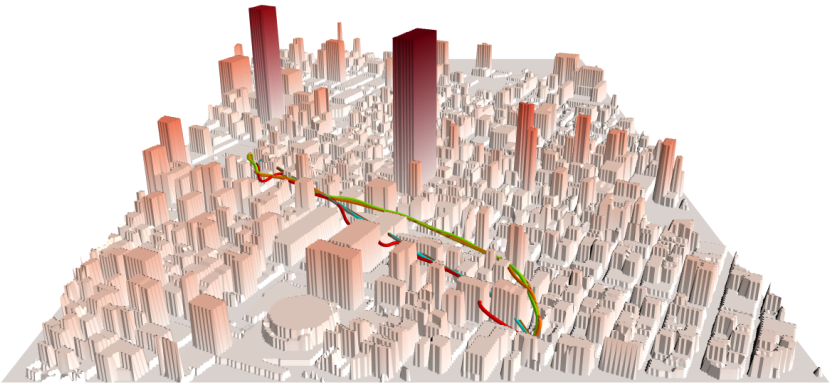


Figure 4.6: Accordingly, the optimal paths calculated by the different solvers concerning the energy consumption objective are shown here. As can be seen, the non-hybrid methods do not succeed in generating collision-free paths.

the Dijkstra algorithm depends on the size of the grid maps and thus on map dimensions and the discretization resolution. Thus, for larger scenarios, the grid resolution must be reduced, or the scenario has to be

decomposed into sub-problems.

Furthermore, constraints can also be considered in step S1. This has a decisive advantage because the initial solutions are already close to or even in the feasible region of the optimization problem. The benefit can be seen in Fig. 4.6. The paths of the non-hybrid approaches still go through static obstacles, while the hybrid methods have found a way around them.

By extending the framework with a resizing functionality and constraints, new tasks are assigned to step S2 of the framework, i.e., the metaheuristic optimization. In step S2, the grid-based objectives are optimized based on their original resolution $\begin{bmatrix}\delta_x & \delta_y & \delta_z\end{bmatrix}$ and the non-grid-based objectives based on their original non-transferred formulation (e.g., the energy consumption model (3.4)). By removing the simplifications made for step S1, there is new potential for improvement in step S2 of the framework. Furthermore, even if in step S1, all boundary conditions can be safely satisfied for the discretized environment and path representation, after the smoothing and approximation step (i.e., the change in the path representation), there may be constraint violations again. The metaheuristic algorithm needs to remove these in step S2. Here, a problem arises, which will be explained and solved in the following.

4.4.3 Problem in the MOPP Framework

The problem originates in the presence of constraints. To explain this further, we look at an example. Let us assume that the Dijkstra algorithm has found two differently shaped discrete paths for the $E = 2$ objectives of a path planning problem. Nevertheless, after the *Smoothing & Approximation* step, one of the two obtained continuous paths violates the obstacle collision constraint, whereas the other does not. This might happen due to the NURBS approximation error, causing the first path to slightly touch a building. In contrast, the remaining parts of the path are nearly optimal regarding one objective. Consequently, this solution's cost functions are assigned a high penalty term. The path is likely Pareto-dominated by the second path that satisfies all constraints. Thus, the metaheuristic algorithm removes the first path from its solution set in the first iteration. This means that the Dijkstra calculation of the first path had no purpose. However, after only minor adaptations to the path's shape, it would probably satisfy the constraints and become a good solution.

To summarize, it is possible that the presence of constraints will undermine some computations in step S1 of the hybrid framework.

4.5 Improvement (Niching)

4.5.1 Explanation

We want to solve this problem by introducing *Niches*. *Niching* is the generic term for a class of techniques commonly used in evolutionary multimodal optimization [74]. By dividing the set of solutions into sub-populations, *Niching* aims to preserve the solution diversity during optimization to find multiple (local) optima. The similarity of different state-of-the-art *Niching* approaches lies in utilizing a distance metric to assign solutions to different sub-populations [75]. The *Niching* approach introduced in the following differs from others [76]–[78] in that it neither needs any additional computations nor extra parameters to build sub-populations. Instead, we can directly use the pre-processed candidate solutions as the origin of a sub-population:

The *Niching* approach proposed for the hybrid MOPP framework is part of the metaheuristic optimization step S2 in the framework shown in Fig. 4.2. During the initialization phase of the Multi-objective Evolutionary Algorithm (MOEA), a separate niche is created for each of the n_{WS} pre-processed Dijkstra solutions. The evolutionary algorithm then applies the reproduction and selection process within each niche separately instead of applying it to the complete population of solutions. The iterative, evolutionary optimization process continues within the niches until at least one candidate solution in every sub-population satisfies all constraints. When this happens, the niches are dissolved into one population, and the evolutionary process continues.

In addition to the n_{WS} regular niches, we also introduce a niche for the initial straight-line solution. This niche is treated the same way as the others, except that the 'constraint satisfaction condition' does not have to be valid for the transition to the entire population.

4.5.2 Evaluation

Observation

Next, the introduced *Niching* feature is evaluated. For 100 scenarios (i.e., start and endpoint pairs) separately, two separate path optimization runs are executed with the same hybrid framework. The difference is that in the first run, the *Niching* feature is switched off, i.e., at the beginning of step S2, all pre-processed solutions are directly put into one population.

In contrast, in the second run, the feature is switched on, i.e., all pre-processed solutions at the beginning of step S2 live in separate niches and are optimized independently. In both runs, the hypervolumes of the generated Pareto sets are measured after every function evaluation. The hypervolume values are normalized to the best and worst hypervolumes achieved throughout all function evaluations. In Fig. 4.7, the mean of the normalized hypervolumes for all scenarios is plotted over the number of cost function evaluations.

Initially, the mean values of the hypervolumes of both strategies increase similarly. After about $n_{\mathrm{FE}} = 100$ evaluations, the curves diverge strongly and reach a difference of 0.19 after $n_{\mathrm{FE}} = 5000$ evaluations. Finally, the method with the applied *Niching* strategy is 29% better than the one without the *Niching* strategy. In other words, the *Niching* strategy requires $\Delta n_{\mathrm{FE}} = 2300$ fewer function evaluations to achieve the same hypervolume as the approach without *Niching*.

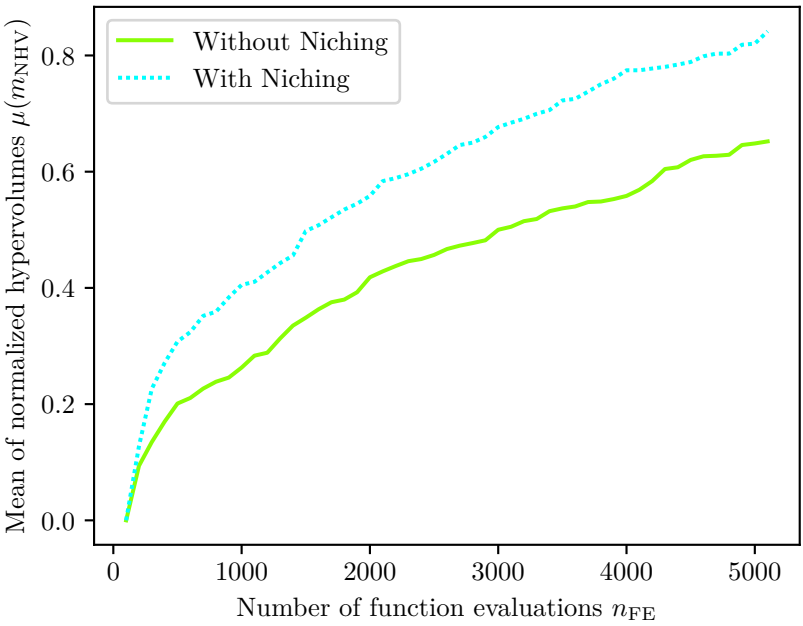


Figure 4.7: Performance test results for the evaluation of the *Niching* feature

Discussion

The results of the described experiment demonstrate the positive effect that *Niching* has on the proposed path planning framework under the presence of constraints. The problem that is solved with the *Niching* feature lies in the different grid scaling and path representation in both stages S1 and S2 of the hybrid framework. This causes good Dijkstra solutions to potentially violate the constraints in stage S2. The better performance of the framework with the *Niching* feature is due to the initial preservation of all pre-processed solutions, constraint-violating or not, assuming that within a few iterations, the constraints are potentially resolved.

Summarizing, *Niching* allows to integrate constraints into the hybrid optimization framework while ensuring that no pre-processed solution will initially be rejected for constraint violation. However, the framework user must ensure that the defined constraints are satisfiable. Otherwise, the niches will never migrate into the complete population, and the intended many-objective optimization will fall back into E single-objective optimizations.

4.6 Summary & New Results

This chapter was about formulating, evaluating, and improving a method to solve the constrained, multi-objective, three-dimensional path planning problem. First, the path planning problem presented in Chapter 3 was extended to a three-dimensional planning space. The resulting increase in complexity was presented.

Two exemplary but realistic constraints were formulated that penalize flying below the minimum flight altitude and colliding with static obstacles in the three-dimensional planning space.

In the next section of the chapter, the hybrid Multi-objective Path Planning (MOPP) framework for the unconstrained, multi-objective, two-dimensional path planning problem from Chapter 3 was extended by modules that can cope with the new challenges.

The new extended MOPP framework was evaluated together with state-of-the-art optimizers on a three-dimensional urban real-world scenario.

To sum up, the statistical results of the comparison revealed the effectiveness of the extended hybrid MOPP framework. The hybrid embedding consistently achieved significantly better results than the standard approaches for all examined metrics. The advantages already observed in

Chapter 3 can now be further enhanced by the fact that constraints are already being taken into account in the Dijkstra optimization stage S1.

Finally, it could be shown that the presence of constraints can, however, in some cases, also hinder the idea of the hybrid method. In order to solve this problem, the integration of a *Niching* process into the metaheuristic optimization stage has been proposed. This feature was evaluated and shown to increase the performance of the path planning framework. In conclusion, this chapter contained the following contributions:

- The computationally realizable extension of the Multi-objective Path Planning (MOPP) framework to the three-dimensional planning space.
- Leveraging the MOPP framework's design and using evolutionary *Niches*, a concept borrowed from the nature-inspired multi-modal optimization community, to enable the efficient integration of constraints into the optimization problem.
- The large-scale statistical analysis of the extended MOPP framework in a cross-domain comparison with existing methods.

5 Many-objective Three-dimensional Path Planning with Constraints

5.1 Introduction

The hybrid Multi-objective Path Planning (MOPP) framework for multiple objectives introduced in Chapter 3 and extended in Chapter 4 has produced promising results in the examinations made for two objectives. Since the number of objective functions was until now $E \leq 3$, the term *multi*-objective optimization was used. However, real-world problems often consist of more than just two optimization criteria. As shown, for example, by Bauranov *et al.* [14], planning flight paths for unmanned aerial vehicles in urban areas requires the consideration of 1) safety, 2) legal, and 3) economic aspects. In addition, attention to 4) social factors for gaining public acceptance becomes increasingly essential. Nevertheless, scientists have rarely considered this crucial social component in their Urban Air Mobility (UAM) applications [14].

Thus, this chapter focuses on investigating the MOPP framework for optimizing paths concerning four criteria. Because $E > 3$ now applies to the number of criteria, we are talking about *many*-objective optimization. The optimization criteria tackled in the following can be seen as representatives for the stated areas of interest, including social acceptability: The paths should be optimized such that 1) residents are exposed to the lowest possible risk of injury (*safety requirement*), 2) there is minimal signal interference between UAVs and base stations (*legal requirement*), 3) the energy consumption is minimized (*economic requirement*), and 4) the noise received by residents is as low as possible (*social requirement*). The already introduced minimum flight height and collision constraints still apply. We now have a many-objective path planning problem, which will be defined in Section 5.2. The new problem is solved utilizing the developed MOPP framework and evaluated in Section 5.3 based on exemplary scenarios set in San Francisco. This chapter pays special attention to analyzing and interpreting the generated Pareto sets. The following questions will be addressed:

- How risky, noisy, and illegal is the (energetically) cheapest path?
- How high are the (energy) costs for a less risky, quieter, and more legal path?

So far, the number of weighted solutions n_{WS} that are initially computed in step S1 of the MOPP framework has been kept small, e.g., set to $n_{\mathrm{WS}} = E$. In Section 5.4, we will discuss the benefits of computing many initial solutions, i.e., $n_{\mathrm{WS}} > E$.

Parts of this chapter have already been published in:

- [N2] N. Hohmann, S. Brulin, J. Adamy, and M. Olhofer, „Three-dimensional urban path planning for aerial vehicles regarding many objectives“, *IEEE Open Journal of Intelligent Transportation Systems*, vol. 4, pp. 639–652, 2023. DOI: 10.1109/OJITS.2023.3299496

5.2 Problem Formulation

The three-dimensional representations for the planning environment and the path correspond to those in Section 4.2.1. The utilized objective functions have already been presented in Section 3.2.2, and the applied constraints in Section 4.2.2. Therefore, we now proceed directly with the formal description of the problem.

5.2.1 Optimization Problem

Applying the *Risk of Injury* objective function $f_{\mathrm{R}}(\mathbf{v})$, the *Noise Immission* objective function $f_{\mathrm{N}}(\mathbf{v})$, the *Energy Consumption* objective function $f_{\mathrm{E}}(\mathbf{v})$, and the *Radio Signal Disturbance* objective function $f_{\mathrm{D}}(\mathbf{v})$, which was introduced in Section 3.2.2, the constrained many-objective three-dimensional path planning problem can be described as

$$\min_{\mathbf{v}\in\mathbb{V}} \begin{cases} f_{\mathrm{D}}(\mathbf{v}), \\ f_{\mathrm{R}}(\mathbf{v}), \\ f_{\mathrm{E}}(\mathbf{v}), \\ f_{\mathrm{N}}(\mathbf{v}), \end{cases}$$
$$\text{s.t.} \begin{cases} g_{\mathrm{H},i}(\mathbf{v}) \geq 0, & \forall i \in \{1,\dots,n_{\mathrm{CP}}-2\} \\ g_{\mathrm{Z},i}(\mathbf{v}) \geq 0, & \forall i \in \{0,\dots,|\Pi|-1\} \\ g_{\mathrm{P},i}(\mathbf{v}) > 0, & \forall i \in \{0,\dots,|\Pi^{\times}|-1\} \end{cases} \tag{5.1}$$

where g_{H} is the height constraint, g_{Z} is the zenith collision constraint, and g_{P} is the plain collision constraint. Again, the search space

$$\mathbb{V} = \underbrace{[\mathbb{D}_{\mathrm{c}}^{3\mathrm{D}}] \times \ldots \times [\mathbb{D}_{\mathrm{c}}^{3\mathrm{D}}]}_{n_{\mathrm{CP}}-2}$$

has $\dim(\mathbb{V}) = 3(n_{\mathrm{CP}} - 2)$ dimensions.

5.3 Evaluation

In order to evaluate the Multi-objective Path Planning (MOPP) framework for a many-objective path planning problem, the experiment's setup is described in Section 5.3.1 and used metaheuristic solvers in Section 5.3.1. Then, the obtained results are presented in Section 5.3.2, emphasizing a discussion of social considerations. Finally, another improvement for the MOPP framework is introduced in Section 5.3.3 that facilitates the generation of many-dimensional Pareto sets.

5.3.1 Setup

Scenario

A section of San Francisco, which is visualized in Fig. A.13 in the Appendix A.2, serves as an example city for the following evaluation. Based on data from OpenStreetMap [67], grid maps are generated that contain semantic data (see Fig. A.14) and information about the height of buildings (see Fig. A.15). Different grid maps are derived using this information, which is necessary for modeling the problem's objective functions. The risk grid map $\mathcal{G}_{\mathrm{R}}^{3\mathrm{D}}$ is visualized as a cross-section at height $z = 0\,\mathrm{m}$ in Fig. A.16, and the noise grid map $\mathcal{G}_{\mathrm{N}}^{3\mathrm{D}}$ is displayed as cross-section at height $z = 0\,\mathrm{m}$ in Fig. A.17.

A table summarizing all necessary scenario parameters is given in Table 5.1.

Moreover, the rendered city map A.13 is overlaid with lines, which indicate the 100 random start and end positions for the statistical evaluation. Moreover, the positions of 4G radio masts, indicated by gray circles in the same Fig. A.13, are extracted from OpenCelliD (OCID) [79]. They are used for the calculation of the radio disturbance map $\mathcal{G}_{\mathrm{D}}^{3\mathrm{D}}$, whose cross-section at height $z = z_{\mathrm{R}}$ is shown in Fig. A.18.

Table 5.1: Scenario Parameters

	Parameter	Symbol	Value
Scenario	Origin latitude	Lat	37.7902°N
	Origin longitude	Lon	-122.4193°E
	Scenario dimensions	$x_{\min}$, $x_{\max}$	0 m, 2000 m
		$y_{\min}$, $y_{\max}$	0 m, 2000 m
		$z_{\min}$, $z_{\max}$	0 m, 300 m
	Min. flight height	$z_{\mathrm{F,min}}$	50 m
Envir.	Discr. resolution	δ_x, δ_y, δ_z	4 m, 4 m, 10 m
	Resizing resolution	$\tilde{\delta}_x$, $\tilde{\delta}_y$, $\tilde{\delta}_z$	10 m, 10 m, 10 m
Path	Num. control points	n_{CP}	ANCP
	Basis function degree	d	2
	Parametrization		chordal
	Waypoint res.	$\Delta\boldsymbol{\pi}$	25 m

Solvers

For the metaheuristic optimization step S2 in the hybrid framework, several state-of-the-art many-objective evolutionary algorithms are utilized that are the Non-dominated Sorting Genetic Algorithm 3 (NSGA3) [23], the Reference Vector Guided Evolutionary Algorithm (RVEA) [80], and the $\mathcal{S}$-Metric Selection Evolutionary Multiobjective Optimization Algorithm (SMS-EMOA) [81] with their default parameter settings. The three algorithms differ primarily in their selection operator, which significantly affects the result's quality (i.e., the Pareto set's convergence and diversity).

In the following evaluation, the performance of the standard metaheuristic solvers NSGA3, RVEA, and SMS-EMOA is compared to their respective performance in the proposed hybrid Multi-objective Path Planning (MOPP) framework (H. NSGA3, H. RVEA, H. SMS-EMOA). Any important parameters are given in Table 5.2, with all algorithms developed in this thesis marked in blue. The same initializing procedure as in Chapter 4 is applied. Moreover, the known metrics Hypervolume (HV), Generational Distance (GD), and Inverted Generational Distance (IGD) are utilized to compare the approaches.

For all solvers, the optimization runs terminate after $n_{\mathrm{FE}} = 5000$ function evaluations.

Table 5.2: The parameters for all evaluated solvers are listed. The algorithms developed in this work are highlighted in blue. The abbreviations stand for Non-dominated Sorting Genetic Algorithm 3 (NSGA3), Reference Vector Guided Evolutionary Algorithm (RVEA), $\mathcal{S}$-Metric Selection Evolutionary Multiobjective Optimization Algorithm (SMS-EMOA), Hybrid Non-dominated Sorting Genetic Algorithm 3 (H. NSGA3), Hybrid Reference Vector Guided Evolutionary Algorithm (H. RVEA), and Hybrid $\mathcal{S}$-Metric Selection Evolutionary Multiobjective Optimization Algorithm (H. SMS-EMOA).

Method	Parameter	Symbol	Value
NSGA3 **RVEA** **SMS-EMOA**	Crossover crowding deg.	η_{x}	20
	Crossover probability	p_{x}	0.9
	Mutation crowding deg.	η_{mut}	20
	Mutation probability	p_{mut}	1
	Individual mut. prob.	$p_{\mathrm{mut,ind}}$	$1/D$
	Population size	s_{pop}	100
H. NSGA3 **H. RVEA** **H. SMS-EMOA**	Num. weighted solutions	n_{WS}	2

5.3.2 Results

Observation

For each of the 100 test scenarios, the Pareto sets' hypervolumes obtained by all solvers are normalized to the best and worst hypervolume over all iterations. In Fig. 5.1, the mean of the normalized hypervolumes for all scenarios is plotted over the number of cost function evaluations. Note that a cost function evaluation evaluates one candidate solution regarding all four objective functions. The large initial performance difference of $\Delta\mu(m_{\mathrm{NHV}}) = 0.986$ between hybrid and standard solvers is striking. In the further course of the optimization runs, the mean values of the three standard approaches increase and converge towards values in the range between $\mu(m_{\mathrm{NHV}}) = 0.76$ and $\mu(m_{\mathrm{NHV}}) = 0.8$, but do not reach the initial performance of the hybrid approaches. The best hybrid approach is H. NSGA3, which achieves a mean normalized hypervolume of $\mu(m_{\mathrm{NHV}}) = 0.998$.

Furthermore, over all scenarios, the mean and the standard deviation of the normalized hypervolume, GD, and IGD values of the final obtained

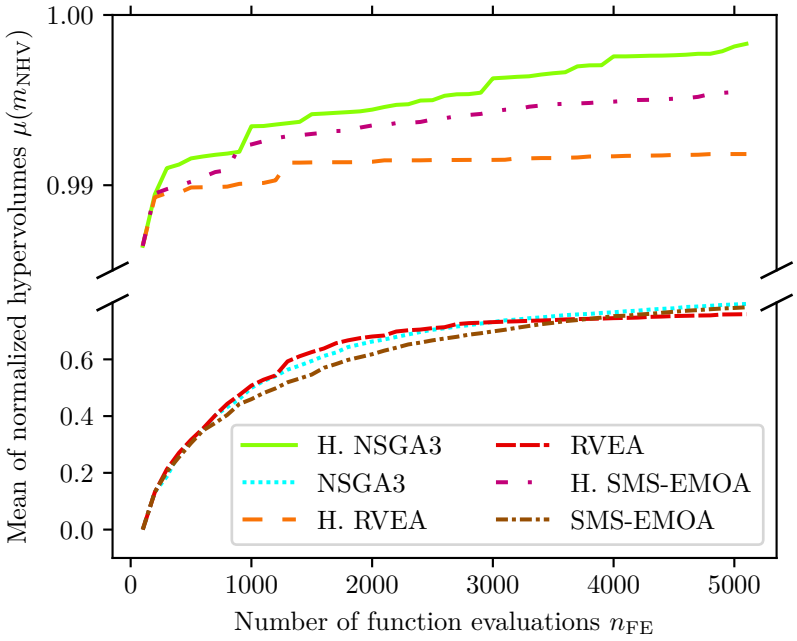


Figure 5.1: The performance of the hybrid and standard algorithms is indicated by the obtained hypervolumes that are normalized regarding the best and worst hypervolume per scenario and averaged over 100 scenarios. The abbreviations stand for Hybrid Non-dominated Sorting Genetic Algorithm 3 (H. NSGA3), Reference Vector Guided Evolutionary Algorithm (RVEA), Non-dominated Sorting Genetic Algorithm 3 (NSGA3), Hybrid $\mathcal{S}$-Metric Selection Evolutionary Multiobjective Optimization Algorithm (H. SMS-EMOA), Hybrid Reference Vector Guided Evolutionary Algorithm (H. RVEA), and $\mathcal{S}$-Metric Selection Evolutionary Multiobjective Optimization Algorithm (SMS-EMOA).

sets of non-dominated solutions are computed for all six solvers. The numeric values can be taken from Table 5.3. Note that the hypervolume is normalized to the worst and best hypervolumes of the last iteration. The set of non-dominated solutions with the highest hypervolume is chosen as a reference set for calculating the GD and IGD metrics. The best GD and IGD values are achieved by the H. SMS-EMOA method, which are 77% and 89% better than those of the best standard method RVEA.

Table 5.3: The performance comparison of the six optimizers is given. The algorithms developed in this work are highlighted in blue. Each column's best mean value is bold. The abbreviations stand for Hybrid Non-dominated Sorting Genetic Algorithm 3 (H. NSGA3), Non-dominated Sorting Genetic Algorithm 3 (NSGA3), Hybrid Reference Vector Guided Evolutionary Algorithm (H. RVEA), Reference Vector Guided Evolutionary Algorithm (RVEA), Hybrid $\mathcal{S}$-Metric Selection Evolutionary Multiobjective Optimization Algorithm (H. SMS-EMOA), and $\mathcal{S}$-Metric Selection Evolutionary Multiobjective Optimization Algorithm (SMS-EMOA).

	$\mathbf{m}_{\mathrm{NHV}}$		$\mathbf{m}_{\mathrm{GD}}$		$\mathbf{m}_{\mathrm{IGD}}$	
	μ	σ	μ	σ	μ	σ
H. NSGA3	**99%**	0.00	0.034	0.00	0.046	0.10
NSGA3	23%	0.17	2.641	25.10	2.77	25.13
H. RVEA	96%	0.10	0.075	0.00	0.109	0.00
RVEA	5%	0.14	0.132	0.00	0.244	0.14
H. SMS-EMOA	98%	0.10	**0.031**	0.00	**0.027**	0.00
SMS-EMOA	17%	0.14	2.628	25.09	2.773	25.12

For a statistical comparison of each hybrid approach with its state-of-the-art counterpart respectively, the two-tailed Mann-Whitney U-Test with $\nu_1 = 100$, $\nu_2 = 100$, and $p < 0.05$ is used. The results indicate that the samples of the normalized hypervolume, the GD, and the IGD metric differ significantly between hybrid and standard approaches.

The influence of the Dijkstra initialization step S1 compared to the standard *straight-line* initialization will be examined further. Therefore, in Fig. 5.2, the four-dimensional Pareto sets obtained by the NSGA3 and the H. NSGA3 solver are visualized. A red line in Fig. A.13 indicates the corresponding scenario. For the visualization, the function values of both solvers were normalized to a maximum and minimum per objective. The area enclosed by the dominated solutions of all generations is shown in dark yellow, and that enclosed by the last iteration's non-dominated solutions is gray. In particular, the extreme points of the Pareto sets for all objectives are highlighted, as is the knee point solution. In the lower Fig. 5.2, we also see the objective function values of the approximated Dijkstra solutions in black, which serve as initial values in the H. NSGA3. It is noticeable that these are very close to the extreme points finally reached.

Comparing the Pareto sets of the standard and the hybrid approach,

the differences in the extreme values in the upper and the lower Fig. 5.2 are in order $\Delta f_{\mathrm{D}} = 0.284$, $\Delta f_{\mathrm{R}} = 0.008$, $\Delta f_{\mathrm{E}} = 0.002$, and $\Delta f_{\mathrm{N}} = 0.022$. Here, a comparatively larger difference in the noise immission and radio disturbance objectives can be observed, and we will take a closer look at these two objectives in the following.

Let us first consider the noise immission objective. The corresponding initial solution in the lower Fig. 5.2 (black line with a dot in the lower right corner) differs only slightly from the noise immission extreme point of the standard method in the upper Fig. 5.2 concerning the noise immission fitness. This solution achieves a significantly worse fitness than the standard method regarding the other objectives. This solution might not be useful for a decision maker because of the high-risk value. However, during the optimization, this solution is likely to be responsible for finding other non-dominated solutions (gray lines with $f_{\mathrm{E}}(\mathbf{v}) \approx 0.4$ in the lower Fig. 5.2), which could be interesting for the decision maker. Thus, the hybrid approach creates more diversity in the computed Pareto set, giving the decision maker more flexibility in choosing an appropriate path.

Second, let us consider the radio signal disturbance objective. In this case, the Dijkstra step S1 provides a much better solution (black line in the lower left corner of the lower Fig. 5.2) than the standard approach. Around this solution, the evolutionary algorithm subsequently provides a high density of non-dominated solutions in the objective space, as seen from the many gray lines in the immediate neighborhood. Also, the knee point solution (violet line), often chosen as a good trade-off solution, is among these.

Discussion

Regarding the many-objective path planning problem examined, we observed that the hybrid framework achieves significantly better results than the standard methods. The reason for this is the pre-processing step. The Dijkstra algorithm finds good approximations for the final Pareto set's extreme points, which can be seen by the initial solutions' objective values (black points) in the lower Fig. 5.2. They almost correspond to the final extreme points. The critical insight is that pre-processing increases the quality of the final extreme points even though the Dijkstra algorithm searches on a different grid scaling and with another path representation than the evolutionary algorithm.

