Imperial College of Science, Technology and Medicine
Dyson School of Design Engineering

Practically Optimal UAV Mission Planning under Uncertainty

Qiuchen Qian

Submitted in part fulfillment of the requirements for
the degree of Doctor of Philosophy and Diploma at
Imperial College London January 2025

Abstract

While Unmanned Aerial Vehicle (UAV) technologies have transformed industrial applications, the practical utility of current mission planners is limited by oversimplified assumptions about operational costs and spatial constraints. The central challenge remains in developing planners that can effectively handle energy consumption uncertainty and spatial optimization requirements while maintaining computational efficiency for online deployment on resource-constrained platforms. This thesis addresses this gap, focusing on UAV-enabled wireless power transfer and communication, by showing that combining probabilistic energy estimation with spatial-aware approaches enables robust mission planning under real-world operational constraints.

We develop two complementary frameworks. First, to manage energy uncertainty, we formulate an Uncertain and Dynamic Orienteering Problem (UDOP), introducing the Rapid Online Metaheuristic-based Planner (ROMP) that combines first principle analysis and wind field segmentation, and then advancing it with an ADaptive Approach for Probabilistic paThs (ADAPT). This framework employs a Bayesian method for real-time energy consumption estimation, adapting to dynamic factors like wind. Our experimental results demonstrate that ADAPT achieves a 100% mission success rate across all tested scenarios while maintaining comparable solution quality and computation time.

Second, for spatial optimization, we propose the Close Enough Orienteering Problem with non-uniform neighborhoods (CEOP-N) and solve it with the CRaSZe-AntS algorithm. This hybrid metaheuristic features a Randomized Steiner Zone Discretization (RSZD) scheme that identifies overlapped sub-regions for Particle Swarm Optimization (PSO) to refine continuous waypoint positioning, while an Ant Colony System (ACS) optimizes the discrete visiting sequence. Results show CRaSZe-AntS significantly outperforms single-neighborhood strategies, increasing prize collection by an average of 140.44% and reducing computation time by 55.18%. We extend this approach with CRaSZe-AntS-3D to efficiently handle non-convex geometric constraints in air-to-ground communication scenarios, achieving near-optimal solutions with

only a 0.38% deviation from the optimal energy cost.

Declaration of Originality

I declare that this thesis was composed by myself, that the work contained here is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified. All else from other published or unpublished work has been appropriately referenced.

Publications and unpublished work included in this thesis:

- Qian, Q., Wang, Y., & Boyle, D. (2025). Adaptive Probabilistic Planning for the Uncertain and Dynamic Orienteering Problem. IEEE Internet of Things Journal.
- Qian, Q., Wang, Y., & Boyle, D. (2024). On Solving Close Enough Orienteering Problems with Overlapped Neighborhoods. European Journal of Operational Research.
- Qian, Q., O'Keeffe, J., Wang, Y., & Boyle, D. (2023). Practical Mission Planning for Optimized UAV-Sensor Wireless Recharging. arXiv preprint arXiv:2203.04595.

Other related publications:

- Wang, Y., Qian, Q., & Boyle, D. (2024). Probabilistic Constrained Reinforcement Learning with Formal Interpretability. In Forty-first International Conference on Machine Learning.
- Wang, Y., O'Keeffe, J., Qian, Q., & Boyle, D. (2023). Quadue-ccm: Interpretable distributional reinforcement learning using uncertain contraction metrics for precise quadrotor trajectory tracking. In Conference on Robot Learning (pp. 2306-2316). PMLR.
- Wang, Y., O'Keeffe, J., Qian, Q., & Boyle, D. (2022). Interpretable stochastic model predictive control using distributional reinforced estimation for quadrotor tracking systems. In 2022 IEEE 61st Conference on Decision and Control (CDC) (pp. 3335-3342). IEEE.

- Wang, Y., O'Keeffe, J., Qian, Q., & Boyle, D. (2022). KinoJGM: A framework for efficient and accurate quadrotor trajectory generation and tracking in dynamic environments. In 2022 International Conference on Robotics and Automation (ICRA) (pp. 11036-11043). IEEE.

- Qian, Q., Pandiyan, A. Y., & Boyle, D. E. (2021). Optimal Recharge Scheduler for Drone-to-Sensor Wireless Power Transfer. IEEE Access, 9, 59301-59312.

Copyright Declaration

The copyright of this thesis rests with the author. Unless otherwise indicated, its contents are licensed under a Creative Commons Attribution-Non Commercial 4.0 International Licence (CC BY-NC). Under this licence, you may copy and redistribute the material in any medium or format. You may also create and distribute modified versions of the work. This is on the condition that: you credit the author and do not use it, or any derivative works, for a commercial purpose. When reusing or sharing this work, ensure you make the licence terms clear to others by naming the licence and linking to the licence text. Where a work has been adapted, you should indicate that the work has been changed and describe those changes. Please seek permission from the copyright holder for uses of this work that are not included in this licence or permitted under UK Copyright Law.

Acknowledgements

First and foremost, I would like to express my deepest gratitude to my Ph.D. adviser, Dr. David Boyle, for his invaluable support, guidance, and inspiring mentorship throughout my Ph.D. journey. What I value most is how he struck that perfect balance - giving me the freedom to pursue new ideas while keeping me anchored to the core objectives of our research. His rigorous standards and thoughtful guidance have been invaluable, and this thesis simply would not exist without his support.

My research has been immeasurably enriched by collaborations with Yanran Wang and Ivan Revenga Riesco. Our discussions about optimization algorithms and system modeling weren't just productive - they were genuinely engaging, often leading to unexpected insights that advanced our work in new directions.

I would like to thank my family and friends for their support. To my parents, Yaping Qian and Wei Zhang, thank you for always trusting me and encouraging my academic pursuits. And to Yunyi Wang - thank you for bringing light and laughter to what could have been a solitary journey.

My gratitude goes to the entire research group for creating such a collaborative and stimulating research environment. Special thanks to Gaoyun Fang, Anna Soligo, Danish Rizvi, Freddie Hong, and Ranya Aloufi for their friendship and academic support throughout this journey.

Looking back, I realize this thesis represents far more than just the words on these pages - it's a milestone that marks both personal and academic growth, made possible by everyone mentioned here and many others. Thank you all for being part of this journey.

Contents

Abstract **i**

Declaration of Originality **iii**

Copyright Declaration **v**

Acknowledgements **vii**

1 Introduction **1**

1.1 Background and Context 1

1.2 Practical Mission Planning Challenges 3

1.3 Research Objectives and Contributions 5

1.4 Thesis Organization 9

2 Optimization of UAV Routing Problems **11**

2.1 Theoretical Foundations of Routing Optimization 11

2.1.1 Classical Routing Models 11

2.1.2 Solution Approaches 13

2.2 Energy management for UAV Routing 15

2.3 Spatial Optimization for UAV Routing 17

3 Hardware Configuration **21**

3.1 Unmanned Aerial Vehicles 21

3.2 Mission-based Hardware Configuration 23

3.2.1 Wireless Power Transfer (WPT) 23

3.2.2 Wireless Communication 24

4 Energy Management for UAV Routing **26**

4.1 Uncertain and Dynamic OP (UDOP) Formulation 26

4.1.1 Charging Scheduling Problem (CSP) Example 27

4.2 ROMP: First-Principle Analysis with Wind Field Segmentation 29

4.2.1 Introduction 29

4.2.2 UAV Power Consumption Model 30

4.2.3 Rapid Online Metaheuristic-based Planner (ROMP) 34

4.2.3.1 Offline Planning Phase 34

4.2.3.2 Online Planning Phase 37

4.2.4 Experiments, Results, and Discussion 40
4.2.4.1 Parameter Setting 40
4.2.4.2 Metaheuristic Options in Google OR-Tools 41
4.2.4.3 Wind Impact on Solution Performance 42
4.2.4.4 Reaction to the Dynamic UAV Residual Energy 43
4.2.5 Conclusion 44
4.3 ADAPT: Bayesian Approach with Prior Linear Regression Model 46
4.3.1 Introduction 46
4.3.2 System Design 47
4.3.2.1 Prior Knowledge and Assumptions 48
4.3.2.2 Adaptive Approach for Probabilistic Paths (ADAPT) 49
4.3.2.3 Inherited Ant Colony System (IACS) 51
4.3.3 Experiments, Results, and Discussion 56
4.3.3.1 Parameter Setting 56
4.3.3.2 Offline Planning with ACS and Gurobi 58
4.3.3.3 Online Planning Phase with a Bayesian Approach 60
4.3.3.4 ADAPT Performance Analysis 61
4.3.4 Conclusion 67

5 Spatial Optimization for UAV Missions 68

5.1 CRaZSe-Ants: on Solving CEOP with Overlapped Neighborhoods 68

5.1.1 Introduction 68

5.1.2 Formulations of Close-Enough Routing Problems 70

5.1.3 Steiner Zone Search Problem 71

5.1.4 CEOP and Uniform Overlapped Neighborhoods 73

5.1.5 CEOP with Non-uniform Neighborhoods (CEOP-$\mathcal{N}$) 76

5.1.6 Randomized Steiner Zone Discretization (RSZD) Scheme 79

5.1.7 Approach to CEOP with Uniform Neighborhoods 81

5.1.8 Approach to CEOP with Non-uniform Neighborhoods 87

5.1.9 Complexity Analysis 91

5.1.10 Experiments, Results, and Discussion 91

5.1.10.1 CEOP with Uniform Neighborhoods 92

5.1.10.2 Overlap of Non-uniform Neighborhoods 101

5.1.11 Limitations of CRaSZe-AntS 107

5.1.12 Conclusion 108

5.2 CRaZSe-AntS-3D: on Wireless Data Collection 109

5.2.1 Introduction 109

5.2.2 Three-dimensional Extension of CEOP-$\mathcal{N}$ 110

5.2.2.1 Data Collection Problem (DCP) Example 111

5.2.3 System Design 117

5.2.4 Experiments, Results, and Discussion 122

5.2.4.1 Parameter Configuration 123

5.2.4.2 Searching within Annulus Space 125

5.2.5 Conclusion and Future Work 126

6 Conclusion and Future Outlook **130**

6.1 Thesis Achievements 130

6.2 Open Challenges and Future Outlook 131

Appendices **149**

A Parallel Search Efficiency of CBHA **150**

B Sensitivity analysis of the minimum safety belief **153**

List of Tables

4.1 Main parameters in ADAPT. 47

4.2 Model coefficient ± bootstrap standard error (Rodrigues et al., 2022). 48

4.3 Offline solution performance comparison. 58

4.4 Performance of solutions with different offline paths. 59

4.5 Solution quality comparison. 65

5.1 Notation in formulations of Close-Enough Routing Problems. 70

5.2 Scores achieved with varying ACS settings 93

5.3 Performance summary of IPD-ACS and RSZD-ACS (first discretize the CEOP and then solve the SOP). 95

5.4 Performance summary of GRASP and CRaSZe-AntS for CEOP. 99

5.5 Performance summary of BE-PSO-IACS and CRaSZe-AntS for TDDP. 105

5.6 PHY configuration parameters and corresponding minimum SNR requirements for different data rates, established for a target BER of 10^{-4}. 124

B.1 Sensitivity analysis of safety belief Θ. 154

List of Figures

2.1 Linear regression models, i.e., Reg-Model-R (Rodrigues et al., 2022) and Reg-Model-A (Alyassi et al., 2023), for estimating real-time power of DJI M100 (DJI, 2016). 16

3.1 Main block diagram of a quadcopter. 22

4.1 Drone to sensor Inductive Power Transfer block diagram (Arteaga et al., 2023). . 28

4.2 An example 2D quiver plot of wind speed over a lake, where the UAV traverses in a wind cube region Ω, formed by eight vertices A, B, C, D, E, F, G, H. The wind velocity vector of Ω is averaged over eight wind velocity vectors at these vertices. 30

4.3 An example of wind field predicted by WindNinja (Forthofer et al., 2014). . . . 32

4.4 Vertical motion (a) and horizontal motion (b) of the M100 drone (DJI, 2016). . 33

4.5 System architecture of ROMP. 34

4.6 Offline planning phase of ROMP. 35

4.7 Online planning phase of ROMP. 38

4.8 Execution results of OR-Tools and the optimization solver with different local search options, including Automatic selection (AUTO), Greedy Descent (GD), GLS, Simulated Annealing (SA), and Tabu Search (TS). 42

4.9 An instance showing changes in route sequence and discharged energy after applying a constant west wind. 43

4.10 Flight plan adaptation under simulated energy depletion scenarios. (a) The originally planned route uses an energy capacity of 99.9 Wh. (b) After artificial energy reduction to 50 Wh following five node visits, the first route is re-planned. (c) Second route adjustment following simulated energy drop to 22 Wh after servicing two more nodes. (d) Composite visualization of the complete executed path showing energy states at key decision points, demonstrating ROMP's adaptive re-planning capabilities under simulated emergency conditions. 44

4.11 ADAPT framework. The UAV follows an initial path, sequentially servicing sensor nodes. During flight, the UAV continuously logs power consumption, which informs subsequent planning triggered upon completing the task at each node. This iterative process continues until the UAV returns to base when all target nodes are recharged, or residual energy cannot continue. 47

4.12 An example of how ADAPT updates posterior distributions using online observations. Nine re-plannings happened during this mission, moving from the most diverged distribution (Post1) to the most centralized one (Post9). 60

4.13 An example of ADAPT solving *California20* with $\Delta\mu_{\bar{\mathrm{P}}^*} = 20\%$ and $\Delta\sigma_{\bar{\mathrm{P}}^*} = 0\%$. When the UAV recharged sensors 4 and 15, its residual energy (budget) is 283.08 kJ. Here we show four typical candidate paths of ADAPT with safety belief $\Theta \in \{70, 80, 90, 99\}\%$. 61

4.14 Two examples of algorithms' execution time. The top and bottom scenarios have the lowest and highest problem complexity among all tested scenarios. 62

4.15 Mission success rate over 50 executions. 63

4.16 Online re-planning processes comparison for a typical execution of *California40* with $\Delta\mu_{\bar{\mathrm{P}}^*} = 20\%$ and $\Delta\sigma_{\bar{\mathrm{P}}^*} = -10\%$. Indices for sensor nodes not in the path are hidden. The value after '/' is the budget, and the initial budget is 359.64 kJ. 66

5.1 A Steiner zone of degree three, $|\Omega| = 3$, formed by three circular arcs $\widehat{v_1'v_2'}(c_3)$, $\widehat{v_1'v_3'}(c_2)$, and $\widehat{v_2'v_3'}(c_1)$. The red cross points refer to SZ vertices v_1', v_2', v_3'. 71

5.2 Sweep line algorithm (Wang et al., 2019b). Figure (b) shows an example where the sweep line algorithm may find a wrong SZ of degree three. The interval $[l_2, l_5]$ is registered at t_2, overlapping with $[l_3, l_6], [l_1, l_4]$ and $[l_3, l_4]$. Three circles intersect pairwise, but only form SZs of degree two. 80

5.3 An example of the first three steps of employing RSZD on *concentricCircles1*. . . 80

5.4 Graphical abstract of CRaSZe-AntS. 82

5.5 Examples of the arc search algorithm for SZ of different degrees. 86

5.6 An example of invoking arc search and add operator for five times on *bubbles1* ($\mathscr{B}_d = 340$). At iteration 1, the arc search operator reduced the path cost. This allows the add operator to insert an additional waypoint. 87

5.7 Discretization results of IPD and RSZD. RSZD and IPD have the same worst-time complexity as $\mathcal{O}(N)$, where N is the number of target circles. Here RSZD needs ∼10 times slower computation time than IPD because we set $N_{\mathrm{iter}}^{\mathrm{RSZD}} = 10$. 96

5.8 Paths of the best-known solution and CRaSZe-AntS in solving CETSP instance *bubbles6*. Compared to CRaSZe-AntS, the best-known path exhibits a notably 'straight' and 'smooth' trajectory. Many SZs of degree four form the 'saw-tooth' pattern in the path. Another relates to extreme scenarios, such as four circles intersecting at a single point. 100

5.9 Paths of GRASP and CRaSZe-AntS in solving two CEOP instances. 101

5.10 Convergence processes of ACS and IACS in the first 20 iterations of PSO when solving *team5_499*. Using the information from the last global-best ants, IACS can generally have a lower starting cost (except for the first iteration). 103

5.11 Visualization of BE-PSO-IACS's and CRaSZe-AntS's solution in solving TDDP instance *bubbles1*, with given budget $\mathscr{B}_t = 6.9826$. 106

5.12 The antenna radiation pattern and angle representation. 111

5.13 The non-trivial communication space bounded by the half-power-beam-width angle θ_{HPBW} and maximum communication altitude $H_{\max}$. 114

5.14 Examples of Annulus Zones (AZs). 119

5.15 The lower bound ($h_{\min}$) of a two-degree annulus space. 120

5.16 Examples that the unique intersection point $v_{\text{vtx}}^{\times}$ is outside of convex polygons. . 121

5.17 Directional antenna characteristics showing radiation pattern and elevation angle-dependent absolute gain. The antenna achieves a peak gain of 14.7 dBi at the main lobe. 123

5.18 Line-of-Sight probability characterization using logistic distributions under varying parameters a and b. 123

5.19 Visualization of an annulus space with degree $|\mathbf{\Lambda_H}| = 3$, illustrating the geometric relationship between top and bottom Annulus Zones (AZs). The top AZ $\Lambda h_{\max}$ at 39.79 m is defined by a Steiner Zone with three vertices and constrained by three inner circles, while the bottom AZ $\Lambda_{h_{\min}}$ at 23.52 m converges to a single point above the minimum communication altitude $H_{\min} = 10$ m. 125

5.20 Energy consumption distribution within the annulus space, generated through an exhaustive search, revealing significant spatial variations in operational costs and identifying the global optimum at $(75.94, 80.94, 35.07)$ with minimum energy consumption of 0.7823 kJ. 126

5.21 Progressive evolution of the PSO algorithm utilizing ten particles, demonstrating convergence from randomly initialized positions at various altitudes toward the optimal solution. The final achieved energy cost of 0.7853 kJ closely approximates the global optimum of 0.7823 kJ identified through the exhaustive search. 127

5.22 Comparative visualization of the simplified annulus space $\mathbf{\Lambda_H}$ against the complete annulus space (extending to the nonlinear boundary curve). 128

A.1 Parallel performance summary. 152

Chapter 1

Introduction

1.1 Background and Context

Unmanned Aerial Vehicle (UAV) technologies have transformed numerous industrial applications, such as precision agriculture and infrastructure inspections (Atyabi et al., 2018). Multi-rotor UAVs, in particular, offer more flexibility and mobility compared to traditional ground vehicles, enabling operations in complex three-dimensional spaces while carrying specialized payloads. This capability has revolutionized previously challenging tasks, such as maintaining large-scale agricultural sensor fields, where traditional approaches often require human operators to physically visit each sensor node for battery replacement or data collection - a time-consuming and labor-intensive process. UAVs can autonomously deliver power to sensor nodes through Wireless Power Transfer (WPT) technologies (Suman et al., 2019; Arteaga et al., 2023) while simultaneously collecting sensor data (Polo et al., 2015), dramatically improving maintenance efficiency and scalability.

Efficiently deploying UAVs for such missions, however, introduces complex planning challenges.

At its core, UAV mission planning must optimize multiple competing objectives: maximizing service delivery (such as recharged energy or collected data) while minimizing resource consumption (such as UAV battery usage) under various operational constraints. These constraints broadly manifest through energy management uncertainty and spatial coverage requirements. Field experiments with the DJI Matrice 100 drone illustrate this energy management challenge, showing that wind conditions varying from 2 to 16 knots can cause significant power consumption fluctuations, even at typical operating altitudes of 25-100 meters (Rodrigues et al., 2022). This uncertainty directly impacts mission planning - the UAV must maintain sufficient energy reserves not only for its planned path but also for ensuring safe return to base under adverse conditions. Similarly, spatial constraints emerge from service delivery requirements. For instance, in data collection missions, the UAV must optimize its positioning to balance coverage and performance - on the one hand, the characteristics of the transmitter, receiver, and their antennas jointly determine the maximum possible communication range (Yang et al., 2021); on the other hand, rational use of overlapping service areas can significantly increase mission efficiency (Li et al., 2023). This creates fundamental trade-offs, e.g., hovering at higher altitudes may lead to better Line-of-Sight (LoS) conditions and, thus, extended communication coverage, but at the cost of increased signal attenuation.

These dual energy management and spatial optimization challenges highlight a fundamental gap in current mission planning approaches, which often pursue theoretical optimality but over-simplify the complexities of real-world operations. This limitation may result in solutions that, while mathematically elegant, prove suboptimal or even infeasible in practical implementation. The need therefore exists for robust mission planning approaches that can explicitly address these real-world operational challenges while maintaining computational tractability for deployment on resource-constrained UAV platforms.

1.2 Practical Mission Planning Challenges

The deployment of UAVs in autonomous missions has highlighted energy management and spatial optimization challenges, fundamentally shaping how we approach mission planning for practical applications.

Energy management has become crucial as UAVs operate under strict battery constraints while facing diverse environmental factors that create uncertainties in power consumption. This is significant in mission-critical applications where energy depletion could lead to operational failures or emergency landings. The Orienteering Problem (OP) model (Golden et al., 1987) naturally aligns with these energy management challenges - its inherent budget constraint effectively models battery limitations, and its prize-collection objective captures mission objective optimization. While recent research has extended classic OPs to incorporate dynamic and stochastic attributes, e.g., (Wang et al., 2019a; Angelelli et al., 2021; Thayer and Carpin, 2021), a notable research gap remains in modeling the complex interplay between uncertain travel costs and their impacts on mission objectives in UAV operations.

To address this gap, we propose the Uncertain and Dynamic Orienteering Problem (UDOP), a new formulation that better approximates the complexities of energy-constrained UAV routing. UDOP advances beyond existing models in two key aspects. First, it handles edge costs that follow distributions with unknown and time-variant parameters, reflecting the fact that power consumption patterns often cannot be fully characterized a priori. Second, UDOP explicitly models the interdependencies between travel edge costs, collectible prizes, and prize-collection costs - a crucial consideration when remaining battery capacity directly constrains operational capabilities. To demonstrate the practical utility of UDOP, we examine the Charging Scheduling Problem (CSP) as a representative application. The CSP presents a challenging scenario in which the charging process consumes substantial battery energy, creating a more restrictive

routing environment than traditional missions like delivery or surveillance. This requires careful optimization of travel paths and charging operations to maintain energy safety while maximizing mission effectiveness.

We develop adaptive approaches that combine power consumption estimation (based on a first-principle analysis of UAV aerodynamics or a Bayesian approach with pre-trained models and online updates) and efficient route planning strategies. Our solution emphasizes maintaining mission safety under uncertainty while maximizing operational efficiency, providing a practical framework for energy-aware UAV routing in dynamic environments.

Spatial optimization presents compelling challenges as UAVs operate in three-dimensional space. Though this spatial flexibility offers significant operational advantages, it introduces geometric optimization requirements, particularly when missions must be performed within specific spatial constraints, such as truck-and-drone delivery (Boysen et al., 2018) and wireless data collection (Samir et al., 2020). Current spatial-aware routing models, particularly Close-Enough Routing Problems (Mennell, 2009), have limitations in addressing these aspects of UAV operations. Real-world missions require optimization models that simultaneously handle variable service priorities and spatial boundaries caused by environmental constraints. Moreover, UAVs can often service multiple targets from a single position. This capability creates new opportunities for mission efficiency but requires other computational approaches that are not well-studied in conventional routing problems like (Gao, 2020; Cariou et al., 2023a).

To tackle these challenges, we develop the Close Enough Orienteering Problem with Non-uniform Neighborhoods (CEOP-$\mathcal{N}$). It explicitly considers the benefit of overlapped service regions when having non-linear cost functions, transforming this traditionally challenging aspect into an optimization advantage. To demonstrate the practical application of our model, we first examine a truck-and-drone delivery scenario, then move to a more challenging wireless

communication scenario[1]. These case studies illustrate how our spatial optimization formulation addresses two-dimensional and three-dimensional planning challenges while incorporating real-world energy limitations and service requirements constraints.

1.3 Research Objectives and Contributions

Current frameworks for UAV mission planning mainly focus on constructing theoretical models that can be optimally solved through various relaxations, such as simplified models for UAV energy consumption and wireless communication (Zeng et al., 2019; Ghorbel et al., 2019; Feng et al., 2022; Lin et al., 2023; Pandiyan et al., 2021; Pasha et al., 2022a). While these approaches have advanced our mathematical understanding, there remains significant gaps between theory and practical application.

In energy management, existing approaches primarily rely on predetermined probability distributions or simplified assumptions about environmental conditions. While these simplifications enable mathematical tractability, they prove inadequate for real-world UAV operations where energy consumption patterns vary unpredictably due to uncertain environmental factors, as stated in Section (2.2). The absence of adaptive frameworks that can continuously refine energy estimates during mission execution severely limits the practical utility of current solutions.

In spatial optimization, current approaches either focus on single-neighborhood optimization or employ basic overlapped region strategies, as stated in Section (2.3). Briefly, the single-neighborhood approach, while guaranteeing optimality, fails to capitalize on opportunities where multiple targets can be served from a single position. Existing overlap-based strategies, though

[1]It is necessary to highlight that three-dimensional Close-Enough Routing Problems have been addressed in the literature (Mennell, 2009; Coutinho et al., 2016). However, the sphere constraint is inaccurate in the UAV-assisted data collection scenarios, where wireless communication conditions can significantly affect the communication range, forming a non-convex feasible region (see Section (5.2.2.1)).

more applicable in practice, face limitations in constructing optimal Steiner Zone layouts and often unnecessarily constrain the search space. Furthermore, both approaches assume uniform service quality within target neighborhoods, ignoring the spatial variations in operational effectiveness that characterize real-world scenarios, such as signal attenuation or risk exposure.

These limitations motivate our research question: Is it possible to design a mission planner that **(1)** can effectively handle uncertain and dynamic energy consumption patterns in real-time mission execution and **(2)** can efficiently optimize service quality within application-specific spatial constraints? Our research makes several key contributions toward addressing this question.

For **Research Objective (1)**

- We formulate the Uncertain and Dynamic Orienteering Problem (UDOP Section (4.1)), a novel model that explicitly accounts for edge costs following distributions with dynamic and unknown parameters. UDOP advances beyond existing models by capturing the complex interactions between uncertain travel costs and their impacts on mission objectives. We demonstrate UDOP's effectiveness through the Charging Scheduling Problem (CSP Section (4.1.1)), simulating an energy management critical scenario.

- We develop a lightweight power consumption model that combines first-principle analysis with wind field segmentation. This reduced-order model estimates instantaneous power requirements across different flight regimes while remaining computationally tractable on resource-constrained UAVs. Based on this model, we propose the Rapid Online Metaheuristic-based Planner (ROMP[1] Section (4.2)). This hybrid offline-online framework enables online path refinement through a dynamic drop-and-add operator.

- We advance ROMP through the **AD**aptive **A**pproach for **P**robabilistic pa**T**hs (ADAPT[2] Section (4.3)), which addressed ROMP's limitations of need for extra sensors and modeling

[1] `https://github.com/sysal-bruce-publication/ROMP.git`
[2] `https://github.com/sysal-bruce-publication/Uncertain-Dynamic-OP.git`

knowledge. ADAPT incorporates a Bayesian approach to estimate the UAV's average power consumption during flight, achieving 100% mission success rate across all tested scenarios while maintaining comparable solution quality and computation time. This represents a significant improvement over alternative approaches (including ROMP) that show unstable performance under challenging conditions.

For **Research Objective (2)**

- We formulate the Close Enough Orienteering Problem with Non-uniform Neighborhoods (CEOP-$\mathcal{N}$, Section (5.1.5)), a novel model that explicitly handles non-uniform spatial variations in service quality that characterize real-world UAV operations. We demonstrate CEOP-$\mathcal{N}$'s effectiveness through two case studies: a straightforward two-dimensional Truck-and-drone Delivery Problem (TDDP Section (5.1.5)) for ease of understanding and a novel three-dimensional Data Collection Problem (DCP Section (5.2.2.1)) that jointly optimizes UAV hovering positions and travel paths. DCP explicitly accounts for the benefits of overlapping space while respecting practical constraints on UAV energy and communication capabilities.

- We develop the Randomized Steiner Zone Discretization (RSZD Section (5.1.6)) scheme to discretize the CEOP and CEOP-$\mathcal{N}$ graph with SZs characterized by multiple circular arcs. Our empirical findings indicate that RSZD outperforms the Internal Point Discretization (IPD) scheme (Carrabs et al., 2020). For instance, the results of solving a *bubbles5* instance (CEOP) indicate that when utilizing RSZD's layout instead of IPD's, the solver achieves notable enhancements in computational efficiency (approximately 30 times faster) and solution quality (a prize gain of 24.32%).

- We demonstrate the significance of employing the SZ-based strategy in solving CEOPs with overlapped neighborhoods and (non-)uniform prize-collecting cost functions. We

develop an efficient and reliable algorithm (CRaSZe-AntS[1]), integrating RSZD and hybrid metaheuristics. CRaSZe-AntS is constituted of different solvers and local search operators when addressing the uniform and non-uniform scenarios, specifically:

- For uniform CEOP, CRaSZe-AntS employs an Ant Colony System (ACS) solver with an arc search algorithm to optimize the path sequence and waypoint positions generated by RSZD. We compare the performance of CRaSZe-AntS against a relevant Greedy Randomized Adaptive Search Procedure (GRASP) algorithm (Štefaníková et al., 2020). Our experimental findings demonstrate that CRaSZe-AntS can yield solutions with comparable quality within remarkably reduced computation time (-40.6% on average) across large-scale CEOP instances.

- For non-uniform CEOP, CRaSZe-AntS combines a Particle Swarm Optimization (PSO) solver that searches waypoints' positions within SZs and an Inherited ACS (IACS) that converges quicker than the classic ACS. We compare the performance of CRaSZe-AntS against a relevant Hybrid Algorithm (HA) (Yang et al., 2018). Our experimental results show that CRaSZe-AntS can significantly reduce the computation time (-55.18% on average) and improve solution quality (140.44% on average) across all instances tested, compared to the HA that considers a single neighborhood only.

- We propose CRaSZe-AntS-3D (Section (5.2.3)), extending the original CRaZSe-AntS to search within complex three-dimensional space efficiently. The algorithm employs novel spatial search strategies specifically designed for non-convex geometric structures. CRaSZe-AntS-3D demonstrates a promising performance by rapidly finding the near-optimal solution with only 0.38% deviation from the optimal energy cost.

[1] `https://github.com/sysal-bruce-publication/CRaSZe-AntS.git`

1.4 Thesis Organization

Chapter 2 establishes theoretical foundations through a literature review on routing optimization models and their applications to UAV mission planning under uncertainty. Beginning with classical routing problems, it discusses their inherent limitations and solution approaches. Section (2.2) examines the theoretical developments in handling uncertainties and the practical strategies for energy management of UAV operations. Section (2.3) overviews the evolution of Close-Enough routing problems, corresponding solution approaches, and their applications.

Chapter 3 provides technical foundations by examining hardware configurations essential for UAV mission planning. The chapter details general UAV hardware and specific configurations required for Wireless Power Transfer and Wireless Communication applications.

Chapter 4 addresses our first research objective through two complementary approaches. Section (4.1) first presents the Uncertain and Dynamic Orienteering Problem formulation and its application to the Charging Scheduling Problem. Section (4.2) introduces the analytical approach based on first-principle analysis of UAV aerodynamics and wind field segmentation. Section (4.3) introduces the empirical approach utilizing a Bayesian approach on real-time power observations.

Chapter 5 addresses our second research objective. Section (5.1) first introduces the Close Enough Orienteering Problem formulations with Uniform and Non-uniform Neighborhoods and a straightforward Truck-and-Drone Delivery Problem (TDDP). It then examines the performance of the proposed CRaSZe-AntS algorithm through classic problems and the TDDP example in later sections. Section (5.2) details a three-dimensional Data Collection Problem, presenting CRaSZe-AntS-3D for handling complex geometric structure constraints characterized by practical communication devices and environments.

Chapter 6 synthesizes our research contributions, discussing key challenges addressed and

findings obtained, and outlining future research directions for advancing practical UAV mission planning under uncertainty.

Chapter 2

Optimization of UAV Routing Problems

This chapter reviews routing optimization models and their applications to UAV mission planning under uncertainty. We examine the theoretical foundations of routing problems, their mathematical formulations, and corresponding solution approaches. Particular attention is given to energy-aware mission planning strategies and spatial optimization techniques that address operational constraints in practical UAV applications.

2.1 Theoretical Foundations of Routing Optimization

2.1.1 Classical Routing Models

The foundation of UAV routing optimization traces its origins to the Traveling Salesman Problem (TSP), first studied by Schrijver (2005). The TSP aims to determine the shortest

path that begins at a designated depot, visits each specified location exactly once, and returns to the initial depot. This problem can be formally represented using a complete graph $\mathbf{G} = \{\mathbf{V}, \mathbf{E}\}$, where $\mathbf{V} = \{v_1, ..., v_N\}$ denotes the set of target nodes and $\mathbf{E} = \{e_{i,j}, ...\}$ represents the edge connecting these nodes. TSP employs Euclidean distance as its cost metric $\mathscr{C}_d\big(v_i,\ v_j\big) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$. Regular TSP assumes an identical starting and ending depot, which we denote by v_0. With binary decision variables $X_{i,j}$ indicating edge selection, the TSP can be formulated as an Integer Linear Program (ILP) (Miller et al., 1960a):

$$\textbf{(TSP)} \qquad \min \sum_{i=0}^{N} \sum_{j=0}^{N} \mathscr{C}_d\big(v_i,\ v_j\big) \cdot X_{i,j} \tag{2.1a}$$

$$\textbf{s.t.} \quad \sum_{\substack{i=0 \\ i \neq k}}^{N} X_{i,k} = \sum_{\substack{j=0 \\ j \neq k}}^{N} X_{k,j} = 1 \qquad k = 1, ..., N \tag{2.1b}$$

$$\sum_{v_i \in \mathbf{S}} \sum_{v_j \in \mathbf{S}} X_{i,j} \leq |\mathbf{S}| - 1, \qquad \forall\, \mathbf{S} \subset \mathbf{V},\ |\mathbf{S}| \geq 3 \tag{2.1c}$$

Objective (2.1a) minimizes the traveling cost of the path. Constraint (2.1b) ensures exactly one edge enters each node, and exactly one edge leaves it. Constraint (2.1c) prevents subtours according to (Gendreau et al., 1998). Note that the practical implementation of Constraint (2.1c) may be challenging due to the exponential number of subtour constraints.

While the TSP provides a fundamental framework for routing problems, it exhibits three significant limitations for practical UAV applications. First, its requirement for identical start and end depots constrains operational flexibility. Second, its 'unconstrained' nature makes any sequence permutation feasible, disregarding practical resource limitations. Third, it assumes uniform importance across all target locations, ignoring mission priorities. The Orienteering Problem (OP) (Tsiligirides, 1984) addresses these limitations by introducing prize collection objectives and budget constraints. Each target node v_i carries an associated prize value $\mathscr{P}(v_i)$. We denote the starting depot by v_0 and the ending depot by v_{N+1} with zero prizes $\mathscr{P}(v_0) = \mathscr{P}(v_{N+1}) = 0$.

Following TSP, OP can be formulated as an ILP (Vansteenwegen et al., 2011):

$$\textbf{(OP)} \qquad \max \sum_{i=0}^{N} \sum_{j=1}^{N} \mathscr{P}(v_j) \cdot X_{i,j} \tag{2.2a}$$

$$\textbf{s.t.} \quad \sum_{j=1}^{N+1} X_{0,j} = \sum_{i=0}^{N} X_{i,N+1} = 1 \tag{2.2b}$$

$$\sum_{\substack{i=0 \\ i \neq k}}^{N+1} X_{i,k} = \sum_{\substack{j=0 \\ j \neq k}}^{N+1} X_{k,j} \leq 1 \qquad k = 1, ..., N \tag{2.2c}$$

$$\sum_{i=0}^{N} \sum_{j=1}^{N+1} \mathscr{C}_d\big(v_i,\, v_j\big) \cdot X_{i,j} \;\leq\; \mathscr{B}_d \tag{2.2d}$$

Constraint (2.1c)

Objective (2.2a) maximizes the collected prizes of all visited target circles. Constraint (2.2b) ensures the path starts at depot v_0 and ends at depot v_{N+1}. Constraint (2.2c) provides the connectivity of the path and enforces that one target node is visited at most once. Constraint (2.2d) implies the path distance cost must not exceed a given distance cost budget $\mathscr{B}_d$.

2.1.2 Solution Approaches

While the ILP models for TSPs and OPs formally define the routing problems, their NP-hard complexity (Jünger et al., 1995; Golden et al., 1987) means that finding optimal solutions is computationally intractable for all but small-scale instances. This fundamental challenge has given rise to a spectrum of solution methodologies, each offering a distinct trade-off between solution quality and computational cost.

Exact methods, including dynamic programming (Bellman, 1962; Gonzales, 1962), Branch-and-Bound (BnB) algorithm (Little et al., 1963; Fischetti et al., 1998) and Integer Programming approaches (Miller et al., 1960b; Leifer and Rosenwein, 1994), guarantee optimality through

systematic exploration of the solution space. These methods typically start from an initial node and construct the complete path by evaluating all promising possibilities, employing various techniques to reduce the search space based on optimality conditions. However, due to computational constraints, their practical application remains limited to small-scale problems.

Approximation algorithms offer an alternative approach, providing polynomial-time solutions with provable performance guarantees relative to optimal solutions. While theoretically significant, these algorithms often face practical limitations due to large approximation ratios and implementation complexity. Notable contributions include algorithms based on tour-to-tour improvements (Reiter and Sherman, 1965; Lin, 1965), which iteratively enhance an initial tour through designed strategies such as interchange operations until meeting termination criteria.

While not guaranteeing optimality, heuristic approaches are widely employed as they can efficiently find solutions, even for large-scale problems. The "high quality" of these solutions is typically supported by extensive empirical studies where results are compared against known optima on benchmark instances to measure the optimality gap, or benchmarked against other state-of-the-art methods to show competitive performance. These methods include constructive heuristics that build solutions from scratch (such as nearest neighbor approaches), improvement heuristics that enhance existing solutions (like k-opt operations (Croes, 1958)), and stochastic metaheuristics frameworks. Metaheuristics, notably Genetic Algorithm (GA) (Potvin, 1996; Tasgetiren and Smith, 2000), Ant Colony Optimization (ACO) (Wang and Han, 2021; Liang and Smith, 2006), have demonstrated particular effectiveness in handling complex variants of TSPs and OPs (Toaza and Esztergár-Kiss, 2023), adapting readily to various operational constraints while maintaining computational tractability.

2.2 Energy management for UAV Routing

Integrating energy considerations into UAV routing has emerged as a critical research direction, driven by the practical limitations of UAV battery capacity and the need for reliable mission execution. Research on OPs with uncertain attributes has gained increasing attention due to its direct applicability to real-world scenarios. Significant contributions include the OP with Stochastic Travel and Service times by Campbell et al. (2011), which employs dynamic programming to address cases with identical distribution patterns in travel and service time. The Dynamic and Probabilistic OP (Angelelli et al., 2021) incorporates visitation probabilities and time window constraints, solved through various heuristics including static approximation and Monte Carlo sampling. Lau et al. (2012)'s Dynamic Stochastic OP advances this framework by considering discrete travel time distributions that vary with arrival time, addressed using a BnB algorithm with local search operators. However, these models share a common limitation as they rely on a priori known probability distributions for uncertain elements. This assumption proves problematic in practical applications where distribution parameters are unknown and evolve dynamically, necessitating new approaches for more complex scenarios such as our Uncertain and Dynamic OP detailed in Section (4.1).

Research on UAV energy management has primarily focused on extending operational endurance through various charging strategies, e.g., mobile utility vehicles (Liu et al., 2024), charging UAVs (Xue et al., 2023), and static charging stations (Yang et al., 2022). However, the critical aspect of optimizing mission efficiency and safety from a planning perspective remains underexplored. While some studies have attempted to incorporate energy considerations into mission planning, they often rely on simplified assumptions. For instance, Li et al. (2023) employed fixed energy consumption rates for hovering and travel in their data collection optimization. While Thibbotuwawa et al. (2019) incorporated weather forecasts into the UAV fleet planning, their assumption of constant wind conditions within time windows limits adaptation

to dynamic variations.

Evers et al. (2014b) made significant progress with their Robust OP (ROP) formulation using uncertainty sets for edge costs and node prizes, highlighting the importance of accurate cost modeling. However, precise energy estimation remains challenging in UAV operations. Recent research has advanced this area through wind dynamics integration (Zhang et al., 2021) and data-driven approaches (Rodrigues et al., 2022; She et al., 2020). Notable work on the DJI M100 drone (DJI, 2016) demonstrates two contrasting approaches to energy estimation: Alyassi et al. (2023)'s linear regression model (Reg-Model-A) incorporating detailed flight dynamics, and Rodrigues et al. (2022)'s simpler model (Reg-Model-R) based on hover power. Our comparison (see Figure (2.1)) using actual M100 flight data (Rodrigues et al., 2021) shows that Reg-Model-R achieves lower overall error while Reg-Model-A better tracks power variations. Moreover, while Reg-Model-R offers practicality when detailed flight parameters are unavailable, it may underperform with limited training data or when applied to smaller UAVs more susceptible to

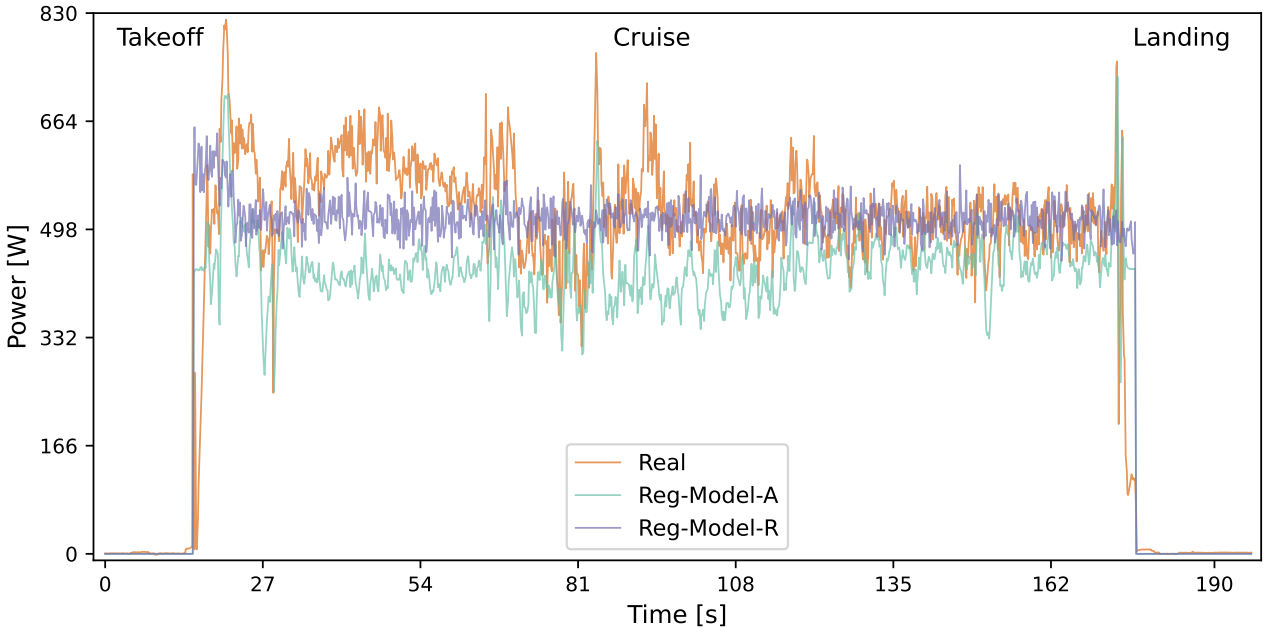


Figure 2.1: Linear regression models, i.e., Reg-Model-R (Rodrigues et al., 2022) and Reg-Model-A (Alyassi et al., 2023), for estimating real-time power of DJI M100 (DJI, 2016).

wind effects. We address these limitations by extending this approach with a Bayesian approach for adaptive distribution updates (see Section (4.3)).

2.3 Spatial Optimization for UAV Routing

In practical UAV applications, missions often do not require the vehicle to visit exact target locations - it is sufficient to come within a certain proximity of each target to accomplish the task. This reality gives rise to Close-Enough routing problems (Mennell, 2009), a class of optimization problems where service can be completed when the agent reaches any point within a specified neighborhood around each target. For instance, in wireless communication scenarios, a UAV can establish reliable data links within the ground nodes' communication range (Chen et al., 2021; Sun et al., 2023). Though this spatial flexibility in mission execution offers more freedom in path planning, it also introduces additional complexity in determining optimal waypoint positions. Research in spatial optimization for Close-Enough routing problems has evolved through two primary approaches. The first focuses on optimizing waypoints within individual service regions, while the second leverages the geometric properties of overlapped neighborhoods.