Providing good initial extreme points, the second-stage evolutionary algorithm does not need to excessively search in unknown areas of the search

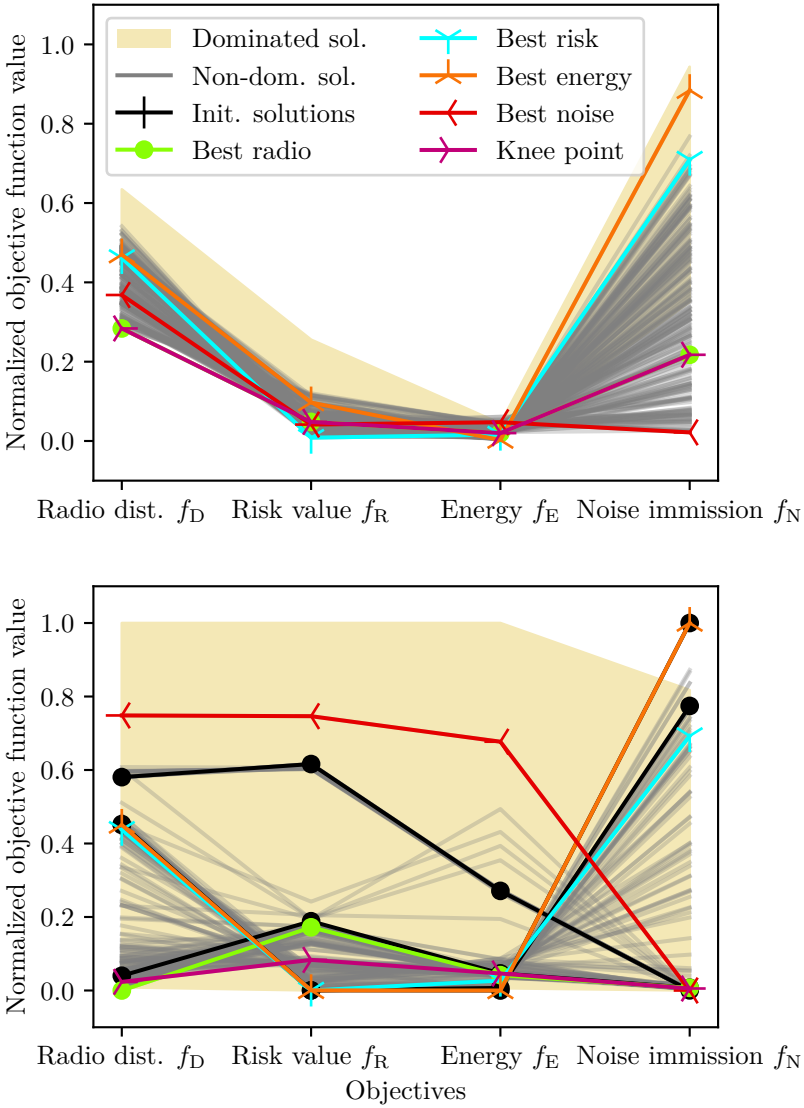


Figure 5.2: Pareto set comparison for a selected scenario

space (exploration) but instead finds non-dominated solutions that compromise the already pre-computed extreme point solutions (exploitation). In other words, although the extreme points of a Pareto front are unlikely to play an essential role in real applications, making them available to the optimizer is advantageous. Thus, the boundaries of the meaningful search space (i.e., the part of the search space where one can expect to find non-dominated solutions) are already better known. Consequently, this improves the chances of the evolutionary algorithm finding non-dominated solutions in highly diverse parts of the search space more efficiently.

We observed from Fig. 5.1 that without those pre-computed extreme point solutions, the standard metaheuristic algorithms need considerably more iterations to explore the search space and find comparably reasonable solutions. During this exploration, the evolutionary algorithm can most likely get stuck in local optima due to the highly multimodal character of the optimization problem and, therefore, achieves a worse convergence to the true Pareto Front, as the higher (i.e., worse) GD values in Table 5.3 suggest.

Moreover, the higher (i.e., worse) IGD values in the same table show that the standard evolutionary algorithms fail to achieve a comparably good diversity of the Pareto set.

To sum up, the hybrid approach for many-objective path planning problems shows several advantages in comparison to standard many-objective optimization approaches, which are the quality of the obtained Pareto sets and the number of iterations (i.e., the computation time) needed to calculate them. The decision maker obtains more diversified paths that satisfy the needs of multiple stakeholders differently.

Social Considerations

In the following section, we will examine the paths obtained by the H. NSGA3 algorithm for the exemplary scenario that is indicated by the red line in Fig. A.13. The focus here is on interpreting the trade-offs made between the four objectives. The motivation for this study lies in the assumption that companies want to maximize their profits in addition to legal and safety criteria, while social criteria are of secondary importance. Here, we want to investigate how expensive it would be for a company to consider social components.

We have already seen the Pareto set for this scenario in lower Fig. 5.2. With corresponding colors, the matching path representations of the Pareto set's extreme points and its knee point are visualized in Fig. 5.3.

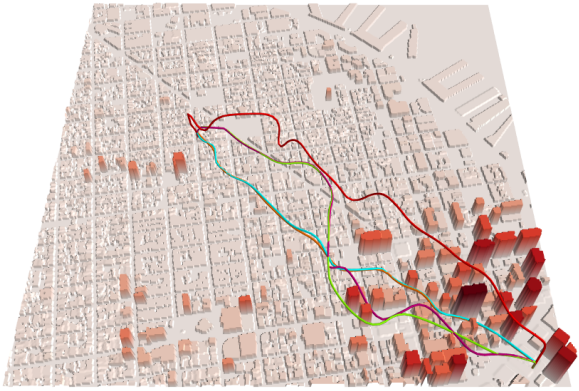


Figure 5.3: For an exemplary scenario, the plot shows the paths belonging to the Pareto set's extreme points (radio: green, risk: blue, energy: orange, noise: red) and the knee point solution (violet) in the three-dimensional height map.

The visible paths have different qualities. The Pareto set's extreme point paths are optimized regarding the respective objective. However, they can achieve sub-optimal function values for the remaining objectives. For example, a noise-optimal path runs at higher altitudes above the city, resulting in high energy consumption (i.e., higher costs for a company).

The first four rows of Table 5.4 approve this coherence. The value in each cell indicates the percentage by which the solution specified in the first column differs from the solution specified in the first row regarding the respective objective. For example, the fourth row and third column of Table 5.4 reads: 'The best noise path is 512% worse than the best energy path in terms of energy consumption'. The fictional UAV delivery company would probably not use this socially acceptable path due to the high monetary costs.

If the company were purely energy-cost-driven, it would choose the energy-optimal path. This, in turn, means a noise increase of 3094% for the residents compared to their favored solution and thus is far from socially acceptable.

From this example, it quickly becomes clear that the endpoints of the Pareto front are usually not reasonable realistic solutions. Instead, a practical compromise solution should be found. For example, a suitable trade-off solution could be the knee point of the Pareto set. The fifth row in Table 5.4 depicts the relative deviations of the knee point solution to the

Table 5.4: The deviation from the best-obtained fitness value is shown.

Deviation between	Best radio solution	Best risk solution	Best energy solution	Best noise solution
Best radio sol.	0%	212%	32%	22%
Best risk sol.	204%	0%	20%	2143%
Best energy sol.	210%	0%	0%	3094%
Best noise sol.	349%	920%	512%	0%
Knee point sol.	11%	103%	35%	17%

respective best solutions. If the fictional company chooses the knee point as a path, this would mean a 35% cost increase to fly a path that is only 17% worse than the noise optimal path in terms of noise. The company's decision maker can also choose from many other compromise solutions using the finely sampled Pareto set visualized in lower Fig. 5.2.

5.3.3 Problem in the MOPP Framework

The example in the last section showed the high relevance of many-objective optimization problems for real-world applications. The goal is a densely sampled Pareto set from which a solution can be chosen freely.

The next section will investigate whether the quality of the Pareto sets computed in the hybrid framework can be further improved by more initial solutions computed in the Dijkstra step S1 of the proposed path planning framework.

5.4 Improvement (MWSP)

5.4.1 Explanation

We recall that during step S1 of the hybrid Multi-objective Path Planning (MOPP) framework shown in Fig. 4.2 a Dijkstra Algorithm (DA) calculates n_{WS} initial solutions from differently weighted aggregations of all objective functions that are grid-based or grid-transferable. This grid aggregation has been described by equation (3.8). So far, in Chapter 4 and Chapter 5, the number of weighted solutions has equaled the number of objectives ($n_{\mathrm{WS}} = E$) without any further explanation. Thus, the aggregation weights λ_i have either been one or zero, resulting in the approximate calculation of the Pareto set's extreme points.

Now provided that the grid maps are of the same size and resolution, it is possible to perform a finer weighted aggregation to obtain more solutions on the convex subset [82] of the Pareto front. In the following, we call this approach Multiple Weighted Start Points (MWSP). How can we calculate the different weight combinations?

- When having an optimization problem with E objectives, a simple approach is to choose n_{WS} uniformly distributed weight vectors on the E-dimensional hyperplane that is spanned by all E-dimensional unit vectors. The weights λ_i then add up to one.
- In an alternative approach, called Adaptive Weight Determination Scheme (AWDS) [73], the weights are generated and refined online while calculating the Pareto set. Due to the iterative process of the AWDS approach, E^i new weight combinations are created in the i^{th} iteration depending on the number of objectives E. In order to handle this exponential increase in combinations, Barth *et al.* [83] suggests sorting out certain weight combinations based on a similarity measure of the generated paths.

Since the aim is to investigate whether the MWSP approach has *any* effect at all on the path planning framework, uniformly distributed weights are used for the reason of simplicity[1].

Each increase of n_{WS} and thus each additional initial solution also means another Dijkstra run and thus an increased computational effort. A compromise must be made.

In the following Section 5.4.2, we want to investigate how the number of weighted solutions n_{WS} affects both the quality of the Pareto sets and the computational effort. However, before that, we need to look closer at the design of the MWSP feature in the following paragraph.

Weighting grids problem So far, grid maps whose cell entries were of arbitrary magnitude have been used. Using only weights $\lambda_i \in \{0, 1\}$ in the aggregation (3.8), this has not been a problem. When using finer weight resolutions, grid maps with entries of different magnitudes should only be added up in a weighted aggregation with prior normalization. Otherwise, the Dijkstra path search would be biased towards the grid maps of higher

[1] Note that there exist advanced techniques to choose the weights and calculate a convex Pareto set, e.g., [84], which are not applicable to the problem dimensions (i.e., grid sizes) tackled in this thesis.

magnitude, superseding the MWSP approach. Accordingly, we want to look at three possibilities for the needed normalization step.

1. The first and most obvious approach would be a standard normalization of all entries of a grid map $\mathcal{G}$ into a value range between a and b

$$\mathcal{G}_{\text{norm}} = (\mathcal{G} - \min(\mathcal{G})) \cdot \frac{b-a}{\max(\mathcal{G}) - \min(\mathcal{G})} + a.$$

   However, the use of this normalization is not meaningful since adding a constant to all weights of a graph (i.e., to the entries of a grid) affects the shortest path calculation of the Dijkstra Algorithm (DA)[2]. Multiplying all edge weights with a constant is solution invariant, which is why two further normalization techniques are proposed in the following.

2. Dividing the grid entries of a grid $\mathcal{G}$ by the sum of all grid entries

$$\mathcal{G}_{\text{norm}} = \mathcal{G} \cdot \frac{1}{\sum_{(i,j,k)\in\mathbb{D}_\text{d}} \mathcal{G}(i,j,k)},$$

   in the following called Divide By Sum (DBS) normalization, yields to $\sum \mathcal{G}_{\text{norm}} = 1$ and thus assures that all grid map entries lie between zero and one $\mathcal{G}_{\text{norm}} : \mathbb{D}_\text{d} \rightarrow [0,1]$. With all grid map values having the same magnitude, the described bias is expected to be diminished.

3. Lastly, we want to apply the normalization

$$\mathcal{G}_{\text{norm}} = \mathcal{G} \cdot \frac{1}{\gamma},$$

   and calculate the grid normalizing constant γ_i for every objective's grid map $\mathcal{G}_i$ in dependence on the other involved grid maps. To quantify γ_i, we look at a small artificial example of two grid maps (i.e., a two-objective path optimization) with two cells each. Let there be two optimal paths, which only cross one cell between their start and endpoint. The optimal path A regarding objective f_1, of course,

[2] Think of a path A from start to goal consisting of two edges with weight $w = 2$ respectively and a path B from start to goal consisting of three edges all assigned with weights $w = 1$. Thus, the shortest path is path B with costs of 3. Now, adding 100 onto all edges, the shortest path is A with costs of 204 vs. path B now costing 303.

leads through the minimum $\min(\mathcal{G}_1)$ of the first grid, and we assume that it also leads through the maximum $\max(\mathcal{G}_2)$ of the second grid. The same applies to the optimal path B regarding objective f_2. Remember that the grid value's magnitudes are arbitrary. Ideally, the costs of both paths on an equally weighted (i.e., $\lambda_1 = \lambda_2 = 0.5$) aggregation of both normalized grid maps are supposed to be equal, leading to the relationship

$$\underbrace{0.5 \cdot \frac{\min(\mathcal{G}_1)}{\gamma_1} + 0.5 \cdot \frac{\max(\mathcal{G}_2)}{\gamma_2}}_{\text{costs for path A}} \stackrel{!}{=} \underbrace{0.5 \cdot \frac{\min(\mathcal{G}_2)}{\gamma_2} + 0.5 \cdot \frac{\max(\mathcal{G}_1)}{\gamma_1}}_{\text{costs for path B}}.$$

If we set γ_2 to an arbitrary positive value, we obtain γ_1 to be

$$\gamma_1 = \frac{\max(\mathcal{G}_1) - \min(\mathcal{G}_1)}{\max(\mathcal{G}_2) - \min(\mathcal{G}_2)} \cdot \gamma_2.$$

This principle can be applied to an arbitrary number of objectives E by assuming that the optimal path regarding objective f_i leads through the minimum of the grid map $\mathcal{G}_i$ and through the maxima of all $E-1$ remaining grid maps. For all two-pair combinations of all objectives, this leads to the relationship

$$\frac{1}{\gamma_i}(\max(\mathcal{G}_i) - \min(\mathcal{G}_i)) - \frac{1}{\gamma_j}(\max(\mathcal{G}_j) - \min(\mathcal{G}_j)) = 0.$$

The resulting system of $\frac{1}{2}E(E-1)$ linear equations is well determined for $E = 3$ and over-determined for $E > 3$ and can be solved for the solution vector $\begin{bmatrix}\gamma_1 & \cdots & \gamma_E\end{bmatrix}$ with a least square optimizer and the non-zero constraint $\gamma_i > 0$. We call this approach Least Square (LS) normalization in the following.

Figure 5.4 shows the evaluation results for a comparison between the DBS normalization, the LS normalization, and no applied normalization on the 100 test scenarios located in San Francisco whose setup had already been discussed earlier in this chapter. The dominance of the LS normalization can be seen.

5.4.2 Evaluation

Observation

Now, the introduced MWSP feature is evaluated. For all 100 start and endpoint configurations of the already presented San Francisco scenario, the

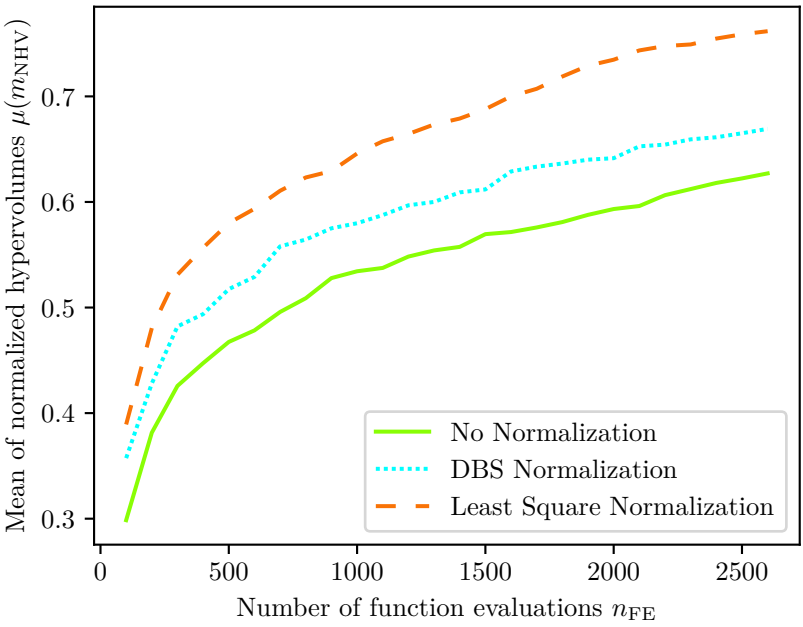


Figure 5.4: Performance test results for the evaluation of different grid map normalization techniques

four-objective optimization problem (5.1) is solved with different numbers of weighted solutions n_{WS} in the pre-processing step of the path planning framework. Using the proposed equidistant weights sampling on a four-dimensional hyperplane, the number of weighted solutions for different sampling resolutions is $n_{\mathrm{WS}} = \{4, 10, 20, 35, 56, 84\}$. The grid maps are normalized before the weighted aggregation using the proposed Least Square (LS) normalization. The problem is solved using H. NSGA3 with the ANCP and the *Niching* feature enabled. The mean of the normalized hypervolumes for all scenarios is plotted over the number of cost function evaluations in Fig. 5.5. The evaluation results indicate that increasing the number of weighted solutions from $n_{\mathrm{WS}} = 4$ to $n_{\mathrm{WS}} = 10$ improves the mean normalized hypervolume by 21%. Furthermore, using a number of weighted solutions of $n_{\mathrm{WS}} = 20$ or higher achieves an even better mean

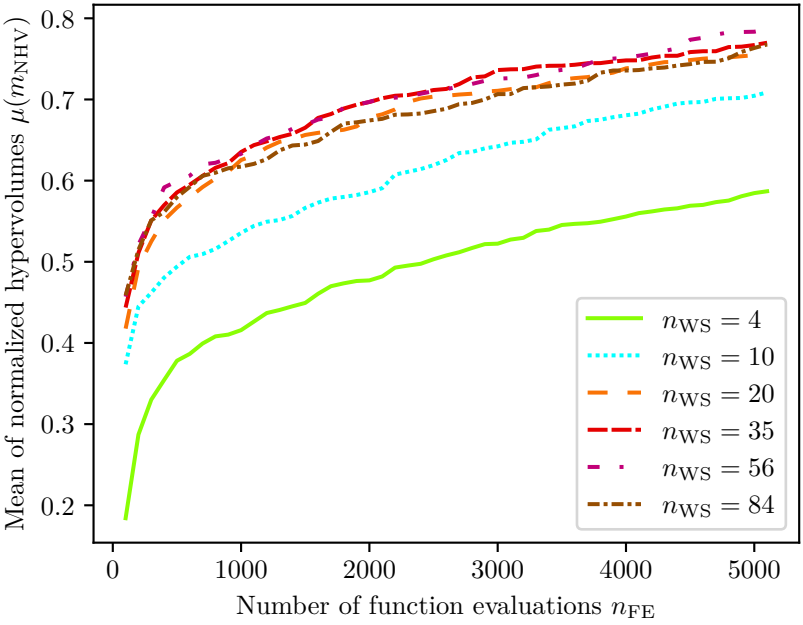


Figure 5.5: Performance test results for the evaluation of the Multiple Weighted Start Points (MWSP) feature

normalized hypervolume, an improvement of 34% compared to $n_{\mathrm{WS}} = 4$. It is also worth noting that increasing the number of weighted solutions from $n_{\mathrm{WS}} = 20$ onward does not considerably increase the performance anymore. We investigate potential reasons for this in the following.

Discussion

Figure 5.6 shows the number of distinct (i.e., unique) Dijkstra solutions $\tilde{n}_{\mathrm{WS}}$ for each scenario as colored horizontal lines. The gray bar illustrates the potential number of distinct solutions (i.e., n_{WS}). For example, among the 100 test scenarios, there were two scenarios where $n_{\mathrm{WS}} = 84$ different weightings resulted in $\tilde{n}_{\mathrm{WS}} = 39$ unique Dijkstra paths as visualized by the upper brown bar in Fig. 5.6. The remaining 45 weightings resulted in Dijkstra paths that have already been calculated.

This shows that increasing the resolution of the weighted aggregation has no effect beyond a certain number of weighted solutions. On the contrary, additional unnecessary computations increase required computing resources, as can be seen in Fig. 5.7. Thus, the user must trade between computation time and performance gain. Ideally, the number of distinct Dijkstra solutions $\tilde{n}_{\mathrm{WS}}$ equals the number of weighted solutions n_{WS}, and each additional weighted aggregation leads to an already calculated Dijkstra path. Figure 5.6 has shown that the number of distinct Dijkstra solutions depends on the chosen scenario and, thus, on the start and endpoint of the path. The characteristics of the grid maps seem to be of crucial importance for the choice of an appropriate weighting resolution.

The generalization of this finding leads to the problem of identifying a correlation between the grid maps' textures and an optimal weighting resolution, which is beyond the scope of this thesis. Therefore, we restrict ourselves to an empirical hyperparameter tuning of n_{WS}. For the San Francisco scenario, one could choose the number of weighted solutions n_{WS} between 20 and 35, which seems to be a suitable compromise between performance maximization and reasonably utilized computation time.

To sum up, given the requirement of all objective's grid maps having the same size and resolution, the use of Multiple Weighted Start Points (MWSP) in the pre-processing step proves to be helpful, where the number of weighted solutions n_{WS} is a hyperparameter to be set by the user in dependence on the scenario at hand.

5.5 Summary & New Results

This chapter examined the hybrid Multi-objective Path Planning (MOPP) framework's performance in solving a many-objective optimization problem with constraints. It was shown that the Pareto sets obtained by the hybrid approaches significantly outperform the results achieved by the standard many-objective optimizers.

Special attention was given to the study of four-dimensional Pareto fronts. The four criteria represented real-world requirements of a UAM system. Legal, safety, economic, and social aspects were included. The results of a huge statistical analysis based on an exemplary city showed that most economic paths are generally socially unacceptable. It could be shown that the hybrid MOPP framework generates reasonable compromise solutions, which entail acceptable monetary costs while considering social criteria.

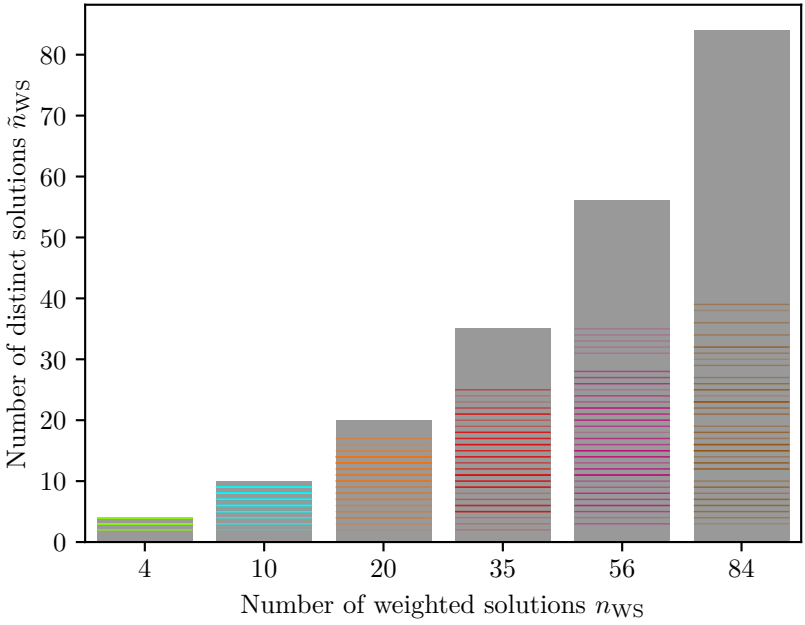


Figure 5.6: Number of distinct weighted solutions $\tilde{n}_{\mathrm{WS}}$ in comparison to all pre-processed weighted solutions n_{WS} on 100 test scenarios. The gray bar visualizes the number of weighted solutions. Ideally, they would all lead to a distinct Dijkstra path. In reality, many weight combinations result in the same Dijkstra paths, which means an unnecessary calculation overhead. The number of distinct Dijkstra paths $\tilde{n}_{\mathrm{WS}}$ is plotted for each scenario as a thin colored line.

Finally, the usefulness of using several weighted start solutions in the Dijkstra step S1 was demonstrated, and several methods for the necessary normalization of the grid maps were presented and compared.

The contributions of this chapter can be summed up as

- The demonstration of the hybrid Multi-objective Path Planning (MOPP) framework to be applicable to four-objective optimization problems.
- The consideration of social factors in an optimization problem that is often only considered economically.

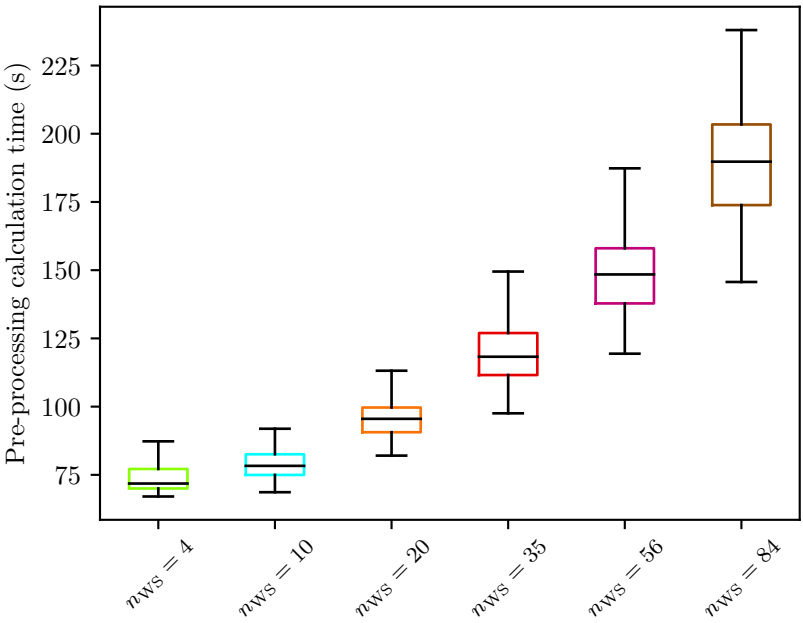


Figure 5.7: Pre-processing step calculation times for different numbers of weighted solutions n_{WS} on 100 test scenarios

- The derivation of an effective normalization method for the weighted aggregation of grid maps to be used in single-objective shortest path search.
- The demonstration of an upper performance bound in weighted aggregation approaches with finer resolved weights.

The presented hybrid MOPP framework for many-objective three-dimensional path planning allows to efficiently find paths that connect a given start and endpoint regarding different objectives. However, aerial transportation services in a city are expected to require more than one single path but rather a complete transportation network. The next chapter concerns generating such connected aerial corridor networks from the optimized single paths.

6 Multi-objective Traffic Network Optimization

6.1 Introduction

So far, only *paths* were optimized, with each path always connecting exactly two spatial points, i.e., vertiports. This seems reasonable as early UAM projects are likely only to realize single routes due to governmental approval processes and the gradual gaining of acceptance by the public. Then, these individual routes could successively be expanded to include more vertiports and new connections. However, if the number of vertiports increases, the airspace will be traversed by many paths, the union of which could be seen as a dense network. It then makes sense to identify frequently used airspace areas, i.e., joint air corridors, and connect them while neglecting less critical routes. Ultimately, this would lead to a transportation network. The following questions arise when deriving such a traffic network from a given set of paths.

1. How can we identify and merge joint paths into a network? In the following, this problem is called *Path-merging*.

2. Which of this network's edges should be kept and which should be discarded to save costs? This problem is referred to as *Network Optimization* in the following. The term *cost* is intentionally kept general here and can be interpreted differently depending on the stakeholder, leading to a multi-criteria *Network Optimization* problem.

The combination of these questions is called a Multi-objective Traffic Network Optimization (MONO) problem whose analysis and solution are the subject of this chapter.

The applicability of the proposed solution is demonstrated by a study conducted for the city of Frankfurt, Germany. The evaluation analyzes the trade-off solutions between social and economic aspects.