The single-neighborhood strategy optimizes the path's waypoints within individual service regions. Carrabs et al. (2020) introduced a discrete point scheme coupled with Second-Order Cone Programming (SOCP), establishing a framework for exact solution finding. Yang et al. (2018) adopted Particle Swarm Optimization (PSO) with a Boundary Encoding (BE) scheme for optimizing waypoint positions on the boundary of arbitrary neighborhoods and a Genetic Algorithm (GA) for optimizing the path sequence and evaluating the fitness of the particles. Similarly, Placido et al. (2022) employed a reverse logic approach, using GA and random sampling to generate a feasible solution and then optimizing continuous positions with the CPLEX

solver. These methods successfully optimize individual waypoints but may not fully capture the benefits of overlapped service regions under UAV mission planning scenarios.

The overlap strategy emerged with Gulczynski et al. (2006)'s pioneering work on CETSP discretization. Their heuristics comprise three primary phases: (1) generation of feasible discrete nodes; (2) construction of a solution path based on the feasible node set by solving a Generalized Traveling Salesman Problem (GTSP) (Henry-Labordere, 1969); (3) local improvements made to the discrete nodes within the solution path. Their noteworthy Steiner Zone Heuristic (SZH) constructs SZs, which represent the intersection of circles (defined in Section (5.1.3)), and examines the discrete nodes' feasibility by checking their intersections with all circles covered by the SZ. This approach was further refined by Mennell (2009) through limited SZ degrees (defined in Section (5.1.3)) and the minimum-angle strategy. For instance, the minimum-angle strategy restricts the insertion circle with a minimum insertion angle threshold. Following the framework in (Gulczynski et al., 2006), Mennell (2009) designates the intersection chord's midpoint as the representative point for the SZ. The path is then improved using another heuristic, known as IP_{phIII}, that iteratively improves the position of a representative point within a certain arc range. However, the enhanced SZH exhibits a slow convergence rate because it iteratively narrows the search range of the arc across all SZs. Wang et al. (2019b) proposed a Variable Neighborhood Search heuristic based on SZs (SZVNS), which replaces the path improvement phase in (Mennell, 2009) with a bisection algorithm. The algorithm identifies the waypoint B on the arc of a SZ (or within a SZ if B falls on line segment $\overline{AB}$) that minimizes the distance of line segments $\overline{AB} + \overline{BC}$, where A, C represent the previous and next waypoints in the path. SZVNS achieved a solution quality comparable to SZH, significantly reducing computational time. Despite their ability to swiftly generate (near-)optimal solutions (compared to the single neighborhood strategy), SZ-based strategies have limitations in addressing CETSP. The SZ strategy can be seen as an approximation of the original graph, but constructing an optimal SZ layout for the input graph remains challenging, as highlighted in (Mennell, 2009; Wang et al.,

2019b). For instance, Mennell (2009) have introduced multiple metrics to regulate the addition of circles to a SZ. SZVNS attempts to iteratively improve layout quality by regenerating three candidate SZ layouts. Additionally, SZs confine the search space of multiple neighborhoods into smaller convex regions, limiting solution quality.

Though SZ-based strategies have been widely employed in addressing CETSP, their application to the CEOP and its variants remains unexplored in the existing literature. GRASP is a notable implementation of a single-neighborhood strategy for CEOP, as detailed by Štefaníková et al. (2020), which utilizes a local search based on the geometric relationship between the circle and the line segment. However, GRASP's efficacy diminishes with increasing problem scales, attributed to its reliance on an 'elite' selection strategy (discussed in Section (5.1.10.1)). Adapting SZ-based strategies for CEOP can effectively reduce the problem complexities by aggregating the prizes of overlapped target circles into a unified SZ. This approach mirrors the discretization process in CETSP, simplifying CEOP to a Set Orienteering Problem (SOP) (Archetti et al., 2018). SOPs can then be efficiently tackled using metaheuristics or exact methods. For example, Pěnička et al. (2019) introduced a Variable Neighborhood Search (VNS) metaheuristic alongside an Integer Linear Programming formulation for SOP. Archetti et al. (2024) proposed another efficient formulation and a branch and bound algorithm for SOP. Local search operators can then enhance the quality of the CEOP's solution by leveraging the SOP's solutions.

Modern UAV mission planning scenarios often demand greater flexibility than traditional CETSP (or CEOP) formulations (detailed in Section (5.1.2)) typically offer: the agent arriving near targets is sufficient to complete the mission. For example, Cariou et al. (2023a) examined CETSP solutions within communication intersections. Given that antenna radiation patterns are not perfectly isotropic, the original CETSP model would yield suboptimal solutions unless penalties are introduced based on the radio wave strength within the circle's

interior. Gao (2020) introduced the Risk and Reward Asset Routing Problem (R^2ARP), which demands a delicate balance between the risk and reward within the neighborhood. These practical examples underscore a fundamental limitation in current approaches: the prevalent focus on boundary searching for neighborhoods (Coutinho et al., 2016; Carrabs et al., 2020; Placido et al., 2022; Faigl, 2019; Placido et al., 2023). While analytically convenient, this emphasis on boundary optimization fails to capture the rich variability of conditions within service regions. Real-world factors such as signal strength degradation and risk exposure often vary continuously throughout a service region, making interior points potentially better than boundary positions.

The growing recognition of these limitations has motivated our development of CEOP with non-uniform neighborhoods (CEOP-$\mathcal{N}$) (see Section (5.1)). This extension explicitly accounts for spatially varying costs and benefits within service regions, providing a more realistic framework for modern UAV applications. CEOP-$\mathcal{N}$ proves promising in three-dimensional wireless communication scenarios (see Section (5.2)), where factors such as antenna orientations and radiation patterns create enormous spatial variations in service quality.

Chapter 3

Hardware Configuration

This chapter introduces the hardware that may be involved in general UAV mission planning, with specific support examples of wireless power transfer and communication.

3.1 Unmanned Aerial Vehicles

This section introduces the key hardware architecture of autonomous quadcopters used for mission planning, referencing the DJI Matrice 100 (M100) (DJI, 2016) as a representative platform. The system's components, as illustrated in Figure (3.1), are organized into flight-critical and mission-level subsystems. At the center of the system are two primary computing units: the UAV flight controller and the UAV companion computer. The flight controller, such as the M100's N1 controller, is responsible for real-time flight stability by processing data from a sensing module, which mainly includes an IMU, barometer[1], and GPS, to translate control inputs into commands for the propulsion system. In parallel, the companion computer, like the

[1]A barometer provides the primary estimate of altitude above sea level, while additional range-finding sensors like LiDAR, vision, or ultrasonic sensors can be employed to provide precise measurements of the height above terrain, enhancing hovering accuracy and altitude hold precision.

DJI Manifold 2-C (DJI, 2019), handles high-level computational tasks such as mission planning and algorithm execution, where its processing power defines the limits of real-time operational complexity.

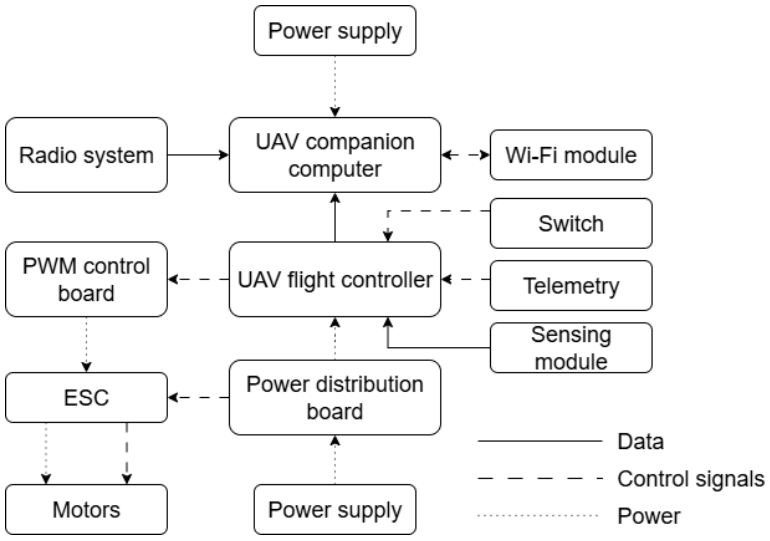


Figure 3.1: Main block diagram of a quadcopter.

The physical flight is managed by the propulsion system, consisting of four sets of motors (typically brushless DC motors), propellers, and ESCs, which receive precise signals from the flight controller to manage the quadcopter's movement. Power for these systems originates from a primary battery, such as the M100's 99.9 Wh TB47D. It is managed by a power distribution board that routes high currents to the ESCs and regulates power to the flight controller. A separate power supply may isolate the companion computer, ensuring its stable operation. This power system's capacity, which affords the M100 approximately 22 minutes of hover time without payload (the net weight of the M100 with a TB47D battery is 2355 g), establishes a fundamental limit on mission duration. Environmental factors, such as temperature and humidity, can significantly impact battery performance, underscoring the importance of considering energy management in UAV routing. These hardware specifications directly inform mission planning parameters. For example, the M100's maximum velocities (5 m/s ascent, 4 m/s descent, 17 m/s cruise under GPS mode) define the feasible solution space for trajectory planning algorithms.

Additionally, the signal transmission system ensures reliable communication between the remote controller and the drone; however, a detailed discussion of this system lies beyond the scope of this hardware overview, as we focus on autonomous operations.

3.2 Mission-based Hardware Configuration

Beyond the general UAV hardware components discussed previously, specific mission applications require additional specialized hardware configurations. We focus mainly on two critical mission applications: wireless power transfer and wireless communication systems.

3.2.1 Wireless Power Transfer (WPT)

In the context of UAV-assisted systems, we primarily focus on scenarios where UAVs serve to extend the operational lifetime of wireless sensor networks rather than scenarios where WPT is used to prolong UAV flight endurance, e.g., (Liu et al., 2024; Xue et al., 2023; Yang et al., 2022). WPT technologies can be categorized into two fundamental approaches: near-field (non-radiative) and far-field (radiative) techniques, each with distinct characteristics.

Far-field techniques, such as Radio Frequency Energy Transfer (RFET), enable power transmission over greater distances. Suman et al. (2019) demonstrates this approach using a Powercast TX91501 transmitter (Powercast, 2024) as the RF source. The system operates with 3 W output power and weighs 454 g, achieving approximately 10 dB path loss at 50 cm distance under agricultural field tests. However, the practical implementation of RFET faces significant challenges in real-world environments, where environmental path loss significantly impacts the system's efficiency. At the same time, UAV payload and energy constraints limit the deployment of high-power transmitters necessary for effective power transfer.

A more practical solution emerges with the Inductive Power Transfer (IPT) systems. Arteaga et al. (2023) presents a high-frequency IPT system where the UAV lands on a designated pad to recharge the supercapacitor bank of sensor nodes through relatively light (250 g) inductive coils. The power transfer chain comprises multiple stages: a DC-DC converter, an inverter operating at 13.5 MHz, an inductive link, a rectifier, and a constant current (CC) charger (Arteaga et al., 2023, Figure 17). The sensor node employs an energy storage system consisting of a bank of 20 SSC-Series 200 F supercapacitors (AVX, 2015) connected in series. Powered by a TB47D battery providing a 22.2 V input, this system achieves a peak receiver power of 30 W with an IPT efficiency exceeding 40%. This configuration forms the basis for our investigation of energy management strategies in UAV charging scheduling problems, detailed in Chapter (4).

3.2.2 Wireless Communication

Wireless communication systems enable information exchange through electromagnetic waves propagating through space. In such systems, the transmitter performs modulation and encoding to convert information into radio waves broadcast via an antenna. The receiver's antenna captures these propagating waves and recovers the original information through corresponding demodulation and decoding.

A practical implementation of such a system can be found in the Ultra-wideband (UWB) based data collection scheme detailed in (Polonelli et al., 2020). This implementation utilizes the Qorvo EVK1000 board (Qorvo, 2016b), which integrates the DW1000 transceiver chip (Qorvo, 2024). The system operates on a MODE3 configuration using channel 2 of the IEEE802.15.4-2011 standard, supporting dual data rates of 0.11 Mbps for ranging operations and 6.8 Mbps for high-speed data transfer. The EVK1000 board is equipped with an omnidirectional antenna, providing uniform gain characteristics with a nearly circular radiation pattern that enables consistent performance regardless of the relative orientation between the transmitter and receiver.

Power consumption analysis of the DW1000 transceiver has been documented in (Polonelli et al., 2022), revealing an average power draw of approximately 150 mW during operation. This power requirement is critical for energy-constrained applications such as battery-powered sensor nodes. Building upon these hardware specifications, we develop a detailed wireless communication simulation framework incorporating practical antenna and channel characteristics. This simulation framework is then applied to analyze and optimize a data collection scenario utilizing UWB techniques, as presented in Section (5.2).

Chapter 4

Energy Management for UAV Routing

4.1 Uncertain and Dynamic OP (UDOP) Formulation

Let $\mathbf{G} = \{\mathbf{V}, \mathbf{E}\}$ be a complete graph with a set of N target nodes $\mathbf{V} = \{v_1, ..., v_N\}$ and corresponding edge set $\mathbf{E} = \{e_{ij}, ...\}$. Each node v_i is characterized by its 3D coordinate (x_i, y_i, z_i) and a time-dependent prize $\mathscr{P}_{f_1}(v_i \mid t)$, $t \in \mathbb{R}^{\geq 0}$. UDOP introduces two distinct cost functions, $\mathscr{C}_{f_2}(v_i,\ v_j \mid t)$ for the travel cost between nodes v_i and v_j when departing from v_i at time t, and $\mathscr{C}_{f_3}(v_i \mid t)$ for the prize-collection cost at node v_i when collection begins at time t. Here, f_1, f_2, and f_3 are 'nominal' functions that can take any continuous form. We denote the start and the end depots by v_0 and v_{N+1}, with $\mathscr{P}(v_0) = \mathscr{P}(v_{N+1}) = 0$ and $\mathscr{C}_{f_3}(v_0 \mid t) = \mathscr{C}_{f_3}(v_{N+1} \mid t) = 0$ respectively. A feasible UDOP path begins at v_0, collects as much prize as possible, and ends at v_{N+1}, subject to a given budget constraint $\mathscr{B}$. Using binary variable $X_{ij} \in \{0, 1\}$ to determine node visitation and continuous time variable t, we formulate

UDOP as a Mixed-Integer Linear Programming (MILP) problem:

$$\textbf{(UDOP)} \qquad \max \sum_{i=0}^{N} \sum_{j=1}^{N} \mathscr{P}_{f_1}\big(v_j \mid t\big)\, X_{ij} \tag{4.1a}$$

$$\textbf{s.t.} \quad \sum_{j=1}^{N+1} X_{0\,j} = \sum_{i=0}^{N} X_{i\,N+1} = 1 \tag{4.1b}$$

$$\sum_{i=1}^{N} X_{ik} = \sum_{j=1}^{N} X_{kj} \leq 1, \qquad k = 2, ..., N \tag{4.1c}$$

$$\sum_{v_i \in \mathbf{S}} \sum_{v_j \in \mathbf{S}} X_{ij} \leq |\mathbf{S}| - 1, \qquad \forall\, \mathbf{S} \subset \mathbf{V},\ |\mathbf{S}| \geq 3 \tag{4.1d}$$

$$\mathscr{P}_{f_1}\big(v_j \mid t\big) = f_1\big(v_j,\, t\big), \qquad j = 1, ..., N,\ t \in \mathbb{R}^{\geq 0}, \tag{4.1e}$$

$$\mathscr{C}_{f_2}\big(v_i,\, v_j \mid t\big) = f_2\big(v_i,\, v_j,\, t\big) \qquad i \neq j = 1, ..., N,\ t \in \mathbb{R}^{\geq 0}, \tag{4.1f}$$

$$\mathscr{C}_{f_3}\big(v_j \mid t\big) = f_3\big(v_j,\, t\big) \qquad j = 1, ..., N,\ t \in \mathbb{R}^{\geq 0}, \tag{4.1g}$$

$$\sum_{i=0}^{N} \sum_{j=1}^{N+1} \mathscr{C}_{f_2}\big(v_i, v_j \mid t_k\big) \cdot X_{ij} + \sum_{i=0}^{N} \sum_{j=1}^{N} \mathscr{C}_{f_3}\big(v_j \mid t_l\big) \cdot X_{ij} \ \leq\ \mathscr{B}, \qquad t_k \neq t_l \in \mathbb{R}^{\geq 0} \tag{4.1h}$$

~~The~~ Objective (4.1a) maximizes the collected prizes. Constraint (4.1b) ensures the path starts at v_0 and ends at v_{N+1}. Constraint (4.1c) maintains path connectivity and restricts each target node to at most one visit. Constraint (4.1d) prevents subtours, ensuring a single continuous path. Constraint (4.1e) confines the prize of v_j must satisfy a continuous function $f_1\big(v_j,\, t\big)$. Constraint (4.1f) enforces the cost to travel from v_i to v_j must satisfy a continuous function $f_2\big(v_i,\, v_j,\, t\big)$. Constraint (4.1f) enforces Constraint (4.1h) stipulates that the total path cost, comprising prize collection and travel costs, must not exceed the given budget $\mathscr{B}$.

4.1.1 Charging Scheduling Problem (CSP) Example

The CSP assigns a single UAV to recharge the maximum energy while ensuring the UAV's safe return to the end depot under uncertain environments. To tackle the UDOP within the CSP

scenario, understanding the analytical form of f_1, f_2, and f_3 is essential. In principle, all three functions can be stochastic and initially unknowable, but we make several assumptions as below to reasonably reduce the problem's complexity based on established research. The recharging process for sensor nodes involves a DC-DC converter, inverter, inductive link, rectifier, and constant current (CC) charger, as illustrated by Figure (4.1).

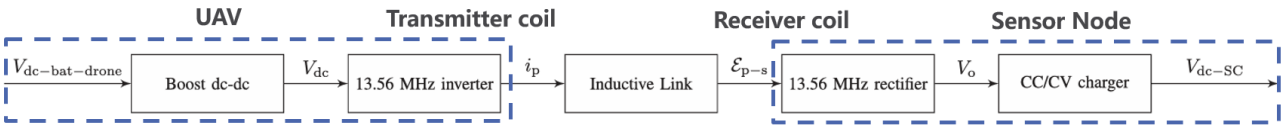


Figure 4.1: Drone to sensor Inductive Power Transfer block diagram (Arteaga et al., 2023).

Because our primary objective is to present effective energy management for UAV routing, we simplify their Inductive Power Transfer (IPT) process using fixed efficiencies, drawing from their experimental results (Arteaga et al., 2023): η_{IPT} for the IPT link and η_{CC} for the constant current (CC) charger. We assume all sensor nodes are identical to those described in (Polonelli et al., 2020) and operate at a constant sampling frequency. This leads to a uniform energy depletion rate, R_{SN}, which linearly increases energy and time requirements for recharging sensor nodes. Based on these assumptions, we define f_1, f_2 and f_3 as follows:

$$\mathscr{P}_{f_1}(v_i \mid t_l) = \frac{1}{2}\,\mathrm{C}\left(\mathrm{V}_{\max}^2 - \mathrm{V}(v_i \mid t_l)^2\right) \tag{4.2a}$$

$$\mathscr{C}_{f_2}(v_i, v_j \mid t_l) = \int_{t_l}^{t_l+t_{\text{tk}}} \mathrm{P}^*_{\text{tk}}(t)\,\mathrm{d}t + \int_{t_l+t_{\text{tk}}}^{t_l+t_{\text{tk}}+t_{\text{cr}}} \mathrm{P}^*_{\text{cr}}(t)\,\mathrm{d}t + \int_{t_l+t_{\text{tk}}+t_{\text{cr}}}^{t_l+t_{\text{tk}}+t_{\text{cr}}+t_{\text{ld}}} \mathrm{P}^*_{\text{ld}}(t)\,\mathrm{d}t \tag{4.2b}$$

$$\mathscr{C}_{f_3}(v_i \mid t_l) = \frac{\mathscr{P}_{f_1}(v_i \mid t_l)}{\eta_{\text{IPT}}} \tag{4.2c}$$

Equation (4.2a) defines a sensor node's chargeable energy as the difference between its maximum energy capacity and current energy level. Equation (4.2b) stipulates the UAV's energy consumption during travel, accounting for three distinct flight regimes: takeoff, cruise, and landing. Equation (4.2c) quantifies the energy needed to recharge a sensor node as the ratio of chargeable energy to the IPT link efficiency.

4.2 ROMP: First-Principle Analysis with Wind Field Segmentation

4.2.1 Introduction

UAVs operating in real-world conditions encounter numerous unpredictable factors, including but not limited to wind gusts, turbulence, and varying atmospheric conditions. These environmental uncertainties significantly impact the UAV's energy consumption patterns and, consequently, its charging capabilities. Therefore, an adaptive online strategy becomes crucial for ensuring successful mission completion by continuously responding to environmental changes and updating the mission parameters accordingly. Accurate modeling of UAV power consumption is essential for mission feasibility assessment under dynamic conditions. The power consumption model is a critical translator between UAV operations and their energy costs, effectively formulating optimization objectives. Efforts to analyze UAV power consumption models in dynamic environments are regularly reported, e.g., (Chen et al., 2018; Liu et al., 2017b; Rodrigues et al., 2021), with wind conditions identified as the primary influencing factor. Notable experimental work by Rodrigues et al. (2021) deployed anemometers and power sensors on the M100 to capture real-time wind patterns and instantaneous power. Their findings demonstrate the feasibility of extending static mission planners to incorporate dynamic updates through time-variant wind data and battery state monitoring. To minimize computational overhead during flight operations, mission re-planning should ideally occur during UAV-sensor interaction periods, which are primarily determined by the charging duration of the target energy storage device.

4.2.2 UAV Power Consumption Model

While first-principle aerodynamic analysis provides accurate power consumption estimates for UAVs, its computational complexity makes it impractical for real-time mission planning. We introduce a novel wind field segmentation approach that significantly reduces computational overhead while maintaining sufficient accuracy for near-optimal mission planning. This approach discretizes the continuous wind field into manageable three-dimensional lattice regions, enabling efficient power estimation during flight. Following (Liu et al., 2017a), our model makes two key assumptions to balance accuracy and computational efficiency:

1. The wind field is discretized into time-variant three-dimensional lattices, where eight vertices form each cubic region Ω. Each vertex v_i' has a wind velocity vector $\mathbf{v}_{\text{wind}}(t, v_i')$, and edge length determines the resolution of the wind field (see Figure (4.2)). Let $\mathbf{x}(t)$ be the coordinate of the UAV at time t. We assume the UAV experiences a constant wind with velocity vector $\mathbf{v}_{\text{wind}}(\mathbf{x}(t)) = \frac{\sum_{i=1}^{8} \mathbf{v}_{\text{wind}}(t, v_i')}{8}$, **iff** $\mathbf{x}(t)$ is in a same region Ω. This allows

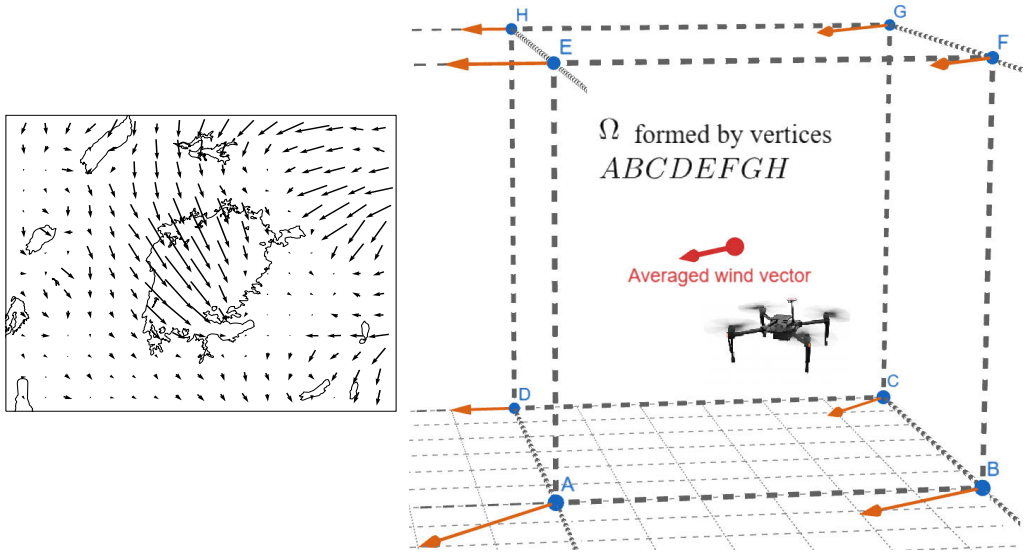


Figure 4.2: An example 2D quiver plot of wind speed over a lake, where the UAV traverses in a wind cube region Ω, formed by eight vertices A, B, C, D, E, F, G, H. The wind velocity vector of Ω is averaged over eight wind velocity vectors at these vertices.

a steady-state motion approximation under force equilibrium conditions, including forces of thrust, gravity, and frictional drag by surrounding airflow (Chen et al., 2018). Wind field forecasts remain valid throughout the relatively short mission duration (typically less than 30 minutes (DJI, 2016; Liu et al., 2017a)), allowing offline computation of initial routes at the base station.

2. UAV operations are categorized into four distinct regimes (takeoff, landing, cruise, and charging), each with specific power requirements derived from first principles (Rodrigues et al., 2022). The UAV is pre-assigned with constant ground speeds $\mathbf{v}_{\text{ground}}$ for takeoff, cruise, and landing when the UAV traverses from the current position to the next.

From first-principle aerodynamics (Rodrigues et al., 2022), the total UAV power consumption can be approximated as:

$$\mathrm{P} \approx \mathrm{P}_i + \mathrm{P}_p + \mathrm{P}_d + \mathrm{P}_a, \tag{4.3}$$

where P_i is induced power overcoming gravity, P_p is profile power handling propeller drag, P_d is parasitic power countering air resistance, and P_a represents constant ancillary power for electronics. The induced power is defined as:

$$\mathrm{P}_i = F_T \cdot \mathbf{v}_i, \tag{4.4}$$

where F_T is the thrust and the induced velocity $\mathbf{v}_i$ can be implicitly computed as (Rodrigues et al., 2022):

$$\mathbf{v}_i = \frac{F_T/(2\rho A_{prop})}{\sqrt{(\mathrm{v}_a^i)^2 + (\mathrm{v}_a^j)^2 + (\mathrm{v}_a^k + \mathbf{v}_i)^2}}. \tag{4.5}$$

Here, ρ is the air density, A_{prop} denotes the overall propeller disk area, v_a^k represents the airspeed vector component flowing perpendicularly to the propeller, and v_a^i and v_a^j are components acting on the propeller plane. The profile power accounts for propeller blade drag (Rodrigues et al.,

2022):

$$\mathrm{P}_p = \frac{4\,c\,c_{d_{blade}}\rho R^4}{8k^{3/2}} F_T^{3/2} + \frac{4\,c\,c_{d_{blade}}\rho R^2}{8k^{1/2}}((\mathrm{v}_a^i)^2 + (\mathrm{v}_a^j)^2) F_T^{1/2}. \tag{4.6}$$

c is the blade chord width, $c_{d_{blade}}$ is the blade drag coefficient, R denotes the propeller radius, and k represents a scaling factor converting rotor angular speed to thrust. The parasitic power P_d handles body frame air resistance (Rodrigues et al., 2022):

$$\mathrm{P}_d = \frac{1}{2}\, C_d^b\, \rho\, A_{ref}^b (\mathbf{v}_a^b)^3. \tag{4.7}$$

C_d^b is the drag coefficient of the drone's body, A_{ref}^b is the reference area, and $\mathbf{v}_a^b$ is the airspeed vector acting on the UAV body frame. Computing instantaneous power consumption during online planning presents significant challenges, particularly in obtaining accurate UAV airspeed and thrust values. These calculations traditionally require computationally expensive wind field predictions and repeated UAV attitude solutions during algorithm evolution. Our wind field segmentation approach approximates wind velocities within discrete three-dimensional regions while maintaining reasonable accuracy, such as Figure (4.3). This segmentation enables efficient

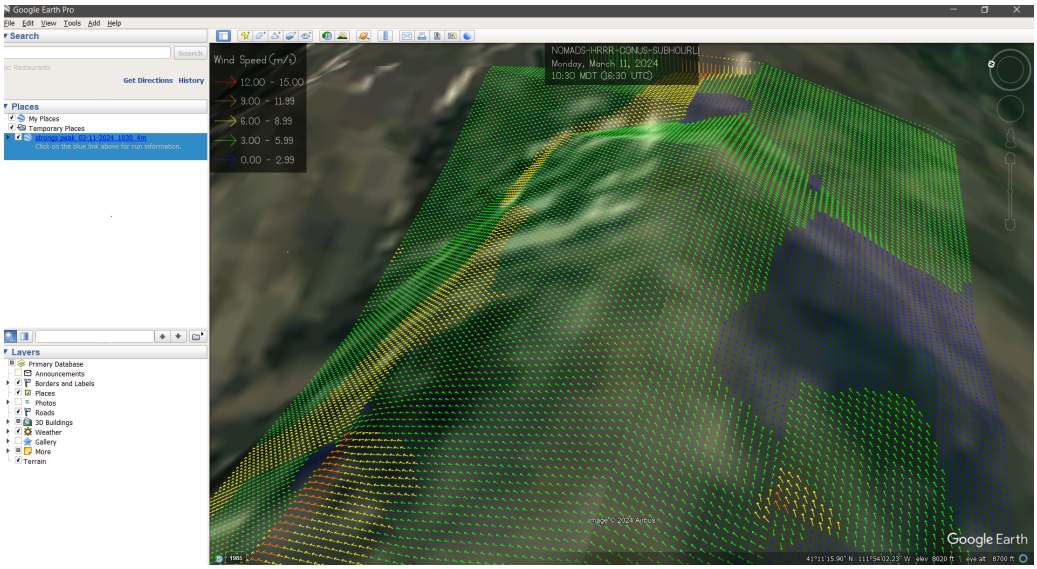


Figure 4.3: An example of wind field predicted by WindNinja (Forthofer et al., 2014).

computation of the required thrust in steady-state conditions:

$$F_T = \begin{cases} F_{D,vert} + F_G, & \text{takeoff} \\ mg - F_{D,vert}, & \text{landing} \\ \sqrt{F_{D,horz}^2 + (mg)^2}, & \text{cruise} \end{cases} \tag{4.8}$$

where thrust requirements are determined by the flight regime and corresponding force equilibrium conditions, as shown in Figure (4.4).

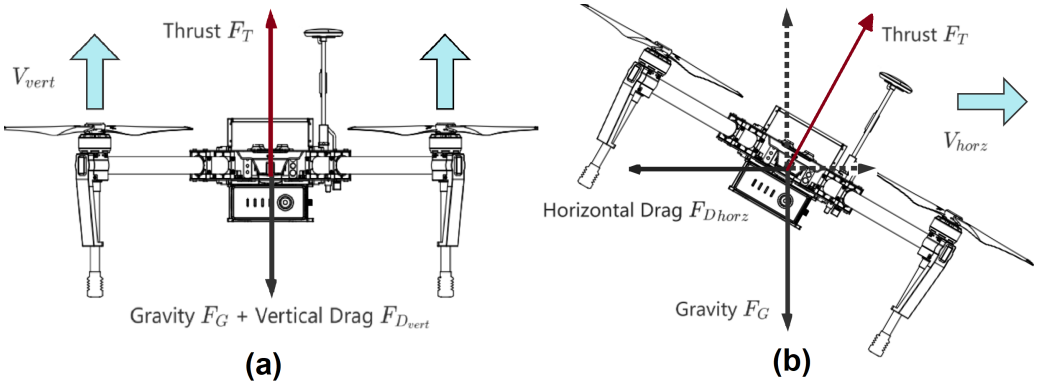


Figure 4.4: Vertical motion (a) and horizontal motion (b) of the M100 drone (DJI, 2016).

Next, we provide the IPT power consumption model. IPT is performed between the coils installed on the UAV and the sensor node. The IPT link efficiency η_{IPT} for each sensor node may vary depending on factors such as the coil diameter, the distance between coils, and disturbances in the air. Although fluctuations in these parameters can impact overall power consumption, they are beyond the scope of this research. Thus, we set a uniform transfer efficiency to simplify computation. Therefore, $\mathrm{E_{IPT}}$ can be derived from the energy requirement of a target sensor node:

$$\mathrm{E_{IPT}} = \frac{C\left(\mathrm{V}_{max}^2 - \mathrm{V}_{now}^2\right)}{2\eta_{IPT}}, \tag{4.9}$$

where C denotes the capacitance, V_{max} and V_{now} are the target sensor's maximum and the present voltage levels.

4.2.3 Rapid Online Metaheuristic-based Planner (ROMP)

ROMP employs a two-stage framework incorporating offline and online phases (see Figure (4.5)). This system maintains autonomy through two key mechanisms: (a) periodic state assessment before the mission, where a UAV collects energy distribution data and estimates energy decay rates for each sensor node; (b) adaptive planning during the mission, triggered either during sensor node charging intervals or when the UAV's remaining energy approaches predefined safety thresholds.

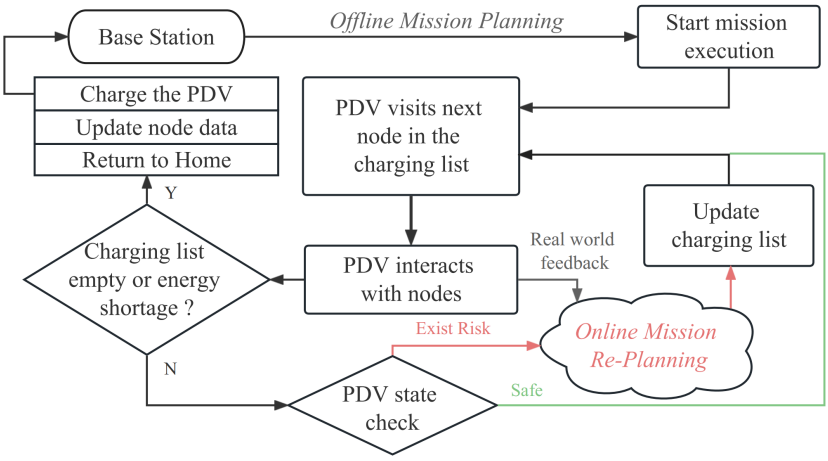


Figure 4.5: System architecture of ROMP.

4.2.3.1 Offline Planning Phase

The offline phase is conducted at the base station, leveraging substantial computational resources to generate a high-quality initial charging plan. The system processes three essential input categories: (a) Sensor node data (Sensor-data): spatial coordinates, sensor specifications, capacitor voltage levels, and charging priorities; (b) UAV data (UAV-data): current position, mission endpoints, and available battery capacity; (c) Environmental data: three-dimensional time-variant wind velocity vectors across the operational space. ROMP's offline component con-

sists of two primary solvers: an *Initial Solver* based on Google OR-Tools (Furnon and Perron, 2024) and an *Optimization Solver* for solution improvement, as visualized in Figure (4.6).

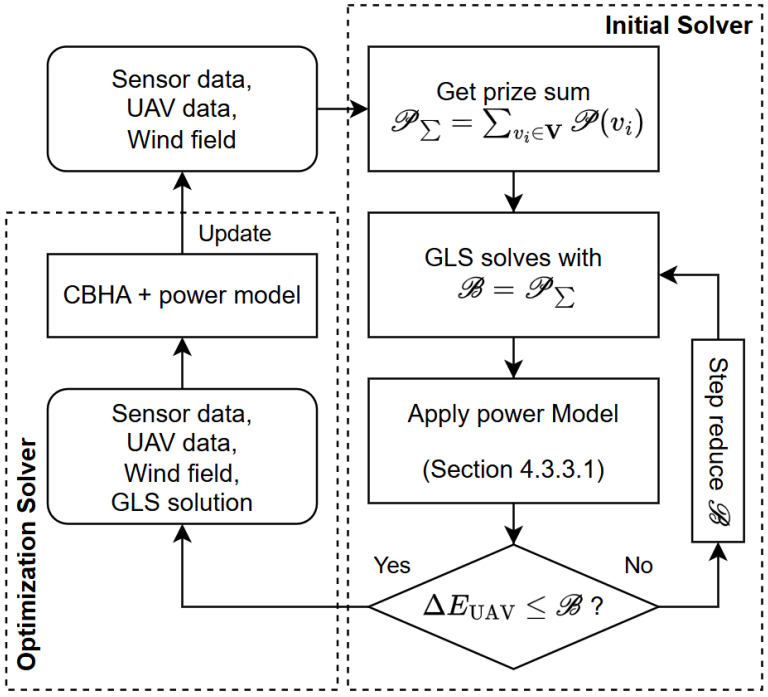


Figure 4.6: Offline planning phase of ROMP.

The initial solver employs a Guided Local Search (GLS) from OR-Tools with a penalty-based mechanism proportional to node prizes (We compare GLS against other heuristics in Section (4.2.4.2)). It begins by calculating the total prize sum $\mathscr{P}_{\sum}$ and iteratively reduces the prize budget $\mathscr{B}$ from $\mathscr{P}_{\sum}$ until identifying the first feasible solution. A speedup strategy is employed to dynamically adjust the budget reduction based on the gap between simulated energy consumption and available capacity.

The optimization solver enhances the initial solution through our Context-aware Black Hole Algorithm (CBHA). Classic BHA (Hatamlou, 2018) represents candidate solutions as particles in the solution space and the best available solution as the *black hole*. The black hole has a probability $\mathcal{P}_a$ to attract other solutions towards it, causing stochastic displacements in the search space. CBHA mainly operates in three phases:

Algorithm 1: Candidate search operator

Input: Feasible node set $\mathbf{A}^v$; Particle path $\mathbf{S} = \{v_1, ..., v_N\}$; Target node $v_i \in \mathbf{S}$; Search number N_{srch}

Record target node v_i's previous node v_{i-1} and next node v_{i+1} in path $\mathbf{S}$;
for *each node* $v_k \in \mathbf{A}^v \setminus \{v_1, v_N, v_{i-1}, v_{i+1}\}$ **do**
    Compute fitness metric $M(v_k)$ by Eq. (4.10) and record pair $\big(M(v_k), v_k\big)$ in list L;
Order L with descending order;
for *each pair* $(M(v_k), v_k) \in L$ **do** add v_k into candidate list L' until $|L'| = N_{\text{srch}}$;
return A candidate L' with $|L'| = N_{\text{srch}}$.

Initialization (Algorithm (1)) generates the population using a probabilistic candidate search operator with a weighted average fitness metric:

$$M(v_j) = W_{re}\Big(\mathscr{P}(v_j) - \mathscr{P}_{\text{lb}}\Big) - W_{de} \cdot \log\Big(\mathscr{C}_d(v_{j-1}, v_j) + \mathscr{C}_d(v_j, v_{j+1})\Big), \quad j = 2, ..., N_{\text{path}} - 1 \tag{4.10}$$

where W_{re} and W_{de} are two weight factors for recharged and discharged energy; $\mathscr{C}_d(v_{j-1}, v_j)$ (or $\mathscr{C}_d(v_j, v_{j+1})$) is the distance between the senor node v_j and the previous (or next) sensor node of v_j in the path. We subtracted $\mathscr{P}(v_j)$ by a constant $\mathscr{P}_{\text{lb}}$ to normalize the weight of prizes. We applied a common logarithm to restrict these two weights within a similar magnitude scale because the prize ranges in $[\mathscr{P}_{\text{lb}}, \mathscr{P}_{\text{ub}}]$, while the distance ranges from meters to thousands of meters. CBHA is designed to prioritize the search for potential sensor nodes with higher prizes, resulting in lower priority assigned to sensor nodes around $\mathscr{P}_{\text{lb}}$.

Attraction (Algorithm (2)) explores search space through position transformation:

$$\mathbf{x}'_i = \mathbf{x}_i + \alpha \cdot (\mathbf{x}_i^{bh} - \mathbf{x}_i), \quad i = 2, ..., N_{\text{path}} - 1 \tag{4.11}$$

where $\mathbf{x}_i$ and $\mathbf{x}_i^{bh}$ are node positions at index i of a particle and a black hole path, respectively. We then approximate the new position $\mathbf{x}'_i$ to the closest node in the feasible sensor node set $\mathbf{A}^v$ to complete this process.

Algorithm 2: Attraction phase

Input: Particle path $\mathbf{S}^i$; Black hole path $\mathbf{S}^{\text{bh}}$

for *each node* $v_j \in \mathbf{S}^i$ **do**
  if *uniform random [0, 1]* $\leq \mathcal{P}_a$ ***and*** $v_i \neq v_i^{bh}$ **then**
    Update new position $\mathbf{x}_i'$ by Eq. (4.11) and confine it to its closest node $v_k \in \mathbf{A}^v$;
    if $v_k \in \mathbf{S}^i$ **then** swap v_j and v_k in $\mathbf{S}^i$;
    else $v_j \leftarrow v_k$;
return The new particle path.

Absorption maintains population diversity through selective re-initialization based on the fitness metric given by:

$$M(\mathbf{S}^i) = \text{penalty} \; + \; W_{re} \, \frac{\mathscr{P}_{re}(\mathbf{S}^i)}{\mathscr{P}_{re}(\mathbf{V})} - W_{de} \, \frac{\mathscr{C}_{de}(\mathbf{S}^i)}{E_{\text{UAV}}(t=0)}, \tag{4.12}$$

where the penalty is negative when the path cannot satisfy the budget constraint; otherwise, it is zero. The second term refers to the ratio of UAV recharged energy to rechargeable energy of all sensor nodes. The third term denotes the ratio of UAV discharged energy to its initial energy, as calculated by Equation (4.3).

4.2.3.2 Online Planning Phase

The online approach (shown as Figure (4.7)) incorporates dynamic updates such as UAV flight time, remaining battery level, and mission progress into the planning process, enabling real-time path adjustments. While our drop-and-add operator builds upon the framework proposed by (Kobeaga et al., 2018), we significantly extend its capabilities for the dynamic CSP context. The key distinction lies in our operator's ability to maintain UAV state information at each waypoint of the adjusted path. This ensures mission feasibility with continuous energy consumption tracking throughout the algorithm and system-level processes.

The online phase begins by re-evaluating the flight plan from the previous computation to

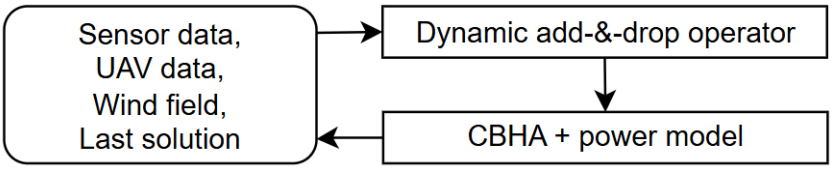


Figure 4.7: Online planning phase of ROMP.

confirm its validity. This 'calibration' step involves re-simulating the previous path using the UAV's current state (position and residual energy) and the latest environmental data. If the previous path cannot comply with the energy budget constraint, ROMP invokes the drop operator iteratively until a feasible solution is reached. The *drop value* to evaluate the behavior of dropping a node v_i (if removed at path index i) is given by:

$$drop(v_i) = \frac{\mathscr{P}_{re}(v_i)}{\int_{t_{v_{i-1}}}^{t_{v_i}} \mathrm{P}\,\mathrm{d}t + \int_{t_{v_i}}^{t_{v_{i+1}}} \mathrm{P}\,\mathrm{d}t - \int_{t_{v_{i-1}}}^{t_{v_{i+1}}} \mathrm{P}\,\mathrm{d}t}. \tag{4.13}$$

where the denominator term refers to the discharged energy difference if node v_i is removed from the path. v_{i-1} and v_{i+1} denote the previous node and next node in the path, respectively. Algorithm (3) demonstrates the drop operator algorithm. Subsequently, the add operator is repeatedly applied until the solution maximizes the allocated energy budget. Specifically, the

Algorithm 3: Dynamic Drop Operator

Input: Particle path $\mathbf{S}^p = \{v_1, ..., v_N\}$; Feasible node set $\mathbf{A}^v$; UAV energy list at each node $L_{\mathrm{E}}^i = \{\mathrm{E}_1, ..., \mathrm{E}_N\}$; UAV flight time list at each node $L_{\mathrm{T}}^i = \{\mathrm{T}_1, ..., \mathrm{T}_N\}$

while $\mathbf{S}^p$ *can* ***not*** *satisfy Constraint (4.1h)* **do**
  for *each node* $v_j \in \mathbf{S}^p$ *(exclude start and end node)* **do**
    Compute drop value $drop(v_j)$ by Eq. (4.13);
  Find the path index $i = \arg\min_j \{drop(v_j), ...\}$ with minimum drop value;
  Update path cost $\mathscr{C}_{de} \leftarrow \mathscr{C}_{de} - \int_{t_{v_{i-1}}}^{t_{v_i}} \mathrm{P}\,\mathrm{d}t - \int_{t_{v_i}}^{t_{v_{i+1}}} \mathrm{P}\,\mathrm{d}t + \int_{t_{v_{i-1}}}^{t_{v_{i+1}}} \mathrm{P}\,\mathrm{d}t$;
  Update path prize $\mathscr{P}_{re} \leftarrow \mathscr{P}_{re} - \mathscr{P}_{re}(v_i)$;
  Remove v_i from $\mathbf{S}^p$ and update feasible node set $\mathbf{A}^v \leftarrow \mathbf{A}^v \cup \{v_i\}$;
  for $n = i$ *to* N **do** update UAV energy $\mathrm{E}_n \in L_{\mathrm{E}}$ and time $\mathrm{T}_n \in L_{\mathrm{T}}$;
return New particle path, UAV energy list, and flight time list.

Algorithm 4: Dynamic Add Operator

Input: Particle path $\mathbf{S}^p = \{v_1, ..., v_N\}$; Feasible node set $\mathbf{A}^v$; UAV energy list at each node $L_{\mathrm{E}}^i = \{\mathrm{E}_1, ..., \mathrm{E}_N\}$; UAV flight time list at each node $L_{\mathrm{T}}^i = \{\mathrm{T}_1, ..., \mathrm{T}_N\}$

while $\exists$ *any node* $v_k \in \mathbf{A}^v$ *inserting to path* $\mathbf{S}^p$ *still satisfies Constraint (4.1h)* **do**
  for *each node* $v_k \in \mathbf{A}^v$ **do**
    Get three closest neighbor nodes $\mathbf{S}_{\mathrm{nbr}} = \{nbr_1, nbr_2, nbr_3\}$ in particle path $\mathbf{S}^p$;
    for *each pair* $(nbr_i, nbr_j) \in \mathbf{S}_{\mathrm{nbr}}$ **do** check "is nbr_i adjacent to nbr_j in path $\mathbf{S}^p$";
    if $\nexists$ *any adjacent pair* **then**
      for $nbr_i \in \mathbf{S}_{nbr}$ **do**
        Find the previous and next node of nbr_i in the path;
        Create new pair (v_{prev}, nbr_i) and (nbr_i, v_{next});
    Find the pair minimizing the visitation cost and insert at index l;
    Compute the add value $add(v_k \mid l)$ by Eq. (4.14);
  Find the node v_j with maximum add value and its insertion index i in the particle path, i.e., $(j, i) = \arg\max_{(k,l)} \{add(v'_k \mid l), ...\}$;
  Update path cost $\mathscr{C}_{de} \leftarrow \mathscr{C}_{de} + \int_{t_{v_{i-1}}}^{t_{v_j}} \mathrm{P}\,\mathrm{d}t + \int_{t_{v_j}}^{t_{v_{i+1}}} \mathrm{P}\,\mathrm{d}t - \int_{t_{v_{i-1}}}^{t_{v_{i+1}}} \mathrm{P}\,\mathrm{d}t$;
  Update path prize $\mathscr{P}_{re} \leftarrow \mathscr{P}_{re} + \mathscr{P}_{re}(v_j)$;
  Insert the node v_j at path $\mathbf{S}^p$'s index i and update feasible node set $\mathbf{A}^v \leftarrow \mathbf{A}^v \setminus \{v_j\}$;
  for $n = i$ *to* N **do** update UAV energy $\mathrm{E}_n \in L_{\mathrm{E}}$ and time $\mathrm{T}_n \in L_{\mathrm{T}}$;
return New particle path, UAV energy list, and flight time list.

add operator (Algorithm (4)) finds the best position for inserting the unvisited node. The *add value* of a node v_j (if inserted at path index i) is given by:

$$add(v_j \mid i) = \frac{\mathscr{P}_{re}(v_j)}{\int_{t_{v_{i-1}}}^{t_{v_j}} \mathrm{P}\,\mathrm{d}t + \int_{t_{v_j}}^{t_{v_{i+1}}} \mathrm{P}\,\mathrm{d}t - \int_{t_{v_{i-1}}}^{t_{v_{i+1}}} \mathrm{P}\,\mathrm{d}t}. \tag{4.14}$$

Note that to guarantee the feasibility of the solution for further optimization, it is necessary to re-simulate the visit of any newly added/dropped sensor node at path index i and all subsequent visits. To better align with the context of the dynamic CSP, we assign two lists to keep track of the energy consumption and the departure time for each visit (see Algorithm (5)).

Algorithm 5: Rapid Online Metaheuristic-based Planner (ROMP)

Input: Last path $\mathbf{S}^{-} = \{v_1, ..., v_N\}$; Sensor-data; UAV-data; Wind field

Calibrate path $\mathbf{S}^{-}$, update UAV energy list and flight time list;
Apply dynamic drop (Alg. (3)) and add operator (Alg. (4)) to path $\mathbf{S}^{-}$;
Initialize N_{pop} populations from path $\mathbf{S}^{-}$ and update their fitness values by Eq. (4.12);
Update black hole $\mathbf{S}^{bh}$ and its fitness metric;
while *not* *reach stopping criteria* **do**
    for *each path* $\mathbf{S}^{i} \in$ *populations* **do**
        Apply attraction phase (Alg. (2)) and 2-opt operator, then update fitness metric;
        if *new fitness metric is better* **then** replace this path and its fitness metric;
    Update black hole $path^{bh}$ and its fitness metric;
    Compute event horizon radius r;
    for *any population fitness metric* $< r$ **do**
        Apply absorption phase and 2-opt operator, then update fitness metric;
    Update black hole $path^{bh}$ and its fitness metric;
return The global-best path $\mathbf{S}^{bh}$, path prize and path cost.

4.2.4 Experiments, Results, and Discussion

This section presents experimental results evaluating both offline and online planning phases. We develop the ROMP implementation in C++20 standard and execute it on a DJI Manifold 2-C onboard computer as detailed in Section (3.1).

4.2.4.1 Parameter Setting

We use the hardware setting introduced in Section (3.2.1). Values of some parameters in Equation (4.5) to Equation (4.9) that experiments can identify are summarized as below: air density $\rho = 1.25$ kg/m^3; we set propeller specifications based on (Brun Castell, 2022): blade radius $R = 0.15$ m, blade chord width $c = 0.01$ m, blade drag coefficient $c_{d_{blade}} = 0.04$, rotating area $A_{prop} = 0.07$ m^2, and scaling factor $k = 3.19 \cdot 10^{-5}$ (Kawamura and Azimov, 2020); drag coefficient $c_D = 0.04$ (Rodrigues et al., 2022), and IPT link efficiency $\eta_{IPT} = 40\%$ (Arteaga

et al., 2023). To prioritize low-energy sensor nodes, we established prize bounds of $[6, 10]$ for unvisited nodes and $[1, 5]$ for charged nodes.

We configure CBHA parameters as follows. We set the population number N_{pop} to 80, generation number N_{gen} to 80, attraction probability P_a to 75%, search number N_{srch} to 10, event horizon radius r at 25%, and aggregate frequency to 10 as a common setting. These settings were chosen to promote active evolution while maintaining computational efficiency. The relatively low aggregate frequency reflects a trade-off between solution quality and execution time, as the global solution update process involves substantial inter-process communication and computation. This configuration was designed for scenarios where the UAV requires tens of seconds for supercapacitor charging operations. While ROMP consistently generates safe routes with these parameters, they may not always outperform GLS solutions. However, our analysis in Chapter (A) demonstrates that increasing the population size to 5000 enables ROMP to surpass GLS performance in 100% of test cases.

4.2.4.2 Metaheuristic Options in Google OR-Tools

OR-Tools provides multiple local search metaheuristics, including Automatic selection (AUTO), Greedy Descent (GD), GLS, Simulated Annealing (SA), and Tabu Search (TS) (Furnon and Perron, 2024). We evaluated these options empirically, averaging results across 20 independent executions (see Figure (4.8)). While AUTO and GD demonstrated rapid computational convergence, they consistently produced solutions with suboptimal fitness metrics. Since the OR-Tools solution is the foundation for our CBHA calculations, a low-quality initial solution, though potentially increasing CBHA's improvement margin, would impede meaningful performance analysis. GLS was, therefore, selected as the benchmark algorithm due to its superior solution quality within reasonable computational bounds.

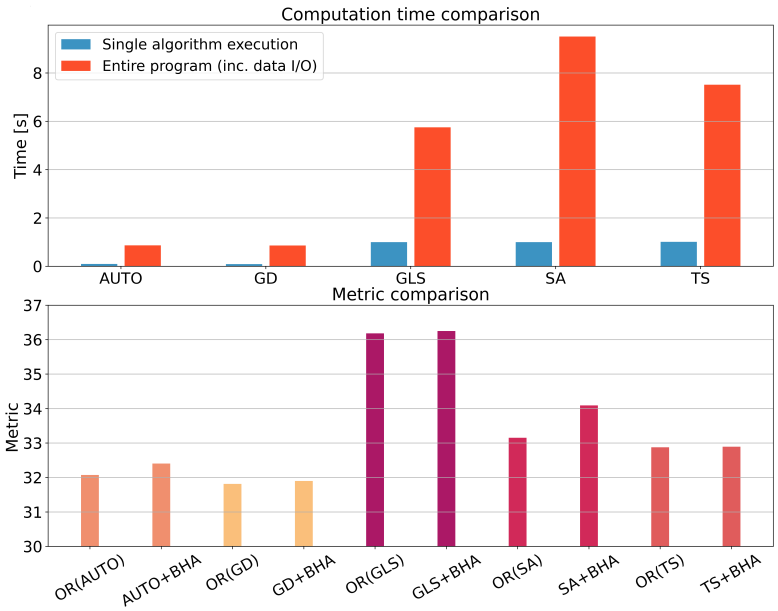


Figure 4.8: Execution results of OR-Tools and the optimization solver with different local search options, including Automatic selection (AUTO), Greedy Descent (GD), GLS, Simulated Annealing (SA), and Tabu Search (TS).

4.2.4.3 Wind Impact on Solution Performance

We selected the TSP scenario to examine wind effects on the optimization solver's performance. This configuration allows for a direct observation focusing on wind-induced energy consumption variations, independent of other confounding factors like changes in node visiting sequences due to prize evaluation. Figure (4.9) demonstrates a typical 15-node TSP instance, where OR-Tools' solution on the left is optimal in terms of minimizing travel distance. However, when a constant westerly wind of 5 m/s is introduced, this distance-optimal path is no longer the most energy-efficient. ROMP, which directly optimizes for energy consumption using its wind-aware power model (Equations (4.3) – (4.7)), suggests an improved path on the right.

The key improvement comes from the re-sequencing of two nodes highlighted in the red boxed region of Figure (4.9). Here, the OR-Tools path contains a long segment almost directly into the headwind. This seemingly small change reduces the energy required for this sub-tour from 1.695

Wh to 1.315 Wh, representing a local saving of over 22%. Although the total energy saving for the entire path is 0.11 Wh, which is not substantial in the context of a full charging mission, this example's purpose is to demonstrate ROMP's capability. It demonstrates that by incorporating environmental factors into the energy cost function, the algorithm can identify solutions that are structurally distinct from and superior to purely distance-based optimizations.

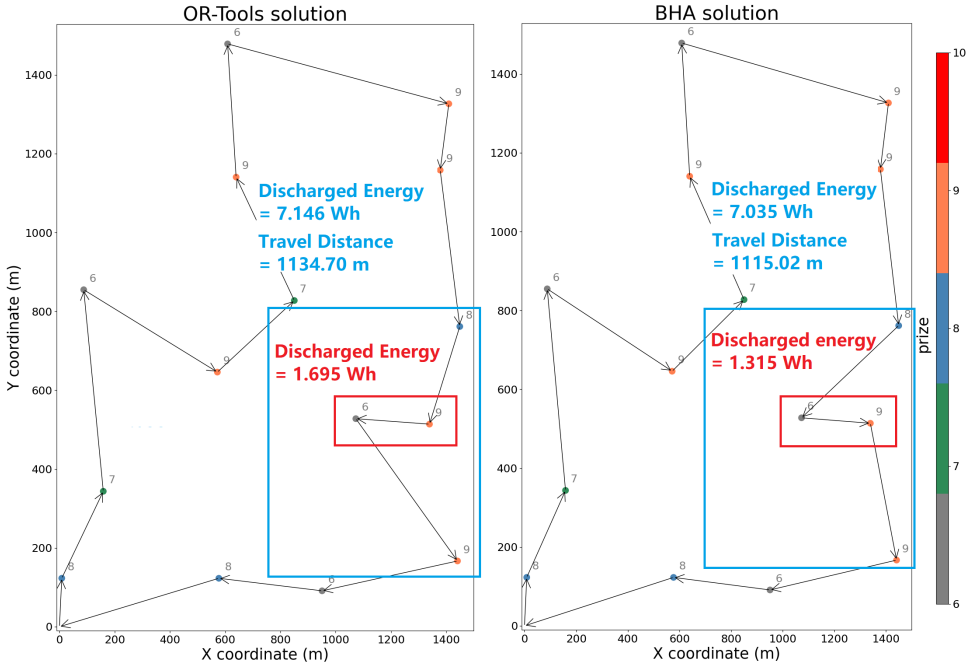


Figure 4.9: An instance showing changes in route sequence and discharged energy after applying a constant west wind.

4.2.4.4 Reaction to the Dynamic UAV Residual Energy

To evaluate ROMP's adaptability to unexpected energy depletion scenarios, we simulated conditions where a UAV encounters unpredictable environmental forces, such as sudden headwinds that accelerate battery consumption. The experiment involved artificially reducing the UAV's remaining energy to critical levels during mission execution to observe the planner's response. The dynamic adaptation process is illustrated through a sequence of route adjustments, shown as Figure (4.10). Initially, the UAV follows its original planned route until five sensor nodes

are charged. At this point, we simulate an energy reduction of 50 Wh, triggering the first route recalculation. After servicing two additional nodes along the revised path, a second energy reduction to 22 Wh is introduced, necessitating another route adjustment. This sequential reduction in available energy renders the original route infeasible, requiring ROMP to reassess the UAV's mission capabilities. The planner successfully adapts to these challenging conditions, generating a final route that requires only 16.18 Wh of discharged energy while maintaining mission objectives within the new energy constraints.

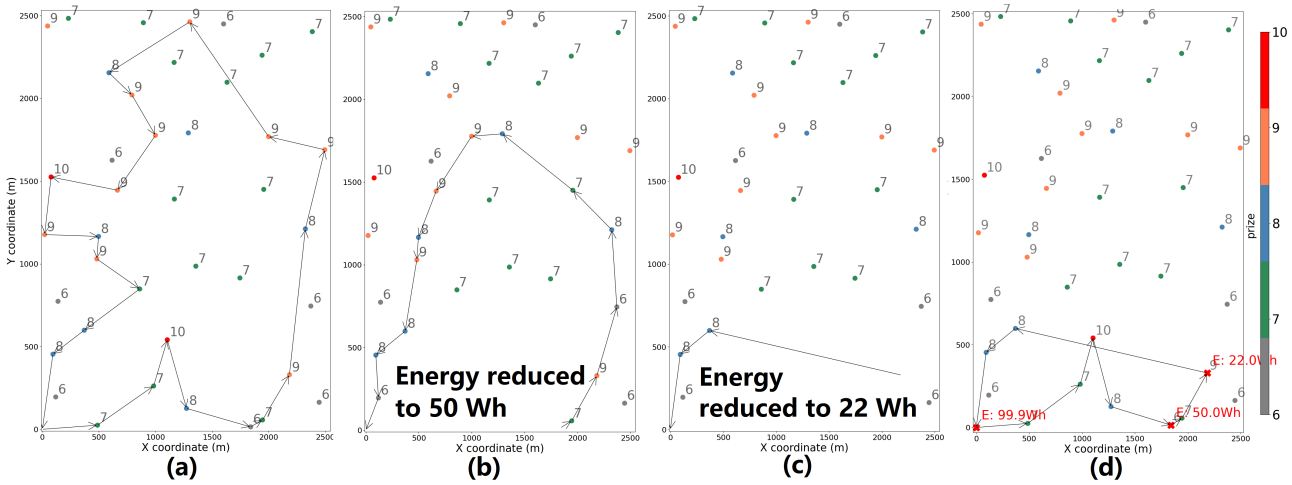


Figure 4.10: Flight plan adaptation under simulated energy depletion scenarios. (a) The originally planned route uses an energy capacity of 99.9 Wh. (b) After artificial energy reduction to 50 Wh following five node visits, the first route is re-planned. (c) Second route adjustment following simulated energy drop to 22 Wh after servicing two more nodes. (d) Composite visualization of the complete executed path showing energy states at key decision points, demonstrating ROMP's adaptive re-planning capabilities under simulated emergency conditions.

4.2.5 Conclusion

We present ROMP, a lightweight and reliable mission planner for the Charging Scheduling Problem. The work aims to maximize sensor node recharging while operating within strict UAV energy constraints. A key contribution is developing a first-principles energy consumption model for the DJI Matrice 100 drone that incorporates wind effects and IPT dynamics. This

model provides a foundation for understanding real-world UAV energy expenditure patterns, though its deterministic nature reveals limitations in handling stochastic environmental factors.

The proposed solution architecture comprises two main components: an initial solver based on Guided Local Search (GLS) and an optimization solver featuring our Context-aware Black Hole Algorithm (CBHA) that refines the initial solutions. To validate the practical deployability of this architecture on typical, resource-constrained flight computers, we analyzed the performance benefits of parallelization. As detailed in Appendix A, implementing a straightforward parallel strategy across multiple processing cores allows the optimization solver to consistently and efficiently find superior solutions compared to the baseline GLS. These results validate the system's practical deployability on typical flight computer hardware.

However, ROMP's deterministic approach to environmental uncertainties has several key limitations. While effective, the wind field segmentation strategy requires substantial infrastructure support for accurate measurements, and the choice of segmentation resolution significantly impacts estimation accuracy. Additionally, the 'dynamic-but-known' assumption of uncertain elements constrains the model's applicability in real-world environments. These limitations highlight the need for a more comprehensive approach to handle stochastic environmental factors and dynamic mission constraints without relying on extensive prior modeling knowledge or sensor infrastructure. To address these challenges, we propose ADAPT (adaptive approach for probabilistic paths) in the following section, introducing a novel framework for handling uncertainties in the UDOP. A detailed comparative analysis between ADAPT and ROMP in Section (4.3.3.4) demonstrates how probabilistic planning can enhance mission efficiency and safety in dynamic environments.

4.3 ADAPT: Bayesian Approach with Prior Linear Regression Model

4.3.1 Introduction

In UAV mission planning, concerns about uncertainty sources concentrate on factors that can cause continuous fluctuations in energy costs during UAV operations. These real-world error sources are generally unpredictable (e.g., moving obstacles, wind gusts, and turbulence). Comprehensively accounting for all unforeseen factors in mission planning is difficult due to the coupled planning between global mission and local trajectory (as in (Wang and Boyle, 2024)), potentially leading to suboptimal paths and uncontrollable computation time. Furthermore, precisely incorporating these factors into energy cost estimation requires additional sensor hardware support (e.g., anemometers (Abichandani et al., 2020)), specialized modeling knowledge (e.g., aerodynamics (Rodrigues et al., 2022) and battery behavior (Chen et al., 2018)). While strategies for efficiently addressing practical CSPs have been widely discussed in the literature, such as the two-stage strategy (Shi et al., 2024), three-dimensional charging schedule (Lin et al., 2023), and joint trajectory and scheduling optimization (Liu et al., 2022), few researchers address the balance between solution efficiency and mission safety. A safety guarantee in the CSP context represents that following a charging plan, the UAV can return to its end depot with sufficient energy, avoiding system failure or emergency actions like forced landing during the mission. Therefore, in contrast to most literature that employs deterministic and static power consumption models for UAV scheduling (Pasha et al., 2022b), we develop adaptive probabilistic planning with continuously calibrated energy cost estimation and assessed solution performance.

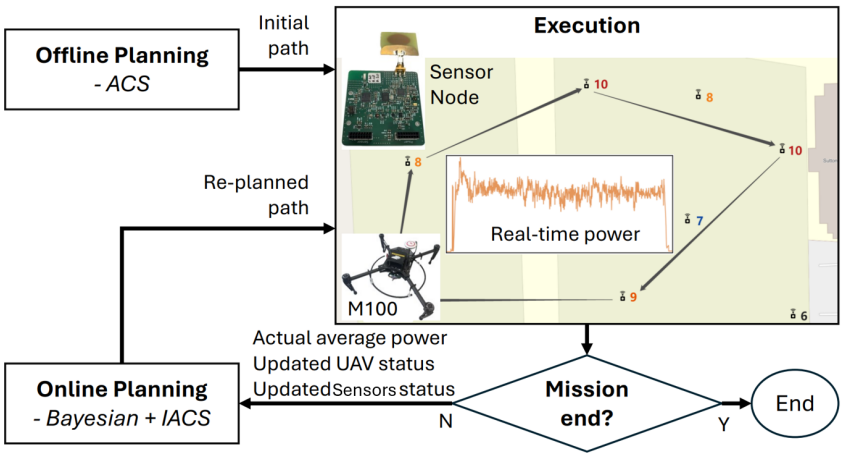


Figure 4.11: ADAPT framework. The UAV follows an initial path, sequentially servicing sensor nodes. During flight, the UAV continuously logs power consumption, which informs subsequent planning triggered upon completing the task at each node. This iterative process continues until the UAV returns to base when all target nodes are recharged, or residual energy cannot continue.

4.3.2 System Design

This section details the framework design of ADAPT for the UDOP. ADAPT dynamically adjusts the visitation sequence in response to environmental changes. It comprises three phases: offline planning, execution, and online planning (shown in Figure (4.11)). The UAV mission initiates with a pre-computed offline path. During the execution phase, it updates the observed travel costs (i.e., via continuous power consumption measurement), node prizes, and prize-collection costs for all unvisited sensor nodes. The execution and online planning phases

Table 4.1: Main parameters in ADAPT.

Parameters	Definition
$\bar{\mathrm{P}}$, $\bar{\mathrm{P}}^*$	Estimated and actual average power during UAV flight (W).
m, ρ	UAV weight (kg) and air density (kg/m^3).
$\mu_{\bar{\mathrm{P}}}$, $\mu_{\bar{\mathrm{P}}^*}$	Mean of distributions for $\bar{\mathrm{P}}$ and $\bar{\mathrm{P}}^*$.
$\sigma_{\bar{\mathrm{P}}}$, $\sigma_{\bar{\mathrm{P}}^*}$	Standard deviation of distributions for $\bar{\mathrm{P}}$, $\bar{\mathrm{P}}^*$.
$\Delta\mu$, $\Delta\sigma$	The shift to original μ and σ (%).
Θ	Using CDF probabilities from the distribution as costs to derive a feasible solution is having a safety belief Θ (%).
$\mathscr{P}$, $\mathscr{C}$	Prize (kJ) and Cost (kJ).

are iteratively conducted until the termination criteria are met: successful charging of all target nodes or mandatory return-to-home due to insufficient energy to continue. Table (4.1) summarizes the main parameters used in ADAPT.

4.3.2.1 Prior Knowledge and Assumptions

Because voltages of sensor nodes remain nearly the same over short intervals, we can assume static prizes and prize-collection costs during planning, but re-plan regularly after each charging operation. ADAPT decouples those dynamics from the planning phases by re-estimating chargeable energy and charging costs in the execution phase. Rodrigues et al. (2022) demonstrates that for UAVs operating at 25-100 m altitudes and ground speeds of 4-12 m/s, wind conditions varying from approximately 3.89 to 8.23 m/s have little impact on **average energy consumption** through field tests. Consequently, the induced power at hover in no-wind conditions serves as an adequate estimate for average power consumption $\bar{\mathrm{P}}$ during flight (Rodrigues et al., 2022):

$$\bar{\mathrm{P}} = b_1\sqrt{\frac{m^3}{\rho}} + b_0, \tag{4.15}$$

where b_1 and b_0 are coefficients derived from linear regression analysis between induced power and actual average power. Constants m and ρ denote the UAV weight and air density, respectively. Table (4.2) presents the trained coefficients for three distinct UAV regimes. For details, we refer the reader to the supplementary files of (Rodrigues et al., 2022).

Table 4.2: Model coefficient ± bootstrap standard error (Rodrigues et al., 2022).

Coefficient	Takeoff	Cruise	Landing
b_1	80.4 ± 2.6	68.9 ± 2.0	71.5 ± 1.7
b_0	13.8 ± 18.9	16.8 ± 15.0	-24.3 ± 12.5

Theorem 4.1. *The estimated average power consumption can be modeled as a normal distribution with mean* $\mu_{\bar{\mathrm{P}}} = \mu(b_1)\sqrt{\frac{m^3}{\rho}} + \mu(b_0)$ *and variance* $\sigma^2_{\bar{\mathrm{P}}} = \sigma^2(b_1)\frac{m^3}{\rho} + \sigma^2(b_0)$.

Proof. Given b_1 and b_0 follow two independent normal distributions, the estimated average power is the linear combination of their independent and identically distributed (i.i.d.) samples. This is mathematically equivalent to stating that the estimated average power follows a normal distribution with mean $\mu_{\bar{\mathrm{P}}} = \mu(b_1)\sqrt{\frac{m^3}{\rho}} + \mu(b_0)$ and variance $\sigma^2_{\bar{\mathrm{P}}} = \sigma^2(b_1)\frac{m^3}{\rho} + \sigma^2(b_0)$. □

While Equation (4.2b) provides the analytical form for the total travel energy via instantaneous power $\mathrm{P}^*(t)$, using such high-frequency data directly would make our estimation overly sensitive to transient noise like wind gusts. Thus, we adopt a more robust approach that averages power consumption over short (e.g., 20 seconds), discrete intervals from real-time measurements. Thus, a static and deterministic graph suffices as input for planners, approximating the complex UDOP to a classic OP under the CSP scenario.

4.3.2.2 Adaptive Approach for Probabilistic Paths (ADAPT)

Offline planning phase executes once at the start depot, aiming to generate a high-quality initial path for the UAV. Because designing a new algorithm for solving the static OP is beyond the scope of this study, we modify a well-established discrete metaheuristic as the main solver for offline and online planning phases. Specifically, we modify the ACS (Dorigo and Gambardella, 1997) due to its straightforward implementation, high adaptability, effectiveness for various instances of OP (Mandal and Dehuri, 2020), and convergence towards the optimal solution (Stutzle and Dorigo, 2002).

It is essential to highlight that even an optimal offline solution cannot guarantee global performance for the whole mission. In practice, ACS exhibits a good balance between solution quality and computational efficiency, aligning with the dynamic nature of our online problem. To validate this, we compare ACS with an exact method implemented by Gurobi (Gurobi Optimization, LLC, 2024) and investigate the potential impact of different offline paths on later

planning in Section (4.3.3.2).

Execution phase continuously monitors and records the battery's real-time power output throughout the flight to calculate the actual average power consumption $\bar{\mathrm{P}}^*$ during the UAV's proceeding to the next target node. Upon completion of the charging process at each node, the UAV updates the estimated chargeable energy $\mathscr{P}_{f_1}(v_i \mid t)$ and charging cost $\mathscr{C}_{f_3}(v_i \mid t)$ for all unvisited sensor nodes based on the actual travel and charging time.

Online planning phase. The online planner takes inputs of power observations, UAV status (coordinate and residual energy), and updated sensor status (chargeable energy and charging cost). If we assume $\bar{\mathrm{P}}^*$ follows a normal distribution with unknown mean $\mu_{\bar{\mathrm{P}}^*}$ and variance $\sigma^2_{\bar{\mathrm{P}}^*}$, whose parameters can be inferred with a conjugate Normal-Gamma (NG) prior distribution through Bayesian Inference (BI) (Friebe et al., 2021; Turlapaty, 2020). Employing BI offers advantages to estimate $\bar{\mathrm{P}}^*$ in the CSP because it: (a) incorporates prior knowledge to make estimation effective in early mission stages; (b) quantifies uncertainty in an interval, enabling robust decision-making under variable conditions; (c) allows continuous updating of estimates as new data becomes available, making it ideal for online scenarios where power consumption fluctuates; (d) offers convergence to the true distribution as observations accumulate (Savchuk and Tsokos, 2011). Moreover, pre-training a regression model as in (Rodrigues et al., 2022) can ease BI's limitation of the need to specify prior distributions and sensitivity to prior. To validate BI performance, we compare it to alternative methods, including frequentist and heuristic-based approaches (see Section (4.3.3.3) and Section (4.3.3.4)).

The hyperparameters of $\mathbf{NG}(\mu_{\bar{\mathrm{P}}^*}\ ,\ \sigma^{-2}_{\bar{\mathrm{P}}^*} \mid \mu_0, \kappa_0, \alpha_0^{\mathrm{BI}}, \beta_0^{\mathrm{BI}})$ can be determined using $\mu_{\bar{\mathrm{P}}}$ and $\sigma_{\bar{\mathrm{P}}}$. Following (Murphy, 2007), the posterior parameters can be updated as:

$$\mu_n = \frac{\kappa_0 \mu_0 + n\bar{\mathbf{x}}}{\kappa_0 + n} \tag{4.16a}$$

$$\kappa_n = \kappa_0 + n \tag{4.16b}$$

$$\alpha_n^{\mathrm{BI}} = \alpha_0^{\mathrm{BI}} + n/2 \tag{4.16c}$$

$$\beta_n^{\mathrm{BI}} = \beta_0^{\mathrm{BI}} + \frac{1}{2}\sum_{i=1}^{n}(x_i - \bar{\mathbf{x}})^2 + \frac{\kappa_0 n(\bar{\mathbf{x}} - \mu_0)^2}{2(\kappa_0 + n)} \tag{4.16d}$$

where n refers to sample size, and $\bar{\mathbf{x}}$ denotes sample mean. We employ a sliding time window to omit out-of-date observations. The window length is determined by two factors: the observed average power consumption may be affected by uncommon extreme values (e.g., significant fluctuations due to wind gusts or turbulence (Kim et al., 2020)), but it should still reflect recent environmental changes. The observed data and NG prior result in a posterior predictive of a Student-t distribution with center at μ_n, precision $\Lambda = \frac{\alpha_n^{\mathrm{BI}}\kappa_n}{\beta_n^{\mathrm{BI}}(\kappa_n+1)}$ and degree of freedom $\nu = 2\alpha_n^{\mathrm{BI}}$ (Murphy, 2007). Therefore, the new estimated average power consumption can be sampled from this location-scale t distribution, i.e.,

$$\bar{\mathrm{P}} \sim \mu_n + \mathbf{t}_{2\alpha_n^{\mathrm{BI}}}\sqrt{\frac{\beta_n^{\mathrm{BI}}(\kappa_n+1)}{\alpha_n^{\mathrm{BI}}\kappa_n}}. \tag{4.17}$$

The posterior's Cumulative Distribution Function (CDF) is used to derive various power consumption levels for takeoff, cruise, and landing, forming potential edge costs for travel. A path planned using average power $\bar{\mathrm{P}}$, obtained with CDF probability $\Theta = p(X \leq \bar{\mathrm{P}})$, is defined as having a *safety belief* Θ to complete the mission. Thus, we reduce UDOP to a deterministic and static problem with a given Θ value.

4.3.2.3 Inherited Ant Colony System (IACS)

The internal solver is a modified version of the Inherited ACS (IACS) (Qian et al., 2024) (see Algorithm (8)). The inheritance mechanism, designed to advance convergence, naturally aligns with the iterative process of execution and online planning. Specifically, the best path from the preceding planning iteration is a superior initialization for the pheromone matrix, surpassing

the conventional nearest neighbor heuristic. Our primary contribution is an inheritance mechanism for the online planning context. Since a path inherited from a prior planning event may be infeasible under an updated energy budget and new cost estimates, our method first calibrates the path to ensure its validity against the current mission state. We employ the corresponding drop operator (see Algorithm (6)) and add operator (see Algorithm (7)) to confine the budget constraint and maximize budget utilization[1]. The drop operator ensures path feasibility, eliminating low-value nodes to adhere to budget constraints. The add operator aims to improve path quality by inserting high-value feasible nodes to maximize prize collection. IACS is then applied multiple times to search candidate paths within a safety belief range of $\left[\Theta_{\min}, \Theta_{\max}\right]$. The final output is the path with the highest weighted score formed by safety belief and solution quality:

$$\boldsymbol{S} = \arg\max_{S_i} \left\{ w_\Theta \frac{\Theta_i - \Theta_{\min}}{\Theta_{\max} - \Theta_{\min}} + w_\mathcal{P} \frac{\mathcal{P}(S_i) - \mathcal{P}_{\min}}{\mathcal{P}_{\max} - \mathcal{P}_{\min}} \right\}, \tag{4.18}$$

where w_Θ and $w_\mathcal{P}$ are factors to balance the weight of safety belief and prize collection.

The drop cost of a node v_i denotes the sum of visit cost and service cost (if the node to be

Algorithm 6: Drop operator

Input: Path of ant m; Feasible node set $\mathbf{A}_m^{\mathbf{V}}$.

while *ant path does* ***not*** *satisfy Constraint* (4.1h) **do**
  for *each node* v_k *at path index* l *(exclude start and end node)* **do**
    Compute drop value $drop(v_k \mid l)$ by Eq. (5.13);
  Find node v_i at path index j such that $i, j = \arg\min_{k,l} \left\{ drop(v_k \mid l), ... \right\}$;
  Update path cost $\mathscr{C}_m \leftarrow \mathscr{C}_m - \mathscr{C}_{\text{drop}}(v_i \mid j)$ and path prize $\mathscr{P}_m \leftarrow \mathscr{P}_m - \mathscr{P}(v_i)$;
  Remove node v_i from ant path and update feasible set $\mathbf{A}_m^{\mathbf{V}} \leftarrow \mathbf{A}_m^{\mathbf{V}} \cup \left\{ v_i \right\}$;
return The feasible path with the new prize, new cost, and updated feasible node set.

[1]It is important to distinguish the drop and add operators here from the 'Dynamic' operators used in ROMP (Section (4.2.3.2)); while the dynamic operators perform stateful modifications on the UAV's single online plan, the operators here are used within the IACS solver to construct and evaluate many candidate solutions based on their final total cost.

Algorithm 7: Add operator

Input: Path of ant m; Feasible node set $\mathbf{A}_m^{\mathbf{V}}$.

while $\exists$ *any node* $v_k \in \mathbf{A}_m^{\mathbf{V}}$ *inserting to ant path still satisfies Constraint (4.1h)* **do**
  for *each node* $v_k \in \mathbf{A}_m^{\mathbf{V}}$ **do**
    Get three closest neighbor nodes $\mathbf{S}_{\text{nbr}} = \{nbr_1,\ nbr_2,\ nbr_3\}$ in the ant path;
    for *each pair* $(nbr_i,\ nbr_j) \in \mathbf{S}_{\text{nbr}}$ **do** check "is nbr_i adjacent to nbr_j in ant path";
    if $\nexists$ *any adjacent pair* **then**
      for $nbr_i \in \mathbf{S}_{\text{nbr}}$ **do**
        Find the previous and next node of nbr_i in the ant path;
        Create new pair $(v_{prev},\ nbr_i)$ and $(nbr_i,\ v_{next})$;
    Find the pair minimizing the visitation cost and insert at index l;
    Compute add value $add(v_k \mid l)$ by Eq. (4.20);
  Find the node v_i with maximum add value and its insertion index j in ant path, i.e., $i, j = \arg\max_{k,l} \{add(v_k \mid l), ...\}$;
  Update path cost $\mathscr{C}_m \leftarrow \mathscr{C}_m + \mathscr{C}_{\text{add}}(v_i \mid j)$ and path prize $\mathscr{P}_m \leftarrow \mathscr{P}_m + \mathscr{P}(v_i)$;
  Insert node v_i into ant path (at index j) and update feasible set $\mathbf{A}_m^{\mathbf{V}} \leftarrow \mathbf{A}_m^{\mathbf{V}} \setminus \{v_i\}$;
return The feasible path with the new prize, new cost, and updated feasible node set.

dropped is at path index j). The drop value of v_i is defined as its prize divided by its drop cost:

$$\mathscr{C}_{\text{drop}}(v_i \mid j) = -\mathscr{C}_{f_2}(v_{j-1},\ v_{j+1}) + \mathscr{C}_{f_2}(v_{j-1},\ v_i) + \mathscr{C}_{f_2}(v_i,\ v_{j+1}) + \mathscr{C}_{f_3}(v_i)$$
$$drop(v_i \mid j) = \mathscr{P}(v_i) \,/\, \mathscr{C}_{\text{drop}}(v_i \mid j) \tag{4.19}$$

Similarly, a node v_i's add value (if inserted at path index j) has the form as drop value:

$$\mathscr{C}_{\text{add}}(v_i \mid j) = -\mathscr{C}_{f_2}(v_{j-1},\ v_{j+1}) + \mathscr{C}_{f_2}(v_{j-1},\ v_i) + \mathscr{C}_{f_2}(v_i,\ v_{j+1}) + \mathscr{C}_{f_3}(v_i)$$
$$add(v_i \mid j) = \mathscr{P}(v_i) \,/\, \mathscr{C}_{\text{add}}(v_i \mid j) \tag{4.20}$$

ACS simulates the foraging behavior of an ant colony, incorporating three fundamental rules: the state transition rule, which decides the next visitation; the local updating rule, responsible for adjusting the pheromone trail visited by all ants; and the global updating rule, which updates the pheromone matrix based on the global-best ant. In our state transition rule, the

probability for the ant m at the node v_r to visit the next node v_s is defined as:

$$p_m(r,s) = \begin{cases} \dfrac{\left[\tau(r,\ s)\right] \cdot \left[\eta(r,\ s)\right]^{\beta}}{\sum_{v_u \in \mathbf{A}_m^{\mathbf{V}}} \left[\tau(r,\ u)\right] \cdot \left[\eta(r,\ u)\right]^{\beta}} , & \text{if } v_s \in \mathbf{A}_m^{\mathbf{V}} \\ 0 , & \text{otherwise} \end{cases} \tag{4.21}$$

where $\tau(r,s)$ is the pheromone deposited on edge $e_{r,s}$. We define the heuristic information $\eta(r,s) = \dfrac{\mathscr{P}(v_s)}{\mathscr{C}_{f_2}(v_r,\ v_s) + \mathscr{C}_{f_3}(v_s)}$ as the ratio of the node v_s's prize to the sum of edge cost between these two nodes and prize-collection cost. β is a parameter to control the relative importance of pheromone versus heuristic information. We denote the feasible set of remaining

Algorithm 8: Inherited Ant Colony System (IACS)

Input: Node set $\mathbf{V}$; Ant number N_{ant}; Iteration number N_{it}; Max no-improvement number N_{impr}; Improvement tolerance ϵ_{ACS}; Previous global-best path; β in Eq. (4.21); q_0 in Eq. (4.22); ρ in Eq. (4.23); α in Eq. (4.24).

if *online planning* **then**
  Apply Add (Alg. (7)) and Drop operator (Alg. (6)) to the previous global-best path;
  Update pheromone matrix with $\tau_0 \leftarrow \mathscr{P}_{\text{gb}}/\mathscr{C}_{\text{gb}}$;
else update pheromone matrix with τ_0 obtained by nearest neighbor heuristic;
Initialize N_{ant} ants and their feasible node set $\mathbf{A}^{\mathbf{V}}$, and reset no-improvement counter;
for $n_{\text{it}} = 1$ *to* N_{it} **do**
  if *no-improvement counter* $\geq N_{\text{impr}}$ **then** break the loop;
  for *each ant* m **do**
    Randomly sample the first node $v \in \mathbf{A}_m^{\mathbf{V}}$ and add it to the ant path;
    while $\mathbf{A}_m^{\mathbf{V}} \neq \varnothing$ ***and*** *ant path satisfies Constraint (4.1h)* **do**
      Select the next node by Eq. (5.9) and add it to the ant path;
      Update the prize, cost, and feasibility of the path;
    Add end node and apply 2-opt operator, then update path cost and feasibility;
    if *ant path* ***not*** *feasible* **then** invoke drop operator (Alg. (6));
    Update the pheromone matrix by Eq. (5.10);
  Record the ant with index of $\arg\max_i \left\{\mathscr{P}_i\right\}$ (or $\arg\min_i \left\{\mathscr{C}_i\right\}$ if $\mathscr{P} = \max_i \left\{\mathscr{P}^i\right\}$);
  if $\left(\ \mathscr{P}_{\text{lb}} \geq \mathscr{P}_{\text{gb}} + \epsilon_{\text{ACS}}\ \right)$ ***or*** $\left(\mathscr{P}_{\text{lb}} = \mathscr{P}_{\text{gb}}$ ***and*** $\mathscr{C}_{\text{lb}} \leq \mathscr{C}_{\text{gb}} - \epsilon_{\text{ACS}}\ \right)$ **then**
    Update the global-best ant and the pheromone matrix by Eq. (5.11);
  else no-improvement counter +1;
  Reset all ants (except the global-best ant);
return The path sequence, path cost, and path prize of the global-best ant.

nodes in the ant m by $\mathbf{A}_m^{\mathbf{V}}$. To balance exploring and exploiting, the state transition rule introduces an additional parameter $q_0 \in \big(0, 1\big)$:

$$s = \begin{cases} \arg\max\limits_{v_s \in \mathbf{A}_m^{\mathbf{V}}} \Big\{ \big[\tau(r,\ s)\big] \cdot \big[\eta(r,\ s)\big]^{\beta} \Big\} , & q \leq q_0 \\ s \sim p_k(r,\ s) \text{ in Eq. (4.21)} , & q > q_0 \end{cases} \tag{4.22}$$

The probability $q \in \mathbb{R}$ is randomly generated from a uniform distribution ranging in $[0, 1]$. Moreover, to reduce the probability of ants constructing the same solution, the local updating rule is applied to edges visited by ants after the solution construction phase:

$$\tau(r, s) \leftarrow (1 - \rho) \cdot \tau(r, s) + \rho \cdot \tau_0(r, s). \tag{4.23}$$

The evaporation rate $\rho \in (0, 1)$ is a constant that limits the accumulated pheromone on edge $e_{r,s}$. In ACS, only the global-best ant, whose solution achieves the highest quality so far (i.e., either maximum prizes or minimum costs when prizes are the same), can deposit the pheromone at the end of each iteration. The global updating rule is defined as:

$$\tau(r,\ s) \leftarrow (1 - \alpha) \cdot \tau(r,\ s) + \alpha \cdot \Delta\tau(r,\ s). \tag{4.24}$$

where $\alpha \in (0, 1)$ is a constant to control the pheromone decay rate, the deposited pheromone can be obtained by:

$$\Delta\tau(r, s) = \begin{cases} \mathscr{P}_{\text{gb}} \,/\, \mathscr{C}_{\text{gb}} , & \text{if } e_{r,s} \in \text{global-best path} \\ 0 , & \text{otherwise} \end{cases} \tag{4.25}$$

$\mathscr{P}_{\text{gb}}$ and $\mathscr{C}_{\text{gb}}$ are the collected prize and cost of the global-best path, respectively. We opted for a straightforward 2-opt local search method for later path sequence improvement.

4.3.3 Experiments, Results, and Discussion

This section presents numerical results, evaluating the performance and robustness of ADAPT and benchmark approaches. We employ three test instances, denoted as *California20*, *California30*, and *California40*, representing sensor nodes randomly deployed in a 1 km^2 area in California with increasing node numbers. To assess the algorithms' robustness under uncertain and variable conditions, we **shift and scale** prior normal distributions with coefficients in Table (4.2) to represent distributions of actual average power consumption $\bar{P}^*$, which is unknown to the planner. For example, a windy scenario might be characterized by distributions with actual mean $\mu_{\bar{P}^*} = 110\%\ \mu_{\bar{P}}$ and SD $\sigma_{\bar{P}^*} = 120\%\ \sigma_{\bar{P}}$. For simplicity, we denote this adjustment as $\Delta\mu_{\bar{P}^*} = 10\%, \Delta\sigma_{\bar{P}^*} = 20\%$, respectively. Unless specifically stated otherwise (e.g., to highlight specific scenarios), we evaluate each approach using the stated test instances under various actual power distributions, i.e., $\Delta\mu_{\bar{P}^*}, \Delta\sigma_{\bar{P}^*} \in \{-10, 0, 10, 20\}\%$. Though 20 individual executions are proven sufficient to examine repeatability (Qian et al., 2024), we increase executions to 50 due to random sampling from distributions. We assess algorithm robustness and performance using four metrics: computation time, Mission Success Rate (MSR), actual collected prize $\mathcal{P}^*$, and actual cost $\mathcal{C}^*$.

4.3.3.1 Parameter Setting

Following (Arteaga et al., 2023), we set the charger operates within a 20-42 V range at CC, charging a C = 10 F supercapacitor bank with an average current of $\bar{I}_{CC} = 0.825$ A, producing a $[0, 6.82]$ kJ node prize range. For instance, with a CC charger voltage at 30 V, the UAV's energy consumption and charging time are given by: $E_{IPT} = 0.5\ C\ (V_{max}^2 - 30^2)/\eta_{IPT}$ and $t_{IPT} = 10 \cdot (V_{max} - 30)/\bar{I}_{CC}/\eta_{CC}$. We set the IPT link efficiency $\eta_{IPT} = 40\%$, the CC charger efficiency $\eta_{CC} = 90\%$ and the energy depletion rate of sensor nodes $R_{SN} = 2.19 \cdot 10^{-6}$ kJ/s (Polonelli et al., 2020).