The content of this chapter has partly been published in:

- [N4] N. Hohmann, S. Brulin, J. Adamy, and M. Olhofer, „Multi-objective optimization of urban air transportation networks under

social considerations“, *IEEE Open Journal of Intelligent Transportation Systems*, vol. 5, pp. 589–602, 2024. DOI: 10.1109/OJITS.2024.3443170

6.2 Related Work

A transportation network consisting of air corridors is only one of the different concepts for urban air infrastructure proposed in the literature. First, Section 6.2.1 presents other approaches. Subsequently, in Section 6.2.2 and 6.2.3, related work is presented that deals with the sub-problems of *Path-merging* and *Network Optimization*. Finally, Section 6.2.4 introduces the findings from the literature research, which form the basis for evaluating the solution framework.

6.2.1 Urban Air Mobility Infrastructure Design

The range of concepts for Urban Air Mobility (UAM) infrastructure design is broad. Over the last years, several different ideas and proposals have emerged from both government-supported (FAA [85], NASA [16], SESAR U-SPACE [86], DLR [87]) and industrial (Airbus [88], Embraer [89], Uber [90]) stakeholders. A good overview is provided in the review paper by Bauranov *et al.* [14].

Jang *et al.* [16] consider three types of airspace structures in their concept: sky-lanes, sky-tubes, and sky-corridors, each with varying degrees of freedom for the aerial vehicles. These structures aim to ensure safety while reducing the need for heavy investment in technology and infrastructure. Tests conducted on various structures demonstrate that a more structured environment increases safety and simplicity at the expense of throughput. Geister *et al.* [87] argue that the greater the number of vehicles in an aerial mobility system, the stronger the requirement for adhering to pre-determined flight paths. Sunil *et al.* [15] propose different types of air spaces that are 1) full mix, 2) layers, 3) zones, and 4) tubes, ranging from entirely unrestricted to fully restricted. Their simulation results show that a moderate separation, like in the layer concept, shows a better overall performance than the others regarding capacity, safety, and efficiency.

Other approaches from the literature examine especially flight *corridor* concepts in more detail. Denham *et al.* [91] investigate different corridor structures (i.e., line segment, intersection, roundabout) geometrically and in simulation concerning their capacity. Cummings *et al.* [92] exam-

ine different corridor network configurations on two scenarios concerning a throughput metric. Fedrigo [93] introduces three urban air traffic flow models. He considers the flow through a corridor structure, through a corridor with tracks, and the traffic flow while entering or exiting a corridor. Pang *et al.* [94] introduce the so-called *AirMatrix* concept that is a separation of the three-dimensional space into cubes. The cubes can be of different sizes, allowing for a denser sampling in crowded low-speed airspace and a sparse sampling in open high-speed airspace. Corridors connect areas of different resolutions.

6.2.2 Path-merging

Trajectory clustering is a subdomain of the broad field of trajectory data mining. For a good overview, the interested reader is referred to the paper of Zheng [95]. The trajectory clustering problem searches for common movement patterns in (segmented) trajectory data, groups them into clusters of similar features, and - in some cases - eventually generates a representative trajectory for a found cluster. The term *trajectory* is not equally defined in all publications. For some, a trajectory is equivalent to a path, which is two or three-dimensional spatial point data. Others use it to refer to a path with time information. Researchers use trajectory clustering algorithms, for example, to analyze aircraft trajectories near airports [96], [97].

One famous trajectory clustering algorithm is TRACLUS by Lee *et al.* [98]. They propose a three-step approach that consists of a segmentation step, a clustering step, and a representation step. First, they partition paths into sub-paths by splitting them apart at designated characteristic points while smoothing the path segments. They minimize the approximation error between path and smoothed sub-paths and the number of sub-paths. A heuristic solves this two-objective optimization problem. Then, the line segments are grouped into clusters by applying a density-based clustering algorithm (adapted DBSCAN [99] algorithm). Last, a representative for each group of line segments is found by a geometrical approach that builds a representative path by computing average coordinates of intersected line segments when sweeping a vertical line alongside the cluster's major axis. However, the TRACLUS algorithm does not apply to the problem at hand, as deriving a graph from the clustered trajectories still requires work.

An example of a trajectory-to-graph conversion is the approach by Chen *et al.* [100] to find the most popular route between two locations

by observing historical GPS data. In the first step, they calculate a transfer network from the trajectory data. The nodes of the transfer graph represent intersections in the trajectory data. The edges of the graph are then assigned transfer probabilities that increase with the edge's frequency of use for a given destination. Finally, the obtained weighted graph can be used to find the most popular route in a breadth-first search.

The map construction problem [101] can be seen as another application-related field of trajectory data mining. Given the GPS data of a vehicle traversing an unknown traffic map, it aims to reconstruct the graph of the underlying traffic map. Trajectory-to-graph conversion and map construction algorithms handle two-dimensional noisy path data to construct a graph. Thus, these approaches use two-dimensional geometric calculations that can not be applied to three-dimensional spatial paths handled here. Nevertheless, the developed *Path-merging* strategy introduced in Section 6.4.2 found inspiration in the presented approaches.

6.2.3 Network Optimization

A good overview of the literature in the domain of network optimization can be found in the article by Ding *et al.* [102]. However, they emphasize urban *ground* traffic networks. For example, Gastner *et al.* [103] take a set of initial logistic hub positions as given and create two-dimensional traffic networks by minimizing the average distance between all node pairs subject to a restricted budget. Li *et al.* [104] also assume a set of initial node positions that are additionally structured in a lattice graph. They examine how the insertion of edges between non-neighboring nodes improves the overall travel time through the network. Instead of the network structure, Chen *et al.* [105] optimize the flow through the network by adapting the edge capacities in a bi-objective optimization problem regarding total travel time and construction costs as objectives.

In all conscience, there is little research on *aerial* traffic network optimization. Related work can be found with He *et al.* [106], who build multiple dependent air traffic routes by sequentially running a shortest path algorithm for different start and endpoints. When planning a new path, they consider the previously planned paths to avoid interfering with them. The paths are optimized regarding UAV energy consumption, risk, and airspace occupancy, which means that the algorithm tries to overlap different air corridors' buffer zones. However, there is no optimization on the network level: A pair of vertiports will always be connected by an isolated path. There is no feature to merge spatially shared path seg-

ments of two paths into a combined air corridor. This work aims for a Pareto-optimal network structure instead of a set of unconnected paths, which has several advantages. First, by identifying common flight paths between several vertiports and replacing direct connections, the aerial corridor length of the complete network and, thus, the infrastructure costs can be reduced. Second, adding a new vertiport to the UAM system does not require planning many new paths for every existing vertiport. Rather, the new vertiport must only be connected to the nearest corridor in the existing traffic network.

In the research area of multi-agent navigation, Mai *et al.* [107] optimize a two-dimensional network. Instead of assuming logistic hubs as starting points, they aim for a homogeneous coverage of the entire design space. As there is no cost for the presence of an edge, the network they generate does not represent a physical transportation network but rather virtual connection possibilities. Here, the design principle of considering the entire airspace as a design space applies.

6.2.4 Findings from the Literature Review

The examination of recent literature about UAM infrastructure design reveals two important observations.

1. *The social is secondary to the economic:* Bauranov *et al.* [14] notice that most evaluated airspace concept studies use idealized network representations that commonly evaluate the proposed airspace structures concerning economic quality criteria. In most cases, the aim is to increase capacity or throughput to allow as many vehicles as possible to fly simultaneously. In some cases, the studies are extended to include the aspect of safety, which leads to trade-off considerations between economic efficiency and safety aspects. Bauranov *et al.* criticize that social factors are subordinate in the examined urban air mobility infrastructure concepts. However, the noise produced by drones constitutes the major acceptance threshold for UAM transportation systems and thus for their success [108].

2. *Straight-line connections are a biased standard:* In network optimization, traffic networks are mostly modeled as graphs, with their edges representing (air)ways. Consequently, the (air)ways are usually assumed to be straight-line connections between logistic hubs [109]. Straight-line networks could be assumed energy-optimal. They are

a sufficient assumption when networks are optimized regarding economic objectives. However, they are biased as soon as a social objective is introduced.

This work considers not only economic aspects but also social requirements in the design of aerial traffic networks. Therefore, three requirements are identified, each of which assesses the traffic network from a different point of view:

- The traffic network provider wants to minimize the maintenance costs of the traffic network.
- The traffic network users prefer short travel times.
- The residents living underneath only accept low-noise traffic networks.

The questions addressed in this chapter are twofold and based on the two stated observations.

1. First, a social objective function is integrated into the network optimization process to investigate the trade-off costs of networks that fulfill economic *and* social objectives.
2. Secondly, a traffic network is represented as a graph, not exclusively consisting of straight-line corridors but also socially pre-optimized and, therefore, in general, curved paths. The performance of this more detailed representation modeling is examined, especially regarding the newly introduced social objective.

6.3 Problem Formulation

The following formally defines the Multi-objective Traffic Network Optimization (MONO) problem. Section 6.3.1 proposes representations for the environment and network structures. Then, the *Path-merging* problem is presented in Section 6.3.2. In Section 6.3.3 and 6.3.4, the objectives and constraints for the subsequent *Network Optimization* are introduced before the optimization is formally defined in Section 6.3.5.

6.3.1 Representations

Environment

We assume a cubic space

$$\mathbb{D} = \begin{bmatrix} x_{\min} & x_{\max} \end{bmatrix} \times \begin{bmatrix} y_{\min} & y_{\max} \end{bmatrix} \times \begin{bmatrix} z_{\min} & z_{\max} \end{bmatrix} \subset \mathbb{R}^3.$$

where $x_{\min} < x_{\max}$, $y_{\min} < y_{\max}$, and $z_{\min} < z_{\max}$ are the bounding coordinates of the operational space.

Path

A path Π is a sequence of $|\Pi|$ three-dimensional points $\boldsymbol{\pi}_i \in \mathbb{D}$

$$\Pi = [\boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \ldots, \boldsymbol{\pi}_{|\Pi|}].$$

Set of Paths

A set of paths P contains $|P|$ unique paths Π_i

$$P = \{\Pi_1, \Pi_2, \ldots, \Pi_{|\mathrm{P}|}\}.$$

Network

A traffic network is represented utilizing a three-dimensional spatial graph representation $G = \{\mathring{N}, \mathring{E}\}$ visualized as a two-dimensional projection in Fig. 6.1. A graph G consists of a set of $|\mathring{N}|$ nodes $\mathring{N} = \{n_1, \ldots, n_{|\mathring{N}|}\}$ and a set of $|\mathring{E}|$ edges $\mathring{E} = \{e_{ij}\}$. Every node n_i is assigned information

- about its three-dimensional spatial position $\mathbf{p}_i = \begin{bmatrix} x_i & y_i & z_i \end{bmatrix} \in \mathbb{D}$,
- whether it represents a vertiport

$$t_i = \begin{cases} 1, & \text{if vertiport,} \\ 0, & \text{else,} \end{cases}$$

- and whether it represents a crossing in the network

$$c_i = \begin{cases} 1, & \text{if crossing,} \\ 0, & \text{else.} \end{cases}$$

An edge $e_{ij} = (n_i, n_j)$ is undirected and connects two nodes. Every edge holds information

- on a path Π_{ij} that connects $\mathbf{p}_i$ and $\mathbf{p}_j$

$$\Pi_{ij} = [\boldsymbol{\pi}_1 = \mathbf{p}_i, \boldsymbol{\pi}_2, \ldots, \boldsymbol{\pi}_{|\Pi_{ij}|} = \mathbf{p}_j],$$

- and on a social cost weight s_{ij}.

There may exist more than one edge between two nodes. Furthermore, the Euclidean distance between two adjacent nodes n_i and n_j is denoted as $|e_{ij}| = ||\mathbf{p}_i - \mathbf{p}_j||_2$, whereas the length of the underlying path Π_{ij} is labeled

$$l_{ij} = \sum_{k=2}^{|\Pi_{ij}|} ||\boldsymbol{\pi}_k - \boldsymbol{\pi}_{k-1}||_2.$$

In general $|e_{ij}| \neq l_{ij}$ yields. Unless otherwise stated, the term *edge length* refers to l_{ij} in the following.

6.3.2 Path-merging

A set of vertiport positions $\mathring{N}_\mathrm{T} = \{\mathbf{p}_{\mathrm{T},1}, \ldots, \mathbf{p}_{\mathrm{T},|\mathring{N}_\mathrm{T}|}\}$ determines where transport agents can take off and land in the network. Defining all $|C| = \binom{|\mathring{N}_\mathrm{T}|}{2}$ vertiport pair combinations (without repetition) as $C = \{(n_k, n_l) \,\forall\, n_k \neq n_l \in \mathring{N}_\mathrm{T}\}$, the MOPP framework presented in Chapter 5 can be used to optimize paths Π_i between every vertiport pair $(n_k, n_l) \in C$. Each path already has an assigned social attribute that quantifies its social compatibility. All paths are contained in the initial set of paths P. *Path-merging* is the problem of transferring this set of paths into a network representation that is a graph $G_0 = \{\mathring{N}_0, \mathring{E}_0\}$. Thereby, the following aspects must be considered.

- All vertiports should appear as nodes in the graph $\mathring{N}_\mathrm{T} \subseteq \mathring{N}_0$.
- Intersecting paths should be broken up into segments and represented as separate edges in the graph.
- Path points of two paths close to each other should be combined to belong to a single edge.

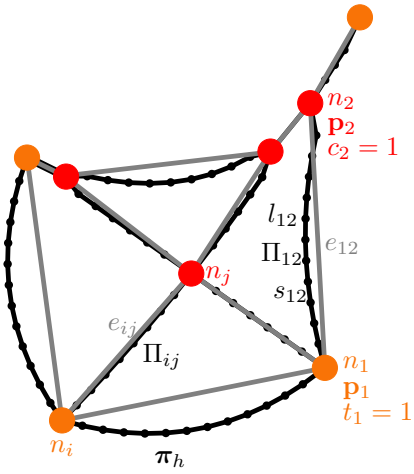


Figure 6.1: Visualization of the graph structure $G = \{\mathring{N}, \mathring{E}\}$ used as traffic network representation. All nodes $n_i \in \mathring{N}$ of the graph are shown as large circles, with vertiport nodes ($t_i = 1$) in orange and crossing nodes ($c_i = 1$) in red. Moreover, each node is assigned a three-dimensional spatial position $\mathbf{p}_i$. The edges $e_{ij} \in \mathring{E}$ of the graph are gray. The 3D path information Π_{ij} stored in each edge is shown in black. Each path has a length l_{ij} and a social attribute s_{ij}. The small black dots represent the waypoints $\boldsymbol{\pi}_h$ of the paths.

- The number of edges $|\mathring{E}_0|$ in G_0 ultimately determines the size of the optimization vector and, therefore, the complexity of the subsequent *Network Optimization* problem. It should, therefore, be as small as possible but as large as necessary to 1) represent all paths from the initial path set P and 2) be the union of all reasonable transport networks that sub-graphs of G_0 can represent.

6.3.3 Objectives

The objective functions for evaluating a traffic network during *Network Optimization* are introduced in the following. They are designed to assess the quality of a traffic network from the perspective of a traffic network operator (maintenance), a traffic network user (travel cost), and the residents living underneath the traffic network (social cost). Beforehand, definitions

are given that are needed to normalize the objective functions. The sum of edge lengths in the initial graph G_0 is denoted as

$$L_0 = \sum_{e_{ij} \in \mathring{E}_0} l_{ij},$$

and the sum of social cost weights as

$$S_0 = \sum_{e_{ij} \in \mathring{E}_0} s_{ij}.$$

Furthermore, given a set of terminal nodes $\mathring{N}_\tau \subseteq \mathring{N}$ in a graph $G = \{\mathring{N}, \mathring{E}\}$ with a set of weighted edges $\mathring{E}$, a Steiner tree [110] $G_{\mathrm{ST}} = \{\mathring{N}_{\mathrm{ST}} \subseteq \mathring{N}, \mathring{E}_{\mathrm{ST}} \subseteq \mathring{E}\}$ is a connected subset of G that contains all terminal nodes $\mathring{N}_\tau \subseteq \mathring{N}_{\mathrm{ST}}$ and minimizes the sum of its edges' weights $\min \sum_{e_{ij} \in \mathring{E}_{\mathrm{ST}}} \omega_{ij}$. When setting the set of vertiport nodes as terminal nodes $\mathring{N}_\tau = \mathring{N}_{\mathrm{T}}$, the initial graph's Steiner tree regarding edge length is denoted $G_{\mathrm{ST,L}} = \{\mathring{N}_{\mathrm{ST,L}}, \mathring{E}_{\mathrm{ST,L}}\}$. Summing its edges' length attributes, yields

$$L_{\mathrm{ST,L}} = \sum_{e_{ij} \in \mathring{E}_{\mathrm{ST,L}}} l_{ij}.$$

Moreover, $\mathcal{R}_{\{\cdot\}}(G, n_k, n_l)$ is an operator (e.g., Dijkstra's algorithm [42]) that finds the shortest route from n_k to n_l regarding an edge attribute $\{\cdot\} \in \{l, s, |e|\}$ in a graph G, resulting in a sequence of edges $R_{G,n_k,n_l,\{\cdot\}}$.

Maintenance cost

A UAM traffic network equipped with monitoring and potentially also navigation technology allows the operation of UAVs with few sensors of their own. Therefore, a larger network means more maintenance and operating costs for the network operator. Consequently, the maintenance cost objective function that evaluates a graph $G = \{\mathring{N}, \mathring{E}\}$ is defined as

$$f_{\mathrm{M}}(G) = \frac{\sum_{e_{ij} \in \mathring{E}} l_{ij} - L_{\mathrm{ST,L}}}{L_0 - L_{\mathrm{ST,L}}}. \tag{6.1}$$

The normalization constants L_0 and $L_{\mathrm{ST,L}}$ guarantee that $0 \leq f_{\mathrm{M}}(G) \leq 1$ applies.

Travel cost

To assess the network's capability to route traffic efficiently, a well-known metric from the graph domain is adopted [111]: The route factor (also detour index) between two nodes n_k and n_l in a graph can be expressed as the quotient of the shortest route cost through the graph

$$r_{|\mathrm{e}|}(G, n_k, n_l) = \sum_{e_{ij} \in R_{G,n_k,n_l,|\mathrm{e}|}} |e_{ij}|$$

and the direct Euclidean distance between n_k and n_l yielding

$$Q(n_k, n_l) = \frac{r_{|\mathrm{e}|}(G, n_k, n_l)}{||\mathbf{p}_k - \mathbf{p}_l||_2}. \tag{6.2}$$

Thus, there is an efficient (i.e., direct) connection between the two nodes if $Q(n_l, n_l) = 1$ applies. The longer the detour, the greater the route factor. The considered traffic networks generally do not have straight connections between two nodes but curved paths. Therefore, the Euclidean distance between n_k and n_l in the denominator of (6.2) is replaced by the shortest route cost between both in the initial graph G_0. Furthermore, by forming the negative reciprocal of (6.2) and adding one, we obtain an objective function having its best value in zero and its worst in one. For the final travel cost objective function, the adapted route factor definitions are averaged for all connections C, yielding the final travel cost objective function

$$f_{\mathrm{T}}(G) = \frac{1}{|C|} \sum_{(n_k,n_l) \in C} \frac{r_{\mathrm{l}}(G, n_k, n_l) - r_{\mathrm{l}}(G_0, n_k, n_l)}{r_{\mathrm{l}}(G, n_k, n_l)} \tag{6.3}$$

which ensures $0 \leq f_{\mathrm{T}}(G) \leq 1$. The function

$$r_{\mathrm{l}}(G, n_k, n_l) = \sum_{e_{ij} \in R_{G,n_k,n_l,l}} l_{ij}$$

sums the length attributes of the shortest path's edges from n_k to n_l through G. The smaller the travel cost values $f_{\mathrm{T}}(G)$ of a graph G, the shorter the established connections between all vertiport pairs compared to the shortest connections in the initial graph G_0.

Social cost

The design of the social cost objective function is similar. For each connection $(n_k, n_l) \in C$, the most socially acceptable route through G is

compared with the social optimum through the initial graph G_0 resulting in

$$f_{\mathrm{S}}(G) = \frac{1}{|C|} \sum_{(n_k,n_l)\in C} \frac{r_{\mathrm{s}}(G,n_k,n_l) - r_{\mathrm{s}}(G_0,n_k,n_l)}{r_{\mathrm{s}}(G,n_k,n_l)}, \tag{6.4}$$

with

$$r_{\mathrm{s}}(G,n_k,n_l) = \sum_{e_{ij}\in R_{G,n_k,n_l,s}} s_{ij}$$

being the totalized social cost values of the most social path from n_k to n_l through G. The relationship $0 \leq f_{\mathrm{S}}(G) \leq 1$ also applies here. A decreasing social cost value $f_{\mathrm{S}}(G)$ means that the graph G establishes connections between vertiport pairs perceived as less noisy. The social cost weight s_{ij}, which is the core of the objective function, can generally be seen as a placeholder for any quantifiable social criterion such as privacy or safety [112]. In this model, s_{ij} represents the aviation noise that city residents perceive near a flight path. For every path Π_{ij}, its social attribute

$$s_{ij} = f_{\mathrm{N}}(\Pi_{ij})$$

with the noise objective function f_{N} that has been introduced in Section 3.2.2. To repeat briefly: a flight path has a lower (i.e., better) social cost weight if it tends to run at higher altitudes or over ground traffic roads as the drone noise then vanishes in the ground traffic noise.

6.3.4 Constraints

As the optimizer will delete edges from the initial graph G_0 to create new traffic networks, this might result in graphs G that do not meet the requirements of a reasonable transportation network. These requirements are formulated using the following equality constraint functions.

Connectivity

A connected component H is a subset of an undirected graph G with a path between any pair of nodes. A graph can thus be written as a union of its $|H|$ connected components $G = H_1 \cup H_2 \cup \ldots \cup H_{|H|}$. The connectivity of the traffic networks at hand is ensured by defining the connectivity equality constraint

$$h_{\mathrm{C}}(G) = |H| - 1 = 0. \tag{6.5}$$

Vertiport Inclusion

When edges are deleted from a graph during optimization, the resulting graph may no longer include important nodes (i.e., vertiport nodes) while still being connected. A binary variable b_i is defined for every node $n_i \in \mathring{N}_0$ indicating whether the node is included in the graph G (i.e., $b_i = 1$) or not (i.e., $b_i = 0$). Then, the $|\mathring{N}_\mathrm{T}|$ equality constraint functions

$$h_{\mathrm{T},j}(G) = b_j - 1 = 0, \quad \forall \arg n_j \in \mathring{N}_\mathrm{T} \tag{6.6}$$

ensure that all vertiport nodes are part of the traffic network.

6.3.5 Optimization Problem

For the optimization, the initial graph G_0 is encoded into a binary optimization vector

$$\mathcal{E}(G_0) = \mathbf{v}_0 = \begin{bmatrix} v_1 = 1 & v_2 = 1 & \dots & v_{|\mathring{E}_0|} = 1 \end{bmatrix} \tag{6.7}$$

of length $|\mathring{E}_0|$. Inversely to the encoding operator $\mathcal{E}$, any solution $\mathbf{v}$ can again be decoded into the respective graph by applying the decoding operator $G = \mathcal{D}(\mathbf{v}) = \mathcal{E}^{-1}(\mathbf{v})$.
In order to obtain a new network G, the optimizer can either delete an edge e_{ij} by setting the corresponding $v_k = 0$ or adding a previously deleted edge again by setting $v_k = 1$ again. The search space is thus defined by $\mathbb{V} = \{0,1\}^{|\mathring{E}_0|}$. The complete optimization problem tackled in this chapter can be described as

$$\begin{aligned} &\min_{\mathbf{v}\in\mathbb{V}} \begin{cases} f_\mathrm{M}(\mathbf{v}), \\ f_\mathrm{T}(\mathbf{v}), \\ f_\mathrm{S}(\mathbf{v}), \end{cases} \\ &\text{s.t.} \begin{cases} h_\mathrm{C}(\mathbf{v}) = 0, \\ h_{\mathrm{T},j}(\mathbf{v}) = 0, \quad \forall \arg n_j \in \mathring{N}_\mathrm{T}, \end{cases} \end{aligned} \tag{6.8}$$

with $\dim(\mathbb{V}) = |\mathring{E}_0|$. For the sake of simplicity, the notations $f(\mathbf{v})$ and $f(G)$ are used interchangeably in the following without explicitly mentioning decoding.

6.4 Solution Approach

As already teased in this chapter's introduction, the Multi-objective Traffic Network Optimization (MONO) problem is divided into a *Path-merging* and a *Network Optimization* problem, which are solved one after the other and independently of each other. The complete solution pipeline will be presented in Section 6.4.2. Before, the actual *Network Optimization* problem is analyzed.

6.4.1 Problem Analysis

The optimization problem (6.8) generally belongs to the class of multi-objective combinatorial optimization problems [113]. The second and third objective functions f_{T} and f_{S} contain shortest path calculations. As a result, the presence or absence of an edge in the graph does not add a static weight to the cost function. Instead, the contribution of an edge to the cost function must constantly be recalculated for each network configuration depending on the presence or absence of all other edges. This makes the optimization problem non-linear. The number of possible network configurations $|\mathbb{V}|$ is finite. The feasible set could, therefore, theoretically be determined offline and the cost functions calculated, but this is computationally infeasible for realistic networks. The problem is, therefore, effectively a black box or derivative-free optimization problem.

In the following, an evolutionary metaheuristic algorithm is used to solve the *Network Optimization* problem [114].

6.4.2 Solution

Given the set of optimized paths $P = \{\Pi_1, \ldots, \Pi_{|C|}\}$, a two-fold approach is pursued in this work to solve the complete MONO problem.

Pipeline

The pipeline visualized in Fig. 6.2 constitutes a *Path-merging* step and the *Network Optimization*.

1. During *Path-merging*, the single paths are combined into a graph that models the traffic network. The graph's nodes represent intersections and vertiports in the traffic network, whereas the edges represent aerial corridors. The detailed steps are described in the next Section 6.4.2.

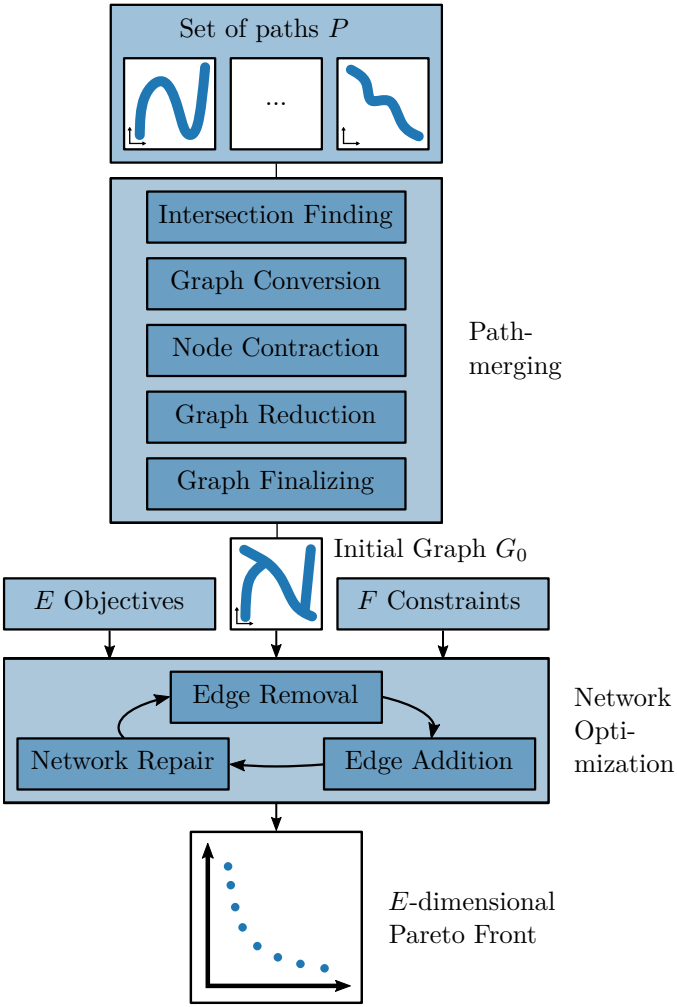


Figure 6.2: Diagram of the proposed network optimization framework. The *Path-merging* stage performs several steps to merge the set of single paths P into an initial graph G_0. Based on this graph, the actual *Network Optimization* stage starts. Here, a multi-objective evolutionary algorithm can remove edges from the graph and add already removed edges again to optimize the graph regarding E objectives. The network repair ensures a proper network structure subject to F constraints. The result is a Pareto set of trade-off networks.

2. By construction, this graph contains redundant connections that might be unnecessary depending on someone's requirements for a traffic network. In the *Network Optimization* step, the traffic network is optimized regarding multiple formulated objectives by removing edges from the graph (or adding already deleted edges). Finally, this results in a Pareto set of traffic networks that satisfy the formulated objectives to varying degrees. A decision maker can then choose a traffic network configuration depending on preference. Details of the optimization routine are given after the following introduction to the *Path-merging* procedure.

Path-merging

The given input set of three-dimensional spatial paths P, as visualized in Fig. 6.3a, consists of connections between all vertiport pairs. Since the paths in P have already been optimized, we see that they often overlap and form a network structure, at least visually (see the red, blue, and brown paths in Fig. 6.3a). From this set of paths, the graph representation G shown in Fig. 6.1 is derived in five steps explained in the following paragraphs. We denote paths in the initial path set $\Pi_i \in P$ with a single index, not to be confused with the path segments Π_{ij} (double index) in the derived graph G.

The paths are assumed to form the centerlines of cylindrical tubes (i.e., corridors) of diameter d_P in which UAVs can move.

Intersection Finding The input is an initial set of paths P as shown in Fig. 6.3a. Where two paths intersect, the later network should be nested. Therefore, the intersection points $\mathbf{p}_{ij} \in \mathbb{D}$ between all paths Π_i and Π_j in P are calculated. Two paths are assumed to intersect when the closest distance between both falls below d_P, meaning that their cylindrical shells intersect. If two paths intersect, the intersection point $\mathbf{p}_{ij}$ is included in both paths as a waypoint $\boldsymbol{\pi}$, and the two intersecting paths are split up into four separate paths as visualized in Fig. 6.3b.

Graph Conversion In the next step, the resulting path set P is transferred into a graph structure by introducing a node n_h with $\mathbf{p}_h = \boldsymbol{\pi}_h$ for every waypoint $\boldsymbol{\pi}_h$. Only one node is created if several paths share the same waypoint (e.g., at a vertiport position). Also, the vertiport flags t_i and crossing flags c_i are set accordingly. An edge e_{ij} is created for every pair of neighboring waypoints $\boldsymbol{\pi}_i$ and $\boldsymbol{\pi}_{i+1}$ for every path in P, whereas

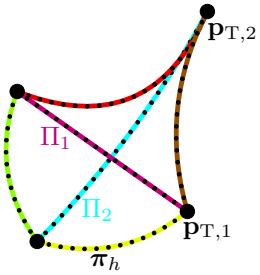
the edge information Π_{ij} and s_{ij} remains empty for now. Figure 6.3c shows the resulting graph.

Node Contraction The graph structure, as shown in Fig. 6.3c, may contain nodes that were assigned to two separate path segments before the graph conversion but whose distance is less than the corridor diameter d_P. Thus, the nodes can be assigned to the same corridor and are, therefore, merged (i.e., contracted). For example, this is visualized by the nodes n_i and n_j in Fig. 6.3c. Two nodes n_i and n_j collapse into a single one n_k, when $||\mathbf{p}_i - \mathbf{p}_j||_2 < d_\mathrm{P}$ applies. If one of the nodes to be contracted is a vertiport node or a crossing node, the position of the new node n_k becomes that of the vertiport or crossing node. Otherwise, $\mathbf{p}_k$ is calculated by the means of

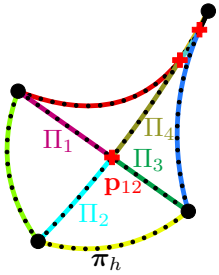
$$\mathbf{p}_k = \bar{X}, \text{ where } X = \{\mathbf{p}_l \,\forall\, n_l \in G[n_i] \cup G[n_j]\},$$

and $G[n]$ denotes the neighborhood of n in G. The exemplary graph after node contraction is visualized in Fig. 6.3d with the contracted nodes marked in blue.

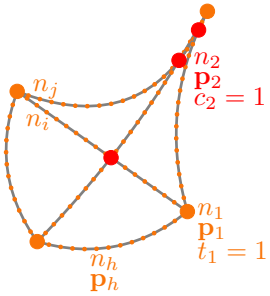
Graph Reduction The graph, as in Fig. 6.3d, now contains sequences of nodes where each inner node n has only two adjacent edges, i.e., degree $d(n) = 2$. Such a sequence is called linear chain $G_\mathrm{C} = \{\mathring{N}_\mathrm{C}, \mathring{E}_\mathrm{C}\}$ with $\mathring{E}_\mathrm{C} = \{(n_{\mathrm{C},1}, n_{\mathrm{C},2}), (n_{\mathrm{C},2}, n_{\mathrm{C},3}), \ldots, (n_{\mathrm{C},k-1}, n_{\mathrm{C},k})\}$ in the following. An exemplary linear chain is visualized with pink nodes in Fig. 6.3d. The optimizer will later remove single, gray visualized edges from the graph. However, removing an edge from a linear chain would result in the unnecessary creation of two dead-ends in the transport network. Instead, it would be meaningful if the optimizer removes a complete linear chain. Therefore, linear chains are resolved into a single new edge $e_{ij} = (n_{\mathrm{C},1}, n_{\mathrm{C},k})$. This means the linear chain's nodes (and thus edges) are deleted. However, it is important to note that the position information of the linear chain's nodes is stored in the new edge's path $\Pi_{ij} = [\mathbf{p}_{\mathrm{C},1}, \mathbf{p}_{\mathrm{C},2}, \ldots, \mathbf{p}_{\mathrm{C},k}]$ of length l_{ij} and is thus still available. Reducing all linear chains in the exemplary graph in Fig. 6.3d ultimately leads to the desired graph structure shown in Fig. 6.1. Its gray-colored edges e_{ij} are subject to the later *Network Optimization*. Its paths' spatial information Π_{ij} (black lines) can be used to retrieve the 'original', i.e., non-reduced, curved network after the *Network Optimization*.



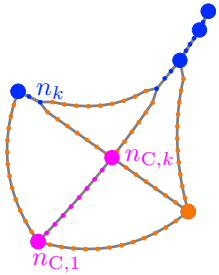
(a) Exemplary colored paths Π_i between all pairs of vertiport positions $\mathbf{p}_{\mathrm{T},k}$ (black circles). The paths build the initial set of paths P, which is input into the *Path-merging* process. A path consists of several waypoints $\boldsymbol{\pi}_h$ shown as black dots.



(b) The same set of paths P after the intersection finding step of the *Path-merging* process. Paths have been cut apart at identified crossing positions $\mathbf{p}_{ij}$ marked as red crosses.



(c) The graph G after the graph conversion step. All round elements are nodes. Every node n_h stores the position information of the former waypoint $\boldsymbol{\pi}_h$ in $\mathbf{p}_h$. Orange circles visualize vertiport nodes ($t_i = 1$), and red circles show crossing nodes ($c_i = 1$).



(d) The graph G after the node contraction step. Nodes that are too close together (e.g., n_i and n_j in Fig. 6.3c) have been contracted and now build a new node (e.g., n_k) visualized in blue. One exemplary linear chain from node $n_{\mathrm{C},1}$ to node $n_{\mathrm{C},k}$ is highlighted in pink.

Figure 6.3: The *Path-merging* process is applied on an exemplary set of paths. Vertiport positions are visualized as black circles, and intersection positions are red crosses. In the set of paths ((a) and (b)), the single paths are colored differently. Their waypoints are black dots. Generally, in graphs ((c) and (d)), vertiport nodes are visualized as orange circles, crossing nodes as red circles. All other nodes are orange dots. The graphs' edges are gray lines.

Graph Finalizing Each path Π_k in the input set of paths P originally had a social attribute s_k. However, the paths were spatially modified in the steps 1) until 4) of the *Path-merging* procedure, so the social attributes s_{ij} of each path segment Π_{ij} must be recomputed. The calculation has already been explained earlier in Section 6.3.3. In the following, we call the final network, i.e., the output of the *Graph-merging*, initial graph $G_0 = \{\mathring{N}_0, \mathring{E}_0\}$ because it serves as input into the *Network Optimization*. The set of vertiport positions can now be expressed as $\mathring{N}_\mathrm{T} = \{\mathbf{p}_i \,\forall\, n_i \in \mathring{N}_0 \,|\, t_i = 1\}$.

Network Optimization

An evolutionary algorithm is applied for the *Network Optimization* stage. The design of its initialization routine, mutation, and repair operator is tailored to the given problem and presented in the following. With the selected binary traffic network encoding (6.7), a crossover operator generates non-feasible solutions (e.g., unconnected networks with dead ends) with a very high probability. These solutions would have to be repaired with high computational effort. As those repaired solutions can also be calculated with the mutation operator, a crossover operation is bypassed, setting its probability of occurrence to $p_\mathrm{x} = 0$.

Initialization The optimization is initialized with five solutions $\{\mathbf{v}_1, \ldots, \mathbf{v}_5\}$ with $\mathbf{v}_i = \mathcal{E}(G_i)$ that are expected to be (near-)optimal for at least one objective:

1. The Steiner tree regarding the length attribute $G_1 = G_{\mathrm{ST,L}}$ obtains optimal values for the maintenance objective function $f_\mathrm{M}(G_1) = 0$.

2. The Steiner tree regarding the social attribute $G_2 = G_{\mathrm{ST,S}}$ is expected to be an attractive initial solution as it contains only socially favored edges while being characterized by small maintenance costs.

3. To obtain the Pareto set's extreme point regarding travel cost (i.e., $f_\mathrm{T} = 0$), a graph is introduced that unites the shortest paths (regarding l) between all vertiport pair combinations C, yielding $G_3 = \{\mathring{N}_3, \mathring{E}_3\}$ with $\mathring{E}_3 = \bigcup_{(n_k, n_l) \in C} R_{G_0, n_k, n_l, l}$.

4. In the same way, the union of all connections' shortest paths through G_0 regarding the social attribute is added to the initial solutions

$G_4 = \{\mathring{N}_4, \mathring{E}_4\}$ with

$$\mathring{E}_4 = \bigcup_{(n_k,n_l)\in C} R_{G_0,n_k,n_l,s}.$$

This results in the extreme point regarding social cost $f_{\mathrm{S}}(G_4) = 0$.

5. Last, another non-dominated solution can be calculated by taking the union of G_3 and G_4 as an initial solution, being $G_5 = \{\mathring{N}_5, \mathring{E}_5\}$ with $\mathring{E}_5 = \mathring{E}_3 \cup \mathring{E}_4$. This solution is characterized by $f_{\mathrm{T}}(G_5) = f_{\mathrm{S}}(G_5) = 0$, but consists of more edges than G_3 and G_4 and is, therefore, worse regarding the maintenance cost.

Mutation The mutation process consists of two parts.

1. Firstly, an individual (i.e., a solution $\mathbf{v}$) is subject to a delete mutation with a probability of $p_{\mathrm{mut,del}}$. The delete mutation removes an edge e_{ij} from $\mathcal{D}(\mathbf{v})$ with a probability of $p_{\mathrm{mut,ind}}$. The probability of deleting an edge is set inversely proportional to the number of edges in the network $p_{\mathrm{mut,ind}} = 1/\sum_{\mathbf{v}} v_k$. This is a reasonable choice, as each additional edge deletion means a larger jump in the search space, potentially overshooting a desired minimum in the objective space.

2. Secondly, an add mutation is applied to an individual with a probability of $p_{\mathrm{mut,add}}$. This mutation randomly draws from the set of connections $(n_k, n_l) \in C$ and then randomly either adds the shortest path $R_{G_0,n_k,n_l,l}$ or the best path regarding the social attribute $R_{G_0,n_k,n_l,s}$ to the graph $\mathcal{D}(\mathbf{v})$.

Repair operator During optimization, the evolutionary algorithm creates solutions that may be improved or made feasible by incorporating domain knowledge. This can be done by defining a repair operator. It repairs infeasible solutions or solutions that must have an obvious (induced from the graph structure) similar solution that Pareto dominates the solution to be repaired.

1. First, dead-ends that emerge in the traffic network during optimization are repaired. Any node n_i in the graph G that is a dead-end (i.e., its degree is $d(n_i) = 1$) but no vertiport ($t_i = 0$) is deleted together with the connected linear chain.

2. Then, solutions that do not satisfy the connectivity constraint (6.5) are repaired by determining the biggest connected component H_0 in terms of number of contained vertiports. Every other connected component H_i is connected to H_0 by choosing a random vertiport node n_i in H_i (or if there is none, any random node) and connecting it via the shortest possible path to the nearest vertiport in H_0. The terms *shortest* and *nearest* apply with a probability of 50% concerning the length or the social attribute of the graph's edges.

3. Finally, solutions that violate the vertiport inclusion constraint (6.6) are repaired. For that to happen, any vertiport that is not part of the current graph $\mathcal{D}(\mathbf{v})$ is again connected to the graph by either the length-related or socially shortest path to the nearest vertiport in $\mathcal{D}(\mathbf{v})$.

6.5 Evaluation

This section describes the experiments conducted to answer the research questions 1) and 2) presented in Section 6.2.4. Therefore, the scenario setup is explained in Section 6.5.1. Then, the different experiments are described in Section 6.5.1. Section 6.5.2 presents and discusses the results of the experiments.

6.5.1 Setup

Scenario

A section of the city of Frankfurt, Germany, visualized in Fig. A.19 in the Appendix A.2, serves as an example for the evaluation. The $|\mathring{N}_\mathrm{T}|$ vertiports, visualized as orange circles, were randomly distributed over the entire area, excluding water areas as possible locations. Any specifications considering the chosen scenario can be drawn from Table 6.1.

Solvers

The metaheuristic evolutionary algorithm NSGA3 [23] is considered one of the state-of-the-art techniques for many-objective optimization problems. The customized mutation and repair operators extend the implementation provided by the `pymoo` framework [115]. The algorithm's parameters are given in Table 6.2.

Table 6.1: Scenario Parameters

	Parameter	Symbol	Value
Scenario	Origin latitude	Lat	50.1002°N
	Origin longitude	Lon	8.666°E
	Scenario dimensions	$x_{\min}$, $x_{\max}$	0 m, 3010 m
		$y_{\min}$, $y_{\max}$	0 m, 3000 m
		$z_{\min}$, $z_{\max}$	0 m, 300 m
	Min. flight height	$z_{\mathrm{F,min}}$	50 m
Envir.	Discr. resolution	δ_x, δ_y, δ_z	10 m, 10 m, 10 m
	Resizing resolution	$\tilde{\delta}_x$, $\tilde{\delta}_y$, $\tilde{\delta}_z$	14 m, 15 m, 10 m
Path	Num. control points	n_{CP}	ANCP
	Basis function degree	d	2
	Parametrization		chordal
	Waypoint res.	$\Delta\boldsymbol{\pi}$	5 m
	Corridor diameter	d_{P}	5 m
Network	Num. vertiports	$\lvert\mathring{N}_{\mathrm{T}}\rvert$	16
	Vertiport pair comb.	$\lvert C\rvert$	120

Table 6.2: Solver parameters

Parameter	Symbol	Value
Add mutation probability	$p_{\mathrm{mut,add}}$	0.5
Delete mutation probability	$p_{\mathrm{mut,del}}$	0.5
Individual delete mutation probability	$p_{\mathrm{mut,ind}}$	$1/\sum_{\mathbf{v}} v_k$
Crossover probability	p_{x}	0
Population size	s_{pop}	100

Experiments

For the same given vertiport locations, three experiments are conducted that are three independent *Path-merging* and *Network Optimization* runs with varying settings. The network optimizations in all three experiments run for $a = 2000$ iterations, but the optimizations converge earlier. The termination criterion from Blank *et al.* [116] is used to determine convergence. Table 6.3 gives an overview of the different runs and the number of iterations until they converge, respectively.

1. In the first experiment, abbreviated **S2** in the following, the initial path set P, which is input into the *Path-merging*, mainly consists of straight-line connections between the vertiports. Figure 6.4a shows a two-dimensional projection of the complete path set after *Path-merging*. The paths were derived by $|C|$ independent energy-optimal path optimizations in the MOPP framework[1]. The curves of some paths are caused by flying around static obstacles (i.e., higher buildings).
   During the *Network Optimization* step, the aerial traffic networks are only optimized regarding the two economic objectives f_{M} (maintenance cost) and f_{T} (travel cost).

2. In the second experiment **S3**, the *Path-merging* is conducted on the same set of straight paths. However, the *Network Optimization* step includes the social criterium f_{S} as a third objective function. The additional dimension in the objective space results in an expected increase in convergence time, which is 15 % in this case.

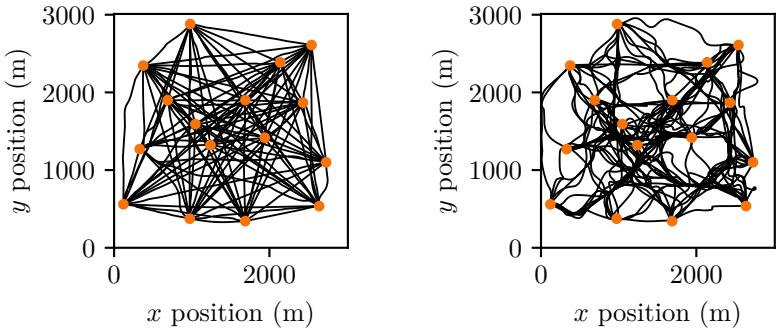
3. In the third experiment **SC3**, the *Path-merging* is initialized with $2|C|$ paths, namely the $|C|$ straight paths as used before and additional $|C|$ paths that had been optimized regarding the social cost weight[2] by the MOPP framework introduced earlier in this thesis in Chapter 5. To differentiate these pre-optimized paths from straight paths, we refer to them as *social* or *curved* paths. A two-dimensional projection of the set of curved paths after *Path-merging* is visualized in Fig. 6.4b. Due to the larger search space compared to the experiment **S3**, the optimization takes 33 % longer to converge.

[1] Each path optimization resulted in a Pareto set of paths from which the energy extreme point was selected.

[2] Each path optimization resulted in a Pareto set of paths from which the noise extreme point was selected.

Table 6.3: Overview of the experiments carried out and the duration until convergence

Exp.	Initial path set	E	Objectives	Iterations until Convergence
S2	**S**traight	**2**	f_M, f_T	705
S3	**S**traight	**3**	f_M, f_T, f_S	809
SC3	**S**traight & **C**urved	**3**	f_M, f_T, f_S	1074



(a) The projected straight-path network obtained after applying the *Path-merging* process to the set of paths that connects all vertiports with straight lines while circumventing static obstacles.

(b) The projected curved-path network obtained after applying the *Path-merging* process to the set of paths that was pre-optimized regarding social cost weights based on the city map shown in Fig. A.19.

Figure 6.4: The two types of networks that were used to initialize the *Network Optimization*

6.5.2 Results

Comparing the results from the three experiments, we now want to obtain answers to the research questions posed in Section 6.2.4. For a fair cross-experiment comparison, the unnormalized objective functions $f_{\mathrm{M,u}}$, $f_{\mathrm{T,u}}$, and $f_{\mathrm{S,u}}$ must be used. These correspond to the objective functions (6.1), (6.3), and (6.4) without the normalization constants. All three of these unnormalized cost functions are given in meters since $f_{\mathrm{M,u}}$ is the sum of all path lengths, $f_{\mathrm{T,u}}$ adds up the path lengths of the shortest paths between all vertiport pairs, and $f_{\mathrm{S,u}}$ sums edge attributes s_{ij} that originate from a path integral over a unitless grid map.

Economic vs. socio-economic network optimization

Observation First, we want to investigate the effects of including the social objective function f_{S} in the multi-objective network optimization problem. This social factor is often neglected in related work. What effect does this have on the Pareto sets of traffic networks that were obtained? We compare the traffic networks obtained in **S2** with those in **S3**. As this means comparing a two-dimensional Pareto set with a three-dimensional Pareto set, the solutions of **S2** are evaluated a posteriori regarding $f_{\mathrm{S,u}}$. The obtained three-dimensional objective space for **S2** and **S3** is visualized as two projections in Fig. 6.5. The upper figure shows the objectives that were used in both experiments. We qualitatively observe that for maintenance costs of $f_{\mathrm{M,u}} < 0.3 \times 10^5$ the solutions of **S2** dominate those of **S3** while otherwise the solutions lie on top of each other. Looking at the second projection in lower Fig. 6.5, we see a large gap between the solutions obtained by both experiments. The **S3** approach has found better solutions regarding the social costs $f_{\mathrm{S,u}}$ than the **S2** approach. We want to quantify the differences between the visualized solutions from **S2** (green) and from **S3** (blue). Therefore, we compare every network obtained in **S2** with every network calculated in **S3** regarding their respective travel and social costs by calculating their relative differences (**S2** is the baseline). The resulting boxplot is shown in Fig. 6.6a. On average (dashed line), the approach **S3**, which includes the social objective, produces solutions that are 13 % worse than those of **S2** regarding travel costs, and 27 % better regarding social costs. In Fig. 6.6a, the left (green) box range indicates that three-quarters of all networks from **S3** are only at most 16 % more expensive, one-fourth of the networks are even less expensive than the networks from **S2**. The right (blue) box of Fig. 6.6a shows that 75 % of the

networks calculated in **S3** are at least 20 % more social, and a quarter of the networks is even at least 42 % more social than the networks from **S2**.

Discussion What are the benefits and costs of integrating social criteria into network optimization? First, we get solutions that are socially more acceptable. On average, the obtained networks achieve a 27 % reduction in noise. However, we also note a 13 % increase in monetary costs, assuming that time and money are proportional. By integrating a further objective function, differently shaped transport networks are created, which, on average, establish longer flight corridors between two vertiports than the shortest possible connection. On the one hand, this causes longer travel times for the network users. On the other hand, these longer corridors are then spanned over areas of the city where they are perceived as less annoying by the residents living underneath.
Choosing a solution from the Pareto set is a crucial decision that can significantly impact social attractiveness and monetary costs. The final decision ultimately lies in the hands of the responsible authorities (i.e., the decision maker). Often, decision makers select the knee point (black-colored markers in Fig. 6.5) as the preferred solution from the Pareto set. When we compare the **S3** knee point with the **S2** knee point (baseline), the travel costs get 6.6 % worse, while the social attractiveness increases by 26.3 %.

Economic vs. socio-economic pre-optimization

Observation Secondly, we want to investigate the influence on network optimization if we deviate from the restriction of straight connections, as is often done in the literature. Initializing with straight connections is an intuitive way to find good economic solutions. However, this strategy tends to neglect the social aspect. What happens when we calculate social paths beforehand and make them available during *Path-merging*? We will compare the resulting networks of experiment **S3** (three-objective optimization on the set of straight paths) with those of **SC3** (three-objective optimization on the set of straight and curved paths) to find an answer. The Pareto set projections of both experiments are visualized in Fig. 6.5 in different colors for **S3** (blue) and **SC3** (orange). Qualitatively, the lower figure already shows the large difference in the social costs of the networks obtained by the two experiments. The quantitative comparison between the networks generated by both approaches results in the boxplot

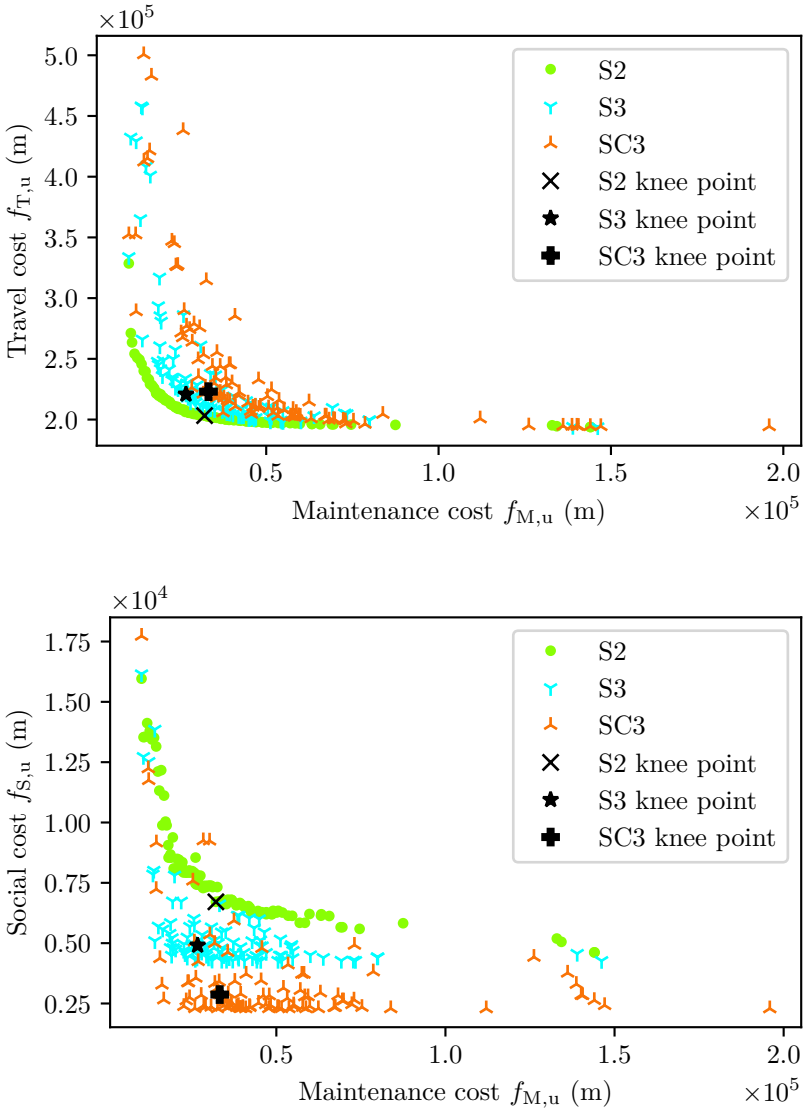
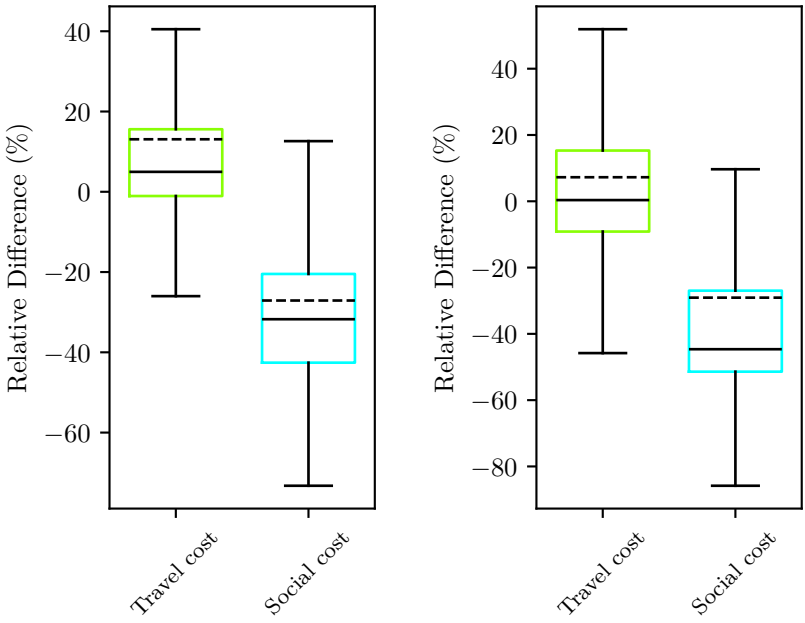


Figure 6.5: The projections of the three Pareto sets obtained by the experiments **S2** (without social objective), **S3** (with social objective), and **SC3** (with social objective and including socially pre-optimized paths).



(a) Results for the comparison between the traffic networks from the optimization run **S2** (two objectives) without the social objective and the traffic networks from the run **S3** (three objectives) including the social objective.

(b) Results for the comparison between the traffic networks from the optimization run **S3** without pre-optimized paths (straight-path network) and the traffic networks from the run **SC3** with pre-optimized paths (straight and curved-path network) paths.