The execution phase models the UAV's energy consumption primarily through travel and service operations within the context of CSP. For travel modeling, parameters are based on (Rodrigues et al., 2021, 2022; DJI, 2016). The total weight of the M100 drone is 3.93 kg, including a TB47D battery (359.64 kJ capacity) and a 0.25 kg payload of induction coil and driving circuits. The air density is set to a common value $\rho = 1.225$ kg/m^3 (Zhang et al., 2021). The UAV flight protocol consists of takeoff (ascend to $H = 30$ m with an average speed $\mathrm{v_{tk}} = 3$ m/s), cruise (travel to the next waypoint with an average speed $\mathrm{v_{cr}} = 10$ m/s) and landing (descend to the ground with an average speed $\mathrm{v_{ld}} = 2$ m/s). The estimated average power consumption $\bar{\mathrm{P}}$ for each regime is sampled from normal distributions with coefficients stated in Rodrigues et al. (2022), i.e., $\bar{\mathrm{P}}_{\mathrm{tk}} \overset{\text{i.i.d.}}{\sim} \mathbf{N}(579.75, 692.16)$, $\bar{\mathrm{P}}_{\mathrm{cr}} \overset{\text{i.i.d.}}{\sim} \mathbf{N}(501.80, 423.20)$, $\bar{\mathrm{P}}_{\mathrm{ld}} \overset{\text{i.i.d.}}{\sim} \mathbf{N}(479.00, 299.45)$. The service simulation models the UAV's recharging process for homogeneous sensor nodes.

During the execution, we assume the period of average power reading as 20 seconds and the sliding time window length as the latest 15 minutes to balance historical observation utilization and temporal sensitivity. For Bayesian Inference, we set $\alpha_0^{\mathrm{BI}} = 2$, $\beta_0^{\mathrm{BI}} = \sigma_{\bar{P}}^2$, $\mu_0 = \mu_{\bar{P}}$, and $\kappa_0 = 1$ to employ a weekly informative prior knowledge about the mean and variance. We assign unbiased weights to the safety belief and prize collection, i.e., $w_\Theta = w_{\mathscr{P}} = 50\%$ in the whole mission. The sensitivity analysis of $\Theta_{\min}$ with a range from 45% to 85% is presented in Table (B.1). In summary, all settings of $\Theta_{\min}$ can have a high mission success rate of over 50 executions under most scenarios. The setting of $\Theta \in [45, 99]\%$ can frequently find higher-quality solutions compared to others because it allows more search space. However, the balance between safety beliefs and prize collection is challenging to maintain, resulting in risky solutions that pursue high prize collection (see *California20* with $\Delta\mu_{\bar{\mathbf{P}}} = 20\%$). In conclusion, the prize advancement achieved by setting a lower $\Theta_{\min}$ is insufficient to compensate for mission safety within the CSP context. The adaptive setting of weights to the safety belief and prize collection may allow a lower $\Theta_{\min}$ to achieve a higher mission success rate. Therefore, we set $\Theta_{\min} = 75\%$ and $\Theta_{\max} = 99.9\%$ to prioritize mission safety with a minor sacrifice of prize collection. Later, we

demonstrate that ADAPT can still outperform other benchmark approaches with $\Theta_{\min} = 75\%$ under some scenarios in Section (4.3.3.4).

Based on Qian et al. (2024), we set solver parameters as: Number of ants $N_{\text{ant}} = 40$, Number of iterations $N_{\text{it}} = 250$, heuristic importance factor $\beta_{\text{ACS}} = 2$, pheromone evaporation rate $\alpha_{\text{ACS}} = \rho = 0.1$. In the offline planning phase, the initial pheromone $\tau_0 = \mathscr{P}_{\text{nn}}/(\mathscr{C}_{\text{nn}} \cdot (|\mathcal{S}_{\text{nn}}| - 1))$, where $\mathscr{P}_{\text{nn}}, \mathscr{C}_{\text{nn}}$, and $|\mathcal{S}_{\text{nn}}|$ are the path prize, path cost, and path length of the solution achieved by nearest neighbor heuristic, respectively. While in the online planning phase, the initial pheromone $\tau_0 = \mathscr{P}^{S^{\text{gb}}(n_{\text{it}}-1)}/(\mathscr{C}^{S^{\text{gb}}(n_{\text{it}}-1)} \cdot (|S^{\text{gb}}(n_{\text{it}} - 1)| - 1))$, where $S^{\text{gb}}(n_{\text{it}} - 1)$ is the global-best solution obtained from previous iteration. We establish a minimum improvement tolerance $\epsilon_{\text{ACS}} = 10^{-4}$, which means ACS would terminate if the fitness difference is less than ϵ_{ACS} for several iterations. To balance the computation time and solution quality, we allow a maximum number of no improvements as $N_{\text{impr}} = N_{\text{ACS}}/10 = 25$.

4.3.3.2 Offline Planning with ACS and Gurobi

We assess the ACS's efficacy in solving *California* instances, comparing it to an exact method, the BnB implemented in Gurobi (Gurobi Optimization, LLC, 2024). To avoid unnecessary computation, we impose a 5-minute execution time limit and a 0.01 minimum improvement tolerance. Our experiments focus on $\Delta\mu_{\bar{\mathbf{P}}} \in \{-10, 0, 10, 20\}\%$ and $\Delta\sigma_{\bar{\mathbf{P}}} = 0\%$ for all *California*

Table 4.3: Offline solution performance comparison.

Instance	Alg.	Execution time (s)		Offline prize (kJ)		Offline cost (kJ)	
		mean	SD	mean	SD	mean	SD
California20	ACS	**4.38**	0.28	47.05	0.00	358.20	0.37
	BnB	12.19	1.55	**47.05**	0.00	358.88	0.50
California30	ACS	**6.91**	0.32	**41.01**	0.14	359.22	0.29
	BnB	301.12	0.04	40.97	0.20	359.23	0.48
California40	ACS	**9.57**	0.36	51.05	0.00	358.15	0.39
	BnB	226.61	53.03	**51.06**	0.01	358.92	0.58

instances. Table (4.3) presents averaged results from 200 individual executions (50 per $\Delta\mu_{\bar{\mathbf{P}}}$ value). The data indicate that ACS achieves solutions comparable to BnB's while significantly reducing computation time. Notably, for the *California30* instance, ACS outperforms BnB, likely due to the 5-minute time constraint being insufficient for Gurobi to identify a high-quality solution. While extended execution time might enable Gurobi to determine the optimal solution, such prolonged computation is impractical during mission execution.

Given the similar performance of ACS's and Gurobi's solutions, we validate the robustness of ADAPT by adjusting ACS parameters stated in Section (4.3.3.1) to $N_{\text{Ant}} = 4$ $N_{\text{ACS}} = 25$, $\epsilon_{\text{ACS}} = 0.01$, and setting the number of no-improvement iterations to 5. This adjustment reduces ACS's performance in computing a lower-quality offline path. Table (4.4) compares the solution quality of ADAPT using offline paths computed by limited ACS and Gurobi over 50 executions. Our findings suggest that the offline path quality may have a weak effect on final solution quality.

Table 4.4: Performance of solutions with different offline paths.

Instance	$\Delta\mu_{\bar{\mathbf{P}}^*}$	$\Delta\mu_{\bar{\mathbf{P}}^*}$	ADAPT using limited ACS path					ADAPT using BnB path				
			MSR	$\mathcal{P}^*$(kJ)		$\mathcal{C}^*$(kJ)		MSR	$\mathcal{P}^*$(kJ)		$\mathcal{C}^*$(kJ)	
	(%)	(%)	(%)	AVG	SD	AVG	SD	(%)	AVG	SD	AVG	SD
California20	-10	0	100.00	**50.54**	0.94	351.35	3.48	100.00	50.40	0.62	350.89	2.83
	0	0	100.00	47.44	0.60	351.81	3.72	100.00	**47.63**	0.54	349.74	4.22
	10	0	100.00	44.43	0.91	352.02	3.98	100.00	**44.46**	0.65	349.99	2.56
	20	0	100.00	41.20	0.89	349.90	5.18	100.00	**41.30**	0.76	350.42	5.52
California30	-10	0	100.00	45.32	0.98	352.14	2.68	100.00	**45.64**	0.38	349.54	3.27
	0	0	100.00	41.39	0.71	352.54	4.13	100.00	**41.57**	0.66	351.49	3.68
	10	0	100.00	**37.74**	1.01	353.39	3.46	100.00	37.43	0.88	353.61	3.00
	20	0	100.00	**34.38**	1.43	348.56	4.32	100.00	33.88	1.64	345.61	5.19
California40	-10	0	100.00	54.08	0.93	344.56	3.96	100.00	**54.28**	0.85	344.45	4.54
	0	0	100.00	51.58	0.85	350.36	4.30	100.00	**51.84**	0.53	348.46	3.39
	10	0	100.00	48.52	1.61	351.97	4.55	100.00	**48.94**	0.95	353.91	2.54
	20	0	100.00	45.75	1.65	351.13	4.43	100.00	**46.14**	1.17	351.28	3.86

4.3.3.3 Online Planning Phase with a Bayesian Approach

We demonstrate the performance of the online planning phase through a working example of a *California20* mission. In this scenario, we set the actual power consumption mean 20% higher than estimated $\Delta\mu_{\bar{P}^*} = 20\%$, while keeping its SD unchanged $\Delta\sigma_{\bar{P}^*} = 0\%$. Figure (4.12) illustrates the evolution of posterior distributions (from Post1 to Post9), based on given prior distributions and continuous online observations. Post1 exhibits a large SD due to insufficient samples, and the observed data significantly differs from the prior mean. As additional observations accumulate, posterior distributions demonstrate increasing centralization, reflecting ADAPT's adaptive learning process. ADAPT subsequently updates the edge cost matrix using these posterior distributions.

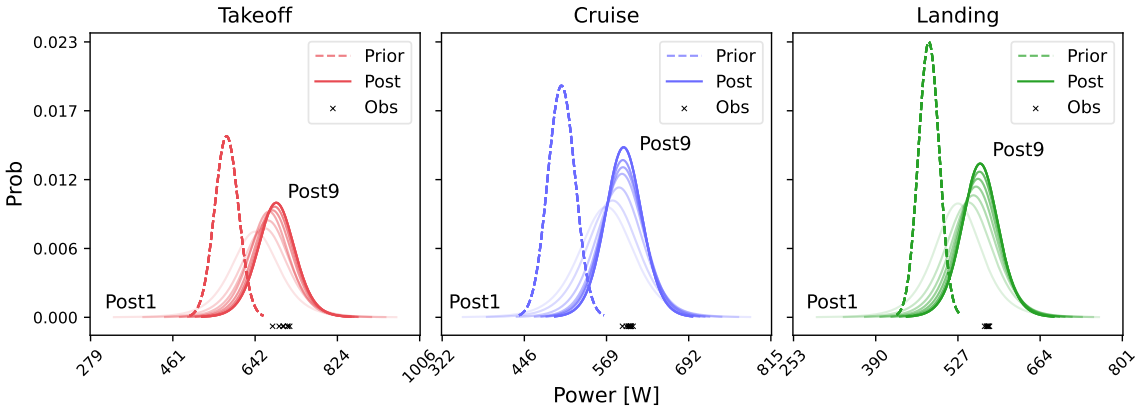


Figure 4.12: An example of how ADAPT updates posterior distributions using online observations. Nine re-plannings happened during this mission, moving from the most diverged distribution (Post1) to the most centralized one (Post9).

Figure (4.13) demonstrates the impact of varying safety belief (Θ) values (representing the confidence level in completing the mission) on solution quality for the above scenario. Paths with higher Θ values tend to be more conservative, while those with lower Θ values expect to charge more nodes. Notably, using Θ values of 80% and 90% yields identical solutions, suggesting this path can accommodate higher travel costs without compromising expected prize collection. This result underscores the ADAPT's capability to balance risk and reward effectively.

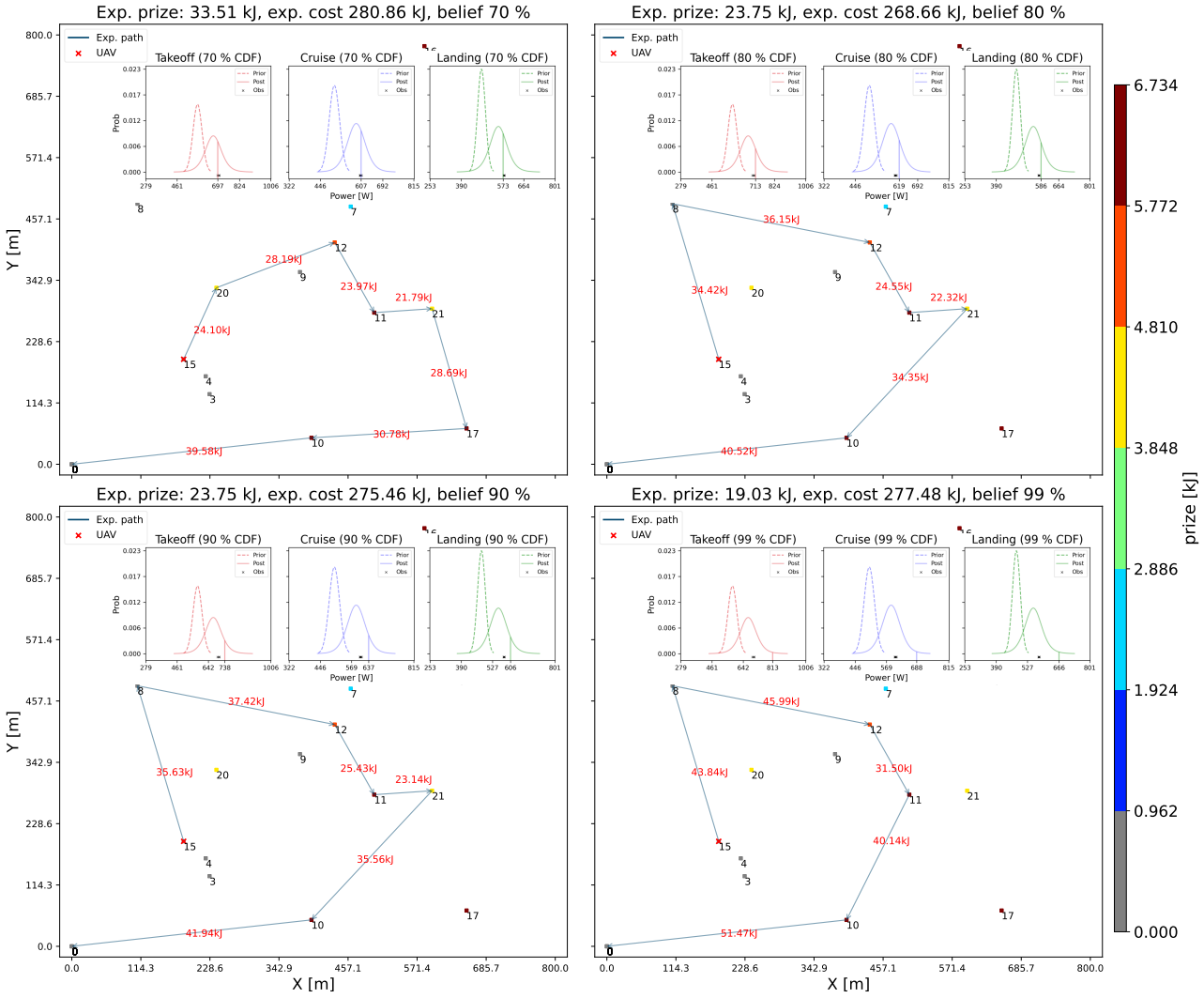


Figure 4.13: An example of ADAPT solving *California20* with $\Delta\mu_{\bar{\mathrm{P}}*} = 20\%$ and $\Delta\sigma_{\bar{\mathrm{P}}*} = 0\%$. When the UAV recharged sensors 4 and 15, its residual energy (budget) is 283.08 kJ. Here we show four typical candidate paths of ADAPT with safety belief $\Theta \in \{70, 80, 90, 99\}\%$.

4.3.3.4 ADAPT Performance Analysis

We evaluate ADAPT's performance against four alternative approaches:

- *Offline* always follows the initial offline path during the whole mission. It shows the performance of a static approach that ignores new information during the mission.

- *ROMP* (Qian et al., 2023) represents a simple adaptive strategy. It re-plans at each node,

using prior travel costs estimated during the offline planning phase.

- *WeightedErr* is a heuristic-based method that dynamically updates energy costs from recent observations. It calculates the weighted error ratio between estimated and actual energy costs from the most recent travel: $R_{\text{err}} = w_{\text{act}}(\frac{\Delta \text{E}_{\text{act}} - \Delta \text{E}_{\text{est}}}{\Delta \text{E}_{\text{est}}} + 1) + w_{\text{est}}$. Energy costs of all feasible edges are then updated as $\text{E}'(v_i, v_j) = R_{\text{err}} \cdot \text{E}(v_i, v_j)$. We set fixed weights $w_{\text{act}} = w_{\text{est}} = 0.5$ to balance sensitivity to estimation errors.

- Monte Carlo Greedy (*MCGreedy* (Angelelli et al., 2021)) randomly samples N_{MC} power levels between the minimum and maximum observed power consumption. The final output is the candidate path with the highest occurrence frequency. We set the number of samples $N_{\text{MC}} = 100$ as in Angelelli et al. (2021). As a frequentist approach, *MCGreedy* compares to ADAPT regarding how uncertainty is quantified and used in decision-making.

Computation time. ADAPT has a worst-case complexity of $\mathcal{O}(N_{\text{SN}}^2)$, where N_{SN} denotes the

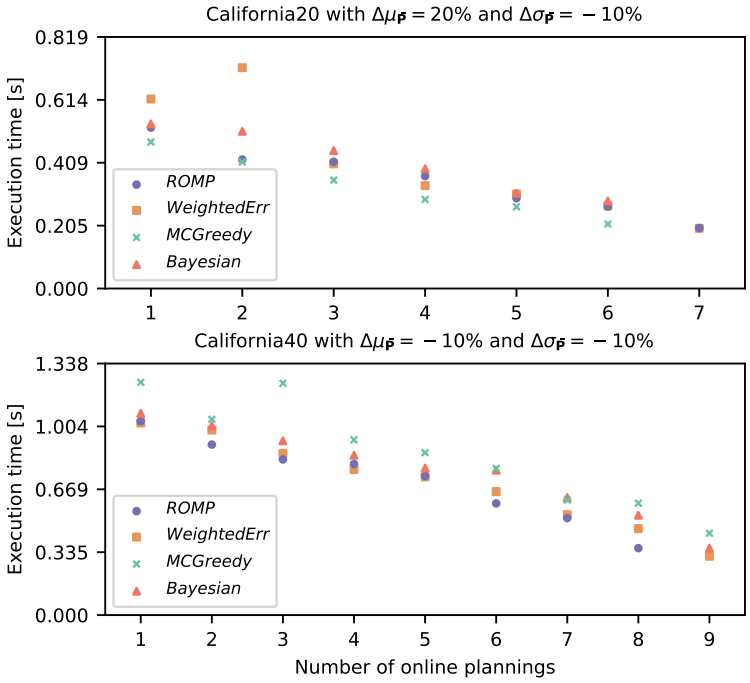


Figure 4.14: Two examples of algorithms' execution time. The top and bottom scenarios have the lowest and highest problem complexity among all tested scenarios.

node number, because the 2-opt operator has a $\mathcal{O}(N_{\text{SN}}^2)$ complexity. Figure (4.14) visualizes the typical computational time of four online approaches under a low and high complexity scenario[1]. All approaches can complete computation within seconds, demonstrating ADAPT's potential for continuous real-time re-planning.

Mission success rate. Figure (4.15) presents the percentage of safe returns to the end depot (i.e., the UAV has more than the minimum allowed energy level) across 50 individual executions for each approach. Within every execution, all approaches utilize the same offline path. Our results show that ADAPT consistently achieves 100% MSR across all test scenarios. This success may be attributed to its ability to identify high-quality paths with strong safety beliefs from the early stages of a mission. *MCGreedy* also performs well on average because the most common path generally has good quality with certain robustness (as shown in Figure (4.13)).

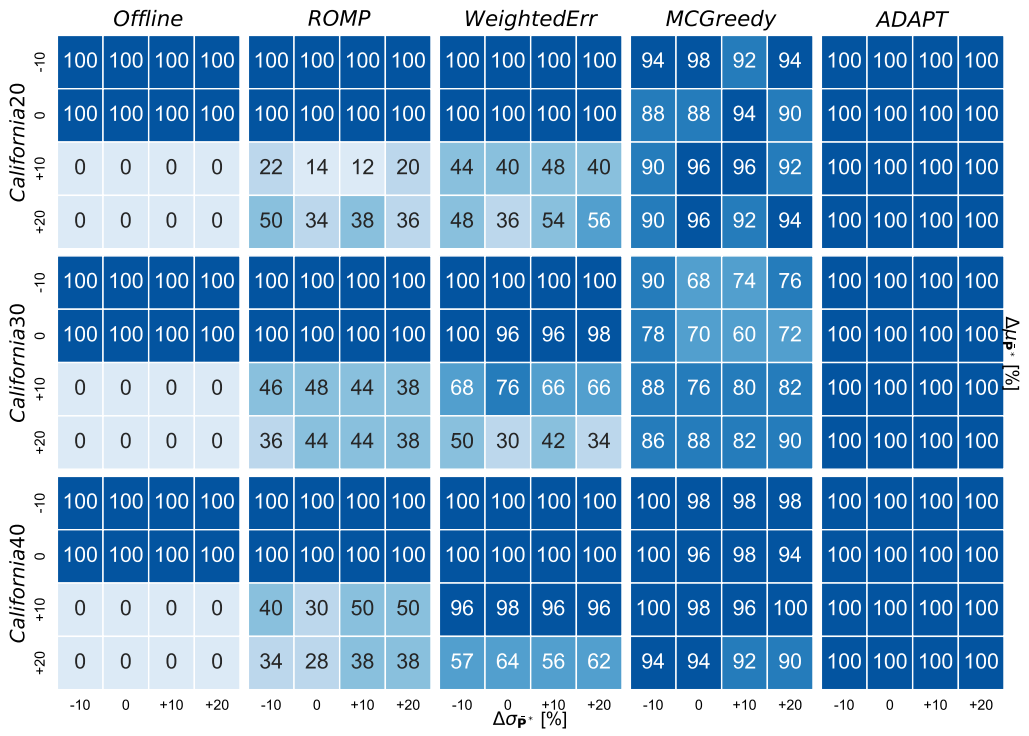


Figure 4.15: Mission success rate over 50 executions.

[1]All experiments were conducted on an Intel NUC11TNK with i7-11657G (2.8 GHz) CPU and 16 GB RAM.

However, the random nature of MC approaches results in unstable performance, as evidenced in the *California30* with low $\Delta\mu_{\bar{\mathrm{P}}^*}$. In contrast, *ROMP* often fails in scenarios with high $\Delta\mu_{\bar{\mathrm{P}}^*}$, as it tends to overestimate the UAV's capacity, leading to delayed recognition of the mission failure risk. Although *WeightedErr* incorporates online information for re-planning, its static weighting $w_{\mathrm{act}} = w_{\mathrm{est}} = 0.5$ proves inadequate to compensate for errors when $\Delta\mu_{\bar{\mathrm{P}}^*}$ is high. This highlights a practical challenge in determining global optimal values for w_{act} and w_{est}, which requires extensive prior knowledge. Note that when solving *California30* with $\Delta\mu_{\bar{\mathrm{P}}^*} = 0$, *WeightedErr* exhibits a few failed paths, despite *Offline* achieving a 100% MSR. This occurs because the estimated path costs are close to the budget at each planning, leading to low error tolerance. The actual costs of these failed paths (360.37, 360.85, 359.72, and 359.82 kJ) all marginally exceed the budget constraint (359.64 kJ).

Path prizes and costs. Table (4.5) presents average performance of *Offline*, *ROMP*, *WeightedErr*, *MCGreedy*, and ADAPT for solving *California20*, *California30* and *California40* with $\Delta\mu_{\bar{\mathbf{P}}}, \Delta\sigma_{\bar{\mathbf{P}}} \in \{-10, 0, 10, 20\}\%$. These results demonstrate the average prizes and costs of successful paths over 50 executions. Note that SDs for these 50 executions are omitted because offline paths can have a weak effect on the final solution quality (as stated in Section 4.3.3.2), and random sampling from distributions (especially for high $\Delta\sigma_{\bar{\mathrm{P}}^*}$) may introduce considerable uncertainty. ADAPT yields high-quality solutions across most scenarios compared to other approaches. In *California20*, long distances between sensor nodes potentially lead to excessive energy costs for achieving high Θ values. Consequently, ADAPT adopts a more conservative strategy, as inserting or changing to new nodes becomes difficult. In the *California40* scenario of Table (4.5), ADAPT tends to generate a high Θ path at early stages, subsequently improving it as more data is observed.

Mission process analysis. Figure (4.16) illustrates typical mission processes of all online approaches for *California40* with $\Delta\mu_{\bar{\mathrm{P}}^*} = 20\%$ and $\Delta\sigma_{\bar{\mathrm{P}}^*} = -10\%$. *MCGreedy* and ADAPT

Table 4.5: Solution quality comparison.

	$\Delta\mu_{\bar{\mathbf{P}}^*}$ (%)	$\Delta\sigma_{\bar{\mathbf{P}}^*}$ (%)	*Offline* $\mathcal{P}^*$(kJ)	*Offline* $\mathcal{C}^*$(kJ)	*ROMP* $\mathcal{P}^*$(kJ)	*ROMP* $\mathcal{C}^*$(kJ)	*WeightedErr* $\mathcal{P}^*$(kJ)	*WeightedErr* $\mathcal{C}^*$(kJ)	*MCGreedy* $\mathcal{P}^*$(kJ)	*MCGreedy* $\mathcal{C}^*$(kJ)	*ADAPT* $\mathcal{P}^*$(kJ)	*ADAPT* $\mathcal{C}^*$(kJ)
California20	-10	-10	47.07	322.48	48.71	341.33	**51.03**	348.92	49.66	351.17	50.48	351.18
	-10	0	47.07	322.44	48.77	342.54	**50.79**	348.87	50.09	351.164	50.27	352.39
	-10	10	47.07	322.52	48.75	341.92	**50.99**	348.54	49.43	351.14	50.42	351.08
	-10	20	47.07	322.46	48.77	343.21	**50.69**	348.18	49.58	351.01	50.29	352.29
	0	-10	47.07	345.15	47.28	346.93	**47.82**	351.91	46.50	350.98	47.53	348.96
	0	0	47.07	345.29	47.38	348.24	**47.95**	352.86	46.14	351.48	47.62	349.72
	0	10	47.07	345.04	47.44	348.21	**47.95**	352.32	46.41	351.73	47.61	349.24
	0	20	47.07	345.22	47.31	347.37	**47.86**	351.99	45.50	351.13	47.61	349.37
	10	-10	N/A	N/A	**45.45**	357.71	44.56	355.27	43.21	351.03	44.57	349.94
	10	0	N/A	N/A	**46.22**	359.27	44.48	355.43	43.15	351.21	44.47	350.05
	10	10	N/A	N/A	**45.84**	358.48	43.96	354.05	42.54	350.43	44.40	349.98
	10	20	N/A	N/A	**45.70**	357.45	45.02	356.91	42.80	350.41	44.72	351.77
	20	-10	N/A	N/A	39.57	357.19	**42.05**	357.94	40.93	351.68	40.98	349.41
	20	0	N/A	N/A	39.80	357.71	**42.25**	357.97	40.97	351.91	41.07	349.95
	20	10	N/A	N/A	39.56	358.14	**42.06**	358.34	40.90	352.05	41.01	349.79
	20	20	N/A	N/A	39.53	357.21	**42.04**	358.28	40.90	351.09	41.07	349.76
California30	-10	-10	41.08	323.25	44.34	347.76	45.52	349.34	**45.90**	353.57	45.64	349.31
	-10	0	41.05	323.02	44.20	347.37	45.56	348.98	**46.08**	352.90	45.56	349.61
	-10	10	41.04	323.20	44.27	347.53	45.53	348.96	45.58	353.41	**45.61**	349.76
	-10	20	41.04	323.26	44.01	346.75	45.62	349.66	45.48	352.45	**45.64**	350.44
	0	-10	41.05	347.54	41.93	353.07	**42.28**	355.93	41.19	352.24	41.93	352.41
	0	0	41.05	347.66	41.85	352.65	**42.50**	355.82	41.16	352.28	41.67	351.29
	0	10	41.05	347.63	41.87	352.68	**42.51**	356.23	40.90	351.29	41.76	351.94
	0	20	41.05	347.73	41.90	353.05	**42.25**	355.70	40.59	349.987	41.60	351.00
	10	-10	N/A	N/A	37.81	357.02	**38.12**	356.69	37.07	352.24	37.66	352.93
	10	0	N/A	N/A	37.88	356.79	**38.22**	356.95	36.89	351.57	37.35	353.06
	10	10	N/A	N/A	37.91	356.44	**37.95**	356.14	37.10	351.46	37.61	352.93
	10	20	N/A	N/A	**37.99**	357.36	37.89	356.70	37.02	351.95	37.63	353.55
	20	-10	N/A	N/A	34.01	352.86	**34.44**	358.72	34.29	351.75	34.24	347.32
	20	0	N/A	N/A	34.23	353.29	**34.34**	357.20	34.17	350.69	33.71	344.79
	20	10	N/A	N/A	34.34	353.62	**34.44**	358.63	34.30	349.62	33.88	346.41
	20	20	N/A	N/A	33.99	352.82	**34.52**	359.04	34.20	350.36	34.35	347.23
California40	-10	-10	51.08	321.77	52.37	330.67	**54.42**	345.34	54.23	349.60	54.26	343.77
	-10	0	51.08	321.75	52.37	330.86	**54.36**	345.07	54.29	348.47	54.22	343.54
	-10	10	51.08	321.80	52.37	330.79	**54.29**	345.14	53.75	347.91	54.11	343.10
	-10	20	51.08	321.73	52.40	331.24	**54.36**	345.18	53.21	347.78	54.19	343.46
	0	-10	51.08	343.32	51.26	347.12	**52.33**	351.47	50.19	347.96	51.70	351.34
	0	0	51.08	343.39	51.26	347.09	**52.33**	351.49	50.60	348.01	51.54	349.95
	0	10	51.08	343.49	51.27	347.16	**52.33**	351.44	50.78	348.73	51.62	350.53
	0	20	51.08	343.41	51.28	347.13	**52.32**	351.06	50.51	347.35	51.76	350.74
	10	-10	N/A	N/A	45.13	338.87	48.52	354.14	46.85	345.28	**49.04**	353.57
	10	0	N/A	N/A	44.57	335.98	48.70	353.89	47.02	345.36	**49.35**	354.43
	10	10	N/A	N/A	44.54	337.14	48.36	354.26	46.91	344.19	**49.18**	353.90
	10	20	N/A	N/A	45.11	337.39	48.30	354.09	46.73	344.42	**49.12**	353.72
	20	-10	N/A	N/A	42.50	337.16	42.63	339.23	45.00	351.71	**46.78**	352.76
	20	0	N/A	N/A	42.72	335.22	43.64	345.10	44.77	349.53	**46.96**	353.16
	20	10	N/A	N/A	42.90	335.65	43.48	343.63	44.02	346.81	**46.85**	352.75
	20	20	N/A	N/A	43.41	337.87	43.15	342.36	44.26	349.39	**46.70**	352.45

only insert node 13 during the mission (they have the same final path). In contrast, *ROMP* and *WeightedErr* underestimate actual power consumption, leading them to include nodes with high prizes (but deviating from the overall path) when the UAV has sufficient residual energy. Consequently, some nodes with low prizes but lower costs (or less risky) are dropped. For instance, in Figure (4.16), *ROMP* drops nodes 13 and 36 to incorporate node 38 in the path. While the solutions of *WeightedErr* and ADAPT suggest that dropping node 36 may be a viable strategy, visiting node 38 results in insufficient energy to visit and recharge node 24, ultimately leading to an inefficient solution. Similarly, *WeightedErr* drops node 13 to incorporate node 38. Although it recognizes the risk of including node 38 when the UAV is at node 34 and attempts to compensate for the prize loss by switching to recharge node 16, the quality of the

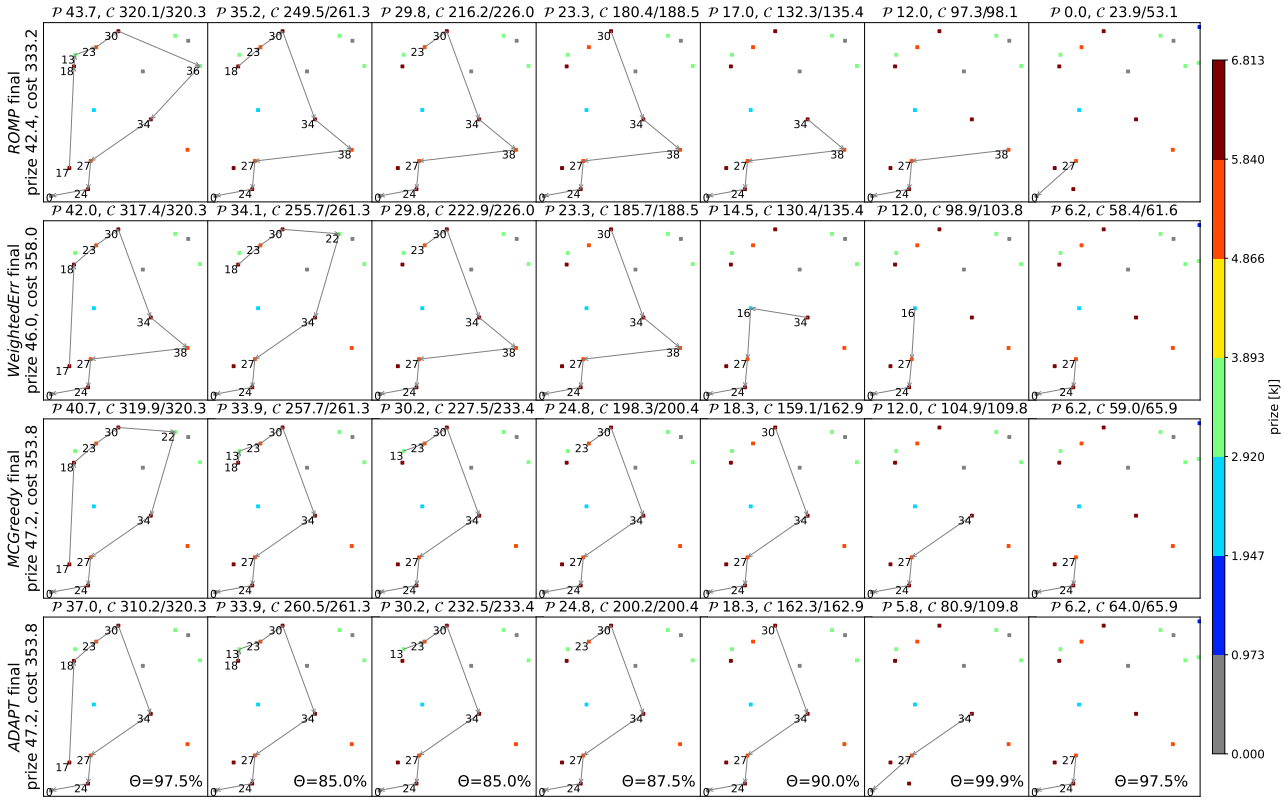


Figure 4.16: Online re-planning processes comparison for a typical execution of *California40* with $\Delta\mu_{\bar{\mathrm{P}}^*} = 20\%$ and $\Delta\sigma_{\bar{\mathrm{P}}^*} = -10\%$. Indices for sensor nodes not in the path are hidden. The value after '/' is the budget, and the initial budget is 359.64 kJ.

final solution is still compromised.

4.3.4 Conclusion

We propose a new problem, the Uncertain and Dynamic Orienteering Problem (UDOP), which aims to identify an optimal path that initiates from a start depot and returns to an end depot, maximizing the prize collection within the budget constraint under uncertain environments. UDOP differs from other OP variants with travel costs following distributions with unknown dynamic parameters and the potential impact of uncertain travel costs on node prizes and associated prize-collection costs. We propose a novel approach, ADAPT, to address the UDOP. In ADAPT, the offline planner generates an initial solution using prior knowledge of edge costs; the execution phase updates the mission execution status and records observations to online costs; the online planner employs a Bayesian approach to infer the parameters of edge cost distributions and determines the cost level (safety belief) for the solver, i.e., the Inherited Ant Colony System (IACS). Because the re-planning happens at each node, the impact of uncertain edge costs is naturally involved in the optimization process. Our experimental results demonstrate that ADAPT can achieve a 100% mission success rate among all test instances. ADAPT can even yield solutions that outperform other benchmark approaches in some scenarios where the expected energy cost is much less than the actual.

Chapter 5

Spatial Optimization for UAV Missions

5.1 CRaZSe-Ants: on Solving CEOP with Overlapped Neighborhoods

5.1.1 Introduction

CETSP aims to determine an optimal path originating from a designated depot, traversing all service regions, and returning to the initial depot. While this framework has been extensively studied through exact methods (Coutinho et al., 2016; Hà et al., 2012) and heuristic approaches (Yang et al., 2018; Wang et al., 2019b; Carrabs et al., 2020), three key limitations constrain its applicability to practical UAV missions. First, the requirement for identical start and end depots conflicts with the dynamic nature of UAV operations, which often necessitate real-time path adjustments in response to environmental changes (Evers et al., 2014a; Bayliss et al., 2020). Second, CETSP, inheriting TSP's unconstrained optimization framework, cannot simulate UAV missions with various constraints (e.g., time limitations and motion capabilities).

Third, CETSP assumes uniform visitation priorities, contradicting practical scenarios where service requirements vary across targets. For instance, different locations may have varying probabilities of successful meter readings (Placido et al., 2023).

We introduce the Close Enough Orienteering Problem (CEOP) to address these limitations. CEOP advances beyond CETSP by allowing different start and end depots, incorporating travel budget constraints, and enabling non-uniform target priorities through a prize collection mechanism. Though approaches to CEOP have been widely discussed in the literature (e.g., heuristics by Pěnička et al. (2017); Štefaníková et al. (2020)), there remains a notable gap in research concerning the intricacies of solving CEOP within *overlapped neighborhoods* – regions where multiple targets can be serviced simultaneously. This concept introduces a novel perspective, turning the limitations of overlapped neighborhoods (see Section 2.3 for more details), a.k.a. Steiner Zones (SZs) employed in the CETSP (Gulczynski et al., 2006), into advantages by leveraging prize-based attributes unique to CEOP. Specifically, CEOP instances can be reduced to Set Orienteering Problem (SOP) (Archetti et al., 2018) or even OP instances by aggregating prizes across overlapped neighborhoods.

However, practical applications often demand more sophisticated treatment of these overlapped service regions through non-linear cost functions. For example, in the Truck-and-Drone Delivery Problem (TDDP) by Boysen et al. (2018), multiple UAVs can be deployed to deliver packages to nearby customers in parallel, and the truck's idle time depends on its stop position and UAV operation time. Similarly, in wireless communication scenarios, data transmission efficiency varies non-uniformly based on antenna orientation and relative positioning between the UAV and ground nodes (Chen et al., 2021). These practical considerations motivate our development of CEOP with non-uniform neighborhoods (CEOP-$\mathcal{N}$), which extends CEOP to accommodate varying service costs within overlapped regions.

We start from the original CEOP formulation, investigating the geometric attributes of single

and overlapped neighborhoods, and subsequently delve into their implications on CEOP-$\mathcal{N}$ when involving non-uniform cost functions. We also discuss the sub-problem formulation from the CEOP discretization, i.e., SOP. We must highlight that we only translate the CEOP-$\mathcal{N}$ discretization to a OP because CEOP-$\mathcal{N}$ requires a continuous interior search of the neighborhood (both single and overlapped) to find the optimal waypoint.

5.1.2 Formulations of Close-Enough Routing Problems

This section presents formulations of Close-Enough Routing Problems with uniform and non-uniform neighborhoods. For ease of reference, we employ regular font abbreviations, such as CEOP and CEOP-$\mathcal{N}$, to denote the problem. We use the bold font **abbreviation** enclosed in **brackets** to represent the mathematical formulations provided in this section, e.g., **(CEOP)** and **(CEOP-$\mathcal{N}$)**. Table (5.1) introduces notation in the following formulations.

Table 5.1: Notation in formulations of Close-Enough Routing Problems.

Geometry related parameters	
$c_i,\ v_i,\ r_i$	Target circle with index i, and its corresponding circle center v_i and radius r_i.
Ω_i	The Steiner Zone with index i.
$\lvert\Omega_i\rvert$	The number of circles covered by the Steiner Zone Ω_i.
$v'(c_i)$, $v'(\Omega_j)$	Any waypoint located in a circle c_i or a Steiner Zone Ω_j.
$v'_{\text{ctr}}(\Omega_i)$	The center of the Steiner Zone Ω_i.
(x_i, y_i), (x'_j, y'_j)	Coordinate of the circle center v_i or the waypoint v'_j.
$\overset{\frown}{v'_i\, v'_j}(c_k)$	The circular arc on the circle c_k that starts from the vertex v'_i to the vertex v'_j.
Problem related parameters	
$\mathscr{C}_{f_1}(v' \mid c_j)$	Prize-collecting cost of the circle c_j at a waypoint v', defined by a continuous function f_1.
$\mathscr{C}_{f_2}(v'_i,\ v'_j)$	Traveling cost from waypoint v'_i to waypoint v'_j, defined by a continuous function f_2.
$\mathscr{P}(c_i)$, $\mathscr{P}(\Omega_j)$	Prize of a circle c_i or a Steiner Zone Ω_j.
$\mathscr{B}_f$	Budget constraint defined in general metric.

5.1.3 Steiner Zone Search Problem

We first introduce the definition of SZs, as it forms the foundation for understanding overlapped neighborhoods. A SZ, denoted by Ω, is a convex region enclosed by one or more circular arcs. The *degree* of a SZ refers to the number of circles it covers. For example, we use $|\Omega| = 3$ to represent a SZ formed by three circles. Given two intersect points (v'_1 and v'_2) on the circumference of two circles (c_1 and c_2), the *circular arc* on c_1 (from v'_1 to v'_2) is denoted as $\widehat{v'_1v'_2}(c_1)$. We use $v'(\Omega_k)$ with coordinate $\big(x'(\Omega_k), y'(\Omega_k)\big)$ to represent any *waypoint* located in the SZ Ω_k. The vertices of a SZ, also belonging to special waypoints, correspond to the starting and ending points of its constituent circular arcs. For consistency, we still use v' to denote a vertex of the SZ. Figure (5.1) demonstrates a SZ of degree three, which is formed by three circular arcs $\widehat{v'_1v'_2}(c_3)$, $\widehat{v'_1v'_3}(c_2)$, and $\widehat{v'_2v'_3}(c_1)$. This SZ has three vertices v'_1, v'_2, v'_3. Moreover, the coordinate of a SZ's *center* $v'_{\text{ctr}}(\Omega_j)$ is defined as:

$$\Big(x'_{\text{ctr}}(\Omega_j),\ y'_{\text{ctr}}(\Omega_j)\Big) = \begin{cases} \text{circle center}\ , & \text{if } |\Omega| = 1 \\ \Big(\dfrac{\sum_{v'_i \in \Omega_j} x'_i}{|\Omega_j|},\ \dfrac{\sum_{v'_i \in \Omega_j} y'_i}{|\Omega_j|}\Big)\ , & \text{otherwise} \end{cases} \tag{5.1}$$

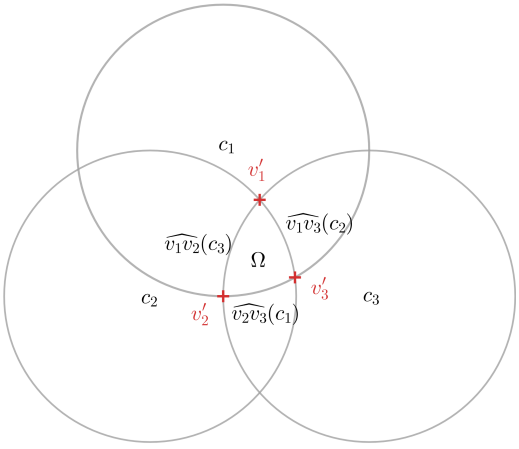


Figure 5.1: A Steiner zone of degree three, $|\Omega| = 3$, formed by three circular arcs $\widehat{v'_1v'_2}(c_3)$, $\widehat{v'_1v'_3}(c_2)$, and $\widehat{v'_2v'_3}(c_1)$. The red cross points refer to SZ vertices v'_1, v'_2, v'_3.