Figure 6.6: Results for the two inter-experiment comparisons **S2** vs. **S3**, and **S3** vs. **SC3**. The traffic networks from the respective first experiment are compared to those from the second experiment in terms of travel and social costs. The relative differences are visualized using boxplots, with the respective first experiment as the baseline. The box spans from the data's first quartile q_1 to the third quartile q_3. The black straight line within the box represents the median, and the dashed line represents the data's mean. The whiskers enclose all data points between the lower bound $2.5q_1 - 1.5q_3$ and the upper bound $2.5q_3 - 1.5q_1$. Since we are comparing the results of a minimization problem, 'the lower, the better' applies here.

in Fig. 6.6b. The experiment **S3** is the baseline for calculating the relative differences between all traffic networks of **S3** and all traffic networks of **SC3**. The results indicate that on average (dashed line), the solutions found on the combined straight-and-curved-path network are 7 % worse regarding travel cost and 29 % better regarding social cost than the solutions found within the only straight-path network. On the one hand, the range of the left (green) box in Fig. 6.6b visualizes that 75 % of all networks from **SC3** are only at most 16 % more expensive than the networks from **S3**. On the other hand, the right (blue) box of Fig. 6.6b shows that three-quarters of the networks calculated in **SC3** are at least 27 % more social, while 25 % are even at least 51 % more social than the networks obtained in **S3**.

Discussion Extra effort is necessary to optimize a traffic network following approach **SC3**. This includes the a priori optimization of paths between all vertiports regarding social costs. In addition, the network optimization itself requires more time to converge (Table 6.3). In terms of social acceptability, the extra effort is worth it, as the increase in social acceptance is greater than the increase in monetary costs. The difference becomes evident when comparing the knee point solutions from both approaches. The knee point solution of approach **SC3** requires 1.7 % more travel time but is 47.7 % better regarding the social criterion than the knee point solution of **S3** (baseline).
In order to provide a comprehensive analysis, the results of the comparison between **S2** and **SC3** are also given: On average, the solutions of a three-criteria optimization with an initial set of both straight and curved paths are 17 % more expensive but 51 % more socially beneficial than the networks obtained with a two-criteria optimization on a set of only straight paths.

Visualization and further examination of SC3

Figure 6.7 shows the visualization of some selected solutions obtained from approach **SC3**. Only solutions $\{\mathbf{v} \mid f_{\mathrm{M}}(\mathbf{v}) < 0.05 \cdot f_{\mathrm{M}}(\mathbf{v}_5)\}$ are considered whose maintenance costs are less than 5 % of the initial solution $\mathbf{v}_5$, otherwise the high number of edges would make it difficult to recognize anything in the networks plotted on top of each other. The solution with the least maintenance costs is visualized in violet, the solution with the best travel costs in black, the solution with the best social costs in green, and the knee point as a trade-off solution in blue. In addition, a video file [117]

visualizing the respective networks (same color coding) is provided in the digital supplementary material of the corresponding publication [N4]. Ta-

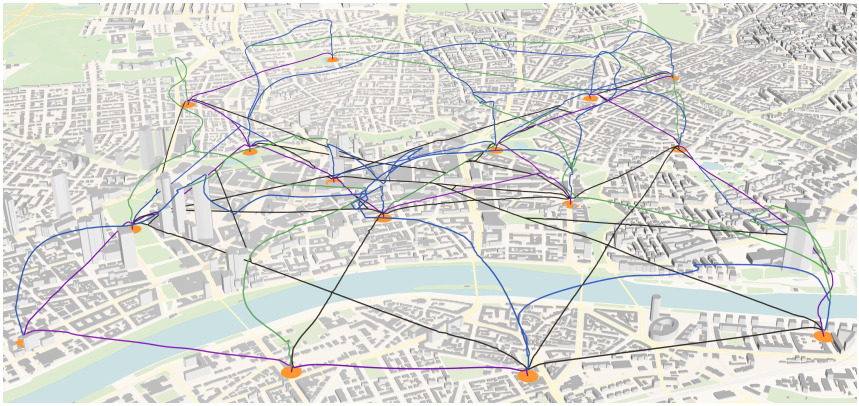


Figure 6.7: Visualization [118] of different aerial traffic networks obtained by selecting differently preferenced solutions from the Pareto set of solutions obtained by **SC3**. The network with the least maintenance costs is displayed in violet, the one with the smallest travel costs in black, the network with the largest social acceptability is shown in green, and the knee point (trade-off between all three objectives) network is visualized in blue.

ble 6.4 illustrates how the (near-)optimal solutions for one objective may not perform optimally regarding the other objectives. It is shown how the traffic networks specified in each row deviate from the best-obtained solution determined by the column. Moreover, the observed correlation between economic investment and social benefit is displayed. For example, we look at the least maintenance cost network, visualized in violet in Fig. 6.7, which equals the Steiner tree $G_{\mathrm{ST,L}}$. It connects all 16 vertiports by a set of edges of minimal maintenance costs (i.e., edge length). Consequently, between any two vertiports, only one route is generally not the shortest or most social one available in the initial graph G_0. Looking at Table 6.4, we thus observe that its objective function value regarding travel time is 56 % worse than this of the least travel cost network, and its social cost value is 672 % worse than that of the best social cost network. Based on this solution, increasing the allowed maintenance costs by 5 % results in an 8 % improvement in social acceptance. Thus, if a decision maker accepts additional maintenance costs of 15 %, the resulting network is 24 % more social than the least maintenance cost network.

Table 6.4: Deviation from the best obtained objective function value and the relation between additional economic effort and resulting social benefit

Solution	$\mathbf{f}_{\mathrm{M,u}}$	$\mathbf{f}_{\mathrm{T,u}}$	$\mathbf{f}_{\mathrm{S,u}}$
Least maintenance cost network	0 %	56 %	672 %
	+5 %		-8 %
	+10 %		-16 %
	+15 %		-24 %
	+20 %		-32 %
Least travel cost network	180 %	0 %	303 %
		+5 %	-22 %
		+10 %	-44 %
		+15 %	-66 %
		+20 %	-74 %
Best social cost network	155 %	94 %	0 %
Knee point solution	123 %	53 %	4 %

The best travel cost network, depicted in black in Fig. 6.7, consists mainly of straight edges that connect the vertiports directly, allowing for the quickest possible travel between the hubs. However, as these fast lanes are located near the ground, this comes at the expense of social acceptance, which is 303 % worse than the best social cost network. If a decision maker were to cause network users to have 10 % higher travel costs, the resulting network would be 44 % quieter than the network with the lowest travel costs.

The best social cost network, visualized in green in Fig. 6.7, mainly includes high-altitude transportation corridors that avoid areas of high social aversion. Due to the longer corridors and the detours taken, both the maintenance and travel costs increase by 155 % and 94 % respectively.

The knee point network, shown in blue in Fig. 6.7, is a possible solution that realizes a balance between the competing objectives. The twelfth row in Table 6.4 displays the relative differences between the knee point solution and the respective best solutions. The knee point traffic network combines straight direct connections and curved corridors at higher altitudes, thus achieving a compromise between established objective functions.

6.6 Summary & New Results

This chapter dealt with developing a framework to find Pareto-optimal three-dimensional urban air traffic networks given a set of vertiport positions and a set of paths connecting them. The assumed given set of paths can, for example, be generated by applying the path planning framework as introduced in the previous Chapter 5.

The literature review at the beginning of this chapter revealed that social aspects have been understudied in related Urban Air Mobility (UAM) infrastructure concepts [14]. Therefore, the focus was on investigating the feasibility and impact of integrating a social perspective into the network design. In the problem formulation of the network optimization problem, particular emphasis was placed on integrating a social objective function, i.e., minimizing the noise that residents are exposed to due to the aerial traffic network. In addition, two further economic objective functions were presented, which evaluate the network from the perspective of traffic participants and network operators. Then, a solution framework was proposed that consists of two parts. The *Path-merging* step generates a three-dimensional graph representation from a set of individual paths, and a *Network Optimization* step optimizes the graph (i.e., the traffic network) by removing edges. This results in a Pareto set of traffic networks that minimize the objective functions to varying degrees. The subsequent evaluation section's main focus was examining the effects of 1) giving a set of socially pre-optimized paths into the framework and 2) integrating a social objective function into the optimization problem. A study based on a real-world scenario in Frankfurt, Germany, has shown that both approaches increase the social attractiveness of a transportation network more than the economic costs. To summarize, this chapter described the following contributions:

- The proposal of an optimization framework for three-dimensional spatial networks based on the conjunction of pre-optimized paths.
- The capability to use non-straight spatial paths as network edges.
- The introduction of a social criterion in network optimization.
- The finding that it is advantageous to integrate social objective functions into Urban Air Mobility (UAM) infrastructure optimization as the gains in social acceptability have shown to be higher than the average increases in economic expenses.

7 Conclusions

7.1 Summary

This work was motivated by the need to develop optimal air transportation infrastructure for urban air mobility to address the challenges of last-mile delivery in congested cities. Due to the involvement of various stakeholders in the city, the requirement to balance objectives like safety, economic interests, and social factors such as noise is central.

However, the study of related work showed that the social point of view, i.e., the human-centered perspective, needs to catch up to economic perspectives. Therefore, the questions in this thesis were always examined in the light of social compatibility.

From a research perspective, several optimization problems need to be addressed on the way to an urban air transportation network [119] including the facility (i.e., landing hub) location problem, optimal infrastructure (i.e., path and corridor network) planning, the seamless integration with existing ground transportation, routing and scheduling, and real-time traffic management and collision avoidance.

In this thesis, two successive questions within the infrastructure optimization were answered. The first problem was to find a Pareto-optimal path between two given vertiport locations in the city. The ultimate problem was to obtain a Pareto-optimal urban aerial corridor network, given several paths in the city.

To sum up, this thesis found its niche in Multi-objective Path Planning (MOPP) and Multi-objective Traffic Network Optimization (MONO), both with the inclusion of social criteria. The answers to the subordinate research questions of both domains initially presented in the outline of this dissertation in Section 1.3 can be summarized as follows.

First in Chapter 3, a simplified version of the MOPP problem was solved.

Answer to Question Q1

The two-objective path planning problem was solved in the two-dimensional operation space using a combination of multiple Dijkstra shortest-path searches and a multi-objective evolutionary algorithm. The drawbacks of each approach were compensated by the other respectively: The evolutionary search, utilizing a smooth Non-uniform Rational B-Spline (NURBS) curve representation in the continuous operation space, was initialized with Dijkstra solutions. Although the Dijkstra paths were calculated on a grid, i.e., a discretized, thus abstracted representation of the environment, they still served the evolutionary algorithm as a reasonable initial basis for further search. In addition, the Dijkstra searches could adaptively and problem-dependently adjust an important hyperparameter of the continuous path representation - the number of control points. Moreover, the evolutionary optimizer corrected approximation errors of the Dijkstra searches, which inevitably arose due to the grid discretization of large operation spaces.
Altogether, in a statistical analysis the presented hybrid framework was shown to be an efficient multi-objective path finding approach, capable of sampling a better Pareto set in less computation time than comparable approaches.

Then, in Chapter 4, the problem was extended by a third dimension for the UAV operation space.

Answer to Question Q2

Increasing the dimension of the planning space is a challenge for both Dijkstra searches and the evolutionary optimizer. The practical feasibility of the Dijkstra algorithm depends on the number of nodes and edges of the environment graph. It increases considerably in three dimensions if the resolution remains the same. Therefore, a coarser resolution for the grid discretization of the planning space had to be chosen, and thus, more considerable approximation errors had to be accepted. These larger errors provided an even stronger argument for using the downstream evolutionary algorithm that could correct the approximation errors of the Dijkstra searches.
Additionally, the dimension of the optimization vector in the evolutionary search also increased by 50 %, which is why it is all the more important to set the number of control points as low as possible but as high as necessary. This is where the developed Adaptive Number of Control Points (ANCP) feature was really at its best, automatically adapting the optimization problem's dimension to the optimization landscape's complexity.

Moreover, in Chapter 4, also constraints were introduced into the MOPP problem.

Answer to Question Q3

Constraints were introduced as soft constraints into the path planning problem. This integration caused a severe drawback in the framework at hand. The initial Dijkstra solutions were sorted out in the evolutionary search if they did not fulfill all constraints from the beginning. In order to prevent this unintended behavior, the idea of evolutionary niching was successfully integrated into the MOPP framework. Its functionality was statistically proven.

Chapter 5 examined the challenges of including more objective functions into the MOPP problem.

Answer to Question Q4

The search for Pareto-optimal solutions becomes more complex with more objective functions. An increased number of Dijkstra searches on weighted and aggregated grid graphs was used to compensate for the increased complexity. In addition, different ways of normalizing grid maps to be aggregated were investigated, and the effects of an increasing number of Dijkstra searches on the problem were considered.
One of the introduced objective functions was a social criterion to minimize noise pollution for the population. Based on real-world data, it was investigated how energy-consuming, risky, and signal-jamming the obtained social, i.e., low-noise, paths are. A good compromise was found in the knee points of the calculated Pareto sets.

Chapter 6 dealt with the derivation of a network representation from the output of the MOPP output and the subsequent optimization of transportation networks. The corresponding questions posed in Section 1.3 can be answered as follows.

Answer to Question Q5

An initial network could be calculated by applying a geometric merging procedure to the set of paths. In a subsequent multi-criteria evolutionary optimization process, this network was then thinned out, i.e., corridors were removed to find a Pareto set of aerial corridor networks for three opposing parties. The objectives were designed to juxtapose the different economic perspectives of network users and network providers, and the social perspectives of the residents.

A literature review revealed that optimizing three-dimensional transportation networks considering social aspects fills a research gap. Therefore, the question of the influence of social criteria on the economic evaluation of transport networks was answered:

Answer to Question Q6

A study based on actual data showed that the consideration of social criteria in optimizing urban aerial corridor networks results in networks that are 51 % quieter than conventional aerial corridor networks. At the same time, these more social networks are only 17 % more expensive.
Thus, it seems worthwhile to include a social criterion in the optimization as the social acceptance increases more than the monetary costs.

At this point, it is worth recalling Fig. 1.1. The figure visualized the fusion of a discrete path representation to leverage planning tasks in *occupied* environments and a continuous path representation allowing for navigation in *unstructured* environments into a combined transportation network. The development and investigation of software frameworks for creating and optimizing such an *aerial corridor network* considering different objectives have emerged as the ultimate goal of this work. This thesis has shown how path planning and network optimization can be brought together to solve the new challenges posed by the era of urban air mobility.

Given the vertiport locations in the city and a set of arbitrary cost functions and boundary conditions, we are now able to optimize the structure of an air corridor network, addressing and optimizing both the course of the curved corridors themselves and their presence in the overall network.

That leaves one to wonder: What comes next?

7.2 Outlook

Looking at optimization techniques from a methodological point of view, all efforts that either increase the quality of a solution or reduce the consumption of computational resources are worthwhile.

In the MOPP framework, rethinking the environment representation could lead to potential improvements. So far, a regular grid structure has been chosen for the Dijkstra search. An adaptive grid size [120] could help to keep the discretization error small and make the Dijkstra computation faster.

Learning as much as possible about a problem before optimizing makes sense. This potential knowledge can be integrated into the optimization algorithm to obtain better solutions more efficiently. So far, we have used

domain knowledge in the optimization, but only in the upstream optimization step (Dijkstra), which has always been separated from the metaheuristic optimization. It would be interesting to explore the benefits of breaking down this separation. It could be done by integrating a local optimization into the evolutionary algorithm if information about gradients is available in the grid maps. Or the integration of the Dijkstra algorithm into the evolutionary optimization cycle, e.g., as a repair operator. To what extent does this mixture of continuous and discrete representation make sense? This remains to be investigated.

From a more strategic point of view, one can ask whether the pure separation of the operational space into free space and obstacle space makes sense. In reality, regulations often prohibit flying in certain free space areas. As long as these can be modeled as static no-fly zones, the MOPP framework can handle them. However, let us talk about temporal restrictions (e.g., differences in day and night flights) or legal restrictions for particular agents (e.g., different flight clearances for different flight licenses). We enter the realm of flight operations and mission design. Within urban air mobility, these are adjacent problems to the problems discussed in this thesis, and it makes sense to transport information across their interfaces. For example, information about legal or time restrictions could be used to drastically reduce the planning space for the path and network optimization problems.

When talking about dynamic, i.e., time-dependent optimization, the following methodological extension of the MONO framework could be interesting: So far, the initial set of paths used for the path-merging and network optimization were obtained by statically selecting solutions from the Pareto set of paths obtained by several MOPP runs with different start and endpoints. Changes in flight operations may now make it necessary to select different paths for network optimization. In this case, a completely new network optimization process must start from scratch. However, it might be interesting to identify a geometric relationship between the old path set and the derived old network, e.g., by graph morphing. This morph information and the coherence between the old path set and a new path set could be used to generate a new network more efficiently or provide a good initial solution for the new network optimization.

A Appendix

A.1 Literature Review: Multi-objective Path Planning for UAVs

In the following two tables, literature from the field of multi-objective and many-objective path planning for UAVs is classified into six defined categories. The features that are also - among others - integrated into the Multi-objective Path Planning (MOPP) framework developed in this thesis are marked in blue.

Table A.1: Part I: Classification of related work in the field of MOPP problems for UAVs

Category	Feature	Publications
C1: UAV environment	Rural terrain	[50], [121]–[125]
	Urban	[51], [53], [126]–[128]
C2: Spatial dimension	2D	[53], [126], [127], [129]
	3D	[50], [52], [121]–[125], [128], [130]
C3: Path representation	Grid-based (i.e., discrete)	[52], [53], [126], [129]–[134]
	Line segments (i.e., continuous)	[51], [121], [122], [125], [127], [128], [135]–[137]
	Polynomial	[123]
	Spline (e.g., NURBS)	[50], [123], [124], [138]–[141]
C4: Objectives & constraints	Path length	[50]–[52], [121]–[125], [127] [128], [130]–[132], [134], [136]–[141]
	Safety, Risk	[50]–[53], [122], [124]–[127], [129], [130], [139]
	Danger zone clearance	[122], [131]–[133], [136]–[138], [140], [141]
	Path feasibility (e.g. altitude, smoothness, curvature)	[122], [123], [132], [137], [140], [141]
	Vision & Visibility	[123], [138]
	Energy cost	[52], [122], [126], [130], [134], [135], [137]
	Travel time	[53], [123], [126], [129], [133], [135]
	Turning angle	[50], [121], [125], [128]
	Flight height	[121]–[125], [128]

Table A.2: Part II: Classification of related work in the field of MOPP problems for UAVs

Category	Feature	Publications
C5: Multi-objective handling	Objective selection and SOO	[123]
	Cascaded SOO	[140]
	Weighted aggregation	[50], [122], [124]–[129], [133], [141]
	Pareto-based	[51]–[53], [121], [130]–[136], [138], [139]
C6: Optimization algorithm	ACO	[126]
	PSO	[121], [125], [136], [137]
	MILP	[135]
	FMM	[53]
	Gradient descent	[53]
	Graph-based (SOO)(e.g., Dijkstra [42], Bellman-Ford [142], [143], A* [43])	[123], [133], [140]
	Graph-based (MOO) (i.e., NAMOA* [144])	[134]
	CFO	[128]
	SOO EA (e.g., GA)	[50], [122], [124], [126], [141]
	MOO EA (e.g., NSGA2)	[51], [52], [127], [130]–[132], [138], [139]
	RRT	[129]

A.2 City Maps and Grid Maps

Darmstadt

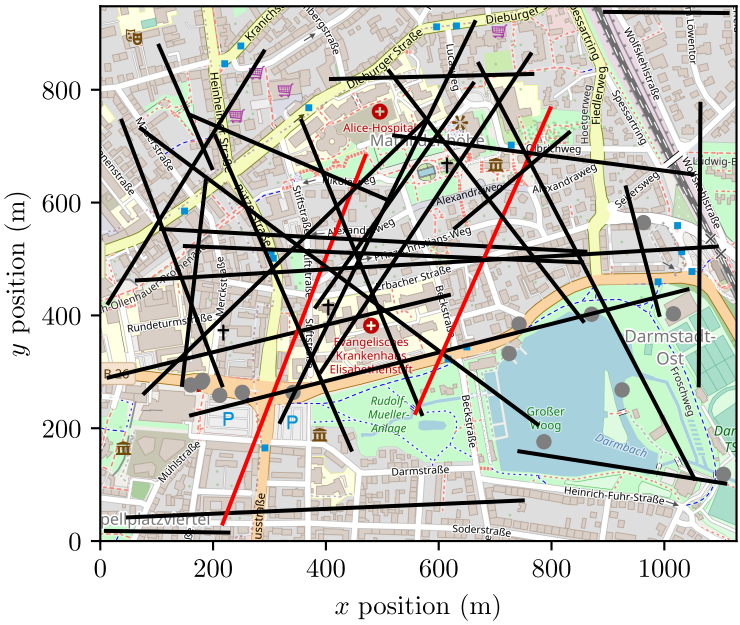


Figure A.1: The OpenStreetMap (OSM) map of the city of Darmstadt, Germany [145] that was used to derive the grid maps given in the following. The lines indicate the start and endpoints of the paths (i.e., scenarios) that were used for the evaluation of the MOPP framework in Section 3.4. Especially for the paths marked in red, Pareto sets of different optimizers and 2D visualizations of the optimized paths are presented in Section 3.4.2.

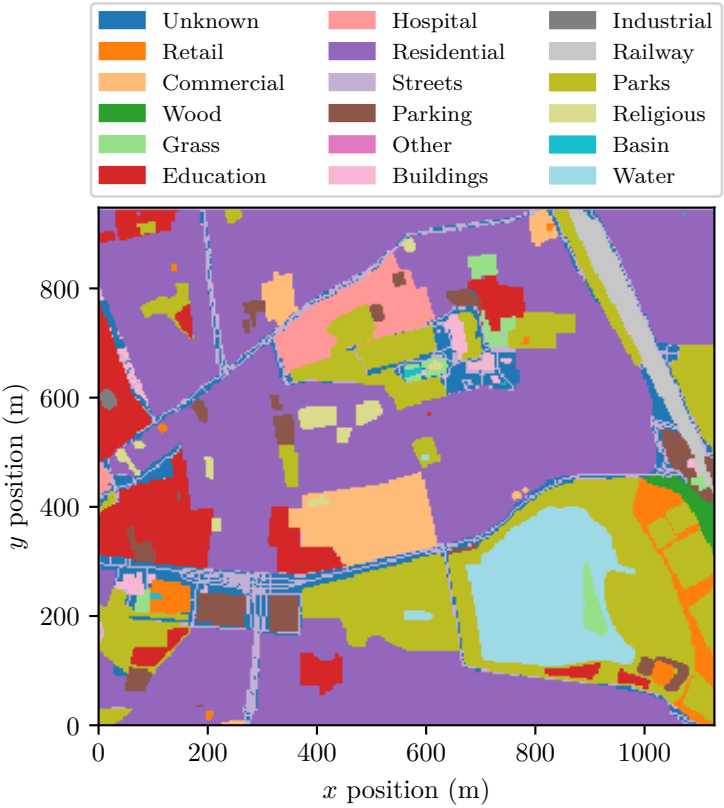


Figure A.2: A visualization of the semantic information extracted from OSM for the Darmstadt map section. This serves as the basis for deriving the grid maps of different objectives given in the following.

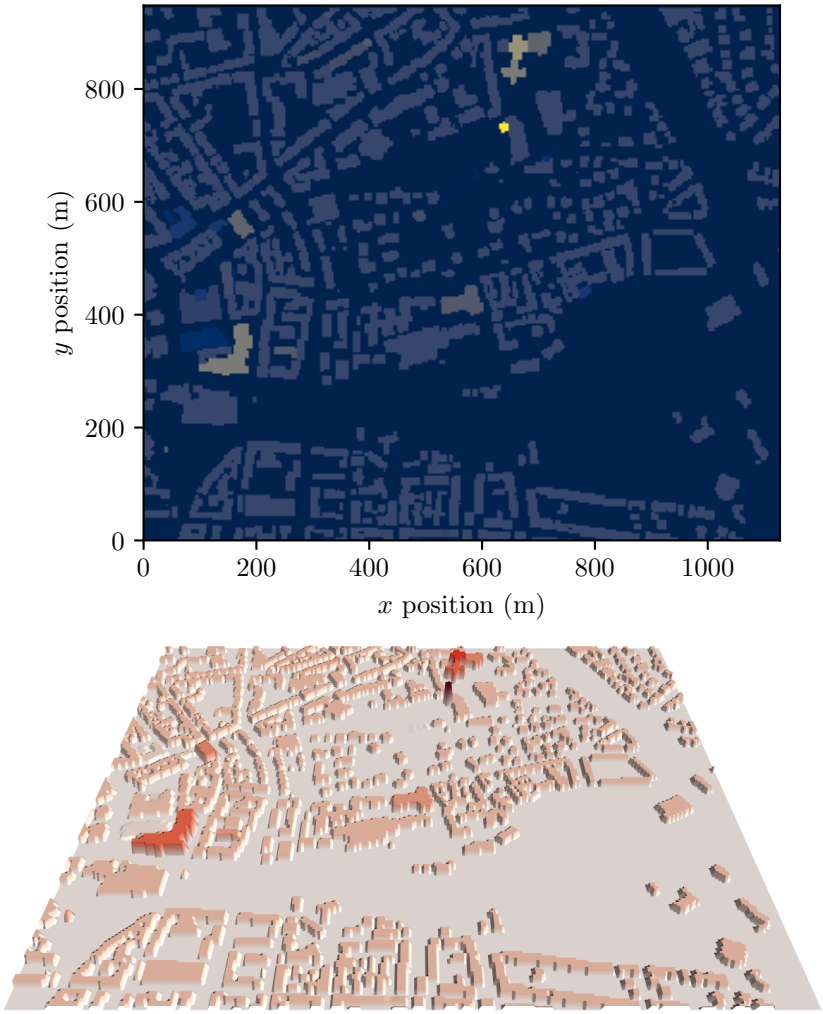


Figure A.3: Height grid map visualization of the Darmstadt map extract derived from OSM data. It contains information on the position of buildings, which is used to derive the risk grid map.

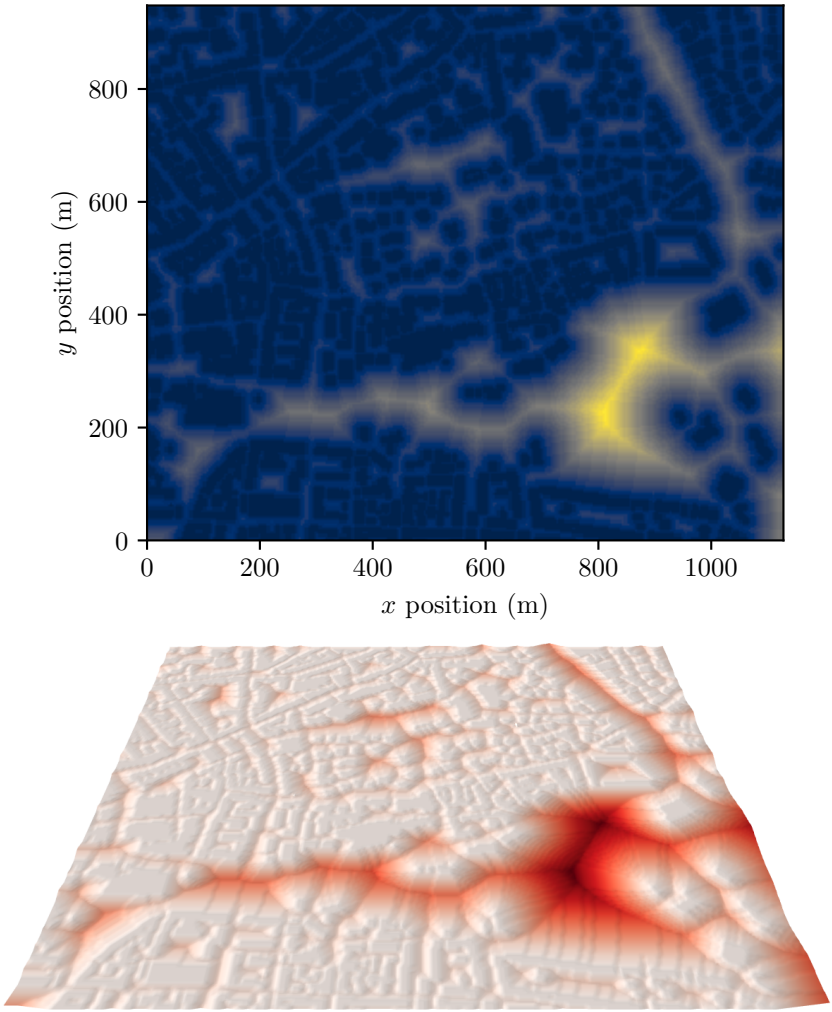


Figure A.4: Risk grid map that was generated with underlying OSM data. According to this risk model, grid cells visualized with yellow/red colors indicate a higher risk, i.e., greater damage, in case of a failure if the particular grid cell belongs to the path. On the contrary, blue/white colored cells indicate a lower risk for residents.

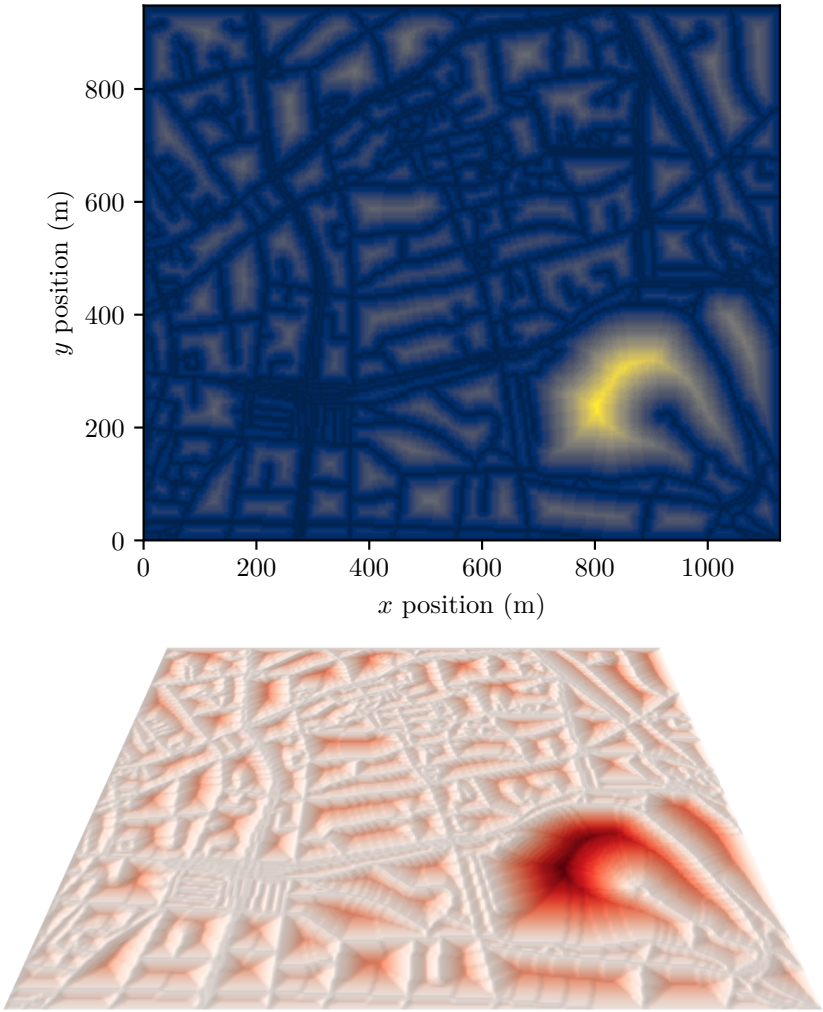


Figure A.5: Noise grid map that was generated with underlying OSM data. According to this noise model, grid cells visualized with yellow/red colors indicate a higher noise immission on humans if a planned path goes through the particular grid cell. On the contrary, blue/white colored cells indicate a lower noise immission on the residents.

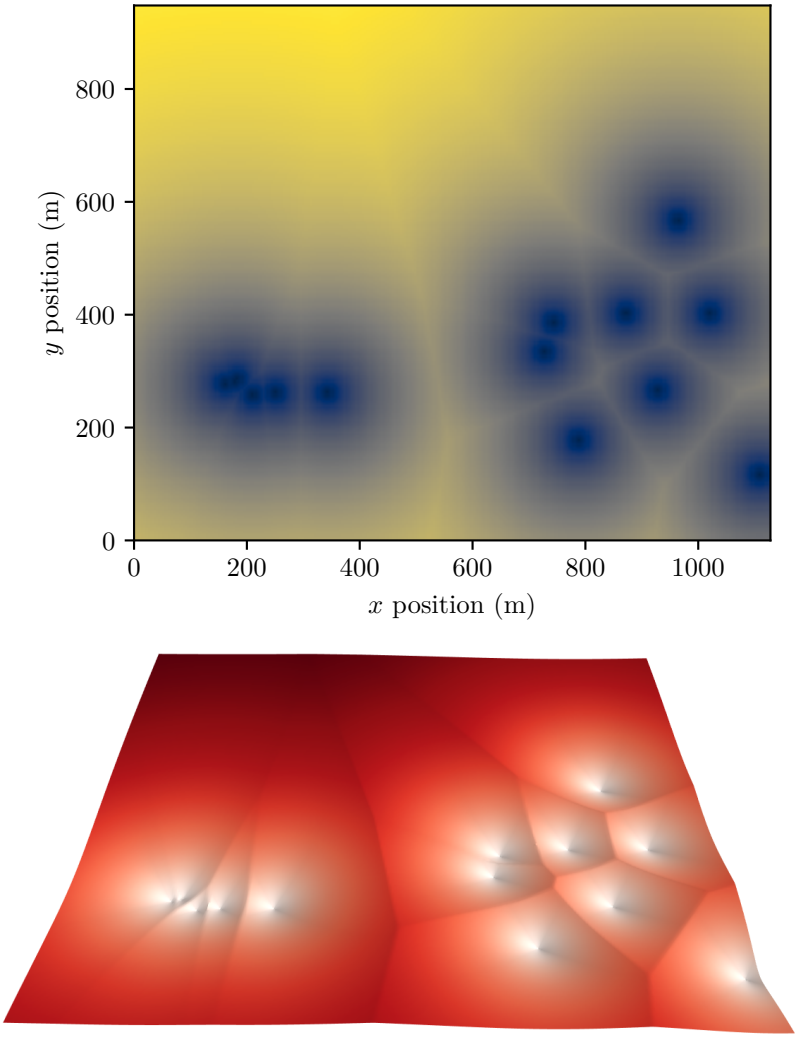


Figure A.6: The radio disturbance information of the Darmstadt map at height $z = z_{\mathrm{R}}$ that was derived from OpenCelliD (OCID) [79] data. Blue/white areas indicate low radio disturbance values around radio masts, and yellow/red areas indicate high radio disturbance values.

New York

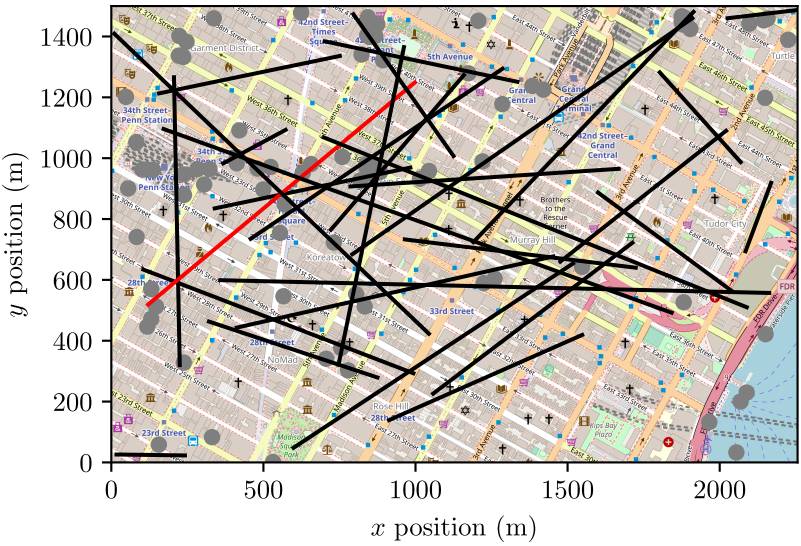


Figure A.7: The OpenStreetMap (OSM) map extract of New York, US [146] that was used to derive the grid maps given in the following. The lines indicate the start and endpoints of the paths (i.e., scenarios) that were used for the evaluation of the MOPP framework in Section 4.4. For the point configuration indicated by the red line, three-dimensional plots of the optimized paths are given in Section 4.4.2. The gray circles denote the positions of radio masts determined by OpenCelliD (OCID) data.

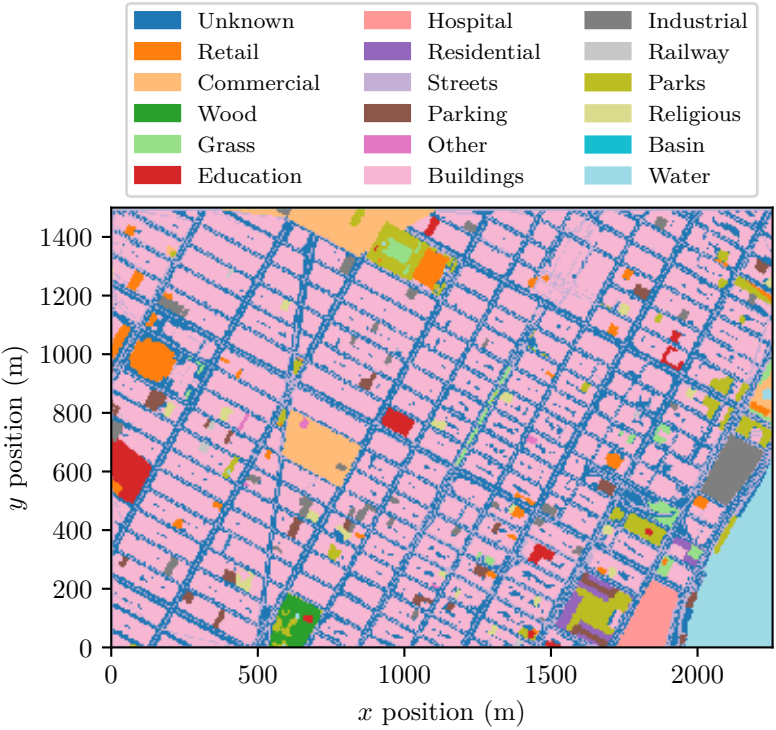


Figure A.8: A visualization of the semantic information extracted from OSM for the New York map section. This serves as the basis for deriving the grid maps of different objectives given in the following.

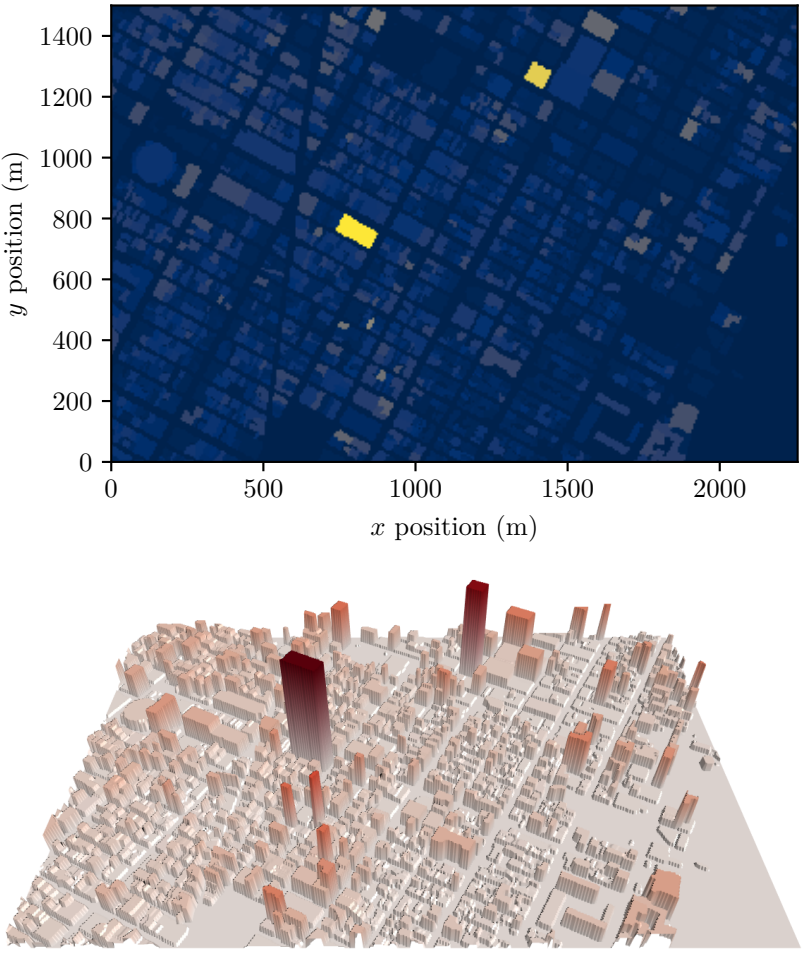


Figure A.9: Height grid map visualization of the New York map extract derived from OSM data. It contains information on the position of buildings used to derive the risk grid map and the building heights used to calculate the obstacle collision avoidance constraint.

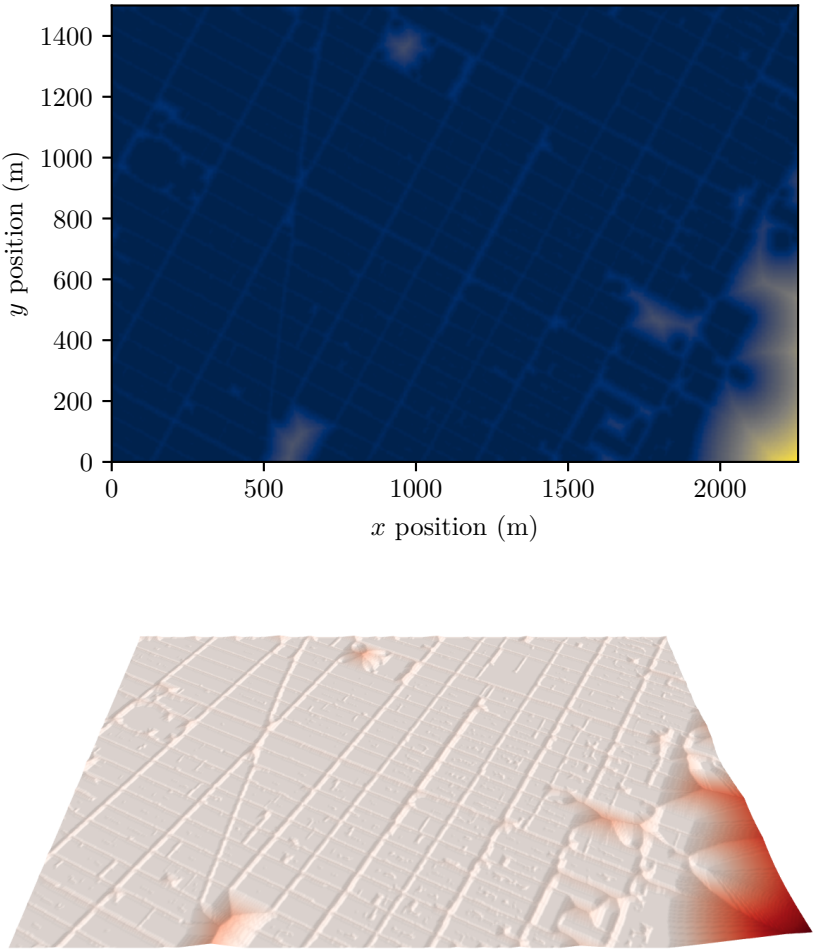


Figure A.10: The risk information of the New York map at height $z = 0\,\mathrm{m}$ that was derived from OSM data. Blue/white areas indicate low-risk values over normal buildings and water surfaces, light-blue/gray areas indicate medium-risk areas over streets, and yellow/red areas indicate high-risk values over educational, medical, railway, military, and worship buildings.

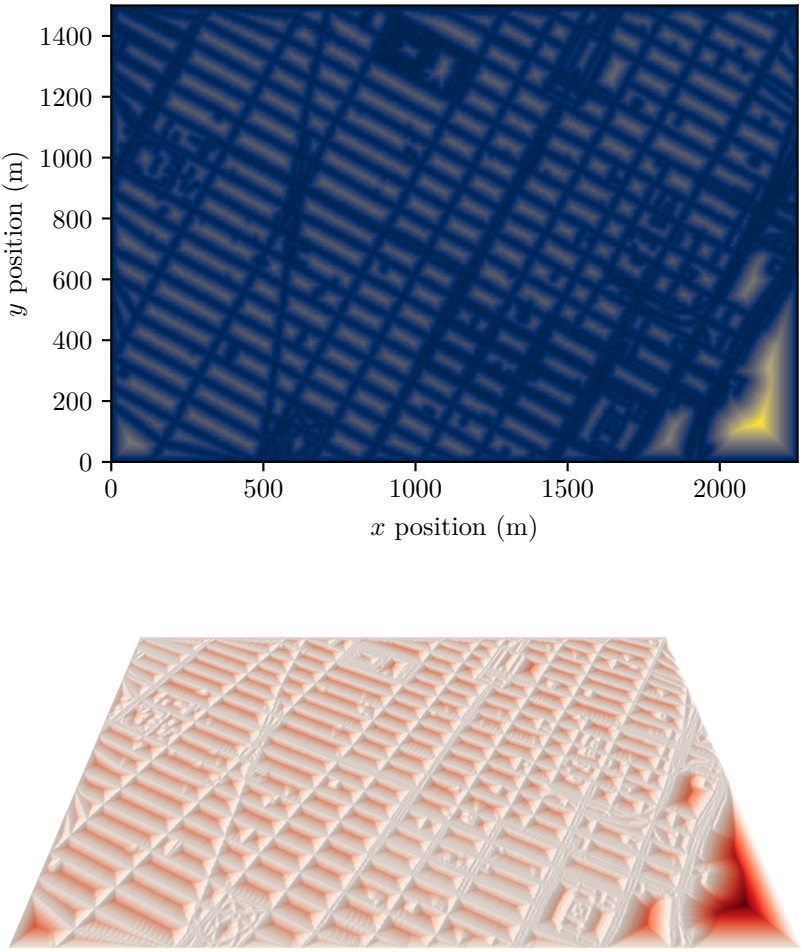


Figure A.11: The noise sensitivity information of the New York map at height $z = 0\,\text{m}$ that was derived from OSM data. Blue/white areas indicate low noise values over streets, industrial areas, and water surfaces, light-blue/gray areas indicate medium-noise values over normal buildings, and yellow/red areas indicate high-noise values over residential areas and parks.

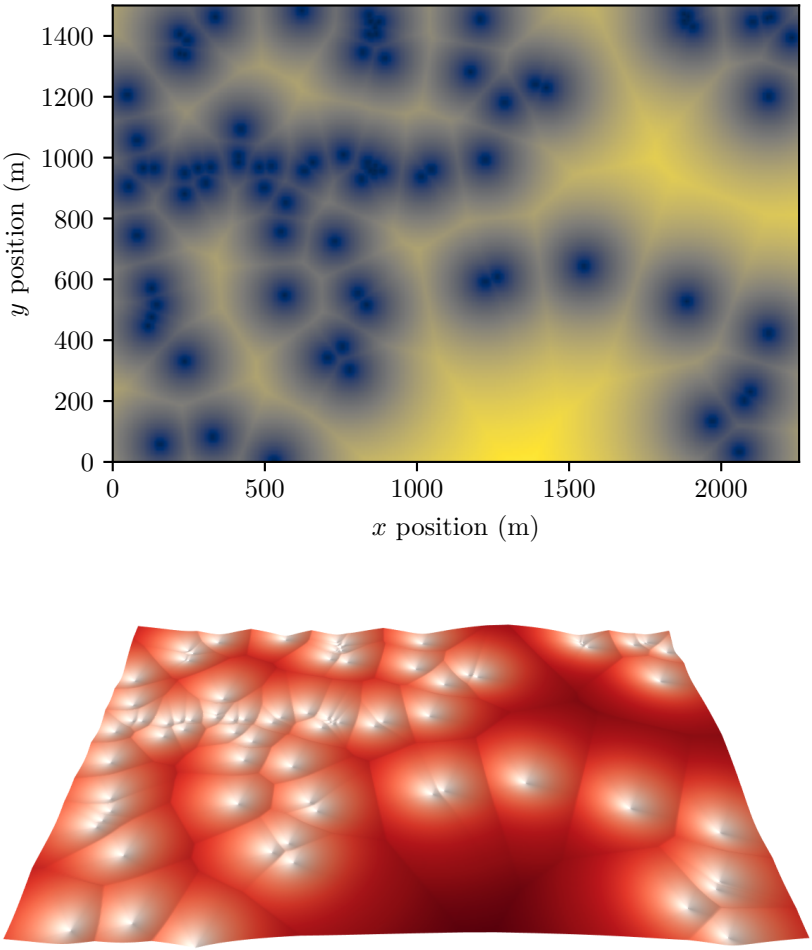


Figure A.12: The radio disturbance information of the New York map at height $z = z_{\mathrm{R}}$ that was derived from OpenCelliD (OCID) [79] data. Blue/white areas indicate low radio disturbance values around radio masts, and yellow/red areas indicate high radio disturbance values.

San Francisco

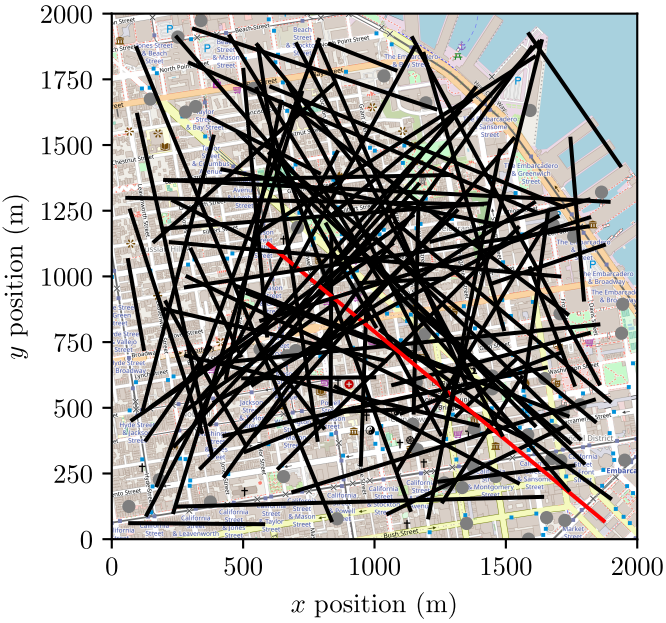


Figure A.13: The OpenStreetMap (OSM) map extract of San Francisco, US [147] that was used to derive the grid maps given in the following. The overlying lines denote the start and endpoints of the paths (i.e., scenarios) that were used for the evaluations in sections 3.5.2, 4.5.2, and Chapter 5. For the red path, more detailed information is provided regarding the calculated paths in Section 5.3.2. The gray circles denote the positions of radio masts.

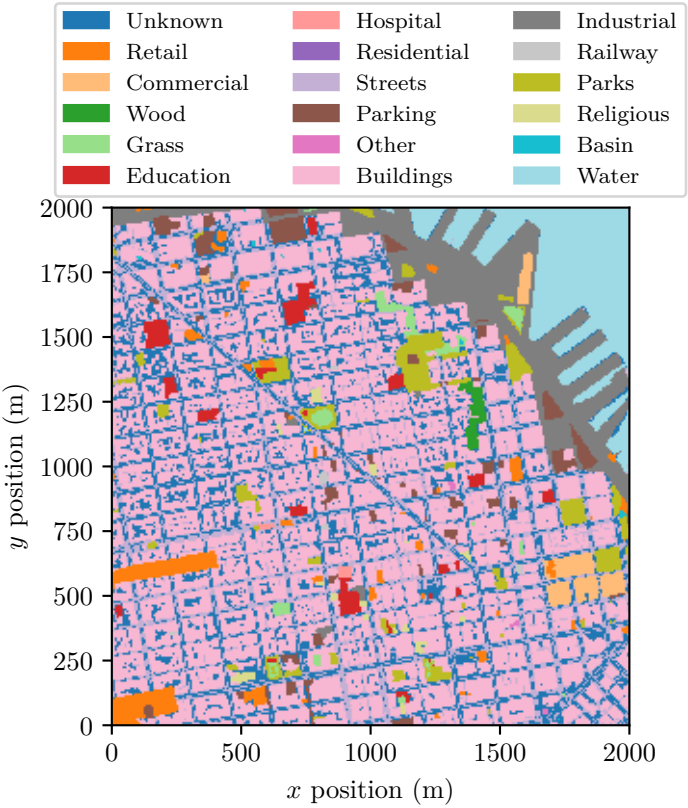


Figure A.14: A visualization of the semantic information extracted from OSM for the San Francisco map section. This serves as the basis for deriving the grid maps of different objectives given in the following.

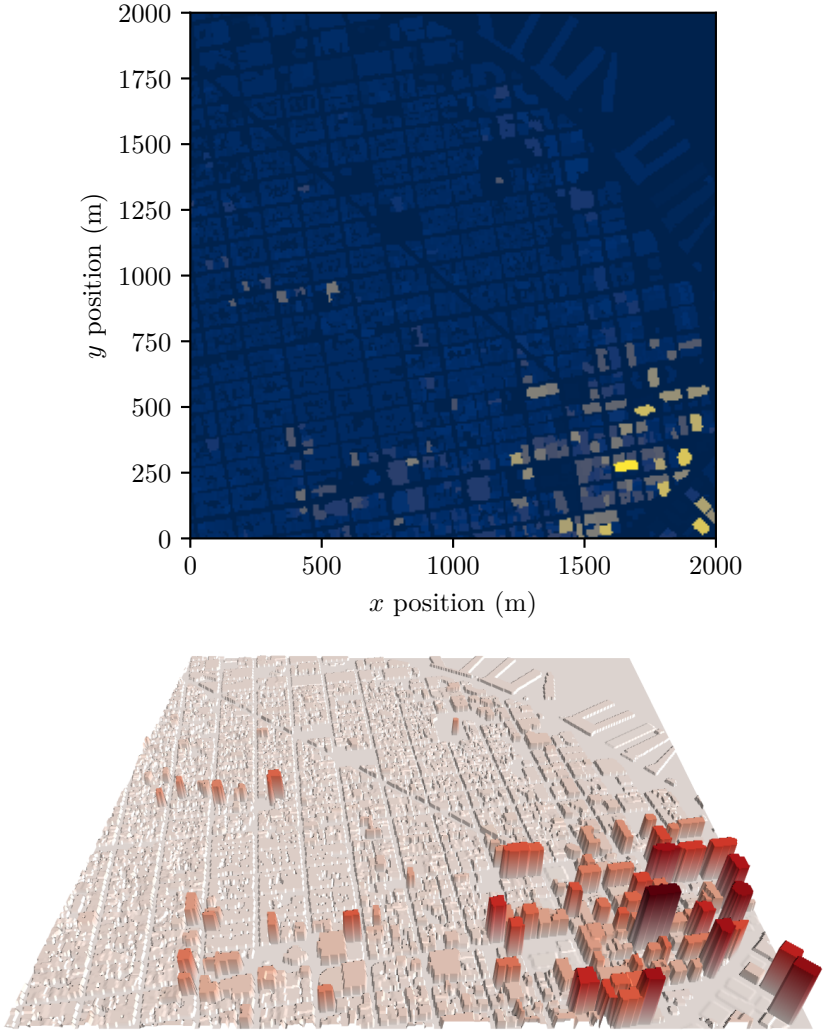


Figure A.15: Height grid map visualization of the San Francisco map extract derived from OSM data. It contains information on the position of buildings used to derive the risk grid map and the building heights used to calculate the obstacle collision avoidance constraint.