In other words, the coordinate of a SZ's center is the average of all SZ vertices. The conditions for adding an external circle c_i to an existing SZ Ω_j are as follows:

Condition 1. *The circle c_i must intersect with all other circles $c_k \in \Omega_j$.*

Condition 2. *For all pairs of intersecting circles, at least one intersection point must reside within the boundaries of those circles or on their circumferences.*

Note that we may add a circle to a SZ if it is contained within another larger circle. However, this specific scenario lies beyond the scope of our study, as it necessitates a trivial preprocessing step to accommodate contained circles. Thus, we assume a uniform circle radius for all circles. Given the circle set $\mathbf{C} = \{c_1, ..., c_N\}$, the Steiner Zone Search Problem (SZSP) aims at reconstructing a SZ layout $\mathbf{Z} = \{\Omega_1, ..., \Omega_M\}$, where $M \leq N$, that covers all circles. With decision variables $Y_{i,j} \in \{0, 1\}$ for each circle c_i to be added in a SZ Ω_j, SZSP can be formulated as:

$$\textbf{(SZSP)} \qquad \min \ |\mathbf{Z}| \tag{5.2a}$$

$$\textbf{s.t.} \quad \sum_{i=1}^{N} \sum_{j=1}^{M} Y_{i,j} = 1 \tag{5.2b}$$

$$\sum_{i=1}^{N} Y_{i,j} \leq N_{\max}^{|\Omega|}, \qquad j = 1, ..., M \tag{5.2c}$$

$$Y_{i,j} \leq \begin{cases} 1, & \text{if } c_i \text{ and } \Omega_j \text{ satisfy } \textbf{Condition } 1, 2 \\ 0, & \text{otherwise} \end{cases} \tag{5.2d}$$

Objective (5.2a) minimizes the number of SZs. Constraint (5.2b) enforces that each circle can be uniquely added into one specific SZ. Constraint (5.2c) restricts a SZ's maximum degree to $N_{\max}^{|\Omega|}$. Constraint (5.2d) requires a circle to be added to a SZ that must satisfy **Condition** 1 and 2.

5.1.4 CEOP and Uniform Overlapped Neighborhoods

Let $\mathbf{G} = \{\mathbf{C}, \mathbf{E}\}$ be a complete graph with a set of N target circles $\mathbf{C} = \{c_1, ..., c_N\}$ and a corresponding edge set $\mathbf{E} = \{e_{i,j}, ...\}$. Each target circle c_i is characterized by its center v_i, radius r_i, and prize $\mathscr{P}(c_i)$. Each edge $e_{i,j}$ represents the connection between two circle centers v_i and v_j. The agent can collect a target circle's prize $\mathscr{P}(c_i)$ at any waypoint v' located within the circle (or on its boundary). We define the distance cost between two circle centers v_i and v_j as Euclidean distance, i.e., $\mathscr{C}_d(v_i, v_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$. We denote the starting depot by v'_0 and the ending depot by v'_{N+1} with two-dimensional coordinates (x'_0, y'_0) and (x'_{N+1}, y'_{N+1}), respectively. Note that $\mathscr{P}(v'_0) = \mathscr{P}(v'_{N+1}) = 0$. A feasible path of CEOP starts at v'_0, collects as much prize as possible, and ends at v'_{N+1}, without violating the given distance budget $\mathscr{B}_d$ constraint. Because CEOP owns decision variables $X_{i,j} \in \{0, 1\}$ (decide the visitation for all target circles and depots), alongside continuous waypoint coordinates and quadratic constraints from circular geometry, we formulate it as a MINLP problem:

$$\textbf{(CEOP)} \quad \max \sum_{i=0}^{N} \sum_{j=1}^{N} \mathscr{P}(c_j) \cdot X_{i,j} \tag{5.3a}$$

$$\textbf{s.t.} \quad \sum_{j=1}^{N+1} X_{0,j} = \sum_{i=0}^{N} X_{i,N+1} = 1 \tag{5.3b}$$

$$\sum_{i=0}^{N+1} X_{i,k} = \sum_{j=0}^{N+1} X_{k,j} \leq 1, \qquad k = 1, ..., N \tag{5.3c}$$

$$\sum_{c_i \in \mathbf{S}} \sum_{c_j \in \mathbf{S}} X_{i,j} \leq |\mathbf{S}| - 1, \qquad \forall\, \mathbf{S} \subset \mathbf{C},\ |\mathbf{S}| \geq 3 \tag{5.3d}$$

$$\sum_{i=0}^{N} \sum_{j=1}^{N+1} \mathscr{C}_d(v'_i,\ v'_j) \cdot X_{i,j} \ \leq \ \mathscr{B}_d \tag{5.3e}$$

$$\mathscr{C}_d(v_i,\ v'_i) \leq r_i, \qquad i = 1, ..., N \tag{5.3f}$$

Objective (5.3a) maximizes the collected prizes of all visited target circles. Constraint (5.3b) ensures the path starts at depot v_0 and ends at depot v_{N+1}. Constraint (5.3c) provides the connectivity of the path and enforces that one target circle is visited at most once. Constraint (5.3d) prevents subtours. Constraint (5.3e) implies the path distance cost must not exceed the given distance budget $\mathscr{B}_d$. Here we exclude $N+1$ for index i and exclude 0 for index j because $\sum_{j=0}^{N+1} X_{N+1,j} = 0$ and $\sum_{i=0}^{N+1} X_{i,0} = 0$. Constraint (5.3f) restricts that the waypoint of a target circle must be located within its circular neighborhood or on its boundary. Additionally, by introducing overlapped neighborhoods into the above CEOP formulation, we can give additional constraints from SZs:

$$v' \in \Omega_k \iff \mathscr{C}_d\big(v',\, v_j\big) \le r_j, \qquad \forall\, c_j \in \Omega_k,\ k = 1, ..., M \quad (5.4a)$$

$$\mathscr{P}(\Omega_k) = \sum_{c_j \in\, \Omega_k} \mathscr{P}(c_j), \qquad k = 1, ..., M \quad (5.4b)$$

Constraint (5.4a) ensures that any waypoint located in a SZ Ω_j is also located within all SZ-covered circles, and vice versa. Constraint (5.4b) defines a SZ's prize as the sum of all SZ-covered circles.

Set Orienteering Problem. We formalize the SOP as an approximation to **(CEOP)** based on the vertices of constructed SZs. Given a set of M Steiner Zones $\mathbf{Z} = \big\{\Omega_1, ..., \Omega_M\big\}$, there are $K = \sum_{i=1}^{M} |\Omega_i|$ SZ vertices. A complete graph $\mathbf{G} = \big\{\mathbf{V}, \mathbf{E}\big\}$ can be constructed with a set of K Steiner Zone vertices $\mathbf{V} = \big\{v'_1, ..., v'_K\big\}$ and a corresponding edge set $\mathbf{E} = \big\{e_{i,j}, ...\big\}$, where the edge $e_{i,j}$ represents the connection between any two SZ vertices v'_i and v'_j. Following **(CEOP)**, the traveling cost between any two SZ vertices v'_i and v'_j is calculated based on the Euclidean distance cost $\mathscr{C}_d\big(v'_i,\, v'_j\big)$, and the prize for visiting a SZ is $\mathscr{P}(\Omega_i)$. To represent the start and end depot, we introduce two additional points, v'_0 and v'_{K+1}. Note that $\mathscr{P}(v'_0) = \mathscr{P}(v'_{K+1}) = 0$. The objective of SOP is to find a path starting from the start depot v'_0, maximizing collected prizes, and ending at the end depot v'_{K+1}, without violating the given distance budget $\mathscr{B}_d$

constraint. Note that in our SOP, each SZ Ω_j must be visited no more than once, with the visitation necessitating a stop at one of its SZ vertices $v'_i \in \Omega_j$. In alignment with the notation used by Pěnička et al. (2019), the decision variables are:

Y_i : binary variable equals 1 if a vertex within the SZ Ω_i is visited, and 0 otherwise;

$X_{i,j}$: binary variable equals 1 if edge $e_{i,j}$ is traversed, and 0 otherwise.

We formulate our SOP as an ILP problem based on (Pěnička et al., 2019):

$$\textbf{(SOP)} \qquad \max \sum_{\Omega_i \in \mathbf{Z}} \mathscr{P}(\Omega_i) \cdot Y_i \tag{5.5a}$$

$$\textbf{s.t.} \quad \sum_{e_{i,j} \in \mathbf{E}} \mathscr{C}_d\big(v'_i,\, v'_j\big) \cdot X_{i,j} \leq \mathscr{B}_d \tag{5.5b}$$

$$\sum_{v'_i \in \mathbf{V} \setminus \{\Omega_k\}} X_{i,j} = \sum_{v'_i \in \mathbf{V} \setminus \{\Omega_k\}} X_{j,i}, \qquad \forall\, \Omega_k \in \mathbf{Z},\ \forall\, v'_j \in \Omega_k \tag{5.5c}$$

$$\sum_{v'_i \in \mathbf{V} \setminus \{\Omega_k\}} \sum_{v'_j \in \Omega_k} X_{i,j} = Y_k, \qquad \forall\, \Omega_k \in \mathbf{Z} \tag{5.5d}$$

$$\sum_{v'_i \in \mathbf{V} \setminus \{\Omega_k\}} \sum_{v'_j \in \Omega_k} X_{j,i} = Y_k, \qquad \forall\, \Omega_k \in \mathbf{Z} \tag{5.5e}$$

$$\sum_{v'_i \in \mathbf{S}} \sum_{v'_j \in \mathbf{S}} X_{i,j} \leq \sum_{\Omega_k \in \mathbf{S} \setminus \{\Omega_l\}} Y_k, \qquad \forall\, \mathbf{S} \subset \mathbf{Z},\ \forall\, \Omega_l \in \mathbf{S} \tag{5.5f}$$

$$\sum_{j=1}^{K+1} X_{0,j} = \sum_{i=0}^{K} X_{i,K+1} = 1 \tag{5.5g}$$

Objective (5.5a) maximizes the collected prizes from SZs. Constraint (5.5b) implies the path distance cost must not exceed the given distance budget $\mathscr{B}_d$. Constraint (5.5c) ensures each SZ vertex has the same number of entering and leaving edges. Constraints (5.5d) and (5.5e) enforces that each visited SZ has exactly one entering and one leaving edge, otherwise no entering and leaving edge. Constraint (5.5f) eliminates subtours. Constraint (5.5g) requires the path starts at v'_0 and ends at v'_{K+1}.

5.1.5 CEOP with Non-uniform Neighborhoods (CEOP-$\mathcal{N}$)

It is important to clarify that 'Non-uniform Neighborhoods' in this context refers to a non-uniform cost function within the service areas, not to varying geometric properties such as the radius. Let $\mathbf{G} = \{\mathbf{C}, \mathbf{E}\}$ be a complete graph with a set of N target circles $\mathbf{C} = \{c_1, ..., c_N\}$ and a corresponding edge set $\mathbf{E} = \{e_{i,j}\}$, where the edge $e_{i,j}$ represents the connection between target circles c_i and c_j. Each target circle c_i is characterized by its center v_i, radius r_i, and prize $\mathscr{P}(c_i)$. The agent can collect prize $\mathscr{P}(c_i)$ at any waypoint v' located within the circular neighborhood (or on its boundary). Similar to **(CEOP)**, we use v'_0 and v'_{N+1} to represent the start and end depot, and $\mathscr{P}(v'_0) = \mathscr{P}(v'_{N+1}) = 0$. The objective of CEOP-$\mathcal{N}$ is to find a path starting from the depot v'_0, maximizing collected prizes, and ending at v'_{N+1}, without violating the given budget $\mathscr{B}_f$ constraint. CEOP-$\mathcal{N}$ introduces two distinct cost functions, $\mathscr{C}_{f_1}(v' | c_j)$ for the prize-collecting cost of the target circle c_j at some waypoint v', and $\mathscr{C}_{f_2}(v'_i, v'_j)$ for the traveling cost between waypoints v'_i and v'_j. Note that f_1 and f_2 are two 'nominal' functions, implying they can adopt any continuous form. For instance, in the standard CEOP, $\mathscr{C}_{f_1}(v' \mid c_j) = f_1(v',\ c_j) = 0$ and $\mathscr{C}_{f_2}(v'_i,\ v'_j) = f_2(v'_i,\ v'_j) = \sqrt{(x'_i - x'_j)^2 + (y'_i - y'_j)^2}$. The overall path cost comprises the prize-collecting and traveling costs, constrained by the budget $\mathscr{B}_f$. Given decision variables $X_{i,j} \in \{0, 1\}$ alongside continuous waypoint coordinates and quadratic constraints, we formulate CEOP-$\mathcal{N}$ as a MINLP problem:

$$\textbf{(CEOP-}\boldsymbol{\mathcal{N}}\textbf{)} \qquad \max \sum_{i=0}^{N} \sum_{j=1}^{N} \mathscr{P}(c_j) \cdot X_{i,j} \tag{5.6a}$$

$$\textbf{s.t.} \quad \mathscr{C}_{f_1}(v' \mid c_j) = f_1(v',\ c_j), \qquad j = 1, ..., N \tag{5.6b}$$

$$\mathscr{C}_{f_2}(v'_i,\ v'_j) = f_2(v'_i,\ v'_j), \qquad i \neq j = 1, ..., N \tag{5.6c}$$

$$\sum_{i=0}^{N} \sum_{j=1}^{N} \mathscr{C}_{f_1}(v' \mid c_j) \cdot X_{i,j} + \sum_{i=0}^{N} \sum_{j=1}^{N+1} \mathscr{C}_{f_2}(v'_i,\ v'_j) \cdot X_{i,j} \leq \mathscr{B}_f \tag{5.6d}$$

Constraints (5.3b) – (5.3d) and (5.3f)

Objective (5.6a) maximizes the collected prizes. Constraint (5.6b) enforces the cost to collect the prize of c_j at waypoint v' must satisfy a continuous function $f_1\big(v',\ c_j\big)$. Constraint (5.6c) enforces the cost to travel from waypoint v'_i to waypoint v'_j must satisfy another continuous function $f_2\big(v'_i,\ v'_j\big)$. Constraint (5.6d) implies the overall path cost, formed by costs to collect prizes and costs to travel, must not exceed the given budget $\mathscr{B}_f$. Moreover, there is no prize-collecting cost for the start and end depot.

Truck-and-drone Delivery Problem (TDDP) Example. Because **(CEOP-$\mathcal{N}$)** is insufficient when introducing overlapped neighborhoods and delving into practical data collection scenarios requires further context, we first support **(CEOP-$\mathcal{N}$)** with a straightforward truck-and-drone delivery example to help the reader understand the nature of CEOP-$\mathcal{N}$. Given the constraint that the delivery UAV possesses a maximum operational range denoted as r_{drone}, TDDP delineates the feasible service region (i.e., neighborhood) for each customer v_i as a target circle c_i. This circle is characterized by its center at coordinates (x_i, y_i) and a radius equivalent to r_{drone}. We construct a complete graph $\mathbf{G} = \big\{\mathbf{C},\ \mathbf{E}\big\}$, where $\mathbf{C} = \big\{c_1, ..., c_N\big\}$ denotes the set of N target circles and $\mathbf{E} = \big\{e_{i,j}, ...\big\}$ is the corresponding edge set. Each edge $e_{i,j}$ represents the connection between target circles c_i and c_j. The proximity between customers may result in an overlapping (i.e., SZ) of their respective serviceable areas. Note that the set of Steiner Zones $\mathbf{Z} = \big\{\Omega_1, ..., \Omega_M\big\}$ can be obtained by solving **(SZSP)**. The delivery truck can halt at a designated waypoint v' within such an overlap region Ω_i. From this position, it can concurrently deploy a maximum of $N_{\max}^{|\Omega|}$ UAVs to serve proximate customers encompassed by Ω_i.

The operational protocol for UAV delivery is as follows: Upon mounting the package intended for customer c_i, the UAV departs from the truck. The UAV initiates the service phase once it arrives above the specified delivery coordinate (x_i, y_i). This phase includes landing, handing over the package, and subsequent takeoff. The duration of this service phase is quantified as a constant t_{serv}. After the service phase, the UAV returns to the truck and prepares for the next

delivery. For the customer v_i, we use $\mathscr{P}(c_i)$ to denote the delivery priority (i.e., prize). Thus, any waypoint $v' \in \Omega_i$ can collect prize $\mathscr{P}\big(v' \mid \Omega_i\big)$ based on Constraint (5.4b). To reflect the heterogeneity of neighborhoods in CEOP-$\mathcal{N}$, TDDP assumes the package weight has a non-trivial impact on the UAV's average flight speed $\mathcal{V}_{\text{drone}}$. The flight efficiency is quantified as λ_i (in percentage). This efficiency parameter ranges between $\big[\lambda_{\min},\ \lambda_{\max}\big]$, and its magnitude is contingent upon the weight of the package requested by customer v_i. As a result, the effective average flight speed of the UAV when delivering the package for customer v_i is calculated as $\lambda_i \cdot \mathcal{V}_{\text{drone}}$. Additionally, we denote the average truck speed as $\mathcal{V}_{\text{truck}}$. Similar to **(CEOP-$\mathcal{N}$)**, we use v'_0 and v'_{N+1} to represent the start and end depot, and $\mathscr{P}(v'_0) = \mathscr{P}(v'_{N+1}) = 0$. TDDP aims to find a path for the truck to start from the depot v'_0, maximize collected prizes, and end at depot v'_{N+1}, without violating the given time budget $\mathscr{B}_t$ constraint. The truck's traveling and idle time defines a path's overall time cost $\mathscr{C}_t$. With decision variables $X_{i,j} \in \big\{0, 1\big\}$ that determine the visitation of all SZs and depots, the TDDP can be formulated as follows:

$$\textbf{(TDDP)} \qquad \max \sum_{i=0}^{N} \sum_{j=1}^{N} \mathscr{P}\big(v' \mid \Omega_j\big) \cdot X_{i,j} \tag{5.7a}$$

$$\textbf{s.t.} \quad |\Omega_i| \leq N_{\max}^{|\Omega|}, \qquad i = 1, ..., M \tag{5.7b}$$

$$\mathscr{C}_t\big(v' \mid \Omega_j\big) = \max \left\{ \frac{\mathscr{C}_d(v', v_i)}{\lambda_i \cdot \mathcal{V}_{\text{drone}}} + t_{\text{serv}} + \frac{\mathscr{C}_d(v_i,\ v')}{\mathcal{V}_{\text{drone}}} : \forall v_i \in \Omega_j \right\}, \qquad j = 1, ..., M \tag{5.7c}$$

$$\mathscr{C}_t\big(v'_i,\ v'_j\big) = \frac{\mathscr{C}_d\big(v'_i,\ v'_j\big)}{\mathcal{V}_{\text{truck}}}, \qquad i \neq j = 1, ..., M \tag{5.7d}$$

$$\mathscr{P}\big(v' \mid \Omega_j\big) = \sum_{c_i \in \Omega_j} \mathscr{P}(c_i), \qquad j = 1, ..., M \tag{5.7e}$$

$$\sum_{i=0}^{N} \sum_{j=1}^{N} \mathscr{C}_t\big(v' \mid \Omega_j\big) \cdot X_{i,j} + \sum_{i=0}^{N} \sum_{j=1}^{N+1} \mathscr{C}_t\big(v'_i,\ v'_j\big) \cdot X_{i,j} \ \leq \ \mathscr{B}_t \tag{5.7f}$$

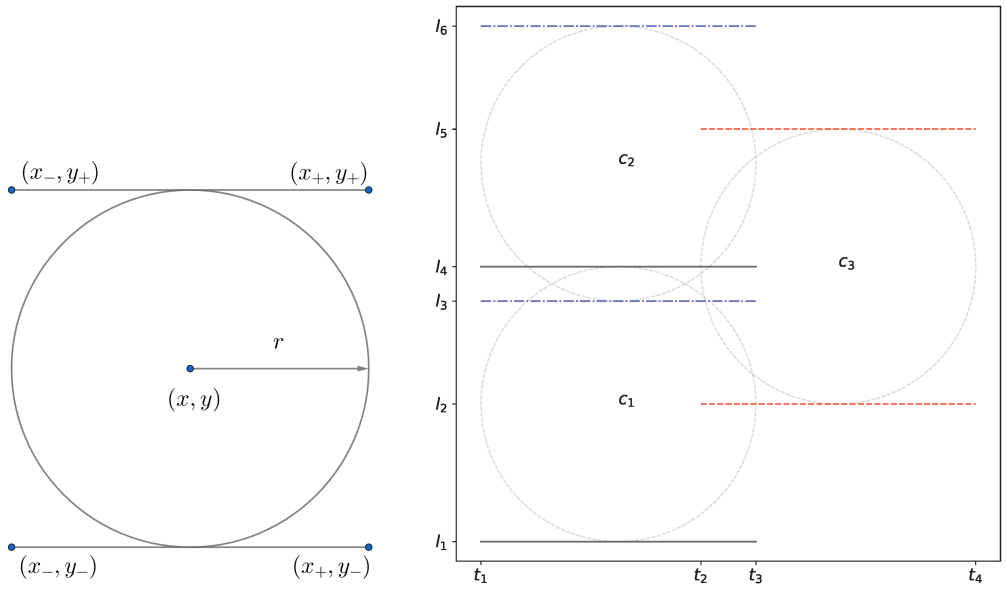
Constraints (5.3b) – (5.3d) and (5.4a)

Objective (5.7a) maximizes the prize of all packages delivered in the path. Constraint (5.7b)

enforces that the maximum degree of any SZ cannot exceed the number of UAVs $N_{\max}^{|\Omega|}$. Constraint (5.7c) defines the time cost of the truck waiting at the waypoint as the maximum time among all UAV deliveries, where t_{serv} is a constant representing the general UAV service time. Constraint (5.7d) defines the time cost for the truck to travel from waypoint v_i' to v_j', where v_i' and v_j' must be located in different SZs. Constraint (5.7e) defines a SZ Ω_k's prize $\mathscr{P}\big(v' \mid \Omega_k\big)$, which can be collected at a waypoint v', as the prize sum of all target customers included in the SZ. Constraint (5.7f) enforces that the overall time cost of the path, formed by a truck's idle and traveling time, must not exceed the given time budget $\mathscr{B}_t$. Moreover, there is no prize-collecting cost for the start and end depot.

5.1.6 Randomized Steiner Zone Discretization (RSZD) Scheme

Following the framework stated in Section (2.3), the first step is to discretize target circles. CRaSZe-AntS employs the RSZD scheme to construct a valid SZ layout and record all SZ vertices as candidates. The sweep line algorithm (Wang et al., 2019b) identifies potential SZ combinations by solving a set covering problem. This algorithm approximates each circle with two line segments $\overline{(x_-, y_+)(x_+, y_+)}$ and $\overline{(x_-, y_-)(x_+, y_-)}$, as illustrated in Figure (5.2a). The algorithm records each circle's leftmost and rightmost x-coordinates (x_- and x_+) as an ordered sequence $\{t_1, t_2, ...\}$, and its lowest and highest y-coordinates (y_- and y_+) as $\{l_1, l_2, ...\}$. A vertical line sweeps from t_1, examining intervals $[t_1, t_2], [t_2, t_3], ...$, to detect overlaps and register corresponding SZs. However, the sweep line algorithm exhibits critical limitations: (1) Computational complexity scales poorly with high-degree SZs (Wang et al., 2019b, Appendix A); (2) Inability to identify precise SZ vertex coordinates, which are crucial for subsequent path optimization; and (3) Insufficient handling of extreme scenarios. Figure (5.2b) presents an example when circles c_1, c_2, c_3 mutually intersect and their y-coordinate ranges $[l_1, l_4], [l_3, l_6], [l_2, l_5]$ overlap within interval $[t_2, t_3]$, the algorithm may incorrectly identify a three-degree SZ.



(a) Circle approximation. (b) Interval registration.

Figure 5.2: Sweep line algorithm (Wang et al., 2019b). Figure (b) shows an example where the sweep line algorithm may find a wrong SZ of degree three. The interval $[l_2, l_5]$ is registered at t_2, overlapping with $[l_3, l_6]$, $[l_1, l_4]$ and $[l_3, l_4]$. Three circles intersect pairwise, but only form SZs of degree two.

We propose the RSZD scheme (Algorithm (9)) to overcome these limitations. RSZD randomly selects candidate circles to establish SZs. For each candidate, it evaluates neighboring circles against **Condition** 1 and 2 to determine SZ membership. Figure (5.3) demonstrates this process on *concentricCircles1* (Mennell, 2009), showing the first three construction steps.

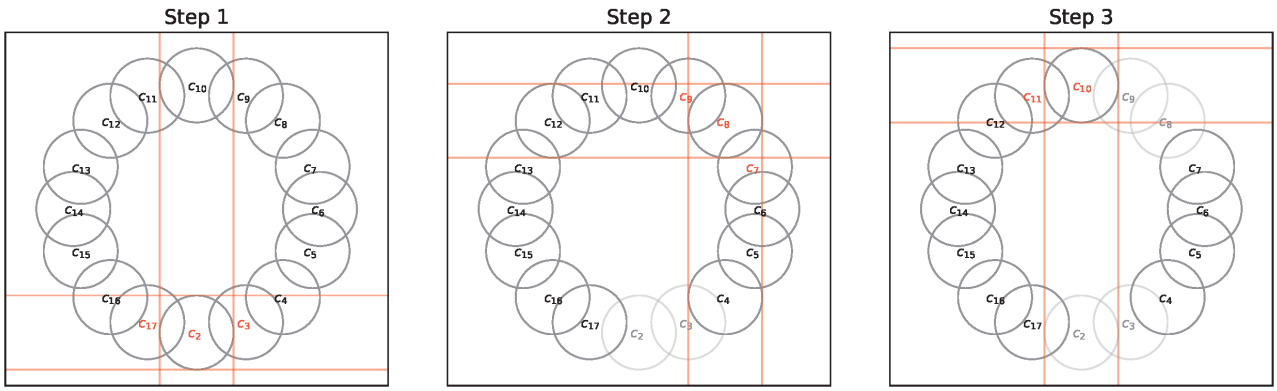


Figure 5.3: An example of the first three steps of employing RSZD on *concentricCircles1*.

Algorithm 9: Randomized Steiner Zone Discretization scheme

Input: Circle set $\mathbf{C} = \{c_1, ..., c_N\}$; Iteration number $N_{\text{iter}}^{\text{RSZD}}$; Max circles in a SZ $N_{\max}^{|\Omega|}$

for $t = 1$ *to* $N_{\text{iter}}^{\text{RSZD}}$ **do**
  Randomly shuffle $\mathbf{C}$ if $t \neq 1$ and update the feasible set $\mathbf{A_C} \leftarrow \mathbf{C}$;
  for *each circle* $c_i \in \mathbf{A_C}$ **do**
    Create a SZ $\Omega_m \leftarrow \{c_i\}$ and remove c_i from feasible circle set $\mathbf{A_C} \leftarrow \mathbf{A_C} \setminus \{c_i\}$;
    for *each circle* $c_j \in \mathbf{A_C}$ **do**
      if $|\Omega_m| < N_{\max}^{|\Omega|}$ **then**
        $\forall c' \in \Omega_m$, if distance $\mathscr{C}_d(c', c_j) < r_{c'} + r_{c_j}$, then c_j passes the necessity condition. Otherwise, skip to the next circle in $\mathbf{A_C}$;
        $\forall c' \in \Omega_m$, if there exists at least one intersection point (from c' and c_j) located in all other circles in the SZ Ω_m, then c_j passes the sufficiency condition. Otherwise, skip to the next circle in $\mathbf{A_C}$;
        if c_j *passes* ***all*** *conditions* **then**
          Update the corresponding vertex list of the SZ Ω_m;
          Update SZ $\Omega_m \leftarrow \Omega_m \cap \{c_j\}$ and feasible circle set $\mathbf{A_C} \leftarrow \mathbf{A_C} \setminus \{c_j\}$;
  Record current SZ set $\mathbf{Z}_t$;
return A SZ set $\mathbf{Z}_i = \{\Omega_1, ..., \Omega_M\}$ with $i = \arg\min_t\{|\mathbf{Z}_t|\},\ t = 1, ..., N_{\text{iter}}^{\text{RSZD}}$.

5.1.7 Approach to CEOP with Uniform Neighborhoods

This section explains the design of CRaSZe-AntS for CEOP (see left part of Figure (5.4)). In CEOP, searching the interior of constructed SZs is not compulsory; CRaSZe-AntS, therefore, utilizes ACS (with additional add/drop operators) to yield a solution from the resultant SOP instance. Finally, the arc search algorithm and the add operator optimize the discrete solution using the geometry of SZs.

While various methods exist for SOP, we select ACS over alternatives like GA due to its demonstrated efficiency in OP scenarios (Liang and Smith, 2006) and its more straightforward adaptation (while GA typically necessitates additional adjustments to its crossover operator) to variable-length solutions (Silva et al., 2008; Mandal and Dehuri, 2020). Our ACS implementation (see Algorithm (10)) incorporates three core mechanisms:

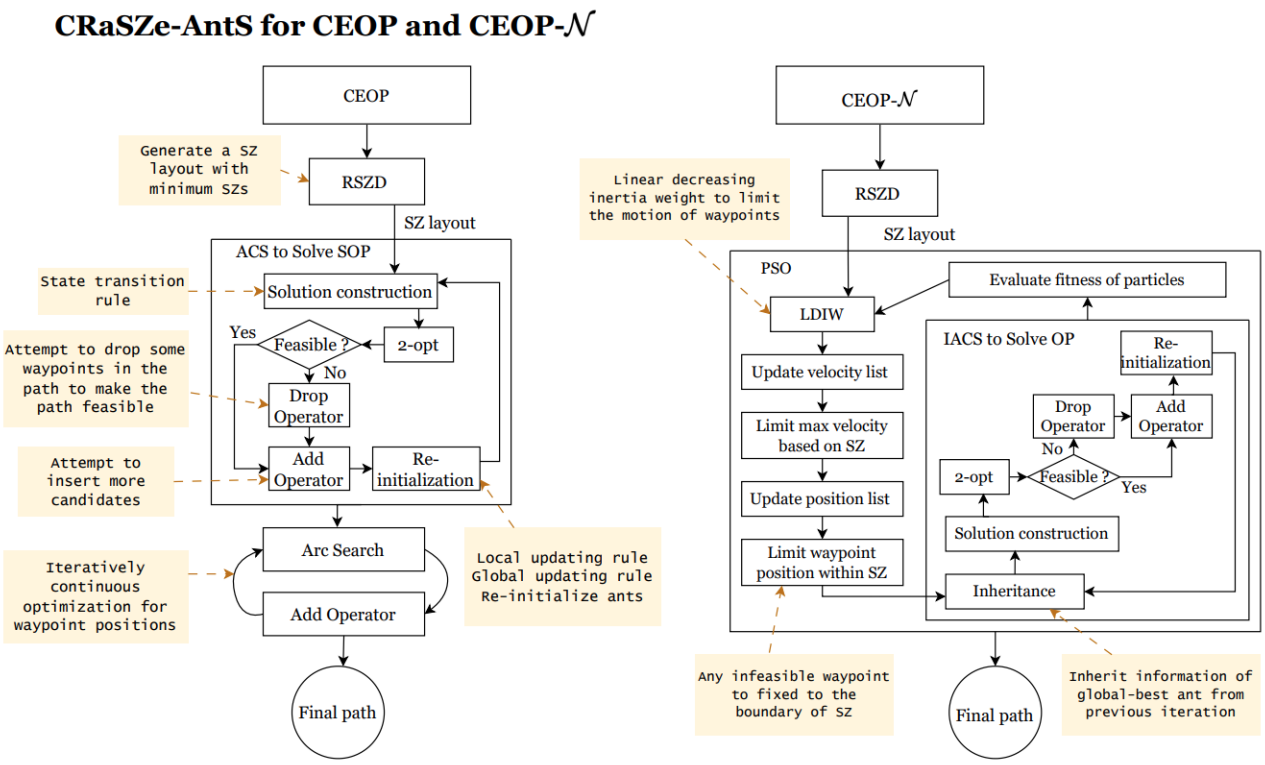


Figure 5.4: Graphical abstract of CRaSZe-AntS.

State transition rule. For ant k at waypoint $v'_r \in \Omega_i$, the probability of visiting waypoint $v's \in \Omega_j$ is:

$$p_k(r,s) = \begin{cases} \dfrac{\left[\tau(r,\ s)\right] \cdot \left[\eta(r,\ s)\right]^{\beta}}{\sum_{v'_u \in \mathbf{A}_k^{\Omega}} \left[\tau(r,\ u)\right] \cdot \left[\eta(r,\ u)\right]^{\beta}}\ , & \text{if } v'_s \in \mathbf{A}_k^{\Omega} \\ 0\ , & \text{otherwise} \end{cases} \tag{5.8}$$

where $\tau(r,\ s)$ is the pheromone deposited on edge $e_{r,s}$, $\eta(r,s) = \dfrac{\mathscr{P}(v'_s \mid \Omega_j)}{\mathscr{C}_d(v'_r,\ v'_s)}$ defines the heuristic information as the ratio of the SZ Ω_j's prize to the distance cost between these two waypoints, and $\mathbf{A}_k^{\Omega}$ denotes feasible set of unvisited SZs. We introduce a parameter $q_0 \in \big(0, 1\big)$ to balance exploration and exploitation:

$$s = \begin{cases} \arg\max\limits_{v'_s \in \mathbf{A}_k^{\Omega}} \left\{ \left[\tau(r,\ s)\right] \cdot \left[\eta(r,\ s)\right]^{\beta} \right\}, & q \le q_0 \\ s \sim p_k(r,\ s) \text{ in Eq. (5.8)}\ , & q > q_0 \end{cases} \tag{5.9}$$

where probability $q \in \mathbb{R}$ is randomly sampled from $[0, 1]$.

Local pheromone updating rule. After solution construction:

$$\tau(r, s) \leftarrow (1 - \rho) \cdot \tau(r, s) + \rho \cdot \tau_0(r, s), \tag{5.10}$$

where $\rho \in (0, 1)$ controls pheromone evaporation rate. We set the initial pheromone $\tau_0 = \mathscr{P}/(N_{wp} \cdot \mathscr{C}_d)$ to maintain consistency with the heuristic information, using path length N_{wp} instead of node count N to accommodate variable-length solutions.

Global pheromone updating rule. For the global-best ant achieving the highest objective

Algorithm 10: ACS to solve SOP within CEOP scenarios

Input: SZ set $\mathbf{Z} = \{\Omega_1, ..., \Omega_M\}$; Ant number N_{ant}; Iteration number $N_{\text{iter}}^{\text{ACS}}$;
Max no-improvement number $N_{\text{impr}}^{\text{ACS}}$; Improvement tolerance ε^{ACS};
β in Eq. 5.8; q_0 in Eq. 5.9; ρ in Eq. 5.10; α in Eq. 5.11.

Compute initial pheromone τ_0 by nearest neighbor heuristic;
Initialize N_{ant} ants and their feasible node sets $\mathbf{A}^{v'}$, reset no-improvement counter to 0;
for $n_{it} = 1$ *to* $N_{\text{iter}}^{\text{ACS}}$ **do**
  if no-improvement counter $\geq N_{\text{impr}}^{\text{ACS}}$ **then** break the loop;
  for *each ant* k **do**
    Randomly sample the first node $v' \in \mathbf{A}_k^{v'}$ to visit;
    while $\mathbf{A}_k^{v'} \neq \varnothing$ ***and*** *ant path satisfies Constraint (5.3e)* **do**
      Select the next node to visit by Eq. (5.9) and add it to the ant path;
      Update the prize, cost, and feasibility of the path;
    Add end node and apply 2-opt operator, then update path cost and feasibility;
    if ant path **not** feasible **then** invoke the drop operator (Alg. (11));
    Invoke the add operator (Alg. (12)) and update pheromone matrix by Eq. (5.10);
  Record the ant with index of $\arg\max_i \{\mathscr{P}^i\}$ (or $\arg\min_i \{\mathscr{C}_d^i\}$ if $\mathscr{P} = \max_i\{\mathscr{P}^i\}$);
  if $\left(\mathscr{P}^{lb} \geq \mathscr{P}^{gb} + \varepsilon^{ACS} \right)$ ***or*** $\left(\mathscr{P}^{lb} = \mathscr{P}^{gb}$ ***and*** $\mathscr{C}_d^{lb} \leq \mathscr{C}_d^{gb} - \varepsilon^{ACS} \right)$ **then**
    Update the global-best ant and the pheromone matrix by Eq. 5.11;
  else no-improvement counter +1;
  Reset all ants (except the global-best ant);
return The path sequence, path cost, and path prize of the global-best ant.

value so far:

$$\tau(r,\ s) \leftarrow (1-\alpha)\cdot\tau(r,\ s) + \alpha\cdot\Delta\tau(r,\ s), \tag{5.11}$$

where $\alpha \in (0, 1)$ controls the pheromone decay rate and

$$\Delta\tau(r,s) = \begin{cases} \mathscr{P}^{gb} \,/\, \mathscr{C}_d^{gb}\ , & \text{if } e_{r,s} \in \text{global-best path} \\ 0\ , & \text{otherwise} \end{cases} \tag{5.12}$$

where $\mathscr{P}^{gb}$ and $\mathscr{C}_d^{gb}$ are the path prize and cost of the global-best ant, respectively.

Beyond the standard 2-opt local search operator for path sequence improvement, additional local search procedures are incorporated to enhance solution quality:

Drop operator (Algorithm (11)). To maintain budget Constraint (5.3e), we remove waypoints with minimal contributions using a *drop value* metric (Kobeaga et al., 2018):

$$drop(v_i') = \frac{\mathscr{P}(v_i' \mid \Omega_j)}{\mathscr{C}_d(v_{i-1}',\ v_i') + \mathscr{C}_d(v_i',\ v_{i+1}') - \mathscr{C}_d(v_{i-1}',\ v_{i+1}')} \tag{5.13}$$

where $v_i' \in \Omega_j$ is the waypoint being evaluated at path index i.

Add operator (Algorithm (12)). Following drop operations, we iteratively insert waypoints

Algorithm 11: Drop operator

Input: Ant path $\mathbf{S}^a = \{v_1', ..., v_{N_{wp}}'\}$; Feasible node set $\mathbf{A}^{v'}$

while *path* $\mathbf{S}^a$ *can* ***not*** *satisfy Constraint (5.3e)* **do**
  for *each waypoint* v' *in ant path (exclude start and end waypoint)* **do**
    Compute drop value $drop(v')$ by Eq. (5.13);
  Find the path index $i = \arg\min_j \{drop(v_j'), ...\}$ with minimum drop value;
  Update path cost $\mathscr{C}_d \leftarrow \mathscr{C}_d - \mathscr{C}_d(v_{i-1}',\ v_i') - \mathscr{C}_d(v_i',\ v_{i+1}') + \mathscr{C}_d(v_{i-1}',\ v_{i+1}')$;
  Update path prize $\mathscr{P} \leftarrow \mathscr{P} - \mathscr{P}(v_i' \mid \Omega_k)$;
  Remove node v_i from $\mathbf{S}^a$ and update feasible set $\mathbf{A}^{v'} \leftarrow \mathbf{A}^{v'} \cup \{v_i' \mid \forall v_i' \in \Omega_k\}$;
return A feasible path with the new prize, new cost, and updated feasible node set.

Algorithm 12: Add operator

Input: Ant path $\mathbf{S}^a = \{v'_1, ..., v'_{N_{wp}}\}$; Feasible node set $\mathbf{A}^{v'}$

while $\exists$ *any node* $v'_k \in \mathbf{A}^{v'}$ *inserting to path* $\mathbf{S}^a$ *still satisfies Constraint (5.3e)* **do**
  for *each node* $v'_k \in \mathbf{A}^{v'}$ **do**
    Get three closest neighbor nodes $\mathbf{S}_{\text{nbr}} = \{nbr_1, nbr_2, nbr_3\}$ in path $\mathbf{S}^a$;
    for *each pair* $(nbr_i, nbr_j) \in \mathbf{S}_{\text{nbr}}$ **do** check "is nbr_i adjacent to nbr_j in path $\mathbf{S}^a$";
    if $\nexists$ *any adjacent pair* **then**
      for $nbr_i \in \mathbf{S}_{\text{nbr}}$ **do**
        Find the previous and next node of nbr_i in path $\mathbf{S}^a$;
        Create new pair (v'_{prev}, nbr_i) and (nbr_i, v'_{next});
    Find the pair minimizing the visitation cost and insert at which index l;
    Compute add value $add(v'_k \mid l)$ by Eq. (5.14);
  Find waypoint v'_j and insert index i in path $\mathbf{S}^a$ such that $(j, i) = \arg\max_{(k,l)} \{add(v'_k \mid l), ...\}$;
  Update path cost $\mathscr{C}_d \leftarrow \mathscr{C}_d + \mathscr{C}_d(v'_{i-1}, v'_j) + \mathscr{C}_d(v'_j, v'_{i+1}) - \mathscr{C}_d(v'_{i-1}, v'_{i+1})$;
  Update path prize $\mathscr{P} \leftarrow \mathscr{P} + \mathscr{P}(v'_j \mid \Omega_m)$;
  Insert waypoint v'_j into path $\mathbf{S}^a$ (at index i) and update feasible set $\mathbf{A}^{v'} \leftarrow \mathbf{A}^{v'} \setminus \{v'_j \mid \forall v'_j \in \Omega_m\}$;
return A feasible path with the new prize, new cost, and updated feasible node set.

with the maximum *add value* metric (Kobeaga et al., 2018):

$$add(v'_j \mid i) = \frac{\mathscr{P}(v'_j \mid \Omega_k)}{\mathscr{C}_d(v'_{i-1}, v'_j) + \mathscr{C}_d(v'_j, v'_{i+1}) - \mathscr{C}_d(v'_{i-1}, v'_{i+1})} \tag{5.14}$$

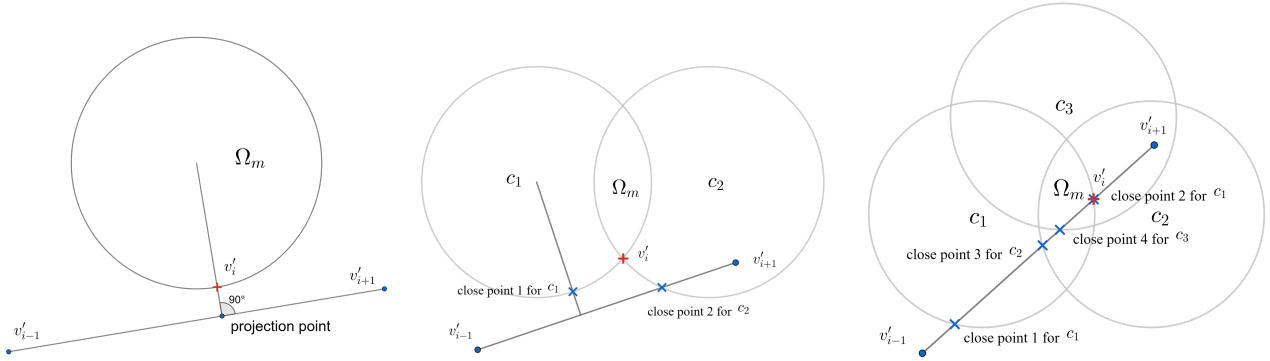
where $v'_j \in \Omega_k$ is evaluated for insertion at index i.

Arc search operator (Algorithm (13)). This operator optimizes the position of each waypoint v'_i within its SZ by fixing its previous waypoint v'_{i-1} and next waypoint v'_{i+1} along the path. For single-circle scenarios ($|\Omega_j| = 1$), it considers two cases between line segment $\overline{v'_{i-1}v'_{i+1}}$ and the circle - (1) No intersection (as shown in Figure (5.5a)): we project circle center onto $\overline{v'_{i-1}v'_{i+1}}$, the optimal position is on the intersection between the project line and circle; (2) One (tangent or intersecting without passing through) or two intersection(s): we use intersection point. For high-degree SZs, the arc search operator evaluates each circle's *close points* against the SZ's

Algorithm 13: Arc search operator

Input: SZ set $\{\Omega_1, ..., \Omega_M\}$; Waypoint sequence $\{v'_1, ..., v'_{N_{wp}}\}$; Iteration number $N_{\text{iter}}^{\text{arc}}$

for $i = 1$ *to* $N_{\text{iter}}^{\text{arc}}$ **do**
  for $j = 2$ *to* $N_{wp} - 1$ **do**
    Record previous waypoint v'_{j-1}, next waypoint v'_{j+1}, and SZ such that $v'_j \in \Omega_k$;
    for *each circle* $c_l \in \Omega_k$ **do**
      Compute close points based on line segment $\overline{v'_{j-1}v'_{j+1}}$ and c_l by Eq. (5.15);
      Compute the feasible arc range for $c_l \in \Omega_k$;
      if $\exists$ *any close point* v'_c *in SZ* Ω_k*'s feasible arc* **then** update waypoint $v'_j \leftarrow v'_c$;
    if $\nexists$ *feasible close point* **then**
      for *each SZ vertex* $v' \in \Omega_k$ **do** compute distance $\mathscr{C}_d(v'_{j-1},\ v') + \mathscr{C}_d(v',\ v'_{j+1})$;
      Update waypoint $v'_j \leftarrow$ the closest vertex;
    Update the path cost;
return The updated waypoint sequence with path cost.



(a) When the line segment has no intersection with the SZ of degree 1.

(b) When the line segment has no intersection with the SZ of degree 2.

(c) When the line segment has two intersection points with the SZ of degree 3.

Figure 5.5: Examples of the arc search algorithm for SZ of different degrees.

valid arc ranges, selecting the best candidate as v'_i's position. We define close points as:

$$\text{close points} = \begin{cases} \text{closest point to } \overline{v'_{i-1}v'_{i+1}}\ , & \text{if } \overline{v'_{i-1}v'_{i+1}} \text{ has no intersection with } c_k \\ \text{intersection points on } c_k\ , & \text{otherwise} \end{cases} \tag{5.15}$$

For each circle, if any close point falls within the feasible range of the circular arc formed by two SZ vertices, this close point is designated as the final position for v'_i. Otherwise, the close

point is infeasible. The arc search algorithm then selects the nearest SZ vertex as the candidate for that particular circle. Traversing all circles allows arc search to determine the optimal position for v_i' among chosen candidates. Figure (5.6) demonstrates this process on *bubbles1*. At iteration 1 (the leftmost sub-figure), arc search relocates waypoints (previously SZ vertices), leaving more budgets for the add operator to insert additional waypoints.

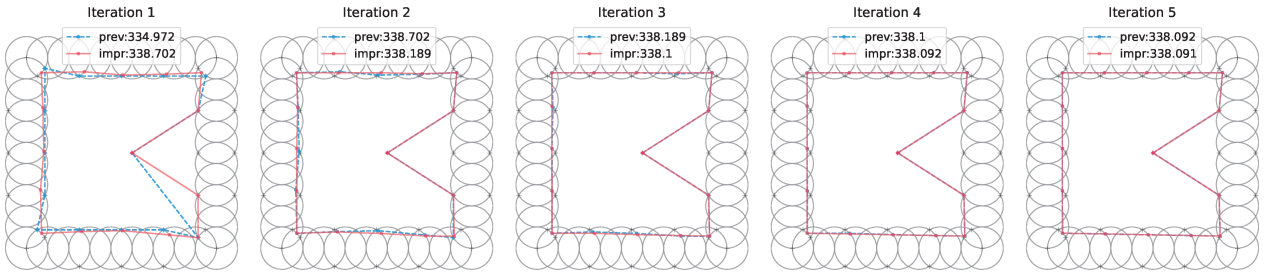


Figure 5.6: An example of invoking arc search and add operator for five times on *bubbles1* ($\mathscr{B}_d = 340$). At iteration 1, the arc search operator reduced the path cost. This allows the add operator to insert an additional waypoint.

5.1.8 Approach to CEOP with Non-uniform Neighborhoods

This section explains the design of CRaSZe-AntS for CEOP-$\mathcal{N}$ (see right part of Figure (5.4)). In CEOP-$\mathcal{N}$, only continuous searching of SZs' interiors can evaluate the minimum cost taken to collect prizes, thereby formulating an OP scenario instead. CRaSZe-AntS leverages PSO to refine waypoint positions based on the constructed SZ layout. IACS (with add/drop operators), which serves as the solver for OP instances, is embedded within all particles of the PSO. This dual-layered strategy facilitates the iterative refinement of solutions, where the outcomes from IACS guide the update processes for both the *positions* and *velocities* of all particles.

Our approach to CEOP-$\mathcal{N}$ employs a hybrid metaheuristics (see Algorithm (14)): RSZD for SZ construction, PSO for waypoint positioning, and IACS for optimizing the sequence of waypoints (by solving the OP).

Algorithm 14: PSO and IACS to solve TDDP

Input: Input arguments in Alg. (10); Particle number N_{ptcl}; Iteration number $N_{\text{ptcl}}^{\text{PSO}}$; Max no-improvement number $N_{\text{impr}}^{\text{PSO}}$; Improvement tolerance ε^{PSO}; Acceleration constants C_1, C_2; Min and max inertia weight $\omega_{\min}, \omega_{\max}$.

for $i = 1$ *to* N_{ptcl} **do**
  Initialize position list L_{pos}^{i} with waypoints from SZs and velocity list $L_{vel}^{i} = \mathbf{0}$;
  Solve OP with list L_{pos}^{i} by IACS, record returned prize $\mathscr{P}^{i}$, cost $\mathscr{C}_t^{i}$, and path $\mathbf{S}^{i}$;
  Update individual best-so-far path $\mathbf{S}^{ib}$, prize $\mathscr{P}^{ib}$, cost $\mathscr{C}_t^{ib}$, and position list L_{pos}^{ib};
Update global best-so-far particle's $\mathbf{S}^{gb}$, $\mathscr{P}^{gb}$, $\mathscr{C}_t^{gb}$, and L_{pos}^{gb};
Reset no-improvement counter to 0;
for *iteration* $n_{it} = 1$ *to* $N_{\text{iter}}^{\text{PSO}}$ **do**
  if *no-improvement counter* $\geq N_{\text{impr}}^{\text{PSO}}$ **then** break the loop;
  Update inertia weight $\omega(n_{it})$ by Eq. (5.17);
  for *each particle* k **do**
    Generate random number R_1, R_2;
    for *index* $j = 1$ *to* M **do**
      Update next velocity $\mathcal{V}_j^k(n_{it}+1) = \min\{\mathcal{V}_j^k(n_{it}+1), \mathcal{V}_j^{\max}\}$ by Eqs. (5.16) and (5.18), and next position $\big(x_j^k(n_{it}+1),\ y_j^k(n_{it}+1)\big)$ by Eq. (5.19);
      Restrict $\big(x_j^k(n_{it}+1),\ y_j^k(n_{it}+1)\big)$ to the SZ Ω_j boundary if infeasible;
    Solve OP with list L_{pos}^{k} by IACS, record returned $\mathbf{S}^{k}$, $\mathscr{P}^{k}$, and $\mathscr{C}_t^{k}$;
    if $\big(\mathscr{P}^{k} > \mathscr{P}^{ib}\big)$ ***or*** $\big(\mathscr{P}^{k} = \mathscr{P}^{ib}$ ***and*** $\mathscr{C}_t^{k} < \mathscr{C}_t^{ib}\big)$ **then**
      Update individual best-so-far particle's $\mathbf{S}^{ib}$, $\mathscr{P}^{ib}$, $\mathscr{C}_t^{ib}$, and L_{pos}^{ib};
  if $\big(\ \mathscr{P}^{lb} > \mathscr{P}^{gb} + \varepsilon^{PSO}\ \big)$ ***or*** $\big(\mathscr{P}^{lb} = \mathscr{P}^{gb}$ ***and*** $\mathscr{C}_t^{lb} < \mathscr{C}_t^{gb} - \varepsilon^{PSO}\ \big)$ **then**
    Update global best-so-far particle's $\mathbf{S}^{gb}$, $\mathscr{P}^{gb}$, $\mathscr{C}_t^{gb}$, and L_{pos}^{gb};
  else no-improvement counter +1;
return The global best-so-far path $\mathbf{S}^{gb}$, prize $\mathscr{P}^{gb}$, cost $\mathscr{C}_t^{gb}$, and position list L_{pos}^{gb}.

Particle Swarm Optimization for continuous position optimization. PSO, originally introduced in (Kennedy and Eberhart, 1995), is a metaheuristic inspired by bird flocking behavior. In the context of CEOP-$\mathcal{N}$, PSO represents a solution as a sequence of moving waypoints confined within the valid boundaries of neighborhoods. We use n_{it} to denote the iteration number. For the particle k, the position of its waypoint at path index i during iteration n_{it} is denoted as $\big(\ x_i^k(n_{it}),\ y_i^k(n_{it})\ \big)$. In PSO, the velocity of the next iteration $\big(\ \mathcal{VX}_i^k(n_{it}+1),\ \mathcal{VY}_i^k(n_{it}+1)\ \big)$ determines the motion of waypoints. The velocity of a waypoint at path index i is affected by

the inertia weight ω, individual best-so-far (ib) position $\big(x_i^{ib},\, y_i^{ib}\big)$, and global best-so-far (gb) position $\big(x_i^{gb},\, y_i^{gb}\big)$, which can be updated as:

$$\begin{aligned}\big(\,\mathcal{VX}_i^k(n_{it}+1),\,\mathcal{VY}_i^k(n_{it}+1)\,\big) \;&=\; \omega(n_{it}+1)\,\big(\,\mathcal{VX}_i^k(n_{it}),\,\mathcal{VY}_i^k(n_{it})\,\big)\\ &+\; C_1R_1\big(\,x_i^{ib}(n_{it})-x_i^k(n_{it}),\,y_i^{ib}(n_{it})-y_i^k(n_{it})\,\big)\\ &+\; C_2R_2\big(\,x_i^{gb}(n_{it})-x_i^k(n_{it}),\,y_i^{gb}(n_{it})-y_i^k(n_{it})\,\big)\end{aligned} \tag{5.16}$$

where C_1 and C_2 are the acceleration constants for the individual best-so-far and global best-so-far solutions; R_1 and R_2 are two random variables generated from a uniform distribution ranging in $[0, 1]$. We adopt the Linear Decreasing Inertia Weight (LDIW) strategy, which leads to the lowest errors among all inertia weight strategies presented in (Marini and Walczak, 2015), to determine ω at each iteration:

$$\omega(n_{it}+1)=\omega_{\max}-\frac{\omega_{\max}-\omega_{\min}}{N_{\text{iter}}^{\text{PSO}}}\cdot n_{it} \tag{5.17}$$

In addition to LDIW, additional velocity restrictions are applied according to the geometric property of the SZ. Given the SZ Ω_k and its vertices $\{v_i', ...\}$, we define the magnitude of maximum velocity for this SZ as:

$$\mathcal{VX}_i^{\max}=\mathcal{VY}_i^{\max}=\begin{cases} r_j \quad (\text{for } c_j\in\Omega_k), & \text{if } |\Omega_k|=1\\ \min\Big\{\mathscr{C}_d\big(v'_{\text{ctr}}(\Omega_k),\,\overline{v_i'v_j'}\big)\Big\},\ \text{for } i,j=1,...,|\Omega_k|, i\neq j, & \text{otherwise}\end{cases} \tag{5.18}$$

where $\mathscr{C}_d\big(v'_{\text{ctr}}(\Omega_k),\,\overline{v_i'v_j'}\big)$ is the shortest distance from the SZ center to the line segment $\overline{v_i'v_j'}$, where $\overline{v_i'v_j'}$ is the edge of the polygon formed by SZ vertices. Note that we set the initial waypoint position as a random point located in a circle with center $v'_{\text{ctr}}(\Omega_k)$ and radius $r = \min\Big\{\mathscr{C}_d\big(v'_{\text{ctr}}(\Omega_k),\,\overline{v_i'v_j'}\big)\Big\}$. As suggested by Engelbrecht (2012), the velocity vector should be

initialized as a zero vector. The position at path index i can then be updated as:

$$\left(x_i^k(n_{it}+1),\ y_i^k(n_{it}+1) \right) = \left(x_i^k(n_{it}) + \mathcal{V}\mathcal{X}_i^k(n_{it}+1),\ y_i^k(n_{it}) + \mathcal{V}\mathcal{Y}_i^k(n_{it}+1) \right) \tag{5.19}$$

To guarantee the feasibility of the solution, PSO confines each waypoint to the boundary of the corresponding SZ if its next position falls outside of the SZ. The algorithm establishes a line segment connecting the current and following positions. It subsequently evaluates each circle covered by the SZ to identify the precise boundary point, i.e., the intersection point where the line segment intersects a specific circle. This boundary point can be uniquely determined due to the convex nature of the SZ.

Inherited Ant Colony System for path sequence optimization. Due to maximum velocity constraints, alterations to waypoint positions between consecutive iterations may be restricted. Consequently, the global-best solution obtained in the previous iteration often remains high quality, and IACS can leverage this prior knowledge to enhance the initialization phase. In IACS, the nearest neighbor heuristic is only used to initialize the pheromone matrix in the first iteration. IACS then 'inherits' the global-best ant from the previous iteration to calculate the initial pheromone:

$$\tau_0(n_{it}) = \frac{\mathscr{P}^{gb}(n_{it}-1)}{\mathscr{C}_f^{gb}(n_{it}-1)}, \qquad n_{it} > 1 \tag{5.20}$$

An additional global updating rule is applied to reinforce the impact of the previous global-best solution. Note that in CEOP-$\mathcal{N}$, IACS is used as the solver for OP instead of SOP because PSO can provide a unique waypoint of each SZ to IACS.

5.1.9 Complexity Analysis

Let the number of target circles N be the primary input. For CEOP, CRaSZe-AntS has a worst-case time complexity of $\mathcal{O}(N^2)$ because the most computationally demanding component is the 2-opt operator, which has a complexity of $\mathcal{O}(N^2)$. On the other hand, the benchmark algorithm, GRASP, is an iterative approach with a **base** time complexity of $\mathcal{O}(N^2)$. Under certain conditions, such as when GRASP identifies a high-quality solution that GRASP's local search operator cannot readily improve, the algorithm may terminate prematurely after only a few executions of the 2-opt operator. However, maintaining a time complexity of $\mathcal{O}(N^2)$ cannot be guaranteed for GRASP in practice, particularly for large-scale instances.

For TDDP, CRaSZe-AntS has a worst-case time complexity of $\mathcal{O}(N^2)$ because RSZD has a complexity of $\mathcal{O}(N_{\text{iter}}^{\text{RSZD}} \cdot N_{\text{max}}^{|\Omega|} \cdot N)$ under a constraint of maximum circles, the solution construct phase of PSO and IACS has a complexity of $\mathcal{O}(N_{\text{iter}}^{\text{PSO}} \cdot N_{\text{ptcl}} \cdot N_{\text{iter}}^{\text{IACS}} \cdot N_{\text{ant}} \cdot N)$ and 2-opt local search can have a complexity of $\mathcal{O}(N^2)$. BE-PSO-IACS has the same worst-case complexity of $\mathcal{O}(N^2)$. Though the benchmark algorithms have the same theoretical complexity as our algorithms for addressing CEOP and TDDP, the actual computation time of CRaSZe-AntS is significantly reduced by SZs.

5.1.10 Experiments, Results, and Discussion

This section investigates the advantages of incorporating overlapped neighborhoods, as revealed through a series of computational experiments conducted on instances of CEOP and TDDP. Our algorithm was implemented using the C++20 standard, and we executed experiments on a Linux machine with AMD EPYC 7742 (2.25 GHz and 8 GB RAM). We derive all CEOP and TDDP instances from CETSP instances presented in Mennell (2009). Yang et al. (2018) examined the performance of their algorithm through 10 runs; we increased this to 20 indi-

vidual executions to examine additional repeatability. The results presented in the subsequent tables represent the mean and standard deviations of outcomes obtained from 20 independent executions, where we denote them as AVG and SD, respectively. The following attributes were considered when evaluating the solution performance:

$\mathcal{T}_{\text{alg}}$ Algorithm execution time (s).

$\mathcal{P}$ Total collected prize of the path.

$\mathcal{C}_d$ Total cost of the path in distance metric.

$\mathcal{C}_t$ Total cost of the path in time metric.

5.1.10.1 CEOP with Uniform Neighborhoods

The significance of incorporating overlapped neighborhoods becomes evident when highlighting the prize attribute in CEOP, enabling the simultaneous collection of adjacent prizes. To illustrate this concept further, we explore how RSZD discretizes CEOP into SOP, how ACS (in Section (5.1.7)) solves SOP, and how the arc search optimizes the waypoints' positions.

ACS parameter setting. Because ACS was initially proposed to tackle TSP (Dorigo and Gambardella, 1997), we first examine the performance of ACS in CETSP instances using different parameters. As suggested by Dorigo and Gambardella (1997), we set heuristic importance factor $\beta = 2$, pheromone evaporation rate $\alpha = \rho = 0.1$; exploitation probability $q_0 = 0.9$, and initial pheromone $\tau_0 = (\mathcal{C}_d^{nn})^{-1}$, where $\mathcal{C}_d^{nn}$ is the distance cost of the path yielded from nearest neighbor heuristic (Rosenkrantz et al., 2009). Sixteen pairs of settings, i.e., $N_{\text{ant}} \in \{10, 20, 40, 80\}$ and $N_{\text{iter}}^{\text{ACS}} \in \{125, 250, 500, 1000\}$, were tested in *bubbles1-9*. For example, setting 1 refers to $N_{\text{ant}} = 10, N_{\text{iter}}^{\text{ACS}} = 125$; Setting 2 refers to $N_{\text{ant}} = 10, N_{\text{iter}}^{\text{ACS}} = 250$, etc. Note that *bubbles1-9* are CETSP instances in (Mennell, 2009), characterized by well-organized

neighborhood structures with sufficiently substantial overlaps. Though other CETSP instances, such as *rotatingDiamonds* and *concentricCircles* series, also possess structured layouts, their complexity is lower than that of *bubbles* series when constructing the SZ layout. For instance, most SZs in *rotatingDiamonds* and *concentricCircles* are consecutive SZs of degree 2. Moreover, *bubbles* series exhibits a progressively increasing computational complexity from 1 to 9, which helps to discern the performance difference among evaluated algorithms. To select a general setting for later experiments, we define the score of setting i based on the path distance cost and computational time:

$$score(i) = \sum_{bubbles} \frac{1}{2} \left(\frac{\max_i \left\{\mathcal{C}_d(i)\right\} - \mathcal{C}_d(i)}{\max_i \left\{\mathcal{C}_d(i)\right\} - \min_i \left\{\mathcal{C}_d(i)\right\}} + \frac{\max_i \left\{\mathcal{T}_{\text{alg}}(i)\right\} - \mathcal{T}_{\text{alg}}(i)}{\max_i \left\{\mathcal{T}_{\text{alg}}(i)\right\} - \min_i \left\{\mathcal{T}_{\text{alg}}(i)\right\}} \right) \quad (5.21)$$

where $i = 1, ..., 16$ refers to the case number of *bubbles*; We denote the maximum (and minimum) achieved path distance cost and computational time among all settings within each *bubbles* instance as $\max_i \left\{\mathcal{C}_d(i)\right\}$ and $\max_i \left\{\mathcal{T}_{\text{alg}}(i)\right\}$ ($\min_i \left\{\mathcal{C}_d(i)\right\}$ and $\min_i \left\{\mathcal{T}_{\text{alg}}(i)\right\}$), respectively.

Table (5.2) summarizes the scores of all settings, which indicates that setting 10 (i.e., $N_{\text{ant}} = 40, N_{\text{iter}}^{\text{ACS}} = 250$) obtained the highest scores among all instances. Therefore, in all experiments of the following sections, except when indicated differently, N_{ant} and $N_{\text{iter}}^{\text{ACS}}$ were set to 40 and 250, respectively. Additionally, we established a minimum improvement tolerance as $\varepsilon^{\text{ACS}} = 10^{-4}$, which means ACS would terminate if the fitness difference is less than ε^{ACS} for several iterations. To balance the computation time and solution quality, we allow a maximum number

Table 5.2: Scores achieved with varying ACS settings

N_{ant} \ N_{iter}	125	250	500	1000
10	4.57	5.78	6.09	5.77
20	6.16	6.21	6.31	5.74
40	6.73	**7.03**	6.14	5.05
80	6.80	6.93	6.08	4.20

The **bold** value indicates the best result.

of no improvements as $N_{\text{iter}}^{\text{ACS}} / 10 = 25$.

Performance of discretization schemes. To illustrate the computational and strategic distinctions between single and overlapped neighborhood strategies for discretizing the CEOP, we compare our RSZD to the IPD scheme (Carrabs et al., 2017). We select the IPD scheme because it represents one of the few strategies that effectively discretizes CETSP into the GTSP, and this can be mapped well to the CEOP scenario. Moreover, as evidenced in (Carrabs et al., 2017), IPD outperforms the perimetral discretization scheme proposed in (Behdani and Smith, 2014). We set the iteration number $N_{\text{iter}}^{\text{RSZD}} = 10$. The selection of this value is determined by tests involving a range of $[5, 10, 15, 20]$. The performance difference is insignificant, mainly because we are testing on organized instances (i.e., *bubbles* series). The budget level for each instance is set to 90%, 60%, and 30% of the best-known cost of its respective CETSP instance (Placido et al., 2022). For example, budgets $\mathscr{B}_d$ for *bubbles1* are set to 314.22, 209.48 and 104.74, respectively (the best-known cost is 349.13). We used three discrete points to represent a single circle, keeping the same configuration as (Carrabs et al., 2020).

Table (5.3) shows the results achieved by ACS when utilizing the waypoint layouts generated by RSZD and IPD. RSZD's layout exhibited superior computational efficiency and solution quality performance across all tested scenarios, outperforming IPD. For instance, in *bubbles5* (with 250 target circles), the RSZD-ACS yielded solutions approximately 30 times faster than IPD-ACS, with a prize increase of 24.32%. Furthermore, ACS can yield more robust solutions by employing RSZD. For example, RSZD-ACS exhibits a Standard Deviation (SD) of 17.81 seconds in computation time and 20.44 in prize collection across 20 individual executions when applied to *bubbles9* with a 90% budget level. In contrast, the SD values for IPD-ACS are substantially higher, with 510.09 seconds for computation time and 35.3 for prize collection. The magnitude of 35.3 remains acceptable because targets' prizes are within a predefined range $[1, 12]$ in *bubbles9*. It is essential to highlight that IPD generates three times as many discrete

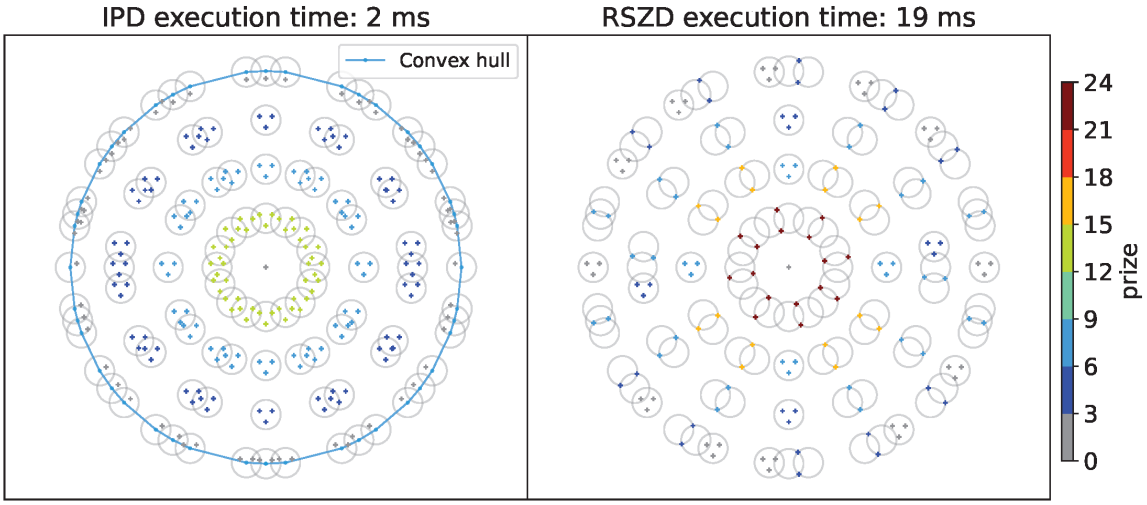
Table 5.3: Performance summary of IPD-ACS and RSZD-ACS (first discretize the CEOP and then solve the SOP).

Instances	$\mathscr{B}_d$	IPD-ACS						RSZD-ACS					
		$\mathcal{T}_{\text{alg}}$		$\mathcal{C}_d$		$\mathcal{P}$		$\mathcal{T}_{\text{alg}}$		$\mathcal{C}_d$		$\mathcal{P}$	
		avg.	SD	avg.	SD	avg.	SD	avg.	SD	avg.	SD	avg.	SD
bubbles1	314.22	1.72	0.03	313.84	0	312	0	**0.05**	0	298.26	0	**348**	0
bubbles2	385.45	10.24	3.14	383.98	1.17	514.4	7.09	**0.21**	0.07	381.59	2.36	**640.6**	5.26
bubbles3	476.96	26.51	7.61	475.86	0.92	639.3	18.8	**1.1**	0.42	474.44	1.77	**842.6**	11.58
bubbles4	724.91	66.23	20.38	722.48	2.13	886.7	7.68	**2.76**	1.12	721.79	1.97	**1121.8**	15.05
bubbles5	934.34	114.75	33.8	932.68	1.08	1024.1	11.58	**4.02**	1.14	930.82	2.73	**1273.2**	7.98
bubbles6	1106.69	204.07	61.13	1104.51	1.43	1133.9	18.32	**6.15**	1.97	1103.57	2.02	**1356.1**	6.11
bubbles7	1446.58	353.65	91.28	1444.15	1.92	1307.45	13.01	**8.75**	4.2	1441.86	4.29	**1554.85**	13.13
bubbles8	1752.05	860.68	344.38	1750.77	0.77	1846.65	37.17	**17.56**	9.22	1747.53	2.35	**2250.9**	24.38
bubbles9	2033.3	1262.54	510.09	2032.03	1.13	2822.2	35.3	**23.9**	17.81	2028.87	3.45	**3384.55**	20.44
bubbles1	209.48	1.2	0.01	206.47	0.55	168	0	**0.04**	0	192.12	0	**192**	0
bubbles2	256.97	6.12	1.42	255.64	0.85	306.2	4.47	**0.14**	0.03	253.98	2.5	**382.8**	4.92
bubbles3	317.98	16.4	5.55	316.53	1.11	392.6	6.3	**0.72**	0.23	314.53	2.44	**520.5**	6.57
bubbles4	483.28	49.63	13.99	481.62	1.16	643.2	22.02	**1.55**	0.57	479.47	3.12	**809**	8.16
bubbles5	622.9	88.61	20.25	621.55	1.18	807.8	16.83	**3.15**	1.22	618.14	2.46	**995.4**	13.93
bubbles6	737.8	162.54	60.77	736.47	1.4	885.7	12.24	**6.29**	3.96	731.72	4.57	**1128.9**	17.35
bubbles7	964.39	238.62	57.1	962.46	1.4	1036	17.23	**8.7**	4.9	957.11	4.68	**1287.1**	10.94
bubbles8	1168.03	483.18	209.57	1166.54	1.1	1264.7	29.63	**13.02**	6.8	1164.08	3.17	**1579.15**	19.04
bubbles9	1355.53	718.75	220.87	1355	0.47	1973.35	39.46	**17.33**	7.26	1352.6	2.35	**2459.35**	24.73
bubbles1	104.74	0.49	0	103.73	0.44	48	0	**0.02**	0	94.86	7.77	**52.8**	5.88
bubbles2	128.48	2.32	0.38	125.57	0.25	84	0	**0.05**	0.01	123.66	5.34	**106.8**	8.77
bubbles3	158.99	5.78	0.99	158.22	0.58	129	2.49	**0.23**	0.06	156.87	0.04	**176**	0
bubbles4	241.64	17.98	6.18	240.62	1.01	268.2	5.86	**0.53**	0.17	238.97	1.93	**339.8**	4.47
bubbles5	311.45	40.37	12.18	310.5	0.95	385.7	6.21	**0.97**	0.32	307.5	2.36	**481.6**	3.67
bubbles6	368.9	68.19	20.04	367.75	1.02	469.2	11.43	**1.8**	0.76	365.76	2.53	**596.2**	6.87
bubbles7	482.19	129.26	41.56	480.91	1.14	641.1	24.92	**3.83**	1.55	477.35	4.36	**805.9**	11.97
bubbles8	584.02	190.93	55.01	582.62	0.99	760.2	18.93	**5.86**	3.35	577.92	4.11	**941.9**	11.99
bubbles9	677.77	285.4	83.29	676.94	1.36	877.7	8.61	**7.83**	3.64	675.54	1.86	**1104.7**	12.41

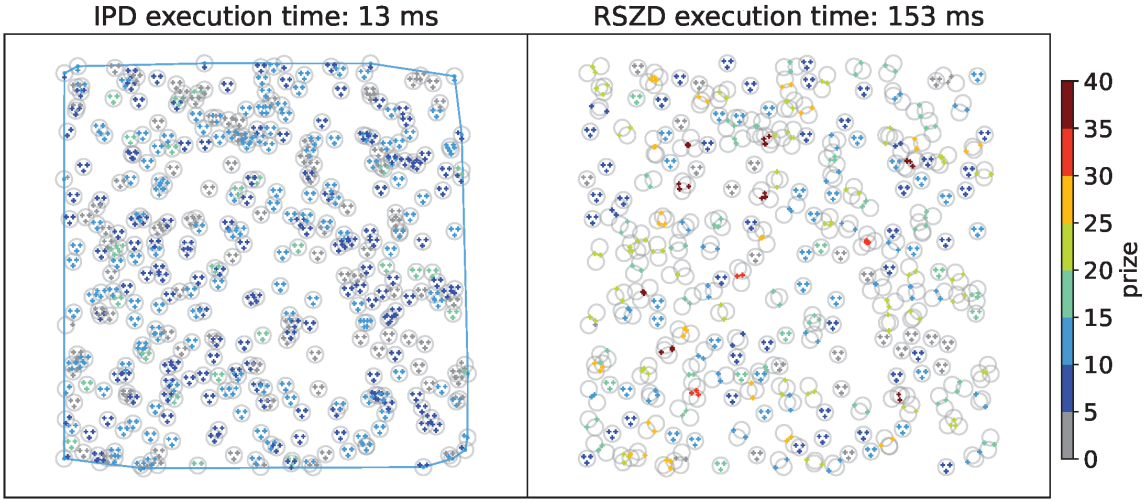
For an OP-related solution, a solution's quality is higher than another when it achieves a higher prize or a lower cost when the prizes are the same. The layout from RSZD outperforms IPD's layout across all instances tested, numerically with an average $\mathcal{T}_{\text{alg}}$ decrease of -96.95% and $\mathcal{P}$ increase of 24.03%. Furthermore, employing RSZD makes the ACS yield more robust solutions because CRaSZe-AntS has significantly reduced SD for $\mathcal{T}_{\text{alg}}$ and $\mathcal{P}$ across *bubbles1-9*.

points as ACS's input, while RSZD extracts SZs to reduce the input size (at worst, the same as the original graph). Furthermore, though IPD is proven effective for CETSP (Carrabs et al., 2017), it exhibits less effective performance in CEOP due to its accumulated discretization

error. This limitation can hinder ACS's ability to incorporate additional targets while adhering to budget constraints. Figure (5.7) visualizes the discretization layouts of IPD and RSZD for *concentricCircles4* and *rd400* with overlap ratio $\phi_{or} = 0.02$. Note that Mennell (2009) proposed *overlap ratio* to describe the complexity of a CETSP instance, $\phi_{or} = r \,/\, l_{contain}$, where $l_{contain}$ is the length of the smallest square to contain all circles. We clarify the definition of overlap



(a) *concentricCircles4*.



(b) *rd400* with overlap ratio $\phi_{or} = 0.02$.

Figure 5.7: Discretization results of IPD and RSZD. RSZD and IPD have the same worst-time complexity as $\mathcal{O}(N)$, where N is the number of target circles. Here RSZD needs ∼10 times slower computation time than IPD because we set $N_{\text{iter}}^{\text{RSZD}} = 10$.

ratio as $\phi_{or} = r \,/\, \max\big(x_{\max} - x_{\min}, y_{\max} - y_{\min}\big)$, where r is the uniform radius for all target nodes; $x_{\min}$, $x_{\max}$, $y_{\max}$, and $y_{\min}$ are the minimum and maximum 2D coordinates among all nodes (including start and end depot).

Impact of overlapped neighborhoods on CEOP solutions. The solution to our SOP can be regarded as an approximate solution to CEOP because the SOP solution must be a feasible CEOP solution with all waypoints being SZs' vertices. To comprehensively assess the influence of incorporating overlapped neighborhoods on CEOP solutions, we re-solve *bubbles1-9*, employing the arc search operator. Although RSZD draws inspiration from the SZ strategy outlined in (Mennell, 2009; Wang et al., 2019b), none of these algorithms have been selected as benchmark references. This decision is based on the fact that these algorithms primarily focus on generating high-quality SZ layouts, which is not the primary objective of this research. Moreover, re-producing SZVNS (Wang et al., 2019b) is not possible due to the absence of specific technical details, particularly concerning determining the optimal waypoint position of a SZ. Also, we aim to present the advantage of considering overlapped neighborhoods as opposed to single-neighborhood strategies in CEOP and CEOP-$\mathcal{N}$ scenarios. Therefore, our comparative analysis used the Greedy Randomized Adaptive Search Procedure (GRASP) (Štefaníková et al., 2020) as the benchmark algorithm. The overall framework (as stated in Section (2.3)) of GRASP is similar to that of CRaSZe-AntS. Both strategies start by discretizing the continuous CEOP, then optimizing the sequence of waypoints and fine-tuning the continuous positions of waypoints. We summarize the principal distinctions between GRASP and CRaSZe-AntS as follows: (1) The solution construction phase of GRASP predominantly relies on local information such as distance between waypoints, the relationship between circle and line segment, etc. In contrast, ACS utilizes the pheromone matrix to represent the balance between global and local information; (2) 'local search' in GRASP optimizes waypoints' positions within individual circles, while our arc search algorithm optimizes waypoints' positions within SZs.

GRASP is characterized by the operators of 'add location' and 'local search'. The 'add location' operator extends the path by greedily selecting nodes with the lowest visitation cost until the budget constraint is met. In cases where the constructed path exceeds feasibility, GRASP iteratively removes a segment of nodes until the path becomes feasible. Whenever a feasible path is achieved, it is added to a candidate list, from which a high-quality path is randomly selected as the starting path for the subsequent iteration. The solution construction phase continues until no better path can be found. GRASP employs a local search operator to refine the global-best path after the construction phase. Initially, 2-opt is applied to improve the path sequence. Subsequently, GRASP optimizes the waypoint position for a single circle (similarly to this element of the arc search algorithm, introduced in Section (5.1.7)). Lastly, the 'add location' operator is re-invoked to replace nodes in the path with potentially higher-profit nodes.

Table (5.4) summarizes the outcomes achieved by GRASP and CRaSZe-AntS. In small-scale scenarios, such as setting the budget as 30% of the best-known cost, GRASP outperforms CRaSZe-AntS. This is because GRASP focuses on optimizing a promising solution locally while ACS employs multiple ants to explore potential solutions across a broader scope. However, as problem instance scales increase, GRASP's 'elite' mode diminishes. This is primarily due to a considerable increase in the time necessary to identify a promising solution and execute the 'local search' operator. While for CRaSZe-AntS, the SZ layout significantly reduced the instance complexity, facilitating ACS to obtain a high-quality solution swiftly. For instance, when considering a 90% budget allocation in *bubbles7*, which comprises 406 circles, GRASP took around 2.57 times longer to obtain the solution than CRaSZe-AntS. While in *bubbles8*, which involves 90 more circles, GRASP demonstrated a nonlinear time increase ($\sim$3.95 times longer) to obtain the solution over CRaSZe-AntS. Furthermore, CRaSZe-AntS performs more robustly than GRASP. For instance, when considering a 60% budget level in *bubbles9*, GRASP has a SD of 131.5 seconds for computation time and 99.28 for prize collection across 20 individual

Table 5.4: Performance summary of GRASP and CRaSZe-AntS for CEOP.

Instances	$\mathscr{B}_d$	GRASP						CRaSZe-AntS (RSZD-ACS-ARC)					
		$\mathcal{T}_{\text{alg}}$		$\mathcal{C}_d$		$\mathcal{P}$		$\mathcal{T}_{\text{alg}}$		$\mathcal{C}_d$		$\mathcal{P}$	
		AVG	SD	AVG	SD	AVG	SD	AVG	SD	AVG	SD	AVG	SD
bubbles1	314.22	**0.02**	0	308.25	0.94	**357.6**	4.8	0.2	0	295.82	0.07	348	0
bubbles2	385.45	**0.36**	0.16	382.57	1.51	**652**	16	0.75	0.26	378.64	2.91	640.2	5.29
bubbles3	476.96	**2.43**	0.65	475.63	1.01	879.6	28.07	4.52	1.68	471.39	2.79	**879.7**	8.08
bubbles4	724.91	**8.57**	2.79	722.86	1.13	1154	17.62	10.67	4.05	720.56	2.81	**1157.4**	8.99
bubbles5	934.34	20.95	5.29	931.82	1.29	1298.3	10.92	**17.05**	7.31	929.36	2.47	**1305.6**	5.99
bubbles6	1106.69	63.44	13.89	1104.64	0.87	**1379.5**	9.13	**24.71**	10.84	1101.94	2.14	1378.3	3.54
bubbles7	1446.58	130.52	29.68	1445.34	0.73	1589.35	16.09	**50.72**	26.67	1439.14	3.14	**1607.55**	12.36
bubbles8	1752.05	332.82	88.75	1751.33	0.46	2262.6	37.34	**84.2**	34.64	1745.95	3.19	**2322**	5.37
bubbles9	2033.3	790.79	180.68	2032.37	0.71	3455.5	30.35	**175.82**	75.68	2026.14	4.62	**3463.9**	10.46
bubbles1	209.48	**0.01**	0.01	203.84	2.49	205.8	4.28	0.14	0	207.89	0.05	**216**	0
bubbles2	256.97	**0.22**	0.08	254.33	2	372.8	23.28	0.53	0.17	251.55	2.54	**380.8**	3.49
bubbles3	317.98	**1.01**	0.45	316.73	1.18	521.2	20.27	2.11	0.7	312.65	2.73	**533.4**	4.78
bubbles4	483.28	**5.39**	1.89	481.7	1.09	**886.3**	24.39	7.22	3.05	478.08	3.08	825.6	4.88
bubbles5	622.9	15.14	4.6	621.18	1.12	**1063.2**	19.94	**11.59**	4.26	617.84	2.57	1027.5	11.36
bubbles6	737.8	32.75	10.18	736.21	1.15	1156.3	26.36	**22.45**	8.86	730.16	3.95	**1161.7**	7.3
bubbles7	964.39	87.1	18.98	962.98	0.89	1315.05	16.12	**38.66**	16.13	957.79	4.66	**1319.5**	7.21
bubbles8	1168.03	272.39	61.14	1167.42	0.37	1631.7	46.48	**48.2**	22.25	1162.4	3.31	**1639.4**	14.54
bubbles9	1355.53	479.8	131.5	1355.09	0.31	**2576.6**	99.28	**62.58**	24.45	1349.69	2.35	2558.35	14.6
bubbles1	104.74	**0**	0	98.21	2.06	51.6	9.37	0.07	0.02	103.54	3.66	**59.4**	2.62
bubbles2	128.48	**0.01**	0.01	125.66	2.12	**109.2**	6.08	0.18	0.01	121.08	5.66	103.4	7.07
bubbles3	158.99	**0.1**	0.08	156.4	2.26	**181.9**	7.22	0.87	0.23	156.62	0.02	176	0
bubbles4	241.64	**0.66**	0.36	239.62	1.72	**349.6**	24.06	2.24	0.62	237.91	1.7	345	6.56
bubbles5	311.45	**2.16**	0.85	309.97	1.16	**499**	21.07	3.52	0.97	306.27	1.03	484.8	2.4
bubbles6	368.9	**4.91**	2.2	367.81	0.73	**629.5**	25.4	6.4	2.02	362	2.35	601	5.2
bubbles7	482.19	18.03	7.9	481.16	0.78	**869.3**	27.25	**14.24**	5.47	475.18	2.63	821.4	7.3
bubbles8	584.02	47.15	15.25	583.32	0.63	941.65	32.93	**27.59**	8.82	578.38	4.19	**983.3**	9.21
bubbles9	677.77	89.95	29.43	677.26	0.44	**1163.35**	55.76	**29.94**	11.43	672.15	3.48	1144.55	15.04

In small-scale instances, GRASP outperforms CRaSZe-AntS because optimizing a promising candidate is efficient. While for larger-scale instances, e.g. *bubbles5-9*, CRaSZe-AntS demonstrated an averaged $\mathcal{T}_{\text{alg}}$ reduction of -40.6% and $\mathcal{P}$ decrease of -0.56%. Furthermore, CRaSZe-AntS also demonstrates a reduced SD in terms of $\mathcal{T}_{\text{alg}}$ and $\mathcal{P}$, compared to GRASP.

executions. Note that in *bubbles9*, the prize for targets has a range of $[1, 12]$; this magnitude of $\mathcal{P}$ SD may represent more than eight targets unvisited in the solution path.

For CRaSZe-AntS, SDs associated with prize collection remain consistently limited in magnitude, explicitly staying below 15 across all instances. On the contrary, this finding indicates

that the SZ strategy narrows the range of path cost, leading to a solution with good quality but limited optimality - the SZ strategy increases the lower bound of the solution quality while potentially reducing its upper bound. This is more evident in CETSP scenarios. We take the *bubbles6* instance as an example (see Figure (5.8)). *bubbles6* features extreme scenarios, such as four circles intersecting at a single point. The investigation of such extreme scenarios falls beyond the scope of this research because this 'boundary problem' is typically associated with real-world constraints, such as data precision from GPS triangulation or sensor readings.

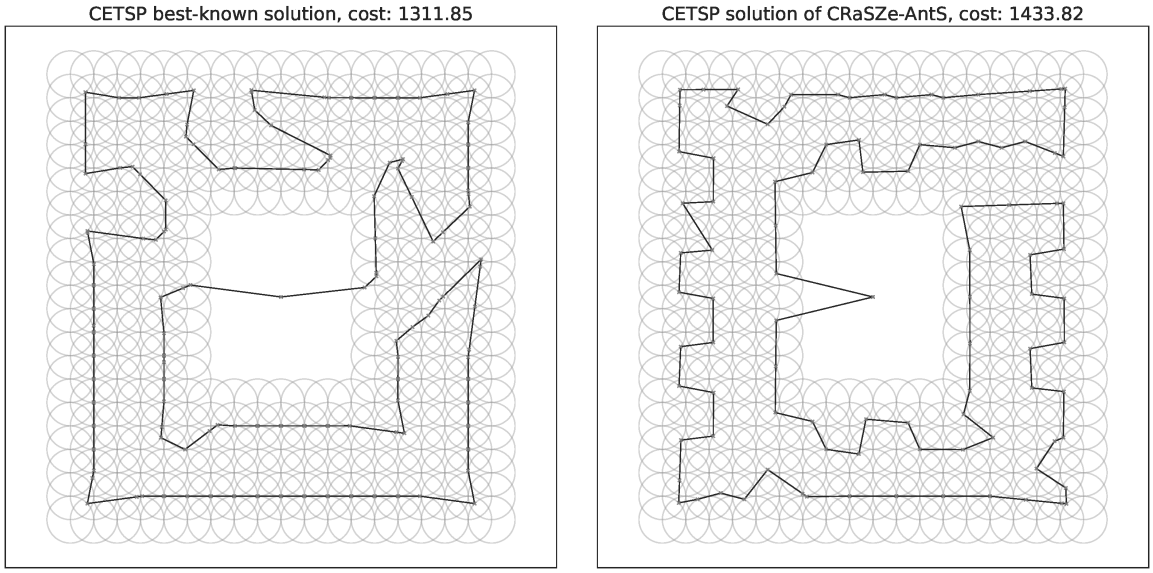
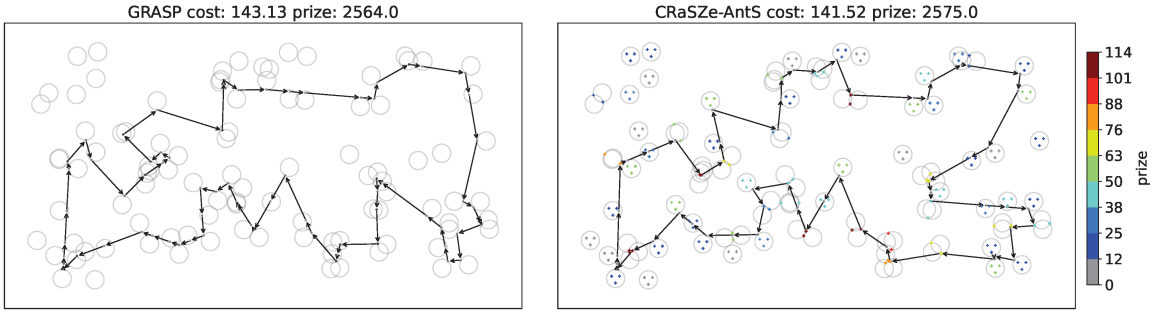
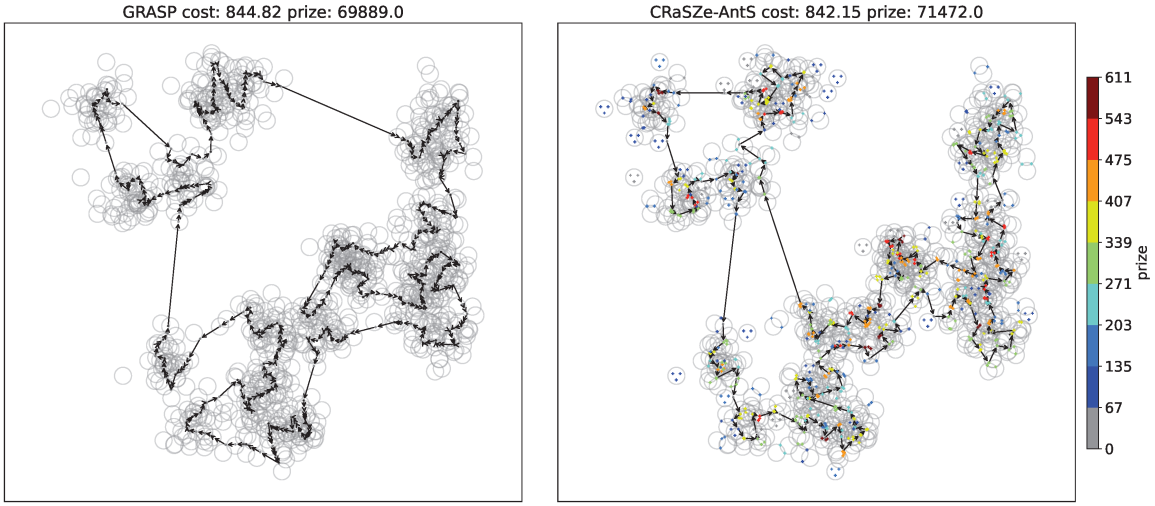


Figure 5.8: Paths of the best-known solution and CRaSZe-AntS in solving CETSP instance *bubbles6*. Compared to CRaSZe-AntS, the best-known path exhibits a notably 'straight' and 'smooth' trajectory. Many SZs of degree four form the 'saw-tooth' pattern in the path. Another relates to extreme scenarios, such as four circles intersecting at a single point.