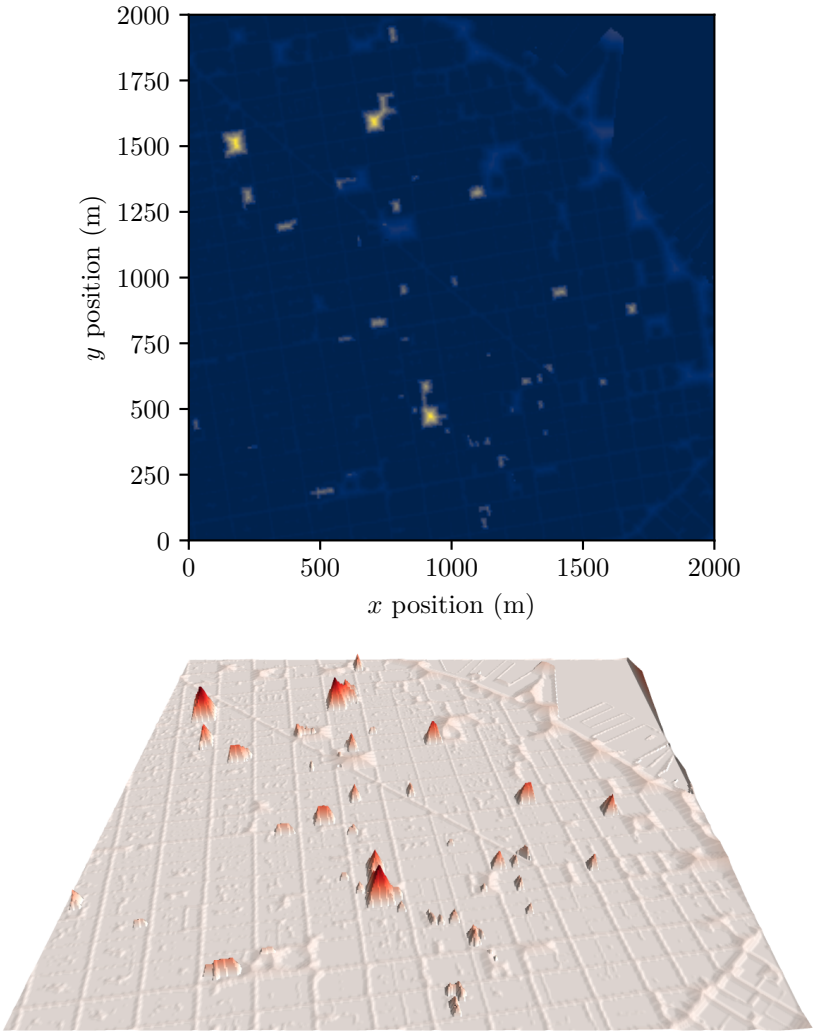


Figure A.16: The risk information of the San Francisco map at height $z = 0\,\mathrm{m}$ that was derived from OSM data. Blue/white areas indicate low-risk values over normal buildings and water surfaces, light-blue/gray areas indicate medium-risk areas over streets, and yellow/red areas indicate high-risk values over educational, medical, railway, military, and worship buildings.

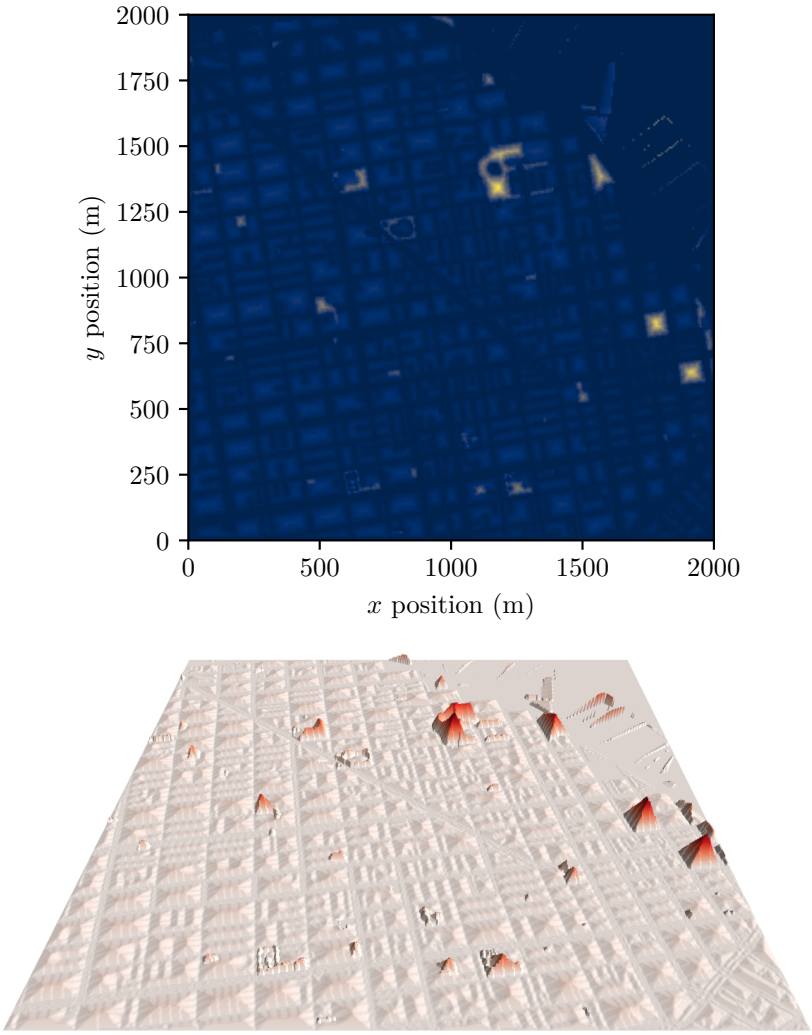


Figure A.17: The noise sensitivity information of the San Francisco map at height $z = 0\,\text{m}$ that was derived from OSM data. Blue/white areas indicate low noise values over streets, industrial areas, and water surfaces, light-blue/gray areas indicate medium-noise values over normal buildings, and yellow/red areas indicate high-noise values over residential areas and parks.

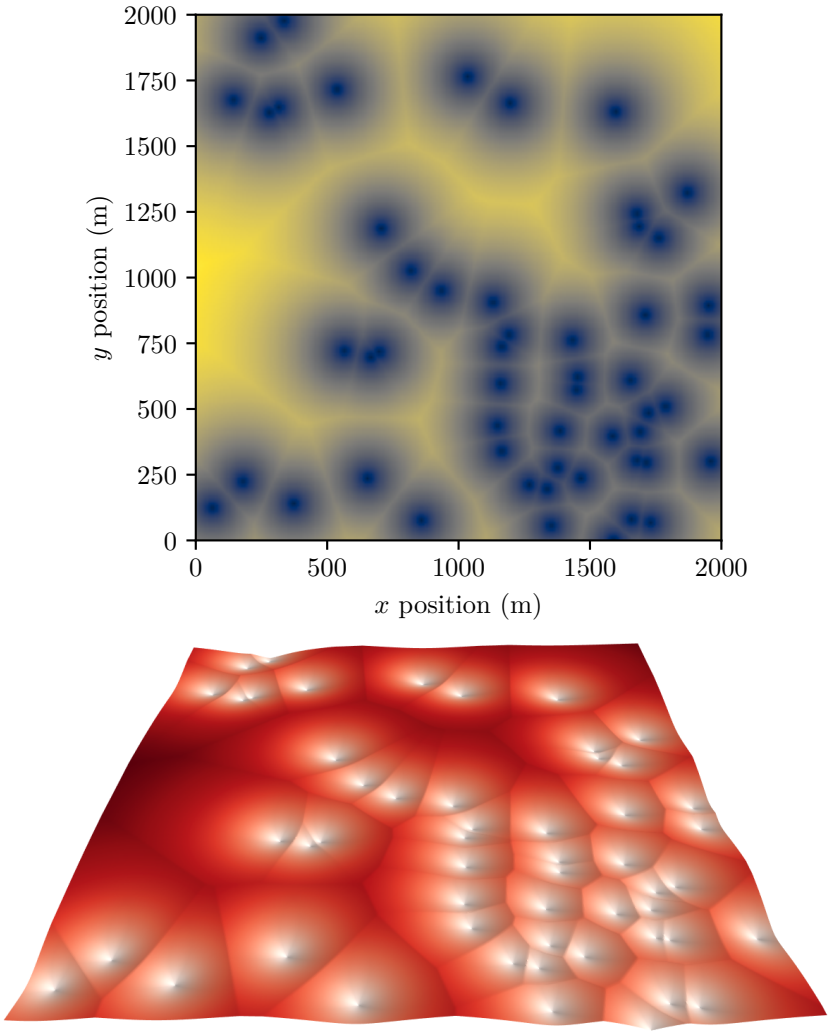


Figure A.18: The radio disturbance information of the San Francisco map at height $z = z_{\mathrm{R}}$ that was derived from OpenCelliD (OCID) [79] data. Blue/white areas indicate low radio disturbance values around radio masts, and yellow/red areas indicate high radio disturbance values.

Frankfurt

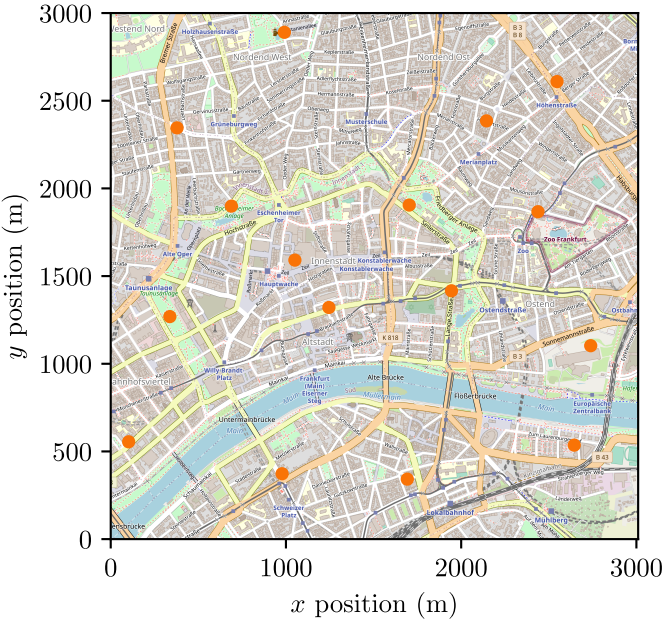


Figure A.19: The OpenStreetMap (OSM) map extract of Frankfurt, Germany [148] that was used to derive the grid maps given in the following. The orange points denote the vertiport positions that were randomly sampled within the map for evaluating the MONO framework in Section 6.5. The z-component of the vertiport positions corresponds to the height of the building at the respective location.

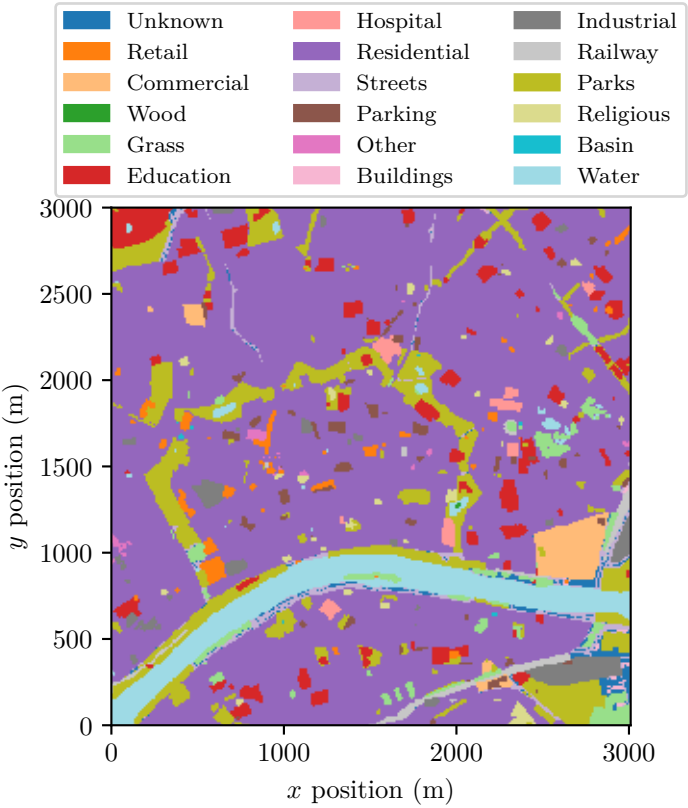


Figure A.20: A visualization of the semantic information extracted from OSM for the Frankfurt map section. This serves as the basis for deriving the grid maps of different objectives given in the following.

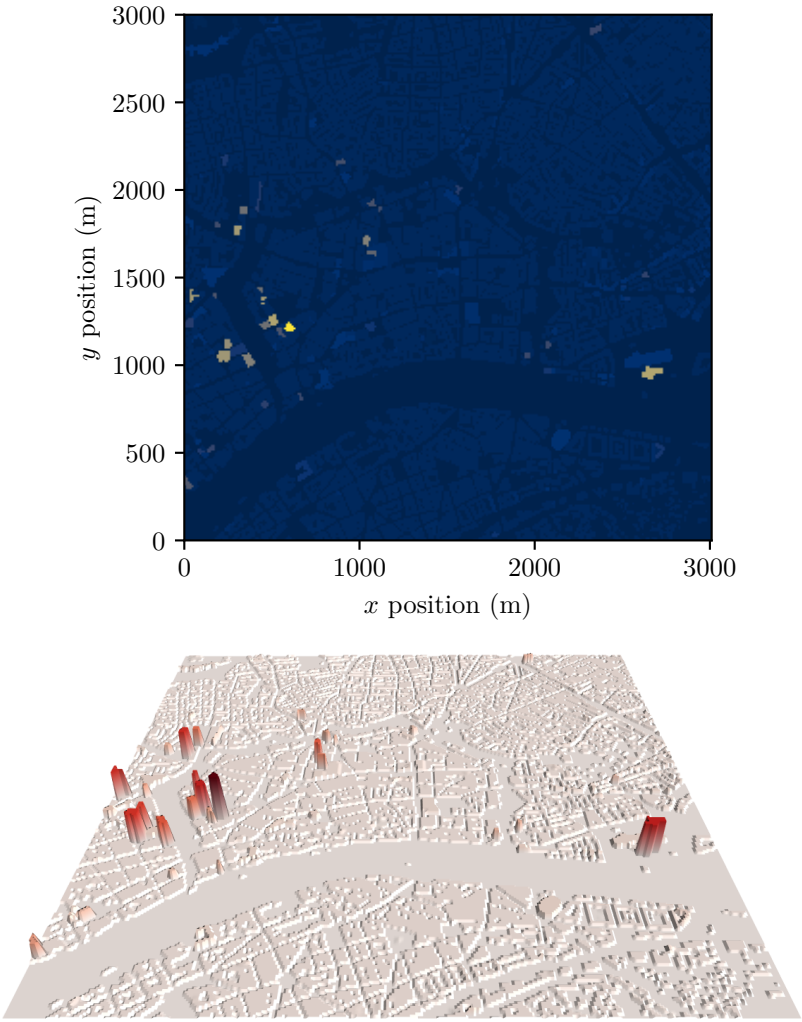


Figure A.21: Height grid map visualization of the Frankfurt map extract derived from OSM data. It contains information on the position of buildings used to derive the risk grid map and the building heights used to calculate the obstacle collision avoidance constraint.

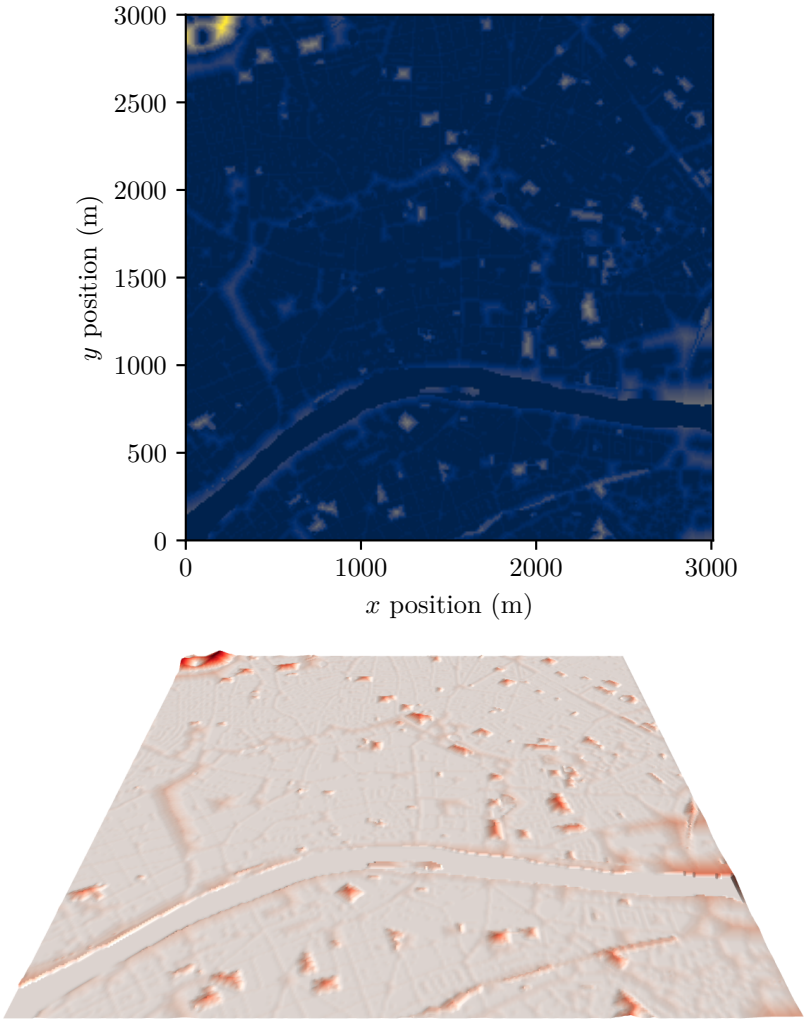


Figure A.22: The risk information of the Frankfurt map at height $z = 0\,\mathrm{m}$ that was derived from OSM data. Blue/white areas indicate low-risk values over normal buildings and water surfaces, light-blue/gray areas indicate medium-risk areas over streets, and yellow/red areas indicate high-risk values over educational, medical, railway, military, and worship buildings.

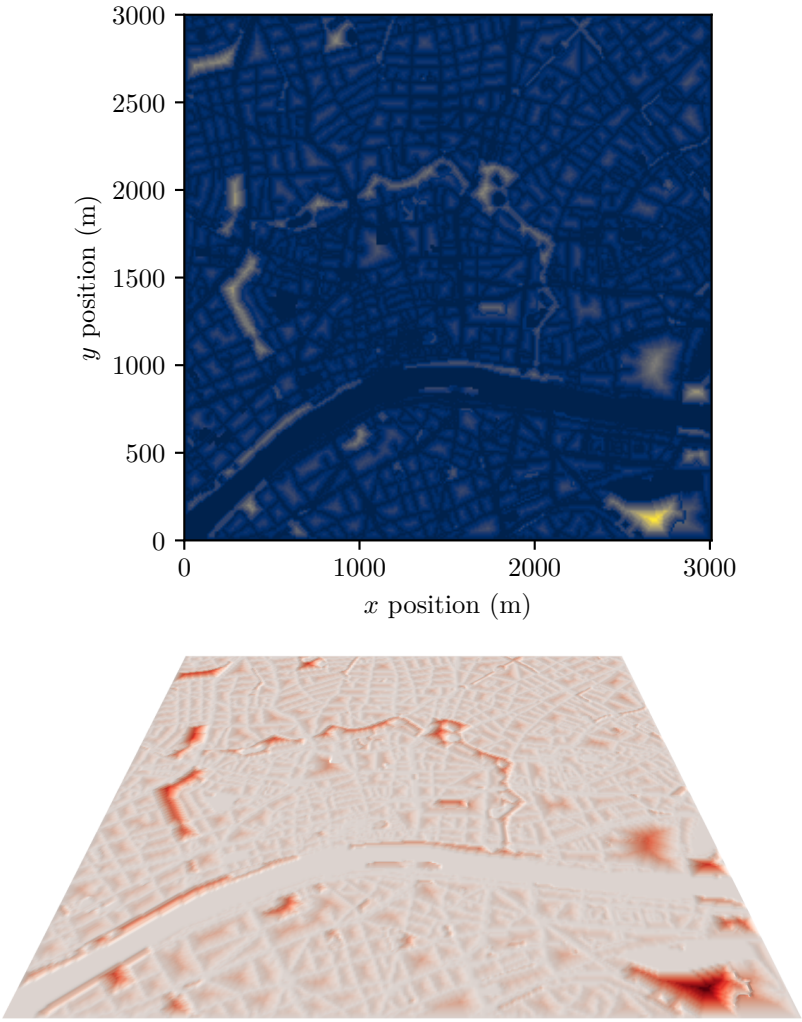


Figure A.23: The noise sensitivity information of the Frankfurt map at height $z = 0\,\mathrm{m}$ that was derived from OSM data. Blue/white areas indicate low noise values over streets, industrial areas, and water surfaces, light-blue/gray areas indicate medium-noise values over normal buildings, and yellow/red areas indicate high-noise values over residential areas and parks.

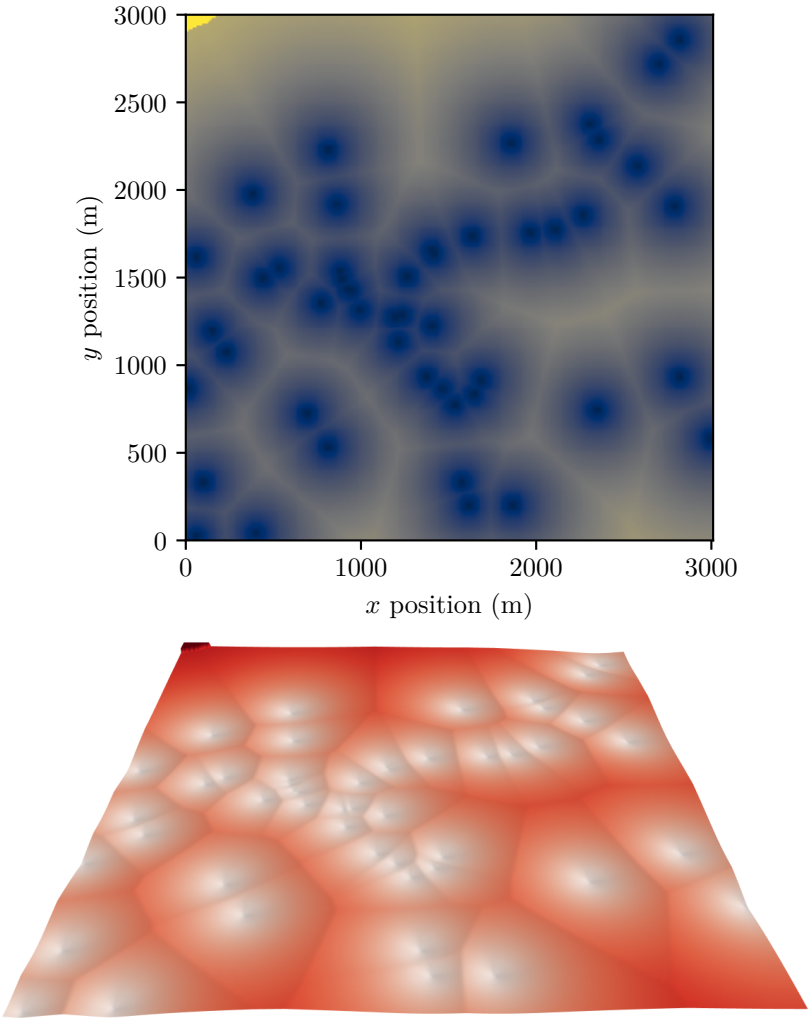


Figure A.24: The radio disturbance information of the Frankfurt map at height $z = z_{\mathrm{R}}$ that was derived from OpenCelliD (OCID) [79] data. Blue/white areas indicate low radio disturbance values around radio masts, and yellow/red areas indicate high radio disturbance values.

List of Own Publications

[N1] N. Hohmann, M. Bujny, J. Adamy, and M. Olhofer, „Hybrid evolutionary approach to multi-objective path planning for UAVs“, in *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*, IEEE, 2021, pp. 1–8. DOI: 10.1109/SSCI50451.2021.9660187.

[N2] N. Hohmann, S. Brulin, J. Adamy, and M. Olhofer, „Three-dimensional urban path planning for aerial vehicles regarding many objectives“, *IEEE Open Journal of Intelligent Transportation Systems*, vol. 4, pp. 639–652, 2023. DOI: 10.1109/OJITS.2023.3299496.

[N3] N. Hohmann, M. Bujny, J. Adamy, and M. Olhofer, „Multi-objective 3D path planning for UAVs in large-scale urban scenarios“, in *2022 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, 2022, pp. 1–8. DOI: 10.1109/CEC55065.2022.9870265.

[N4] N. Hohmann, S. Brulin, J. Adamy, and M. Olhofer, „Multi-objective optimization of urban air transportation networks under social considerations“, *IEEE Open Journal of Intelligent Transportation Systems*, vol. 5, pp. 589–602, 2024. DOI: 10.1109/OJITS.2024.3443170.

Bibliography

[1] H. Choset, K. M. Lynch, S. Hutchinson, G. A. Kantor, W. Burgard, L. E. Kavraki, and S. Thrun, *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT press, 2005.

[2] W. Hönig, J. A. Preiss, T. S. Kumar, G. S. Sukhatme, and N. Ayanian, „Trajectory planning for quadrotor swarms“, *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 856–869, 2018.

[3] V. Usenko, L. Von Stumberg, A. Pangercic, and D. Cremers, „Real-time trajectory replanning for MAVs using uniform B-splines and a 3D circular buffer“, in *Proceedings of the International Conference on Intelligent Robots and Systems*, IEEE, 2017, pp. 215–222.

[4] H. Oleynikova, Z. Taylor, R. Siegwart, and J. Nieto, „Safe local exploration for replanning in cluttered unknown environments for microaerial vehicles“, *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1474–1481, 2018.

[5] R. Stern, N. Sturtevant, A. Felner, S. Koenig, H. Ma, T. Walker, J. Li, D. Atzmon, L. Cohen, T. Kumar, *et al.*, „Multi-agent pathfinding: Definitions, variants, and benchmarks“, in *Proceedings of the International Symposium on Combinatorial Search*, vol. 10, 2019, pp. 151–158.

[6] G. Sharon, R. Stern, A. Felner, and N. R. Sturtevant, „Conflict-based search for optimal multi-agent pathfinding“, *Artificial intelligence*, vol. 219, pp. 40–66, 2015.

[7] R. Z. Farahani, E. Miandoabchi, W. Y. Szeto, and H. Rashidi, „A review of urban transportation network design problems“, *European Journal of Operational Research*, vol. 229, no. 2, pp. 281–302, 2013.

[8] S. F. W. Lim, X. Jin, and J. S. Srai, „Consumer-driven e-commerce: A literature review, design framework, and research agenda on last-mile logistics models“, *International Journal of Physical Distribution & Logistics Management*, vol. 48, no. 3, pp. 308–332, 2018.

[9] N. Boysen, S. Fedtke, and S. Schwerdfeger, „Last-mile delivery concepts: A survey from an operational research perspective“, *OR Spectrum*, vol. 43, no. 1, pp. 1–58, 2020.

[10] M. Heutger and M. Kückelhaus, „Unmanned aerial vehicles in logistics a DHL perspective on implications and use cases for the logistics industry“, DHL Customer Solutions & Innovation, Troisdorf, GER, Tech. Rep., 2014, [Online] Available: `https://www.dhl.com/discover/content/dam/dhl/downloads/interim/full/dhl-trend-report-uav.pdf`. Accessed on Oct. 31, 2022.

[11] G. Grandl, M. Ostgathe, J. Cachay, S. Doppler, J. Salib, H. Ross, J. Detert, and R. Kallenberg, „The future of vertical mobility sizing the market for passenger, inspection, and goods services until 2035“, Porsche Consulting, Munich, GER, Tech. Rep., 2018, [Online] Available: `https://fedotov.co/wp-content/uploads/2018/03/Future-of-Vertical-Mobility.pdf`. Accessed on Oct. 31, 2022.

[12] Wingcopter, *Wingcopter provides worldwide drone solutions for commercial and humanitarian applications to improve and save lives*, [Online]. Available: `https://wingcopter.com`, Accessed on Nov. 1, 2022.

[13] P. D. Vascik, H. Balakrishnan, and R. J. Hansman, „Assessment of air traffic control for urban air mobility and unmanned systems“, in *Proceedings of the International Center for Air Transportation*, FAA and EUROCONTROL, 2018.

[14] A. Bauranov and J. Rakas, „Designing airspace for urban air mobility: A review of concepts and approaches“, *Progress in Aerospace Sciences*, vol. 125, p. 100 726, 2021.