Figure (5.9) visualizes the solution paths of GRASP and CRaSZe-AntS for *kroD100* (Mennell, 2009) with $\phi_{or} = 0.02$ and *dsj1000* (Mennell, 2009) with $\phi_{or} = 0.02$, respectively. CRaSZe-AntS attained solutions with higher prizes in both instances.



(a) *kroD100* with overlap ratio 0.02 and budget 143.136.



(b) *dsj1000* with overlap ratio 0.02 and budget 844.84.

Figure 5.9: Paths of GRASP and CRaSZe-AntS in solving two CEOP instances.

5.1.10.2 Overlap of Non-uniform Neighborhoods

The advantage of considering overlap is further highlighted when dealing with non-uniform neighborhoods. This section presents solutions obtained through the BE scheme (Yang et al., 2018) (in conjunction with PSO and IACS) and CRaSZe-AntS.

Algorithm parameter setting. We first demonstrate the limitations of the parameter settings recommended in (Yang et al., 2018) for addressing the truck-and-drone delivery problem. Yang et al. (2018) suggested that setting the acceleration constants to $C_1 = C_2 = 2$ is generally effective. However, in the context of overlapped neighborhoods, such settings can lead to a phenomenon termed 'velocity explosion' due to the limited search space in SZs. While

it is possible to confine the waypoint to the SZ's boundary, large acceleration constants can increase the likelihood of the waypoint wandering around the boundary, thereby defeating the purpose of an interior search. To mitigate this issue, we opted for a more conservative setting $C_1 = C_2 = 1.33$, which reduces the probability of a particle overflying its momentary target to $\sim$25% (Marini and Walczak, 2015). Additionally, to prevent the PSO from converging prematurely, we increase the minimum inertia to $\omega_{\min} = 0.4$ while maintaining the maximum inertia at $\omega_{\max} = 0.9$. It is worth noting that having a large number of particles (more than 50) can lead to unnecessary and premature computational efforts, as stated in (Marini and Walczak, 2015). Large population size may also conflict with the early stopping strategy, given the increased probability of constructing a distinct solution that outperforms others, particularly within the context of stochastic search. Therefore, we set the number of particles as $N_{\text{ptcl}} = 40$ and iterations $N_{\text{iter}}^{\text{PSO}} = 100$. Additionally, we set the minimum improvement tolerance as $\varepsilon^{\text{PSO}} = 10^{-4}$, so PSO would halt if the fitness difference $< 10^{-4}$ lasts for several iterations. Considering the computational time may be long in large-scale instances because PSO is coupled with ACS, we reduce the maximum iterations allowable without improvements. Specifically, we set this limit for PSO and IACS to $N_{\text{impr}}^{\text{PSO}}/20 = 5$ and $N_{\text{impr}}^{\text{IACS}} = 13$. Though $N_{\text{impr}}^{\text{PSO}}$ and $N_{\text{impr}}^{\text{IACS}}$ are reduced, we employ the same PSO and IACS solver for both the benchmark algorithm and CRaSZe-AntS. This should ensure a fair comparison between the two algorithms. Other settings remain the same as the ACS settings in Section (5.1.10.1).

IACS convergence analysis. To understand the convergence speed of IACS, we conducted a visual analysis of the first 20 iterations of PSO in solving a relatively large instance *team5_499* (Mennell, 2009). Given that the optimization process of path prize may not show a distinct difference, we selected the path cost as the metric to represent convergence. We set $N_{\text{ptcl}} = N_{\text{ant}} = 20$, $N_{\text{iter}}^{\text{PSO}} = 20$, and $N_{\text{iter}}^{\text{ACS}} = N_{\text{iter}}^{\text{IACS}} = 100$ because solution quality is not the main focus in this experiment. Figure (5.10) illustrates the convergence dynamics of IACS and the classic ACS within three randomly selected particles of the PSO swarm. Compared to classic

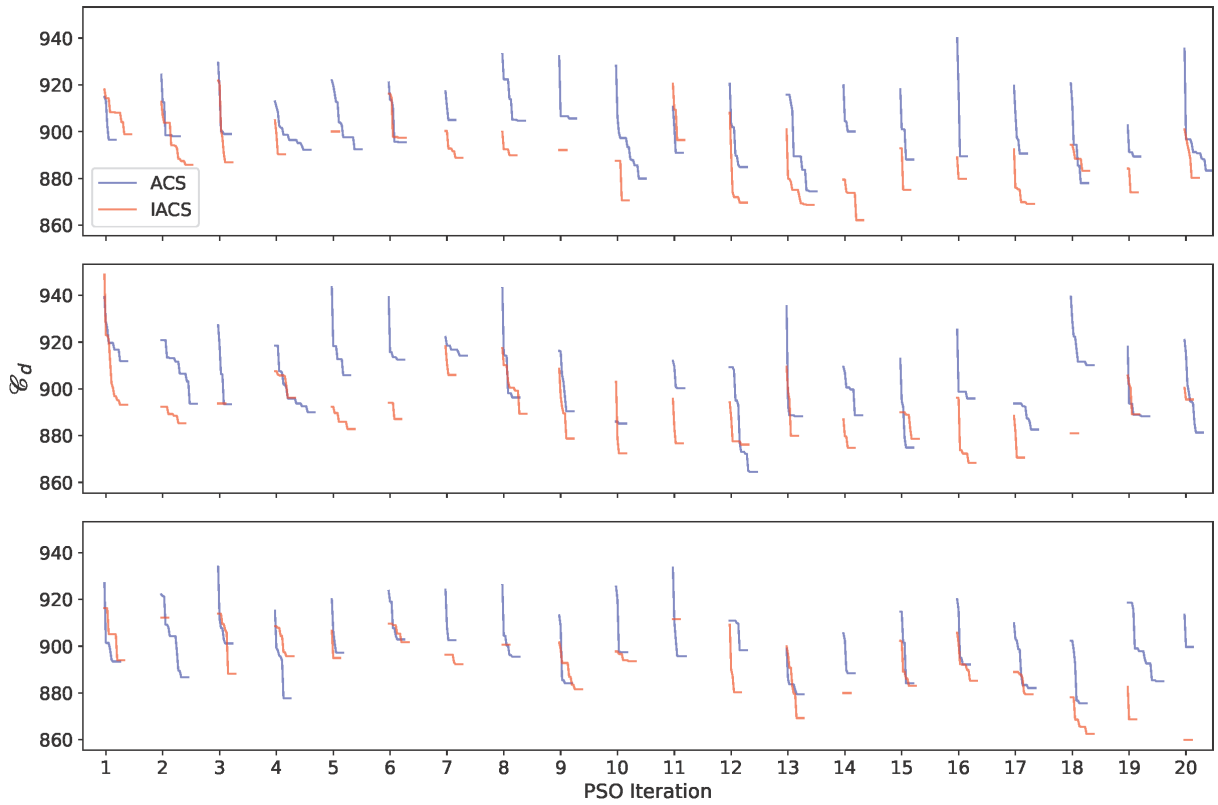


Figure 5.10: Convergence processes of ACS and IACS in the first 20 iterations of PSO when solving *team5_499*. Using the information from the last global-best ants, IACS can generally have a lower starting cost (except for the first iteration).

ACS, IACS can have a lower starting cost and a faster convergence across most iterations.

CRaSZe-AntS performance in TDDP instances. Following (Chang and Lee, 2018), we set parameters of the TDDP model (as stated in Section (5.1.5)) as follows: the UAV average flight speed $\mathcal{V}_{\text{drone}} = 90$ km/h; the UAV average service time $t_{\text{serv}} = 5$ minutes; the maximum service range for the UAV $r_{\text{drone}} = 10$ km; the number of UAVs deployed on a truck $N_{\max}^{|\Omega|} = 5$; the truck average driving speed $\mathcal{V}_{\text{truck}} = 60$ km/h; UAV flight efficiency bounds $\lambda_{\min} = 80\%$, $\lambda_{\max} = 100\%$. The original prizes assigned to targets in CEOP instances are normalized within the interval $[0.1, 0.9]$. This adjustment aims to prevent the pheromone levels from being excessively elevated. We converted distance budgets $\mathscr{B}_d$ into time budgets $\mathscr{B}_t$ by dividing the best-known distance costs by the truck's average speed. Given that TDDP entails additional time costs associated

with prize collection, we set budgets at $60\%, 90\%$, and 120% levels. Note that we set a higher budget level for TDDP instances because here $\mathscr{B}_t$ = best-known CETSP cost / $\mathcal{V}_{\text{truck}}$, which ignores the truck idle time.

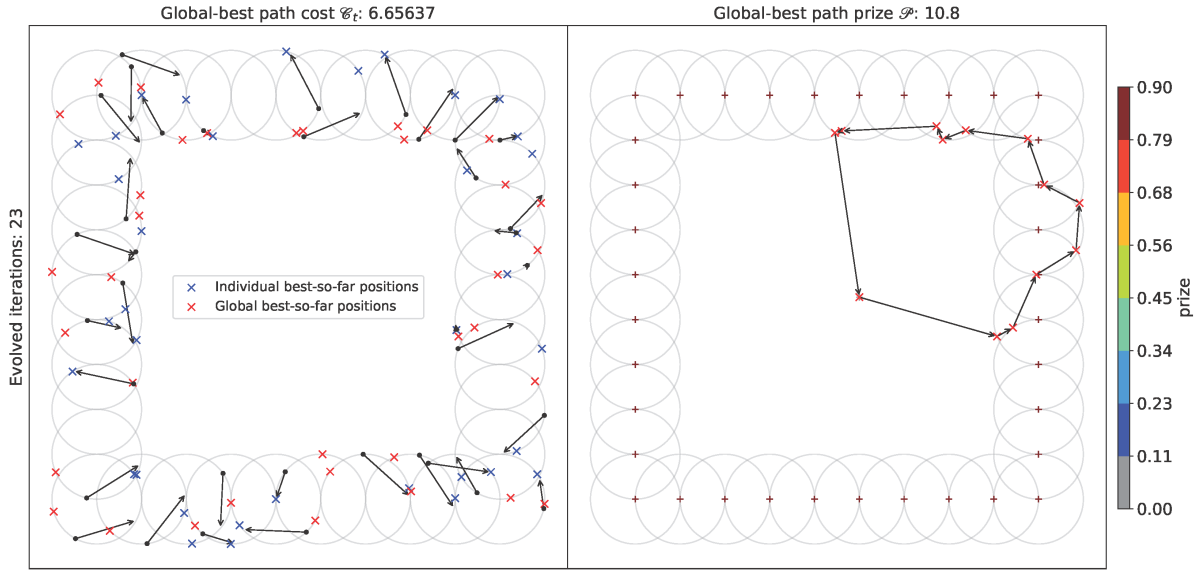
In our comparative analysis, the Hybrid Algorithm (HA) proposed by Yang et al. (2018) emerges as the most analogous algorithm to CRaSZe-AntS for solving our TDDP. HA incorporated PSO with the LDIW strategy for continuous waypoint position optimization and GA with partial mapping crossover for discrete path sequence optimization. Yang et al. (2018) proposed the Boundary Encoding (BE) scheme for arbitrary neighborhoods, simplifying the two-dimensional waypoint coordinates into polar angles, allowing velocity vectors to be confined within the range $\mathcal{V}_j \in [-\pi, \pi]$. In this experiment, we still adopted the BE scheme during PSO optimization while replacing GA with IACS for sequence optimization. This is because a) some parameter configurations of original PSO are inapplicable under TDDP scenarios, as stated in Section (5.1.10.2), which means direct comparison between original HA and CRaSZe-AntS would be unfair; and b) any disparities between GA and IACS can be mitigated.

Table (5.5) presents the outcomes of BE-PSO-IACS and CRaSZe-AntS for *bubbles1-9*. An early termination mechanism has been implemented whereby the system monitors the cumulative execution time before updating the velocity of individual particles. In other words, the maximum allowed execution time is 10 minutes due to the extended computation time for 20 executions of every TDDP instance. Our experimental results reveal marked distinctions between BE-PSO-IACS and CRaSZe-AntS. For example, CRaSZe-AntS significantly reduces computation time and delivers prizes approximately twice as high as those achieved by BE-PSO-IACS (an increasing gap can be observed). Both CRaSZe-AntS and BE-PSO-IACS demonstrate stable performances across all executions. In some large-scale instances, such as *bubbles8-9* with 120% budget level, CRaSZe-AntS and BE-PSO-IACS fail to converge within the time limit. This is because the time required for their initialization phase exceeds 10 minutes.

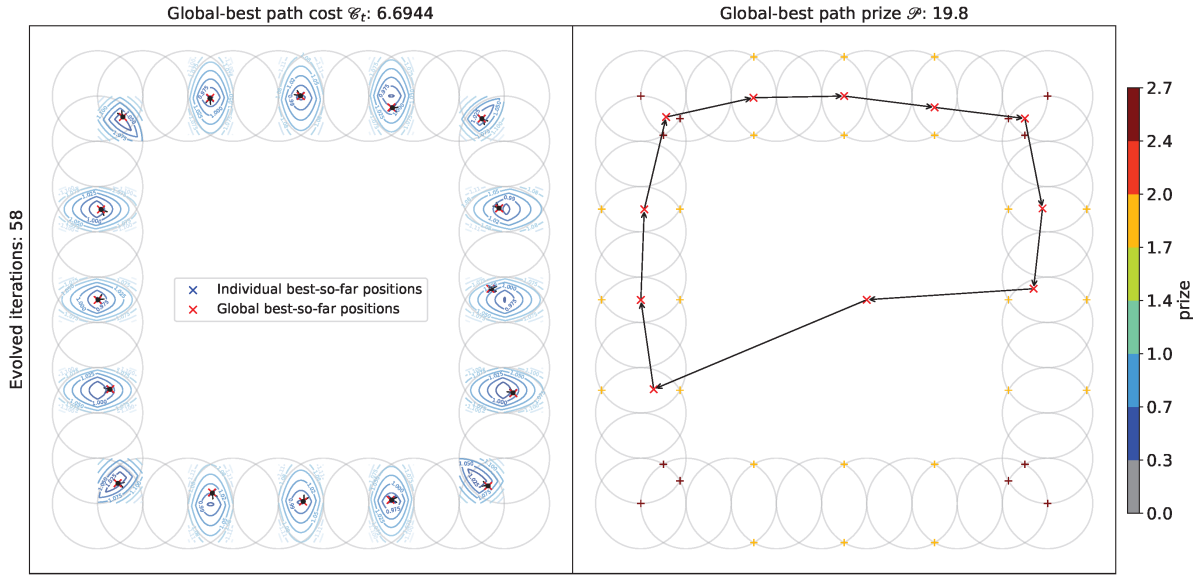
Table 5.5: Performance summary of BE-PSO-IACS and CRaSZe-AntS for TDDP.

Instances	$\mathscr{B}_t$	BE-PSO-IACS						CRaSZe-AntS (RSZD-PSO-IACS)					
		$\mathcal{T}_{\text{alg}}$		$\mathcal{C}_t$		$\mathcal{P}$		$\mathcal{T}_{\text{alg}}$		$\mathcal{C}_t$		$\mathcal{P}$	
		AVG	SD	AVG	SD	AVG	SD	AVG	SD	AVG	SD	AVG	SD
bubbles1	6.98	188.09	48.08	6.72	0.2	10.44	0.44	17.37	2.49	6.78	0.02	**19.8**	0
bubbles2	8.57	276.01	72.83	8.45	0.07	9.38	0.22	42.83	12.09	8.44	0.07	**21.82**	0.86
bubbles3	10.6	369.18	138.59	8.49	1.16	10.18	1.3	163.29	42.36	10.48	0.08	**33.03**	0.16
bubbles4	16.11	591.83	1.41	16.02	0.06	19.8	0.34	539.28	63.46	15.99	0.11	**52.33**	1.24
bubbles5	20.76	595.14	2.5	20.67	0.08	25.87	0.35	596.8	1.09	20.67	0.07	**65.41**	0.48
bubbles6	24.59	598.98	8.23	24.49	0.09	30.09	0.36	597.15	1.41	24.51	0.07	**73.67**	0.46
bubbles7	32.15	734.99	23.02	32.07	0.07	40.56	0.35	598.47	2.81	32.05	0.07	**93.06**	0.36
bubbles8	38.93	1732.97	158.43	38.82	0.09	49.44	0.28	603.3	4.11	38.83	0.08	**113.39**	0.69
bubbles9	45.18	2813.15	228.54	45.07	0.1	61.31	0.34	638.53	41.32	45.07	0.09	**155.05**	1.18
bubbles1	5.24	68.4	14.05	5.08	0.04	8.1	0	16.96	1.83	5.14	0.02	**14.4**	0
bubbles2	6.42	141.42	46.45	6.26	0.06	8.22	0.04	31.38	8.05	6.28	0.09	**15.86**	0.31
bubbles3	7.95	251.09	80.9	5.85	1.12	8.27	0.58	128.87	38.35	7.84	0.09	**22.75**	0.68
bubbles4	12.08	591.08	0.89	12.03	0.06	13.59	0.25	462.7	120.56	12.02	0.06	**39.67**	0.51
bubbles5	15.57	592.54	1.57	15.46	0.09	17.65	0.39	590.81	17.79	15.49	0.07	**50.56**	0.9
bubbles6	18.44	595.34	3.36	18.35	0.07	20.84	0.45	596.34	0.76	18.34	0.06	**56.2**	1.44
bubbles7	24.11	596.14	3.18	23.98	0.09	29.44	0.36	597.56	1.3	24.02	0.07	**74.92**	0.54
bubbles8	29.2	744.29	35.67	29.09	0.09	36.7	0.33	599.33	3.06	29.12	0.06	**90.92**	0.58
bubbles9	33.89	2068.68	178.73	33.78	0.07	45.13	0.5	599.84	3.78	33.74	0.13	**118.07**	1.03
bubbles1	3.49	29.05	4.88	3.23	0.04	4.5	0	7.14	0.78	3.27	0.01	**8.1**	0
bubbles2	4.28	86.07	17.21	4.19	0.08	5.98	0.4	13.87	3.88	4.19	0.07	**9.82**	0.55
bubbles3	5.3	205.73	76.6	5.13	0.07	7.89	0.26	51.29	4.82	5.21	0.02	**16.27**	0
bubbles4	8.05	580.96	25.51	7.95	0.06	10.08	0.2	282.78	104.28	7.95	0.08	**25.45**	0.75
bubbles5	10.38	591.61	1.28	10.28	0.06	10.98	0.27	439.05	107.1	10.28	0.08	**33.79**	0.53
bubbles6	12.3	592.16	1.72	12.18	0.09	12.73	0.4	492.25	91.07	12.22	0.08	**36.9**	1.41
bubbles7	16.07	592.58	2.11	15.97	0.08	17.51	0.37	596.43	0.81	15.99	0.06	**52.07**	0.91
bubbles8	19.47	593.72	3.47	19.37	0.09	22.43	0.33	598.15	1.9	19.35	0.09	**63.21**	0.72
bubbles9	22.59	862.47	45.03	22.49	0.07	27.26	0.42	598.72	2.27	22.51	0.06	**76.95**	0.68

CRaSZe-AntS outperforms BE-PSO-IACS across all instances tested, with an average $\mathcal{T}_{\text{alg}}$ reduction of -55.18% and $\mathcal{P}$ increase of 140.44%.



(a) Solution path generated by BE-PSO-IACS. The left sub-figure refers to the waypoint positions of a randomly selected particle at the last iteration (before break). Some waypoints converged on the global best-so-far positions. The right sub-figure shows the global best-so-far path with a corresponding color bar for circles' prizes.



(b) Solution path generated by CRaSZe-AntS. The left sub-figure refers to the waypoint positions of a randomly selected particle at the last iteration (before break). Most waypoints converged to the global best-so-far positions. The blue contour lines represent the time cost of the SZ. The right sub-figure shows the global best-so-far path with a corresponding color bar for SZs' prizes.

Figure 5.11: Visualization of BE-PSO-IACS's and CRaSZe-AntS's solution in solving TDDP instance *bubbles1*, with given budget $\mathscr{B}_t = 6.9826$.

Figure (5.11) demonstrates the waypoint positions of a randomly selected particle at the last iteration and the global-best solutions generated from BE-PSO-IACS and CRaSZe-AntS. Figure (5.11a) reveals that less than half of the waypoints converged to the global best-so-far positions because of the significant differences between the individual and global best-so-far solutions. This observation indicates the inefficiency of boundary-searching techniques when applied within TDDP. Furthermore, as seen in the right part of Figure (5.11b), the final solution path demonstrates a good balance between accommodating the truck's travel time and waiting time within the budget constraint.

5.1.11 Limitations of CRaSZe-AntS

As the primary focus of this work is to explore the benefits of considering overlaps in solving CEOP and CEOP-$\mathcal{N}$ rather than to generate a high-quality SZ layout, the RSZD scheme may benefit from a minimum-angle strategy and decomposition of high-degree SZs, as presented in (Mennell, 2009). Further enhancement can be achieved through incorporating the boundary encoding scheme (Yang et al., 2018). This scheme facilitates the extension of current SZs into generalized Steiner Zones, i.e., the overlapped regions formed by arbitrary convex neighborhoods. Moreover, a comprehensive sensitivity analysis of CRaSZe-AntS, including understanding the impact of RSZD iterations on *bubbles* series and RSZD's performance on more cluttered or highly overlapping instances, is left for future work. Additionally, while our PSO and ACS solver involves strategies including maximum velocity constraint and inheritance mechanism to improve efficiency, embedding ACS within PSO may be computationally expensive in large-scale CEOP-$\mathcal{N}$ instances. Therefore, considering alternative approaches, such as replacing either PSO or ACS with machine learning techniques like attention mechanisms and reinforcement learning (Kool et al., 2019) may offer computational latency improvements. Though we only demonstrate CRaZSe-AntS to solve two-dimensional routing problems, it is

worth noting that it can be easily extended to solve three-dimensional problems, as presented in Section (5.2).

5.1.12 Conclusion

Although the Close Enough Orienteering Problem (CEOP) has been receiving attention within the research community, few studies explore the interior of the target neighborhood and discuss the potential impact of overlapped neighborhoods within this context. This work introduces an efficient Steiner Zone (SZ) based strategy, CRaSZe-AntS, for addressing the CEOP, specifically emphasizing handling uniform and non-uniform cost functions for neighborhood prize collection. CRaSZe-AntS swiftly constructs a valid SZ layout and identifies SZs' vertices using the proposed Randomized Steiner Zone Discretization (RSZD) scheme. The (near-)optimal positions of waypoints can then be determined quickly by the arc search algorithm based on the geometry of constructed SZs and their vertices. This information can be further utilized for scenarios of non-uniform neighborhoods to define the boundary condition of continuous interior search offered by the Particle Swarm Optimization (PSO) algorithm and maximum velocity constraints in the PSO. Additionally, our solver for determining the sequence order of waypoints, i.e., Ant Colony System (ACS), employs a new unique inheritance mechanism. IACS utilizes prior knowledge between consecutive iterations, which is empirically proven to be generally faster than classic ACS in this scenario. Our experimental results demonstrate that CRaSZe-AntS is computationally efficient and reliable in addressing both CEOP and CEOP-$\mathcal{N}$, compared to state-of-the-art single-neighborhood-based strategies.

5.2 CRaZSe-AntS-3D: on Wireless Data Collection

5.2.1 Introduction

Recent advances in UAV technology have prompted extensive research on Air-to-Ground (A2G) communication scenarios, particularly in data collection from sparse sensor networks deployed in infrastructure-constrained environments. UAVs offer significant advantages over traditional terrestrial wireless networks due to their three-dimensional mobility, which enables extended communication range and improved link quality through higher line-of-sight (LoS) probability and gain. Moreover, UAVs with directional antennas can optimize channel gain by adjusting their attitudes, including steering and tilting operations, to maximize radiation directivity.

While extensive research has investigated UAV positioning for data collection, existing approaches often employ simplified communication models that fail to capture the complex spatial characteristics of directional antenna radiation patterns. For instance, Li et al. (2023); Cariou et al. (2023b) treat the data transmission rate as a binary constant based solely on whether the UAV operates within a predefined circular geometry. Though Feng et al. (2022); Zeng et al. (2019); Ghorbel et al. (2019) introduce SNR-based calculations, their coverage spaces are over-simplified as hemispheres. Guo et al. (2020); Yang et al. (2021) present more sophisticated models incorporating radiation patterns based on cosine functions, but they primarily focus on optimizing UAV deployments instead of exploring full spatial optimization potential and coverage overlaps.

A key insight often overlooked in the existing literature is how receiver sensitivity fundamentally shapes the effective communication region (Detterer et al., 2021). When properly considered, this creates a space similar to a hollow concentric cone rather than the commonly assumed sphere-like or spherical sector-like shapes (see Section (5.2.2)). While more complex, this geo-

metric structure provides opportunities for optimization - particularly in scenarios with overlapping regions (as presented in Section (5.1)) that emerge even in sparse sensor deployments. The challenge then becomes efficiently searching within these three-dimensional conic spaces to find optimal UAV hovering positions that maximize data collection efficiency.

Building upon these insights, we present a comprehensive modeling of A2G wireless communication for our Data Collection Problem (DCP) scenario and introduce CRaZSe-AntS-3D, a novel algorithm designed to handle three-dimensional spatial complexity. Unlike existing approaches that simplify the search space in DCP scenarios, our method directly addresses the challenge of optimizing UAV positions within non-convex cones while considering overlapping coverage space.

5.2.2 Three-dimensional Extension of CEOP-$\mathcal{N}$

Following the convention of three-dimensional CETSP (Coutinho et al., 2016), the extension of CEOP-$\mathcal{N}$ to three-dimensional space transforms the problem from circular to spherical neighborhoods, where waypoints can be positioned anywhere in target spheres. Consequently, the Euclidean distance cost between two sphere centers v_i and v_j is calculated as $\mathscr{C}_d(v_i, v_j) = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}$. Though this extension is relevant for the UAV-assisted data collection scenarios, it is insufficient to describe practical wireless communication scenarios. Therefore, we propose a novel Data Collection Problem below based on three-dimensional CEOP-$\mathcal{N}$ and significantly optimize the feasible solution space utilizing the characteristics of wireless communication devices and environments.

5.2.2.1 Data Collection Problem (DCP) Example

Following the configuration by Guo et al. (2020), the UAV is equipped with a directional antenna, while all Ground Nodes (GNs) are equipped with homogeneous omnidirectional antennas. We first introduce the derivation of the theoretical maximum effective communication region and transfer data rate between the UAV and the GN.

Wireless channel modeling between the UAV and GNs considers device characteristics, environmental factors, and signal propagation effects. Following (Guo et al., 2020), the radiation pattern of UAV's directional antenna, $U(\kappa, \theta_{\text{rad}}) = \cos^{\kappa}(\theta_{\text{rad}})$, is characterized by an antenna parameter κ controlling the significance of directivity and a radiation angle θ_{rad} between the UAV v_{UAV} and the GN v_i. Here, $\theta_{\text{rad}} = \left|\arctan\left(\sqrt{(x_{\text{UAV}} - x_i)^2 + (y_{\text{UAV}} - y_i)^2}/h_{\text{UAV}}\right) - \theta_{\text{atn}}\right|$, where θ_{atn} is the antenna orientation angle to the ground (e.g., $\theta_{\text{atn}} = \pi/2$ in (Guo et al., 2020)), as shown in Figure (5.12). The antenna's beam solid angle $\omega_A(\kappa)$ is given by:

$$\omega_A(\kappa) = \int_0^{2\pi} \int_0^{\pi} U(\kappa, \theta_{\text{rad}}) \sin(\theta)\, \mathrm{d}\theta\, \mathrm{d}\phi = \frac{2\pi}{\kappa + 1} \tag{5.22}$$

where azimuth angle ϕ is the horizontal angle from a cardinal direction. The maximal directivity

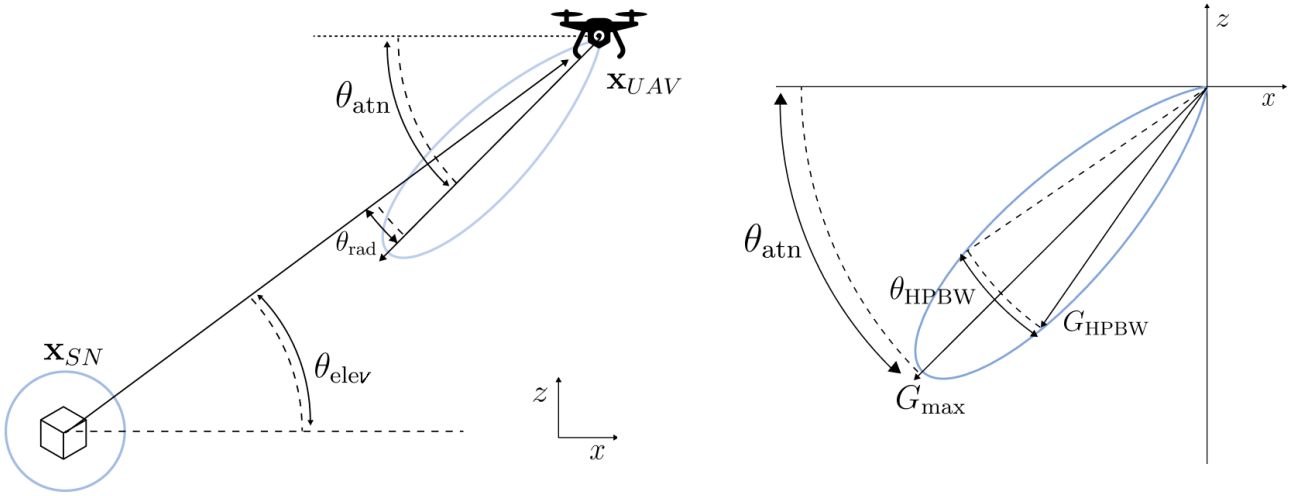


Figure 5.12: The antenna radiation pattern and angle representation.

D_0 and radiation pattern $U(\kappa, \theta_{\text{rad}})$ decides a directional gain:

$$G(\theta_{\text{rad}}, \kappa) = D_0(\kappa)\, U(\kappa, \theta_{\text{rad}}) = 2(\kappa+1)\cos^{\kappa}(\theta_{\text{rad}}) \tag{5.23}$$

$$= D_0 \cos^{\kappa}\left(\left|\arctan\left(\frac{\sqrt{(x_{\text{UAV}}-x_i)^2+(y_{\text{UAV}}-y_i)^2}}{h_{\text{UAV}}}\right) - \theta_{\text{atn}}\right|\right) \tag{5.24}$$

Each GN is equipped with a homogeneous omnidirectional antenna with radiation pattern $U(\kappa, \theta_{\text{rad}}) = |\sin^{\kappa}(\theta_{\text{rad}})|$. This choice addresses practical limitations in aligning radiation directivity, which is generally unknown a priori. Therefore, we approximate the GN antenna gain as a constant $\bar{G}_{\text{TX}}^{\text{GN}}$, neglecting radiation dynamics.

For signal propagation, we consider large-scale fading $\xi_{\mathcal{C}}(v_{\text{UAV}}, v_i)$ between the UAV at position $\mathbf{x}_{\text{UAV}} = (x_{\text{UAV}}, y_{\text{UAV}}, h_{\text{UAV}})$ and GN at $\mathbf{x}_i = (x_i, y_i, 0)$. Under Line-of-Sight (LoS) and Non-LoS (NLoS) conditions, the fading effect is calculated as (Azari et al., 2019):

$$\xi_{\mathcal{C}}(\mathbf{x}_{\text{UAV}},\, \mathbf{x}_i) = A_{\mathcal{C}} \left(\mathscr{C}_d(\mathbf{x}_{\text{UAV}},\, \mathbf{x}_i)\right)^{-\alpha_{\mathcal{C}}}, \quad \mathcal{C} \in \{\text{LoS},\, \text{NLoS}\} \tag{5.25}$$

where $A_{\mathcal{C}}$ is the path loss at outdoor reference distance and $\alpha_{\mathcal{C}} \in [2, 4]$ is the path loss exponent. Small-scale fading effects are negligible because they generally have unit expectation (Azari et al., 2019) and are not dominant in rural environments (Yang et al., 2021). The received power is then given by:

$$\text{P}_{\text{RX}} = \text{P}_{\text{TX}}\, \bar{G}_{\text{TX}}^{\text{GN}}\, G_{\text{RX}}^{\text{UAV}}(\theta_{\text{rad}}, \kappa)\, \xi_{\mathcal{C}}(v_{\text{UAV}}, v_i), \quad \mathcal{C} \in \{\text{LoS},\, \text{NLoS}\} \tag{5.26}$$

Following (Ghorbel et al., 2019; Guo et al., 2020; Zeng et al., 2019), we assume the LoS probability follows a logistic distribution dependent on the environmental parameters a, b and elevation

angle θ_{elev} in radians:

$$\mathcal{P}_{\text{LoS}}(\theta_{\text{elev}}) = \frac{1}{1 + a \exp\Big(- b \big(\frac{180}{\pi}\theta_{\text{elev}} - a\big) \Big)} \tag{5.27}$$

where environment-specific parameters satisfy $0 < a < 90$ and $b > 0$. The signal-to-noise ratio (SNR) incorporating both LoS and NLoS components is then given by:

$$\text{SNR} = \frac{\text{P}_{\text{TX}}\, \bar{G}_{\text{TX}}^{\text{GN}}\, G_{\text{RX}}^{\text{UAV}}(\theta, \kappa) \Big(\mathcal{P}_{\text{LoS}}\, \xi_{\text{LoS}}(v_{\text{UAV}}, v_i) + (1 - \mathcal{P}_{\text{LoS}})\, \xi_{\text{NLoS}}(v_{\text{UAV}}, v_i) \Big)}{N_0} \tag{5.28}$$

where N_0 is the average noise power at the receiver. Finally, the Shannon–Hartley theorem gives the theoretical achievable data rate:

$$\mathcal{R} = \text{B}\ \log_2(1 + \frac{\text{SNR}}{\Gamma}) \tag{5.29}$$

where B is the channel bandwidth (Hz), and $\Gamma > 1$ is the discount factor accounting for practical modulation and coding scheme effects (Zeng et al., 2019).

Communication scenarios. We consider downlink and uplink communication scenarios between the UAV and GNs, but with a focus on the uplink case for data collection. In the downlink scenario (UAV as transmitter, GN as receiver), we model a constant energy consumption $\text{E}_{\text{TX}}^{\text{UAV}}$ for the UAV. This simplification is justified by the brief duration (approximately 100 ms (Niculescu et al., 2023)) of the GN wake-up signal transmission. The uplink scenario involves data transmission from a GN v_i at position $\mathbf{x}_i$ to the UAV at position $\mathbf{x}_{\text{UAV}}$, leading to a data rate $\mathcal{R}$ given by Equation (5.29). The receiver's sensitivity and received signal power determine the effective communication region (Qorvo, 2024; Center, 1997; Detterer et al., 2021). We assume each GN transmits at its maximum allowed power level $\text{P}_{\text{TX}}^{v_i}$. Successful signal reception requires the received power to exceed the receiver sensitivity, which is equivalent to

requiring a lower bound for the SNR at the receiver. We denote it as $\text{SNR}_{\min}$.

Our DCP formulation relies on three fundamental assumptions: (a) The UAV equips a directional antenna with a prior-known and deterministic pattern that remains constant during operations; (b) GNs utilize omnidirectional antennas, producing circular radiation patterns; (c) The UAV can establish asynchronous communication links with multiple GNs from a single position through physical adjustments. These assumptions, particularly (a) and (b), indicate that optimal antenna gain positions lie along a ray pattern. To approximate realistic antenna behavior, the framework incorporates the half-power-beam-width (HPBW) angle θ_{HPBW}, resulting in a concentric cone structure that maintains acceptable gain levels. As illustrated in Figure (5.13), this structure is bounded by the maximum allowed communication altitude $H_{\max}$, inner (blue) cone, and outer (black) cone. This geometry varies based on antenna orientation. An edge case occurs with vertical antenna alignment toward the ground, which produces a solid rather than a hollow cone formation. This vertical configuration, however, significantly constrains the UAV's positional flexibility by requiring a hover near the area directly above the GN.

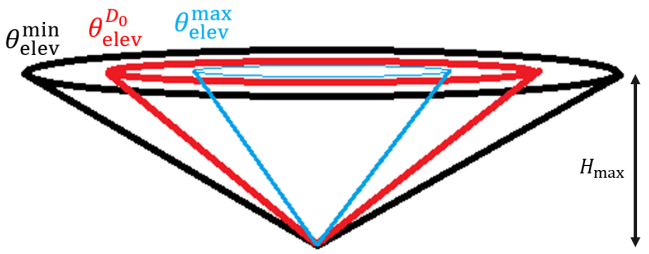


Figure 5.13: The non-trivial communication space bounded by the half-power-beam-width angle θ_{HPBW} and maximum communication altitude $H_{\max}$.

UAV energy consumption model in the DCP comprises two main components: flight operations and wireless communications. The flight operations include takeoff, cruise, landing, and hovering regimes. For a UAV at position $\mathbf{x}_{\text{UAV}}$ targeting a hovering coordinate $\left(x_{\text{hov}}, y_{\text{hov}}, z_{\text{hov}}\right)$, the flight path consists of three phases: ascending to the cruise altitude H_{crs}, cruising to po-

sition $\big(x_{\text{hov}}, y_{\text{hov}}, H_{\text{crs}}\big)$, and finally descending to $\big(x_{\text{hov}}, y_{\text{hov}}, z_{\text{hov}}\big)$. The power required during hovering in no-wind conditions is given by Rodrigues et al. (2022):

$$\mathrm{P}_{\text{hov}} = \sum_{i=1}^{4} \left\{ \frac{Nc\, c_d \rho R^4}{8} \omega_i^3 \right\} + \sqrt{\frac{(mg)^3}{2\rho A}}, \tag{5.30}$$

where N is the number of blades per propeller, c is the blade chord width, c_d is the blade drag coefficient, ρ is the air density, R is the blade radius; ω_i is the angular speed of rotor i; A is the total disk area of rotating blades. This hovering power serves as a parameter estimator for the average power consumption across different flight regimes. Following (Rodrigues et al., 2022), we model the average power consumption using a linear regression:

$$\bar{\mathrm{P}}_{\mathcal{F}} = b_1^{\mathcal{F}} \sqrt{\frac{m^3}{\rho}} + b_0^{\mathcal{F}}, \quad \mathcal{F} \in \{\text{tkf, crs, ldn}\} \tag{5.31}$$

where $b_1^{\mathcal{F}}$ and $b_0^{\mathcal{F}}$ are regression coefficients relating induced power to actual average power, and m is the UAV weight.

For scenarios requiring communication with multiple GNs from a single position, we consider the energy consumed during body steering (yaw axis rotation). Assuming trajectory planning such as (Wang et al., 2022) ensures proper antenna orientation for the UAV arrives at the first GN, the steering energy consumption can be approximated as $\mathrm{E}_{\text{yaw}} \approx \mathrm{P}_{\text{hov}}\, t_{\text{yaw}}$, where t_{yaw} depends on the steering angle and UAV's steering velocity.

The energy consumption for wireless communications varies between downlink and uplink scenarios. In the downlink case, we assume a fixed energy consumption $\mathrm{E}_{\text{TX}}^{\text{UAV}}$. For uplink communications, the energy consumption depends on the UAV receiver's power consumption $\mathrm{P}_{\text{rcv}}^{\text{UAV}}$ and the data collection duration for each GN v_i:

$$\mathrm{E}_{\text{RX}}^{\text{UAV}}(v_i) = \frac{\mathrm{P}_{\text{rcv}}^{\text{UAV}} D_i}{\mathcal{R}\big(\mathbf{x}_{\text{UAV}}, \mathbf{x}_j\big)}. \tag{5.32}$$

DCP formulation. The DCP assigns a single UAV to collect data packages from a set of GNs $\mathbf{V} = \{v_1, ..., v_N\}$. Each GN v_i is characterized by its coordinate $\mathbf{x}_i$, residual voltage level V_i, maximum transmit power P_{TX} and data package size D_i. The key distinction between DCP and 3D CEOP-$\mathcal{N}$ lies in their feasible space constraints. We assume data collection can occur anywhere within an effective communication region, but with varying data-rate throughput and hovering duration. Let v_0 and v_{N+1} denote the start and end depot, respectively. With binary decision variables $Y_{i,j} \in \{0, 1\}$ indicating visitation and continuous time variable t, we formulate DCP as a MINLP problem:

$$\textbf{(DCP)} \qquad \max \sum_{i=0}^{N} \sum_{j=1}^{N} D_j Y_{i,j} \tag{5.33a}$$

$$\text{s.t.} \quad \sum_{j=1}^{N+1} Y_{0,j} = \sum_{i=0}^{N} Y_{i,N+1} = 1 \tag{5.33b}$$

$$\sum_{i=0}^{N+1} Y_{i,k} = \sum_{j=0}^{N+1} Y_{k,j} \leq 1, \qquad k = 2, ..., N \tag{5.33c}$$

$$\sum_{v_i \in \mathbf{S}} \sum_{v_j \in \mathbf{S}} Y_{i,j} \leq |\mathbf{S}| - 1, \qquad \forall\, \mathbf{S} \subset \mathbf{V},\ |\mathbf{S}| \geq 3 \tag{5.33d}$$

$$t^j_{\mathrm{hov}} \geq \frac{D_j Y_{i,j}}{\mathcal{R}\big(\mathbf{x}_{\mathrm{UAV}}(t^j_{\mathrm{start}}), \mathbf{x}_j\big)}, \qquad i \neq j = 1, ..., N \tag{5.33e}$$

$$\mathrm{SNR}\big(\mathbf{x}_{\mathrm{UAV}}(t), \mathbf{x}_j\big) \geq \mathrm{SNR}_{\min} Y_{i,j} \qquad \forall v_j \in \mathbf{V} \tag{5.33f}$$

$$\theta^{\min}_{\mathrm{elev}}\, Y_{i,j} \leq \theta_{\mathrm{elev}}(\mathbf{x}_{\mathrm{UAV}}(t), \mathbf{x}_j),\ \theta_{\mathrm{elev}}(\mathbf{x}_{\mathrm{UAV}}(t), \mathbf{x}_j)\, Y_{i,j} \leq \theta^{\max}_{\mathrm{elev}} \qquad \forall v_j \in \mathbf{V} \tag{5.33g}$$

$$H_{\min} \leq h_{\mathrm{UAV}}(t) \leq H_{\max} \qquad \forall t \in [0, T] \tag{5.33h}$$

$$\sum_{i=1}^{N} \sum_{j=1}^{N} \left\{ \sum_{\mathcal{A} \in \{\text{yaw, TX, RX}\}} \left\{ \int_{t_{\mathcal{A}}} \mathrm{P}_{\mathrm{hov}}\, \mathrm{d}t \right\} + \mathrm{E}^{\mathrm{UAV}}_{\mathrm{TX}} + \mathrm{P}^{\mathrm{UAV}}_{\mathrm{rcv}} t_{\mathrm{RX}} \right\} Y_{i,j} + \sum_{i=0}^{N} \sum_{j=1}^{N+1} \sum_{\mathcal{F} \in \{\text{tkf, crs, ldn}\}} \left\{ \int_{t_{\mathcal{F}}} \mathrm{P}_{\mathcal{F}}\, \mathrm{d}t \right\} Y_{i,j} \ \leq E^{\max}_{\mathrm{bat}} \tag{5.33i}$$

Objective (5.33a) maximizes the total collected data. Constraint (5.33b) ensures the path starts at depot v_0 and ends at depot v_{N+1}. Constraint (5.33c) maintains path connectivity and limits

each GN to at most one visit. Constraint (5.33d) prevents subtours in the solution path. Constraint (5.33e) ensures the UAV collects all data from a target GN. Constraint (5.33f) enforces minimum SNR requirements at hovering positions. Constraint (5.33g) requires the UAV's elevation angle to each GN during communication must range in $\left[\theta_{\text{elev}}^{\min}, \theta_{\text{elev}}^{\max}\right]$. Constraint (5.33f) and Constraint (5.33g) implicitly defines the feasible communication space. Constraint (5.33h) limits the UAV's altitude during the mission must range in $[H_{\min}, H_{\max}]$. Constraint (5.33i) ensures the overall energy consumption from UAV communication and flight operations must not exceed the maximum energy cost.

5.2.3 System Design

DCP requires the optimization algorithm to explore interior spaces of overlapped neighborhoods in $\mathbb{R}^3$. Our approach, CRaSZe-AntS-3D, extends the framework presented in Section (5.1.8), adapting its core components (boundary construction and waypoint positioning) for three-dimensional optimization.