[15] E. Sunil, J. Hoekstra, J. Ellerbroek, F. Bussink, D. Nieuwenhuisen, A. Vidosavljevic, and S. Kern, „Metropolis: Relating airspace structure and capacity for extreme traffic densities“, in *Proceedings of the ATM R&D Seminar*, FAA and EUROCONTROL, 2015.

[16] D.-S. Jang, C. A. Ippolito, S. Sankararaman, and V. Stepanyan, „Concepts of airspace structures and system analysis for UAS traffic flows for urban areas“, in *Proceedings of the Infotech@Aerospace*, AIAA, 2017.

[17] Single European Sky ATM Research 3 Joint Undertaking, *U-space concept of operations (ConOps) – Third edition*. SESAR, 2019, [Online] Available: `https://www.sesarju.eu/sites/default/files/documents/u-space/CORUS%20ConOps%20vol2.pdf`. Accessed on March 25, 2024.

[18] L. Piegl and W. Tiller, „Rational B-spline curves and surfaces“, in *The NURBS book*, 2nd ed., Berlin, GER: Springer, 1997, ch. 4, pp. 117–138.

[19] M. G. Cox, „The numerical evaluation of B-splines“, *IMA Journal of Applied mathematics*, vol. 10, no. 2, pp. 134–149, 1972.

[20] C. De Boor, „On calculating with B-splines“, *Journal of Approximation theory*, vol. 6, no. 1, pp. 50–62, 1972.

[21] C. De Boor, *A practical guide to splines*, 2001.

[22] G. Farin, J. Hoschek, and M.-S. Kim, „Curve and surface constructions“, in *Handbook of Computer Aided Geometric Design*, 1st ed., Amsterdam, The Netherlands: Elsevier, 2002, ch. 7, pp. 165–192.

[23] K. Deb and H. Jain, „An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints“, *IEEE Transactions on Evolutionary Computing*, vol. 18, no. 4, pp. 577–601, 2014.

[24] T. Schmitt, „A posteriori decision making strategies“, in *Multi-objective building energy management optimization with model predictive control*, Ph.D. dissertation, [Online] Available: `http://tuprints.ulb.tu-darmstadt.de/22344/`, Technische Universität Darmstadt, 2022, ch. 4.2.2, pp. 81–84.

[25] G. Sun, G. Li, S. Zhou, H. Li, S. Hou, and Q. Li, „Crashworthiness design of vehicle by using multiobjective robust optimization“, *Structural and Multidisciplinary Optimization*, vol. 44, pp. 99–110, 2011.

[26] E. Zitzler and L. Thiele, „Multiobjective optimization using evolutionary algorithms—a comparative case study“, in *Proceedings of the International Conference on Parallel Problem Solving from Nature*, Springer, 1998, pp. 292–301.

[27] E. Zitzler and L. Thiele, „Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach“, *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.

[28] A. P. Guerreiro, C. M. Fonseca, and L. Paquete, „The hypervolume indicator: Computational problems and algorithms“, *Computing Surveys*, vol. 54, no. 6, pp. 1–42, 2021.

[29] D. A. Van Veldhuizen and G. B. Lamont, „On measuring multiobjective evolutionary algorithm performance“, in *Proceedings of the Congress on Evolutionary Computation*, IEEE, vol. 1, 2000, pp. 204–211.

[30] C. A. C. Coello and N. C. Cortés, „Solving multiobjective optimization problems using an artificial immune system“, *Genetic Programming and Evolvable Machines*, vol. 6, no. 2, pp. 163–190, 2005.

[31] O. Souissi, R. Benatitallah, D. Duvivier, A. Artiba, N. Belanger, and P. Feyzeau, „Path planning: A 2013 survey“, in *Proceedings of the International Conference on Industrial Engineering and Systems Management*, IEEE, 2013, pp. 1–8.

[32] S. M. LaValle, „Feedback motion planning“, in *Planning algorithms*, Cambridge university press, 2006, ch. 8, pp. 369–494.

[33] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, „Probabilistic roadmaps for path planning in high-dimensional configuration spaces“, *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[34] S. LaValle, „Rapidly-exploring random trees: A new tool for path planning“, *Res Rep 9811*, 1998.

[35] S. M. LaValle, *Planning algorithms*. Cambridge university press, 2006.

[36] T. Bäck and H.-P. Schwefel, „An overview of evolutionary algorithms for parameter optimization“, *Evolutionary Computation*, vol. 1, no. 1, pp. 1–23, 1993.

[37] T. Bäck, *Evolutionary algorithms in theory and practice: evolution strategies, evolutionary programming, genetic algorithms*. Oxford university press, 1996.

[38] H.-G. Beyer and H.-P. Schwefel, „Evolution strategies–a comprehensive introduction“, *Natural computing*, vol. 1, no. 1, pp. 3–52, 2002.

[39] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, „A fast and elitist multiobjective genetic algorithm: NSGA-II“, *IEEE Transactions on Evolutionary Computing*, vol. 6, no. 2, pp. 182–197, 2002.

[40] M. Emmerich and A. H. Deutz, „A tutorial on multiobjective optimization: Fundamentals and evolutionary methods“, *Natural Computing*, vol. 17, no. 3, pp. 585–609, 2018.

[41] E. Zitzler, M. Laumanns, and S. Bleuler, „A tutorial on evolutionary multiobjective optimization“, *Metaheuristics for multiobjective optimisation*, pp. 3–37, 2004.

[42] E. W. Dijkstra, „A note on two problems in connexion with graphs“, *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[43] P. E. Hart, N. J. Nilsson, and B. Raphael, „A formal basis for the heuristic determination of minimum cost paths“, *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[44] S. J. Russell and P. Norvig, „Informed (heuristic) search strategies“, in *Artificial intelligence: a modern approach*, Fourth Global, London, UK: Pearson, 2021, ch. II 3, pp. 102–115.

[45] Z. M. Tarapata, „Selected multicriteria shortest path problems: An analysis of complexity, models and adaptation of standard algorithms“, *International Journal of Applied Mathematics & Computer Science*, vol. 17, no. 2, 2007.

[46] E. Q. V. Martins, „On a multicriteria shortest path problem“, *European Journal of Operational Research*, vol. 16, no. 2, pp. 236–245, 1984.

[47] F. Guerriero and R. Musmanno, „Label correcting methods to solve multicriteria shortest path problems“, *Journal of optimization theory and applications*, vol. 111, pp. 589–613, 2001.

[48] J. Hrnčíř, P. Žileckỳ, Q. Song, and M. Jakob, „Practical multicriteria urban bicycle routing“, *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 3, pp. 493–504, 2016.

[49] M. Iori, S. Martello, and D. Pretolani, „An aggregate label setting policy for the multi-objective shortest path problem“, *European Journal of Operational Research*, vol. 207, no. 3, pp. 1489–1496, 2010.

[50] I. K. Nikolos, K. P. Valavanis, N. C. Tsourveloudis, and A. N. Kostaras, „Evolutionary algorithm based offline/online path planner for UAV navigation“, *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 33, no. 6, pp. 898–912, 2003.

[51] J. Rubio-Hervas, A. Gupta, and Y.-S. Ong, „Data-driven risk assessment and multicriteria optimization of UAV operations“, *Aerospace Science and Technology*, vol. 77, pp. 510–523, 2018.

[52] S. Ghambari, M. Golabi, J. Lepagnot, M. Brévilliers, L. Jourdan, and L. Idoumghar, „An enhanced NSGA-II for multiobjective UAV path planning in urban environments“, in *Proceedings of the International Conference on Tools with Artificial Intelligence*, IEEE, 2020, pp. 106–111.

[53] N. Sadallah, S. Yahiaoui, A. Bendjoudi, and N. Nouali-Taboudjemat, „Multi-objective offline and online path planning for UAVs under dynamic urban environment“, *International Journal of Intelligent Robotics and Applications*, vol. 6, no. 1, pp. 119–138, 2021.

[54] D. Gelperin, „On the optimality of A*“, *Artificial Intelligence*, vol. 8, no. 1, pp. 69–76, 1977.

[55] T. T. Mac, C. Copot, D. T. Tran, and R. De Keyser, „Heuristic approaches in robot path planning: A survey“, *Robotics and Autonomous Systems*, vol. 86, pp. 13–28, 2016.

[56] C. A. Tovey, „Nature-inspired heuristics: Overview and critique“, *Recent Advances in Optimization and Modeling of Contemporary Problems*, pp. 158–192, 2018.

[57] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, „A limited memory algorithm for bound constrained optimization“, *SIAM Journal on Scientific Computing*, vol. 16, no. 5, pp. 1190–1208, 1995.

[58] L. Piegl and W. Tiller, „Standards and data exchange“, in *The NURBS book*, 2nd ed., Berlin, GER: Springer, 1997, ch. 12, pp. 571–580.

[59] D. Mellinger and V. Kumar, „Minimum snap trajectory generation and control for quadrotors“, in *Proceedings of the International Conference on Robotics and Automation*, IEEE, 2011, pp. 2520–2525.

[60] C. Richter, A. Bry, and N. Roy, „Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments“, in *Proceedings of the International Symposium on Robotics Research*, Springer, 2016, pp. 649–666.

[61] A. Weiser and S. E. Zarantonello, „A note on piecewise linear and multilinear table interpolation in many dimensions“, *Mathematics of Computation*, vol. 50, no. 181, pp. 189–196, 1988.

[62] R. Fabbri, L. D. F. Costa, J. C. Torelli, and O. M. Bruno, „2D euclidean distance transform algorithms: A comparative survey“, *Computing Surveys*, vol. 40, no. 1, pp. 1–44, 2008.

[63] A. J. Torija, Z. Li, and R. H. Self, „Effects of a hovering unmanned aerial vehicle on urban soundscapes perception“, *Transportation Research Part D: Transport and Environment*, vol. 78, p. 102 195, 2020.

[64] J. S. Reid, *Drone flight — what does basic physics say?*, [Online]. Available: `https://homepages.abdn.ac.uk/nph120/meteo/DroneFlight.pdf`, [Accessed on Feb. 11, 2022], 2020.

[65] K. Deb, „Multi-objective optimisation using evolutionary algorithms: An introduction“, in *Multi-objective Evolutionary Optimisation for Product Design and Manufacturing*, Springer, 2011, pp. 3–34.

[66] L. Piegl and W. Tiller, „Curve and surface fitting“, in *The NURBS book*, 2nd ed., Berlin, GER: Springer, 1997, ch. 9, pp. 410–413.

[67] OpenStreetMap contributors, [Online]. Available: `https://www.openstreetmap.org`, Planet dump retrieved from `https://planet.osm.org`.

[68] D. C. Liu and J. Nocedal, „On the limited memory BFGS method for large scale optimization“, *Mathematical programming*, vol. 45, no. 1, pp. 503–528, 1989.

[69] C. A. C. Coello, „Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art“, *Computer Methods in Applied Mechanics and Engineering*, vol. 191, no. 11-12, pp. 1245–1287, 2002.

[70] F. Hoffmeister and J. Sprave, „Problem-independent handling of constraints by use of metric penalty functions“, *Evolutionary Programming*, vol. 870, 1996.

[71] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*. MIT press, 2022.

[72] T. Voß, N. Hansen, and C. Igel, „Improved step size adaptation for the MO-CMA-ES“, in *Proceedings of the Conference on Genetic and Evolutionary Computation*, 2010, pp. 487–494.

[73] N. Ryu and S. Min, „Multiobjective optimization with an adaptive weight determination scheme using the concept of hyperplane“, *International Journal for Numerical Methods in Engineering*, vol. 118, no. 6, pp. 303–319, 2019.

[74] S. Das, S. Maity, B.-Y. Qu, and P. N. Suganthan, „Real-parameter evolutionary multimodal optimization — a survey of the state-of-the-art“, *Swarm and Evolutionary Computation*, vol. 1, no. 2, pp. 71–88, 2011.

[75] X. Li, M. G. Epitropakis, K. Deb, and A. Engelbrecht, „Seeking multiple solutions: An updated survey on niching methods and their applications“, *IEEE Transactions on Evolutionary Computing*, vol. 21, no. 4, pp. 518–538, 2016.

[76] K. Deb and S. Agrawal, „A niched-penalty approach for constraint handling in genetic algorithms“, in *Proceedings of the International Conference on Artificial Neural Nets and Genetic Algorithms*, Springer, 1999, pp. 235–243.

[77] D. J. Poole and C. B. Allen, „Constrained niching using differential evolution“, *Swarm and Evolutionary Computation*, vol. 44, pp. 74–100, 2019.

[78] K. Deb and A. Saha, „Multimodal optimization using a bi-objective evolutionary algorithm“, *Evolutionary Computation*, vol. 20, no. 1, pp. 27–62, 2012.

[79] OpenCellid contributors, [Online] Available: `https://opencellid.org`, Accessed on May 23, 2022.

[80] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff, „A reference vector guided evolutionary algorithm for many-objective optimization“, *IEEE Transactions on Evolutionary Computing*, vol. 20, no. 5, pp. 773–791, 2016.

[81] N. Beume, B. Naujoks, and M. Emmerich, „SMS-EMOA: Multiobjective selection based on dominated hypervolume“, *European Journal of Operational Research*, vol. 181, no. 3, pp. 1653–1669, 2007.

[82] I. Das and J. E. Dennis, „A closer look at drawbacks of minimizing weighted sums of objectives for pareto set generation in multicriteria optimization problems“, *Structural optimization*, vol. 14, pp. 63–69, 1997.

[83] F. Barth, S. Funke, and S. Storandt, „Alternative multicriteria routes“, in *Proceedings of the Twenty-First Workshop on Algorithm Engineering and Experiments*, SIAM, 2019, pp. 66–80.

[84] M. Shekelyan, G. Jossé, and M. Schubert, „Linear path skylines in multicriteria networks“, in *Proceedings of the International Conference on Data Engineering*, IEEE, 2015, pp. 459–470.

[85] P. Fontaine, „Urban air mobility (UAM) concept of operations“, Federal Aviation Administration, Washington, DC, Tech. Rep., 2023, [Online] Available: `https://www.faa.gov/sites/faa.gov/files/Urban%20Air%20Mobility%20%28UAM%29%20Concept%20of%20Operations%202.0_1.pdf`. Accessed on Feb. 27, 2024.

[86] Single European Sky ATM Research 3 Joint Undertaking, *U-space concept of operations (ConOps) – Fourth edition*. Publications Office of the European Union, 2023, [Online] Available: `https://op.europa.eu/s/zg7g`. Accessed on Feb. 28, 2024. DOI: `10.2829/207917`.

[87] D. Geister and B. Korn, „DLR blueprint - concept for urban airspace integration“, German Aerospace Center, Cologne, GER, Tech. Rep., 2017, [Online] Available: `https://www.dlr.de/de/medien/publikationen/sonstige-publikationen/2017/blueprint-concept-for-urban-airspace-integration_2933/@@download/file`. Accessed on Feb. 27, 2024.

[88] K. Balakrishnan, J. Polastre, J. Mooberry, R. Golding, and P. Sachs, „Blueprint for the sky: The roadmap for the safe integration of autonomous aircraft“, A^3 Airbus, San Francisco, CA, Tech. Rep., 2018, [Online] Available: `https://www.airbus.com/sites/g/files/jlcbta136/files/2022-07/Airbus-whitepaper-integration-autonomous-vehicles.pdf`. Accessed on Feb. 28, 2024.

[89] „Flight plan 2030: An air traffic management concept for urban air mobility“, EmbraerX, Fort Lauderdale, FL, Tech. Rep., 2019, [Online] Available: `https://daflwcl3bnxyt.cloudfront.net/m/4e5924f5de45fd3a/original/embraerx-whitepaper-flightplan2030.pdf`. Accessed on Feb. 28, 2024.

[90] J. Holden and N. Goel, „Fast-forwarding to a future of on-demand urban air transportation“, Uber, San Francisco, CA, Tech. Rep., 2016, [Online] Available: `https://d1nyezh1ys8wfo.cloudfront.net/static/PDFs/Elevate%2BWhitepaper.pdf?uclick_id=688eb213-8550-46cd-b1a9-4ba0fe26fb6d`. Accessed on Feb. 28, 2024.

[91] C. L. Denham, W. G. Cummings, and J. C. Smith, „Theoretical and simulated capacity of urban air mobility airspace characteristics“, in *Proceedings of the SciTech Forum*, AIAA, 2023, p. 0546.

[92] W. G. Cummings, C. L. Denham, and J. C. Smith, „Effect of airspace characteristics on urban air mobility airspace capacity“, in *Proceedings of the SciTech Forum*, AIAA, 2023, p. 0545.

[93] A. Fedrigo, „A series of macroscopic models for urban air mobility traffic flow“, in *Proceedings of the SciTech Forum*, AIAA, 2023, p. 0785.

[94] B. Pang, W. Dai, T. Ra, and K. H. Low, „A concept of airspace configuration and operational rules for UAS in current airspace“, in *Proceedings of the Digital Avionics Systems Conference*, IEEE, 2020, pp. 1–9.

[95] Y. Zheng, „Trajectory data mining: An overview“, *Transactions on Intelligent Systems and Technology*, vol. 6, no. 3, pp. 1–41, 2015.

[96] M. Gariel, A. N. Srivastava, and E. Feron, „Trajectory clustering and an application to airspace monitoring“, *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1511–1524, 2011.

[97] P. Tampakis, N. Pelekis, N. Andrienko, G. Andrienko, G. Fuchs, and Y. Theodoridis, „Time-aware sub-trajectory clustering in Hermes@PostgreSQL“, in *Proceedings of the International Conference on Data Engineering*, IEEE, 2018, pp. 1581–1584.

[98] J.-G. Lee, J. Han, and K.-Y. Whang, „Trajectory clustering: A partition-and-group framework“, in *Proceedings of the SIGMOD International Conference on Management of Data*, ACM, 2007, pp. 593–604.

[99] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, „A density-based algorithm for discovering clusters in large spatial databases with noise“, in *Proceedings of the International Conference on Knowledge Discovery and Data Mining*, AAAI, vol. 96, 1996, pp. 226–231.

[100] Z. Chen, H. T. Shen, and X. Zhou, „Discovering popular routes from trajectories“, in *Proceedings of the International Conference on Data Engineering*, IEEE, 2011, pp. 900–911.

[101] M. Ahmed, S. Karagiorgou, D. Pfoser, and C. Wenk, „A comparison and evaluation of map construction algorithms using vehicle tracking data“, *Geoinformatica*, vol. 19, pp. 601–632, 2015.

[102] R. Ding, N. Ujang, H. B. Hamid, M. S. A. Manan, R. Li, S. S. M. Albadareen, A. Nochian, and J. Wu, „Application of complex networks theory in urban traffic network researches“, *Networks and Spatial Economics*, vol. 19, pp. 1281–1317, 2019.

[103] M. T. Gastner and M. E. Newman, „The spatial structure of networks“, *The European Physical Journal B-Condensed Matter and Complex Systems*, vol. 49, pp. 247–252, 2006.

[104] G. Li, S. D. S. Reis, A. A. Moreira, S. Havlin, H. E. Stanley, and J. S. Andrade Jr., „Towards design principles for optimal transport networks“, *Physical Review Letters*, vol. 104, no. 1, p. 018 701, 2010.

[105] A. Chen, J. Kim, S. Lee, and Y. Kim, „Stochastic multi-objective models for network design problem“, *Expert Systems with Applications*, vol. 37, no. 2, pp. 1608–1619, 2010.

[106] X. He, F. He, L. Li, L. Zhang, and G. Xiao, „A route network planning method for urban air delivery“, *Transportation Research Part E: Logistics and Transportation Review*, vol. 166, p. 102 872, 2022.

[107] S. Mai, M. Deubel, and S. Mostaghim, „Multi-objective roadmap optimization for multiagent navigation“, in *Proceedings of the Congress on Evolutionary Computation*, IEEE, 2022, pp. 1–8.

[108] V. Bulusu, V. Polishchuk, and L. Sedov, „Noise estimation for future large-scale small UAS operations“, in *Proceedings of the Inter-Noise and Noise-Con Congress*, I-INCE, vol. 254, 2017, pp. 864–871.

[109] S. Brulin and M. Olhofer, „Bi-level network design for UAM vertiport allocation using activity-based transport simulations“, in *Proceedings of the International Electric Vehicle Technology Conference*, JSAE, 2023.

[110] F. K. Hwang and D. S. Richards, „Steiner tree problems“, *Networks*, vol. 22, no. 1, pp. 55–89, 1992.

[111] M. Barthélemy, „Spatial networks“, *Physics reports*, vol. 499, no. 1-3, pp. 1–101, 2011.

[112] B. Rao, A. G. Gopi, and R. Maione, „The societal impact of commercial drones“, *Technology in Society*, vol. 45, pp. 83–90, 2016.

[113] M. Ehrgott, „Multiobjective combinatorial optimization“, in *Multicriteria Optimization*, vol. 491, Springer, 2005, ch. 8, pp. 197–220.

[114] M. Ehrgott and X. Gandibleux, „Approximative solution methods for multiobjective combinatorial optimization“, *Top*, vol. 12, no. 1, pp. 1–63, 2004.

[115] J. Blank and K. Deb, „Pymoo: Multi-objective optimization in python“, *IEEE Access*, vol. 8, pp. 89 497–89 509, 2020.

[116] J. Blank and K. Deb, „A running performance metric and termination criterion for evaluating evolutionary multi-and many-objective optimization algorithms“, in *Proceedings of the Congress on Evolutionary Computation*, IEEE, 2020, pp. 1–8.

[117] Google Earth Studio, *Animation tool for 3D and satellite images from Google Earth*, [Online] Available: `https://earth.google.com/studio/docs/`, Accessed on Feb. 26, 2024.

[118] kepler.gl, *Geospatial analysis tool*, [Online] Available: `https://kepler.gl/demo`, Accessed on Feb. 26, 2024.

[119] N. Moradi, C. Wang, and F. Mafakheri, „Urban air mobility for last-mile transportation: A review“, *Vehicles*, vol. 6, no. 3, pp. 1383–1414, 2024.

[120] D. Meagher, „Geometric modeling using octree encoding“, *Computer graphics and image processing*, vol. 19, no. 2, pp. 129–147, 1982.

[121] X. Zhen, Z. Enze, and C. Qingwei, „Rotary unmanned aerial vehicles path planning in rough terrain based on multi-objective particle swarm optimization“, *Journal of Systems Engineering and Electronics*, vol. 31, no. 1, pp. 130–141, 2020.

[122] V. Roberge, M. Tarbouchi, and G. Labonté, „Comparison of parallel genetic algorithm and particle swarm optimization for real-time UAV path planning“, *IEEE Transactions on industrial informatics*, vol. 9, no. 1, pp. 132–141, 2012.

[123] L. Babel, „Three-dimensional route planning for unmanned aerial vehicles in a risk environment“, *Journal of Intelligent & Robotic Systems*, vol. 71, no. 2, pp. 255–269, 2013.

[124] I. Hasircioglu, H. R. Topcuoglu, and M. Ermis, „3-D path planning for the navigation of unmanned aerial vehicles by using evolutionary algorithms“, in *Proceedings of the Annual Conference on Genetic and Evolutionary Computation*, 2008, pp. 1499–1506.

[125] Y. Fu, M. Ding, and C. Zhou, „Phase angle-encoded and quantum-behaved particle swarm optimization applied to three-dimensional route planning for UAV“, *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 42, no. 2, pp. 511–526, 2011.

[126] Q. Yang and S.-J. Yoo, „Optimal UAV path planning: Sensing data acquisition over IoT sensor networks using multi-objective bio-inspired algorithms“, *IEEE Access*, vol. 6, pp. 13 671–13 684, 2018.

[127] Q. Ren, Y. Yao, G. Yang, and X. Zhou, „Multi-objective path planning for UAV in the urban environment based on CDNSGA-II“, in *Proceedings of the International Conference on Service-Oriented System Engineering*, IEEE, 2019, pp. 350–3505.

[128] Y. Chen, J. Yu, Y. Mei, Y. Wang, and X. Su, „Modified central force optimization (MCFO) algorithm for 3D UAV path planning“, *Neurocomputing*, vol. 171, pp. 878–888, 2016.

[129] S. Primatesta, L. S. Cuomo, G. Guglieri, and A. Rizzo, „An innovative algorithm to estimate risk optimum path for unmanned aerial vehicles in urban environments“, *Transportation research procedia*, vol. 35, pp. 44–53, 2018.

[130] M. Golabi, S. Ghambari, J. Lepagnot, L. Jourdan, M. Brévilliers, and L. Idoumghar, „Bypassing or flying above the obstacles? A novel multi-objective UAV path planning problem“, in *Proceedings of the Congress on Evolutionary Computation*, IEEE, 2020, pp. 1–8.

[131] M. Davoodi, F. Panahi, A. Mohades, and S. N. Hashemi, „Multi-objective path planning in discrete space“, *Applied Soft Computing*, vol. 13, no. 1, pp. 709–720, 2013.

[132] F. Ahmed and K. Deb, „Multi-objective optimal path planning using elitist non-dominated sorting genetic algorithms“, *Soft Computing*, vol. 17, no. 7, pp. 1283–1299, 2013.

[133] C. Yin, Z. Xiao, X. Cao, X. Xi, P. Yang, and D. Wu, „Offline and online search: UAV multiobjective path planning under dynamic urban environment“, *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 546–558, 2017.

[134] N. Ganganath, C.-T. Cheng, and K. T. Chi, „Multiobjective path planning on uneven terrains based on NAMOA*“, in *Proceedings of the International Symposium on Circuits and Systems*, IEEE, 2016, pp. 1846–1849.

[135] Y. Hao, B. Li, L. Shao, Y. Zhang, and J. Cui, „Multi-objective path planning for unmanned aerial vehicle based on mixed integer programming“, in *Proceedings of the Chinese Automation Congress*, IEEE, 2017, pp. 7035–7039.

[136] Y. Zhang, D.-w. Gong, and J.-h. Zhang, „Robot path planning in uncertain environment using multi-objective particle swarm optimization“, *Neurocomputing*, vol. 103, pp. 172–185, 2013.

[137] Y. Ma, M. Hu, and X. Yan, „Multi-objective path planning for unmanned surface vehicle with currents effects“, *ISA transactions*, vol. 75, pp. 137–156, 2018.

[138] F. Ahmed and K. Deb, „Multi-objective path planning using spline representation“, in *Proceedings of the International Conference on Robotics and Biomimetics*, IEEE, 2011, pp. 1047–1052.

[139] S. Mittal and K. Deb, „Three-dimensional offline path planning for UAVs using multiobjective evolutionary algorithms“, in *Proceedings of the Congress on Evolutionary Computation*, IEEE, 2007, pp. 3195–3202.

[140] S. Jalel, P. Marthon, and A. Hamouda, „A new path generation algorithm based on accurate NURBS curves“, *International Journal of Advanced Robotic Systems*, vol. 13, no. 2, p. 75, 2016.

[141] S. Jalel, P. Marthon, and A. Hamouda, „NURBS based multi-objective path planning“, in *Proceedings of the Mexican Conference on Pattern Recognition*, Springer, 2015, pp. 190–199.

[142] R. Bellman, „On a routing problem“, *Quarterly of applied mathematics*, vol. 16, no. 1, pp. 87–90, 1958.

[143] L. R. Ford, „Network flow theory“, 1956.

[144] L. Mandow, J. P. De la Cruz, *et al.*, „A new approach to multiobjective A* search“, in *Proceedings of the International Joint Conferences on Artificial Intelligence*, IJCAI, vol. 8, 2005.

[145] OpenStreetMap, *Darmstadt city*, [Online] Available: `https://www.openstreetmap.org/export#map=16/49.8748/8.6663`, Accessed on April. 29, 2024.

[146] OpenStreetMap, *New york manhattan*, [Online] Available: `https://www.openstreetmap.org/export#map=16/40.7483/-73.9818`, Accessed on April. 29, 2024.

[147] OpenStreetMap, *San francisco area around telegraph hill*, [Online] Available: `https://www.openstreetmap.org/search?query=san20francisco#map=16/37.8035/-122.4068`, Accessed on Sept. 6, 2022.

[148] OpenStreetMap, *Frankfurt city center*, [Online] Available: `https://www.openstreetmap.org/export#map=14/50.1138/8.6869`, Accessed on Dec. 1, 2020.