While CRaSZe-AntS defines neighborhood boundaries using planar circles or SZs, CRaSZe-AntS-3D employs curved surfaces to delineate these boundaries in $\mathbb{R}^3$. The key enhancement lies in our three-dimensional PSO formulation. Let $n_{it} \in \mathbb{N}$ denote the current iteration number. For particle k, its waypoint position at path index i during iteration n_{it} is represented as:

$$\mathbf{x}_i^k(n_{it}) = \left(x_i^k(n_{it}), y_i^k(n_{it}), z_i^k(n_{it})\right) \tag{5.34}$$

The particle's motion is governed by its velocity vector for the next iteration:

$$\boldsymbol{\mathcal{V}}_i^k(n_{it}+1) = \left(\mathcal{VX}_i^k(n_{it}+1), \mathcal{VY}_i^k(n_{it}+1), \mathcal{VZ}_i^k(n_{it}+1)\right) \tag{5.35}$$

The velocity vector depends on three components: the inertia weight ω, attraction to the individual best-so-far position $\boldsymbol{x}_i^{ib}(n_{it})$, and attraction to the global best-so-far position $\boldsymbol{x}_i^{ib}(n_{it})$. The velocity update equation is given by:

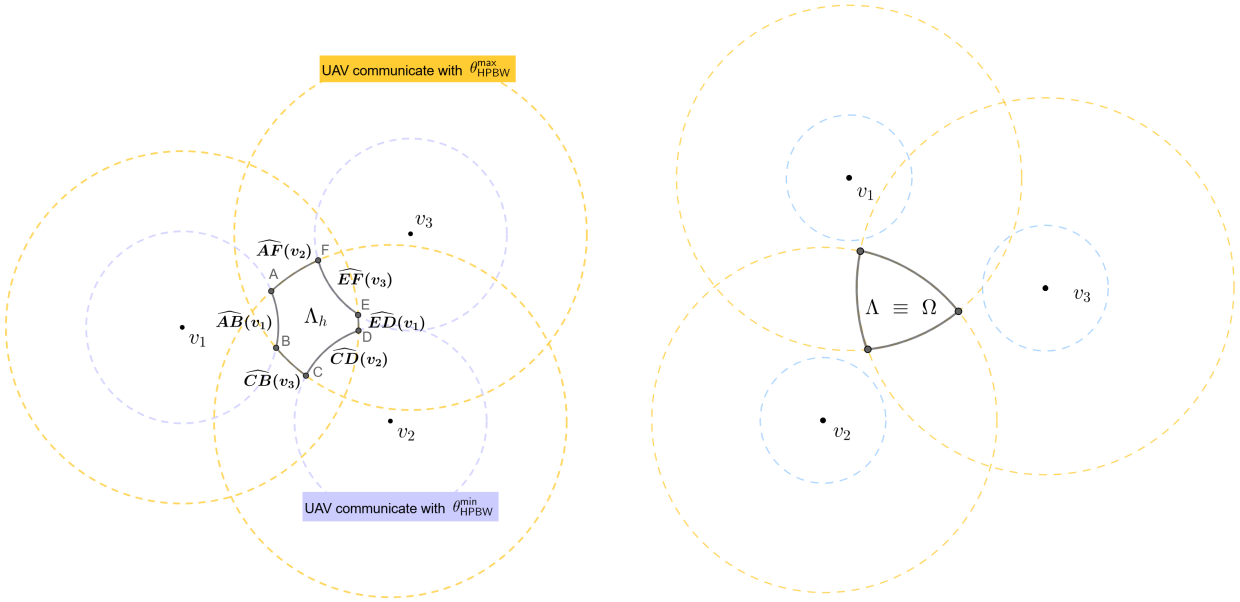
$$\boldsymbol{\mathcal{V}}_i^k(n_{it}+1) = \omega(n_{it}+1)\boldsymbol{\mathcal{V}}_i^k(n_{it}) + C_1R_1\Big(\boldsymbol{x}_i^{ib}(n_{it}) - \boldsymbol{x}_i^k(n_{it})\Big) + C_2R_2\Big(\boldsymbol{x}_i^{gb}(n_{it}) - \boldsymbol{x}_i^k(n_{it})\Big) \quad (5.36)$$

where $C_1, C_2 \in \mathbb{R}^+$ are acceleration constants and $R_1, R_2 \sim \mathbf{U}[0,1]$ are independent uniform random variables. Following the Linear Decreasing Inertia Weight (LDIW) strategy (Marini and Walczak, 2015), ω is updated as:

$$\omega(n_{it}+1) = \omega_{\max} - \frac{\omega_{\max} - \omega_{\min}}{N_{\text{iter}}^{\text{PSO}}} \cdot n_{it} \quad (5.37)$$

where $\omega_{\max}, \omega_{\min} \in \mathbb{R}^+$ denote the maximum and minimum inertia weights.

We introduce a novel spatial discretization approach based on UAV positioning and antenna characteristics for the DCP. This approach extends the concept of SZs into $\mathbb{R}^3$ through discretizing *Annulus Zones* (AZs) at different altitudes, i.e., $\Lambda_h \in \mathbb{R}^2$. Given GN v_i and an altitude h, the effective communication region forms an annulus bounded by concentric circles with radii determined by half-power-beam-width angle θ_{HPBW} and antenna orientation angle θ_{atn}. An AZ Λ_h is the intersection of one or more communication annuli at altitude h. Due to geometric complexity, we decompose an AZ into a SZ and an inner circle set. The feasible space over range $[h_{\min}, h_{\max}]$ is then $\mathbf{\Lambda_H} = \{\Lambda_h : h \in [h_{\min}, h_{\max}]\}$. As illustrated in Figure (5.14a), an AZ Λ_h formed by v_1, v_2, and v_3 is bounded by six circular arcs: $\widehat{AF}(v_2)$, $\widehat{EF}(v_3)$, $\widehat{EF}(v_1)$, $\widehat{CD}(v_2)$, $\widehat{CB}(v_3)$, and $\widehat{AB}(v_1)$. The search space expands with altitude because $\Delta r = r_{\text{out}} - r_{\text{in}}$ increases. Under certain conditions (e.g., Figure (5.14b)), an AZ may reduce to a conventional SZ.



(a) An AZ of degree 3, $|\Lambda_h| = 3$, formed by six circular arcs $\widehat{AF}(v_2)$, $\widehat{EF}(v_3)$, $\widehat{ED}(v_1)$, $\widehat{CD}(v_2)$, $\widehat{CB}(v_3)$, and $\widehat{AB}(v_1)$.

(b) An extreme scenario that the AZ is equivalent to a SZ.

Figure 5.14: Examples of Annulus Zones (AZs).

Given boundary elevation angles (determined by θ_{HPBW} and θ_{atn}) and the minimum SNR requirement in Constraint (5.33f), we can rewrite Equation (5.28) to obtain the communication range limits:

$$\frac{\text{SNR}_{\min} N_0}{\text{P}_{\text{TX}} \bar{G}^{\text{GN}}_{\text{TX}}} = G^{\text{UAV}}_{\text{RX}}(\theta_{\text{rad}}, \kappa) \cdot \sum_{\mathcal{C}} \mathcal{P}_{\mathcal{C}}(\theta_{\text{elev}}) A_{\mathcal{C}} \big(\mathscr{C}_{d,\max}(v_{\text{UAV}}, v_i)\big)^{-\alpha_{\mathcal{C}}}, \ \mathcal{C} \in \{\text{LoS, NLoS}\} \tag{5.38}$$

Given θ_{elev} and θ_{atn}, we can determine θ_{rad}, $G^{\text{UAV}}_{\text{RX}}$, $\mathcal{P}_{\text{LoS}}$, and $\mathcal{P}_{\text{NLoS}}$. Therefore, the maximum distance $\mathscr{C}_{d,\max}$ can be derived by solving a α-power equation (considering $\alpha_{\text{NLoS}} > \alpha_{\text{LoS}}$):

$$\text{A} \cdot (\mathscr{C}_{d,\max})^{\alpha_{\text{NLoS}}} + \text{B} \cdot (\mathscr{C}_{d,\max})^{\alpha_{\text{NLoS}} - \alpha_{\text{LoS}}} + \text{C} = 0 \tag{5.39}$$

$$\text{where constants} \quad \text{A} = \frac{N_0 \, \text{SNR}_{\min}}{\text{P}_{\text{TX}} \, \bar{G}^{\text{GN}}_{\text{TX}} \, D_0 \cos^{\kappa}(\theta_{\text{rad}})}$$

$$\text{B} = \frac{-A_{\text{LoS}}}{1 + a \exp\big(-b\big(\frac{180}{\pi}\theta_{\text{elev}} - a\big)\big)}$$

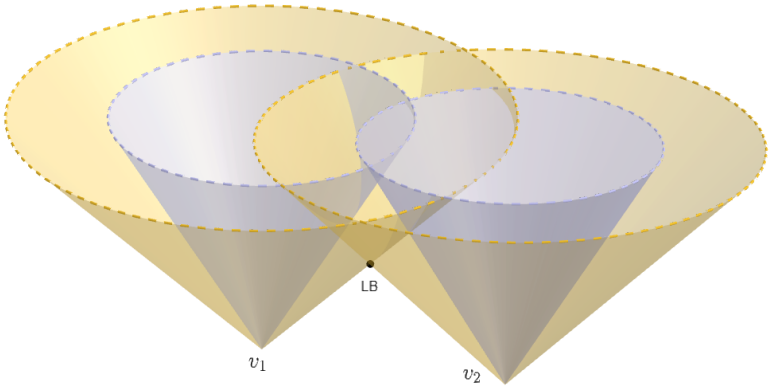


Figure 5.15: The lower bound ($h_{\min}$) of a two-degree annulus space.

$$\mathrm{C} = \frac{-A_{\mathrm{NLoS}}\, a \exp\left(-b\left(\frac{180}{\pi}\theta_{\mathrm{elev}} - a\right)\right)}{1 + a \exp\left(-b\left(\frac{180}{\pi}\theta_{\mathrm{elev}} - a\right)\right)}$$

For an Annulus Zone Λ_h^k at altitude h with vertex set $\mathbf{V}'_{\Lambda_h^k} = \{v'_i, ...\}$, the maximum horizontal velocity components are bounded by:

$$\mathcal{VX}_i^{\max} = \mathcal{VY}_i^{\max} = \begin{cases} \Delta r_k/2, & \text{if } |\Lambda_h^k| = 1 \\ \left(\max\limits_{v' \in \Omega_h^k}\{v'\} - \min\limits_{v' \in \Omega_h^k}\{v'\}\right)/2, & \text{otherwise} \end{cases} \tag{5.40}$$

Theorem 5.1. *For an annulus space $\mathbf{\Lambda_H}$ defined by GNs $\mathbf{V} = \{v_1, v_2, ..., v_N\} \subset \mathbb{R}^2$, let $L_{xy}(p) \perp \overline{v_i v_j}$ denote the perpendicular line at midpoint $p = \frac{v_i + v_j}{2}$ of segment $\overline{v_i v_j}$, and $L_{xyz}(\theta_{elev}, v)$ denote the three-dimensional line through a GN v at elevation angle θ_{elev}. The critical altitude $h_{\min}$ is determined by:*

1. *If there exists a unique intersection point $v_1^{\times}$ of perpendicular lines $\left\{L_{xy}(\frac{v_i+v_j}{2}) : i, j = 1, 2, ..., N, i \neq j\right\}$ locates within the convex hull of $\mathbf{V}$, then $h_{\min} = d_{xy}(v_i, v_1^{\times}) \cdot \tan(\theta_{\mathrm{elev}}^{\min})$.*

2. *Otherwise, for node pair $v_i, v_j = \arg\max_{v_p, v_q \in \mathbf{V}} \left\{d_{xy}(v_p, v_q)\right\}$, let $v_2^{\times}$ be the intersection of $L_{xyz}(\theta_{elev}^{\min}, v_i)$ and $L_{xyz}(\pi - \theta_{elev}^{\min}, v_j)$, then $h_{\min} = z_2^{\times}$.*

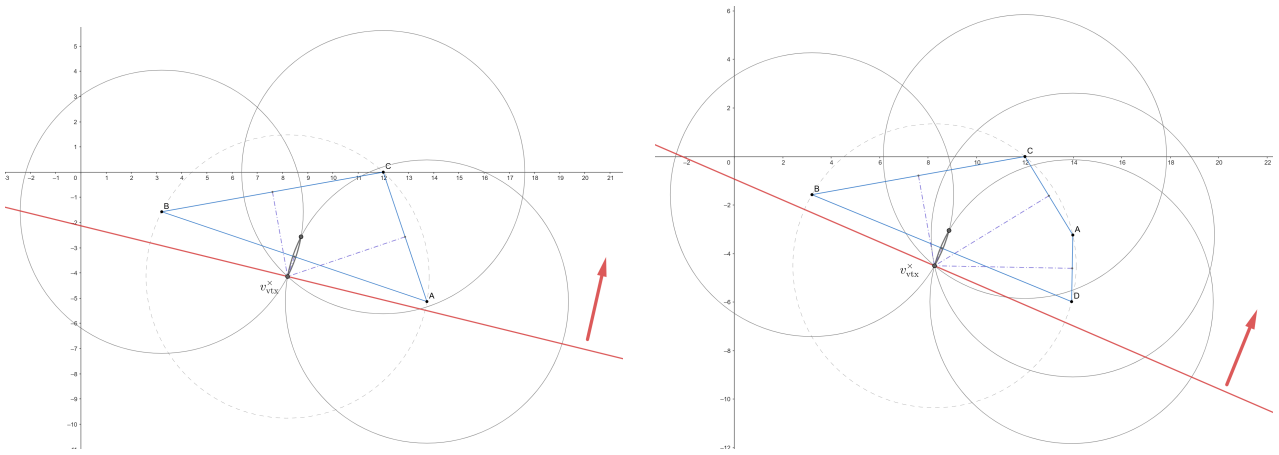
Proof. The critical altitude $h_{\min}$ is determined by the configuration of outer circles, as these circles form the bottom boundary of $\mathbf{\Lambda_H}$. This observation reduces our analysis to a SZ construction problem at $h_{\min}$. For the base case $|\Omega_{h_{\min}}| = 2$, the critical point occurs at the intersection of two conical surfaces with elevation angle $\theta_{\text{elev}}^{\min}$. By symmetry, this represents externally tangent equal-radius circles, where $v_1^{\times}$ and $v_2^{\times}$ necessarily coincide.

For the general case $|\Omega_{h_{\min}}| \geq 3$, we establish:

Lemma 1. *At the critical altitude $h_{\min}$, at least one circle must be tangent to the SZ at either a vertex or an arc point.*

This derives from Conditions 1 and 2 for the SZ construction. If no such tangency exists, we could decrease the altitude while maintaining a valid SZ, contradicting $h_{\min}$'s minimality.

For vertex tangency, all equal-radius circles intersect at a point $v_{\text{vtx}}^{\times}$. By contradiction, if $v_{\text{vtx}}^{\times}$ lies outside the convex hull of $\mathbf{V}$, the separating hyperplane theorem ensures there exists a line L strictly separating $v_{\text{vtx}}^{\times}$ from all points in $\mathbf{V}$, e.g., Figure (5.16). This implies a direction exists in which we can move $v_{\text{vtx}}^{\times}$ to reduce its distance to all nodes simultaneously while maintaining the intersection property, contradicting $h_{\min}$'s minimality. Thus, $v_{\text{vtx}}^{\times}$ must lie within the convex



(a) The example of a polygon with four edges. (b) The example of a polygon with five edges.

Figure 5.16: Examples that the unique intersection point $v_{\text{vtx}}^{\times}$ is outside of convex polygons.

hull of $\mathbf{V}$, coinciding with $v_1^{\times}$ in case-(1).

For arc tangency, only the most distant node pair in $\mathbf{V}$ can establish this scenario, as any other pair's separation is strictly less than this distance, precluding external tangency. The intersection point $v_2^{\times}$ of the corresponding conical surfaces determines the $h_{\min}$ for case-(2).

The theorem is complete if and only if exactly one of these cases occurs. However, in situations where multiple case-(1) and/or case-(2) exist, they necessarily yield identical $h_{\min}$ values. □

The initialization process proceeds as follows. We first select a random altitude $h_0 \sim \mathbf{U}(h_{\min}, h_{\max})$ for each particle in the swarm. At this altitude, the initial position $\mathbf{x}_i^k(0)$ is randomly placed within the feasible region of this AZ Λ_{h_0}. Following initialization with a zero velocity vector in Section (5.1.8), the position is updated as:

$$\mathbf{x}_i^k(n_{it}+1) = \mathbf{x}_i^k(n_{it}) + \boldsymbol{\mathcal{V}}_i^k(n_{it}+1) \tag{5.41}$$

To maintain feasibility, when $\mathbf{x}_i^k(n_{it}+1) \notin \Lambda_h$, we keep projecting the position back until the AZ's boundary. Specifically, we determine the intersection point between the movement vector $\boldsymbol{\mathcal{V}}i^k(n_{it}+1)$ and the AZ boundary by evaluating each constituent arc of Λ_h.

5.2.4 Experiments, Results, and Discussion

This section evaluates CRaSZe-AntS-3D's spatial optimization capabilities. The algorithm was implemented with the C++20 standard, and all experiments were conducted on a Linux platform equipped with an AMD EPYC 7742 processor (2.25 GHz and 8GB RAM). The performance evaluation metrics include total collected data measured in megabytes (MB) and energy-based data collection cost measured in kilojoules (kJ).

5.2.4.1 Parameter Configuration

Our wireless communication simulation parameters are inspired by UWB technology specifications, e.g., the Qorvo DWM1000 (Qorvo, 2016a) operating on channel 5 of the IEEE 802.15.4-a standard with a carrier frequency of 6489 MHz and channel bandwidth $B = 500$ MHz. The UAV employs a directional antenna whose radiation characteristics are illustrated in Figure (5.17). This antenna is configured with specific parameters: maximum directivity gain of 14.7 dBi (12.55 dB), control parameter $\kappa = 17.75$, antenna orientation angle $\theta_{\text{atn}} = 45°$, and HPBW angle $\theta_{\text{HPBW}} = 32°$. Each sensor node is equipped with a homogeneous omnidirectional antenna maintaining a fixed gain $\bar{G}_{\text{SN}} = 4$ dB and transmit power $\text{P}_{\text{TX}} = -10.33$ dBm.

The Line-of-Sight (LoS) probability follows a logistic distribution parameterized by $a = 19$ and $b = 0.4$, as visualized in Figure (5.18). The UAV's communication altitude limits in Constraint (5.33h) are bounded between $H_{\min} = 10$ m and $H_{\max} = 100$ m. Following the path loss model proposed by Azari et al. (2019), we establish path loss at reference distance for LoS and NLoS conditions as $A_{\text{LoS}} = -41.1$ dB and $A_{\text{NLoS}} = -32.9$ dB, with corresponding path loss

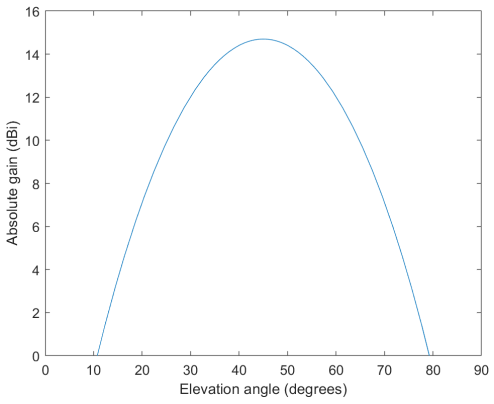


Figure 5.17: Directional antenna characteristics showing radiation pattern and elevation angle-dependent absolute gain. The antenna achieves a peak gain of 14.7 dBi at the main lobe.

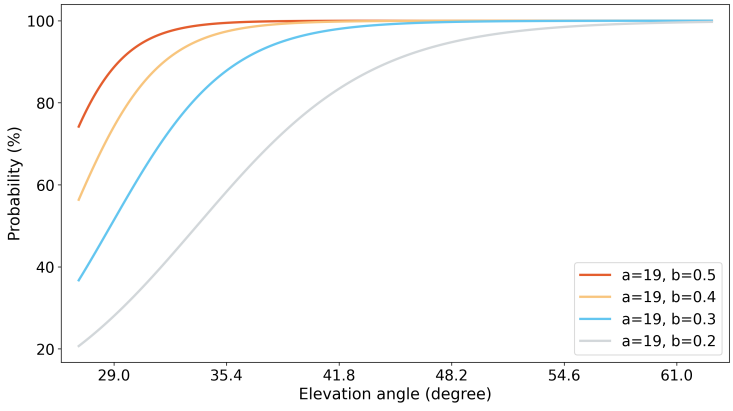


Figure 5.18: Line-of-Sight probability characterization using logistic distributions under varying parameters a and b.

exponents $\alpha_{\text{LoS}} = 2$ and $\alpha_{\text{NLoS}} = 3$. The receiver noise power N_0 is given by (Qorvo, 2016a):

$$N_0 = kTB = 10\log_{10}(1.38065 \times 10^{-23} \cdot 300 \cdot 5 \times 10^{8} \cdot 10^{3}) = -86.84 \text{ dBm}$$

To enhance real-world applicability, we consider regulated data rates according to the IEEE 802.15.14a standard instead of using a constant discount factor Γ. These data rates are 0.11, 0.85, 1.7, 6.81, and 27.24 Mbps, each requiring specific mean Pulse Repetition Frequencies (PRFs), preamble code, and minimum SNRs to achieve a reliable Bit Error Rate (BER). End-to-end UWB physical layer (PHY) simulations were performed on MATLAB (The MathWorks Inc., 2024) to determine the minimum SNR required for each configuration. These simulations were performed on channel 5 (carrier frequency 6489 MHz and bandwidth 499.2 MHz) with a 1024-symbol preamble. Table (5.6) demonstrates different configurations simulated and the resulting minimum SNR for a target BER of 10^{-4}. Therefore, we implement a piecewise data rate function, taking the achieved SNR at the UAV receiver as the input.

Table 5.6: PHY configuration parameters and corresponding minimum SNR requirements for different data rates, established for a target BER of 10^{-4}.

Data rate (Mbps)	Mean PRF (MHz)	Preamble Code	minimum SNR (dB)
0.11	15.6	3	9.45
0.85	15.6	3	13.6
6.81	62.4	9	15.6
27.24	62.4	9	20

The system's power consumption parameters are derived from experimental data. The average hovering power $\bar{\text{P}}_{\text{hov}} = 493.78$ W is obtained from the M100 drone flight dataset (Rodrigues et al., 2021). The DWM1000's power consumption, primarily affecting the UAV's receiver operations, is set to 21.76 dBm based on (Polonelli et al., 2022). The PSO configuration parameters align with those specified in Section (5.1.10.2).

5.2.4.2 Searching within Annulus Space

To demonstrate CRaSZe-AntS-3D's three-dimensional search capabilities in wireless communication scenarios with complex geometric constraints, we focus our investigation on a single annulus space $\mathbf{\Lambda_H}$ as **preliminary results**. The experimental setup consists of three sensor nodes positioned at $\mathbf{x}_{SN1} = (50, 55, 0)$, $\mathbf{x}_{SN2} = (50, 115, 0)$, and $\mathbf{x}_{SN3} = (110, 55, 0)$, each configured to upload an identical data volume of 5 MB. Figure (5.19) demonstrates the structure of an annulus space with degree $|\mathbf{\Lambda_H}| = 3$. Its upper boundary, represented by the top AZ $\Lambda_{h_{\max}}$ at altitude $h_{\max} = 39.79$ m, comprises a feasible area formed by a SZ $\Omega_{h_{\max}}$ with vertices at $(115.22, 85), (80, 120.22), (39.05, 44.05)$ and three inner circles further limits the searchable area. Its lower boundary, defined by $\Lambda_{h_{\min}}$ at altitude $h_{\min} = 23.52$ m, reduces to a single point as it remains above the minimum allowed communication altitude $H_{\min}$ in Constraint (5.33h).

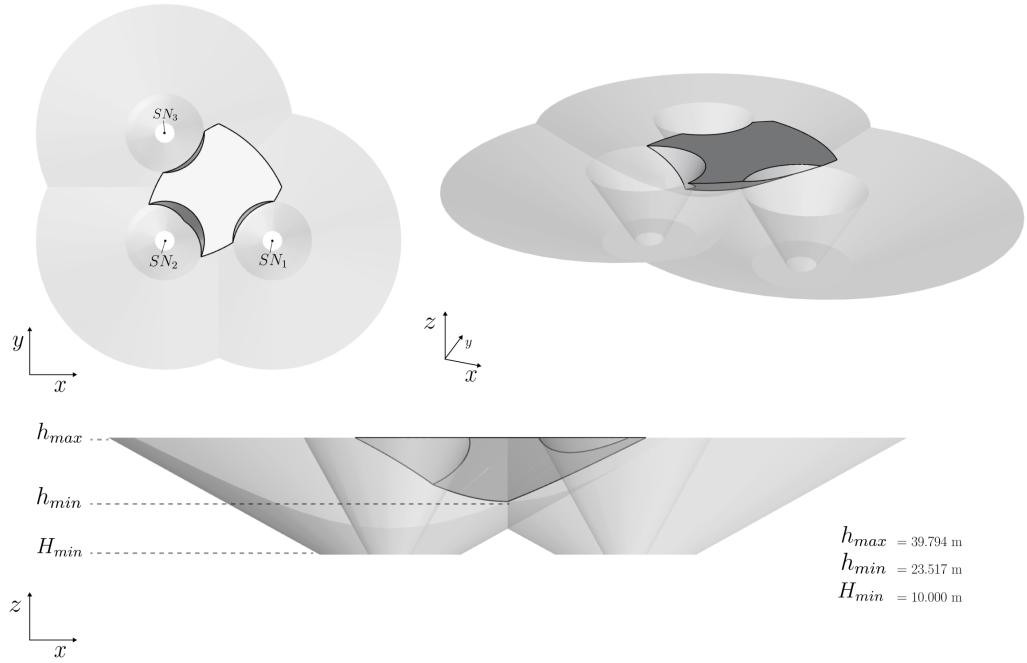


Figure 5.19: Visualization of an annulus space with degree $|\mathbf{\Lambda_H}| = 3$, illustrating the geometric relationship between top and bottom Annulus Zones (AZs). The top AZ $\Lambda h_{\max}$ at 39.79 m is defined by a Steiner Zone with three vertices and constrained by three inner circles, while the bottom AZ $\Lambda_{h_{\min}}$ at 23.52 m converges to a single point above the minimum communication altitude $H_{\min} = 10$ m.

To evaluate the algorithm's search performance, we first establish a comprehensive energy consumption map through an exhaustive search, as shown in Figure (5.20). This analysis reveals substantial variations in energy requirements across different positions, with the optimal cost to collect 15 MB of data from all GNs. The optimal position is $(75.94, 80.94, 35.07)$, leading to an energy consumption of 0.7823 kJ.

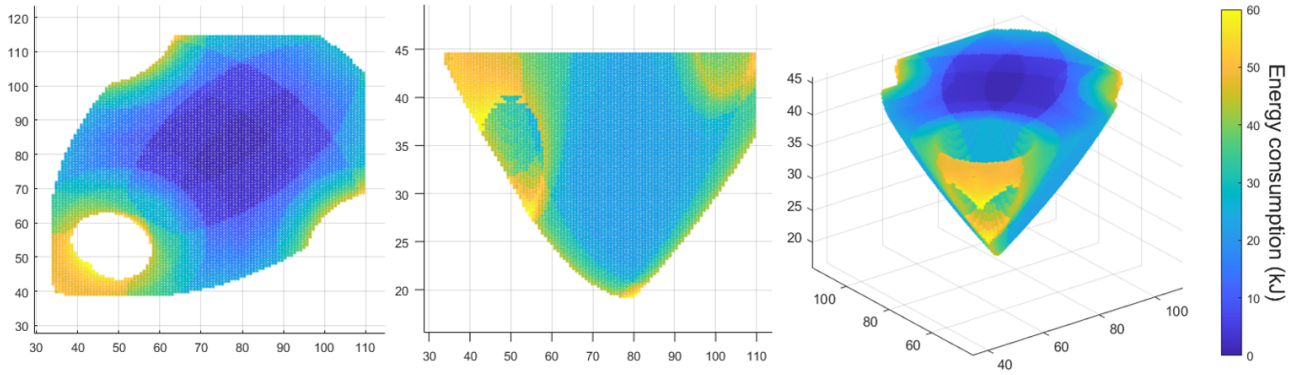


Figure 5.20: Energy consumption distribution within the annulus space, generated through an exhaustive search, revealing significant spatial variations in operational costs and identifying the global optimum at $(75.94, 80.94, 35.07)$ with minimum energy consumption of 0.7823 kJ.

The PSO evolution process is visualized in Figure (5.21), which tracks the movement of ten particles throughout the search iterations. The particles are initially distributed randomly across different altitudes, leading to varied starting positions with different AZs and their corresponding feasible regions. As the optimization progresses, the swarm exhibits collective convergence toward the optimal position, ultimately achieving an energy cost of 0.7853 kJ, closely approaching the global optimum (0.7823 kJ) identified through exhaustive search.

5.2.5 Conclusion and Future Work

This work presents CRaSZe-AntS-3D, an innovative extension of the original CRaSZe-AntS designed to solve CEOP-$\mathcal{N}$, enhancing three-dimensional search capabilities for complex geometric constraints. We evaluate the algorithm's performance through a practical UAV-assisted Data

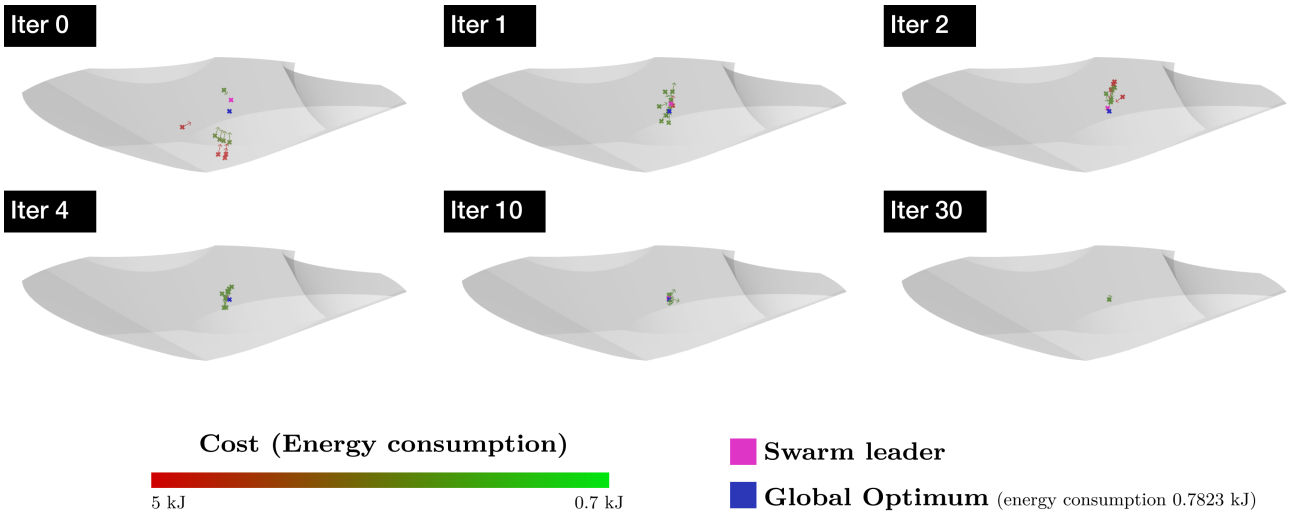


Figure 5.21: Progressive evolution of the PSO algorithm utilizing ten particles, demonstrating convergence from randomly initialized positions at various altitudes toward the optimal solution. The final achieved energy cost of 0.7853 kJ closely approximates the global optimum of 0.7823 kJ identified through the exhaustive search.

Collection Problem (DCP), where directional antenna characteristics create three-dimensional effective communication spaces that can be modeled as hollow concentric cones for individual sensor nodes. The algorithm's continuous interior search capability, implemented through Particle Swarm Optimization (PSO), effectively determines boundary conditions of Annulus Zones (AZs) at varying altitudes and utilizes them to move particles. Our experimental results indicate that CRaSZe-AntS-3D's ability to efficiently navigate and optimize within these complex annulus spaces, consistently approaching the global optimal position.

While this study presents promising initial results, several simplifications were made to facilitate analysis. The level concentric cone examined in Section (5.2.4.2) represents a simplified model of the complete feasible communication space. The antenna's radiation characteristics create more complex spatial patterns in practice, where the gain at maximum directivity can exceed the gain at HPBW angle bounds by 3 dB, resulting in extended communication ranges at specific radiation angles. This phenomenon is illustrated in Figure (5.22), which shows the X-Z cross-sectional view of a complete annulus space using parameters specified in Section (5.2.4.1).

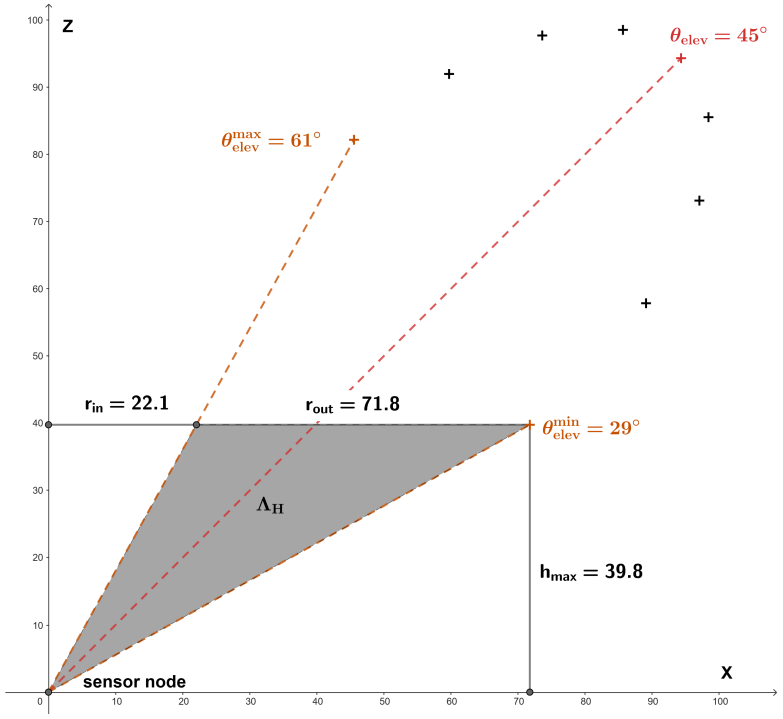


Figure 5.22: Comparative visualization of the simplified annulus space $\mathbf{\Lambda_H}$ against the complete annulus space (extending to the nonlinear boundary curve).

The maximum effective communication range varies with elevation angles, creating a nonlinear and asymmetric boundary curve more accurately representing the actual communication space. Future work will evaluate CRaSZe-AntS-3D's performance with such considerations in more challenging scenarios, particularly in dense sensor node deployments where inner circle intersections may occur. Moreover, our assumption of prior knowledge regarding sensor node data volumes may not hold in practical field deployments, where high-volume sensor nodes may represent high priority, potentially leading to different positioning strategies. A two-stage optimization framework may address this limitation: an offline planning phase utilizing estimated data volumes and deterministic communication modeling, followed by an online optimization phase that enhances mission efficiency and safety by dynamically prioritizing sensor nodes based on actual data volumes and observed communication performance. This approach would better accommodate the uncertainties and variations inherent in real-world data collection tasks. A

limitation of the spatial optimization work presented is that it was validated through simulation. While the simulation models were parameterized with data from real hardware, future work should focus on field experiments to validate the performance of the CRaSZe-AntS framework in real-world conditions. We provide a potential field experiment setup in Section (6.2).

Chapter 6

Conclusion and Future Outlook

6.1 Thesis Achievements

This research tackles three fundamental challenges in practical UAV mission planning. The first is the challenge of real-time energy management under uncertain environments. Traditional approaches often rely on simplified or statistical power consumption models that fail to accurately capture the complexities of real-world operations. The second is the challenge of spatial optimization with complex geometric boundary constraints. While UAVs offer spatial flexibility in mission execution, existing literature lacks comprehensive explorations of interior search and overlap consideration of service spaces. Third is the challenge of developing algorithms that respect onboard computing resource constraints and online requirements, avoiding interference with local controllers.

For energy management under uncertainty, this thesis introduced novel approaches through two key frameworks. The ROMP framework presented a hybrid offline-online planning scheme with wind field segmentation to estimate energy consumption in dynamic environments. Building on

this, the ADAPT framework advanced significantly by employing Bayesian Inference to continuously update power consumption estimates during mission executions, further generalizing to handle uncertain power consumption variations. ADAPT proved valuable when initial energy estimates may be inaccurate or environmental conditions change unexpectedly. The framework demonstrated robust performance by maintaining 100% mission success rates across diverse testing scenarios while delivering comparable solution quality and computation time to existing benchmark approaches.

For spatial optimization, this research developed new methodologies starting from two-dimensional circular constraints with non-uniform prize-collection costs. The CEOP-$\mathcal{N}$ framework revealed how considering overlaps and enabling interior search can improve solutions for specific applications. Our CRaSZe-AntS algorithm demonstrated superior performance in handling these spatial constraints and being computationally efficient. This approach was extended to a practical UAV mission of wireless data collection, highlighting how three-dimensional positioning significantly impacts mission effectiveness, especially in scenarios involving directional antennas and overlapping service spaces.

6.2 Open Challenges and Future Outlook

The impact of this research extends beyond UAV mission planning, opening several critical research directions.

Energy management. While this research provides adaptive approaches for reliable autonomous operations, developing a general robust approach for diverse UAV types (such as wind-sensitive UAVs) remains challenging. Future work could incorporate wind effect parametrization into the pre-trained regression models like (Rodrigues et al., 2022) to enhance the generality of the ADAPT framework.

Spatial optimization. The spatial-aware frameworks developed here, including non-uniform, overlapping Steiner Zones and more complex Annulus Zones, contribute to both Operations Research and Communication communities. Applying the three-dimensional CEOP-$\mathcal{N}$ framework in wireless communication scenarios provides new possibilities to analyze theoretically feasible and non-trivial search regions, potentially leading to improved positioning for air-to-ground communications.

Integration of energy and spatial optimization. While this thesis advanced both areas independently, their integration presents new challenges, e.g., developing unified frameworks that simultaneously optimize positioning with particular energy management under uncertain environments.

Multi-agent systems. The frameworks developed in this research, such as ADAPT and CRaSZe-AntS, could be extended to multi-UAV scenarios. This extension would require reformulations as multi-agent routing problems and new strategies to handle inter-agent coordination while maintaining the robust performance demonstrated in single-agent scenarios.

Hardware-algorithm integration. The gap between theoretical frameworks and practical implementation on resource-constrained platforms remains significant. Future work could focus on developing more lightweight versions of these algorithms with improved performance while operating within hardware limitations.

Field experiments. Field experiments could be conducted to verify the practical performance of the algorithm. For example, in the Data Collection Problem, a potential field experiment would involve:

- Setup: A fully functional quadcopter (e.g., DJI M100 as presented in (Polonelli et al., 2020)) equipped with a directional antenna and UWB transceivers (e.g., EVK1000 (Qorvo, 2016b)) and several[1] UWB-equipped ground sensor nodes are placed at known locations.

[1]Three to five sensor nodes to form a cluster should be enough for this experiment.

These sensor nodes should store a certain amount of data.

- Procedure: The UAV departs from the base station, ascends to the cruise altitude, approaches, and lastly descends to the 3D hovering positions computed by CRaSZe-AntS-3D to collect data from these sensor nodes. After data collection, the UAV follows the same procedure to return to the base. We should record the actual data throughput and energy consumption during transmission. Multiple comparison hovering positions should be tested.

- Validation: The primary validation would be to compare the predicted data throughput and energy consumption with the actual measured data. Note that the validation must also analyze the hovering accuracy error during the mission and its impact on data throughput and energy consumption.

Bibliography

Abichandani, P., Lobo, D., Ford, G., Bucci, D., and Kam, M. (2020). Wind measurement and simulation techniques in multi-rotor small unmanned aerial vehicles. *IEEE Access*, 8:54910–54927.

Alyassi, R., Khonji, M., Karapetyan, A., Chau, S. C.-K., Elbassioni, K., and Tseng, C.-M. (2023). Autonomous recharging and flight mission planning for battery-operated autonomous drones. *IEEE Transactions on Automation Science and Engineering*, 20(2):1034–1046.

Angelelli, E., Archetti, C., Filippi, C., and Vindigni, M. (2021). A dynamic and probabilistic orienteering problem. *Computers & Operations Research*, 136:105454.

Archetti, C., Carrabs, F., and Cerulli, R. (2018). The set orienteering problem. *European Journal of Operational Research*, 267(1):264–272.

Archetti, C., Carrabs, F., Cerulli, R., and Laureana, F. (2024). A new formulation and a branch-and-cut algorithm for the set orienteering problem. *European Journal of Operational Research*, 314(2):446–465.

Arteaga, J. M., Sanchez, J., Elsakloul, F., Marin, M., Zesiger, C., Pucci, N., Norton, G. J., Young, D. J., Boyle, D. E., Yeatman, E. M., Hallett, P. D., Roundy, S., and Mitcheson,

P. D. (2023). High-frequency inductive power transfer through soil for agricultural applications. *IEEE Transactions on Power Electronics*, 38(11):13415–13429.

Atyabi, A., MahmoudZadeh, S., and Nefti-Meziani, S. (2018). Current advancements on autonomous mission planning and management systems: An auv and uav perspective. *Annual Reviews in Control*, 46:196–215.

AVX (2015). *SCC Series Automotive Grade Cylindrical SuperCapacitors*, Rev 4 edition. Available at `https://datasheets.kyocera-avx.com/SCCT20E106SRBQ.pdf`.

Azari, M. M., Rosas, F., and Pollin, S. (2019). Cellular connectivity for uavs: Network modeling, performance analysis, and design guidelines. *IEEE Transactions on Wireless Communications*, 18(7):3366–3381.

Bayliss, C., Juan, A. A., Currie, C. S., and Panadero, J. (2020). A learnheuristic approach for the team orienteering problem with aerial drone motion constraints. *Applied Soft Computing*, 92:106280.

Behdani, B. and Smith, J. C. (2014). An integer-programming-based approach to the close-enough traveling salesman problem. *INFORMS Journal on Computing*, 26(3):415–432.

Bellman, R. (1962). Dynamic programming treatment of the travelling salesman problem. *J. ACM*, 9(1):61–63.

Boysen, N., Briskorn, D., Fedtke, S., and Schwerdfeger, S. (2018). Drone delivery from trucks: drone scheduling for given truck routes. *Networks*, 72(4):506–527.

Brun Castell, D. A. (2022). *Studies of Aerodynamic Performance of Propeller Blades for Sensor Implementation to Study Plumes in Extreme Environments*. PhD thesis, The University of Texas at San Antonio.

Campbell, A. M., Gendreau, M., and Thomas, B. W. (2011). The orienteering problem with stochastic travel and service times. *Annals of Operations Research*, 186(1):61–81.

Cariou, C., Moiroux-Arvis, L., Pinet, F., and Chanet, J.-P. (2023a). Evolutionary algorithm with geometrical heuristics for solving the close enough traveling salesman problem: application to the trajectory planning of an unmanned aerial vehicle. *Algorithms*, 16(1).

Cariou, C., Moiroux-Arvis, L., Pinet, F., and Chanet, J.-P. (2023b). Evolutionary algorithm with geometrical heuristics for solving the close enough traveling salesman problem: Application to the trajectory planning of an unmanned aerial vehicle. *Algorithms*, 16(1).

Carrabs, F., Cerrone, C., Cerulli, R., and Gaudioso, M. (2017). A novel discretization scheme for the close enough traveling salesman problem. *Computers & Operations Research*, 78:163–171.

Carrabs, F., Cerrone, C., Cerulli, R., and Golden, B. (2020). An adaptive heuristic approach to compute upper and lower bounds for the close-enough traveling salesman problem. *INFORMS Journal on Computing*, 32(4):1030–1048.

Center, N. A. W. (1997). Electronic warfare and radar systems engineering handbook. *Electronic Warfare Division, Pont Mugu, CA*.

Chang, Y. S. and Lee, H. J. (2018). Optimal delivery routing with wider drone-delivery areas along a shorter truck-route. *Expert Systems with Applications*, 104:307–317.

Chen, M., Liang, W., and Das, S. K. (2021). Data collection utility maximization in wireless sensor networks via efficient determination of uav hovering locations. In *2021 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 1–10.

Chen, Y., Baek, D., Bocca, A., Macii, A., Macii, E., and Poncino, M. (2018). A case for a battery-aware model of drone energy consumption. In *2018 IEEE International Telecommunications Energy Conference (INTELEC)*, pages 1–8.

Coutinho, W. P., Nascimento, R. Q. d., Pessoa, A. A., and Subramanian, A. (2016). A branch-and-bound algorithm for the close-enough traveling salesman problem. *INFORMS Journal on Computing*, 28(4):752–765.

Croes, G. A. (1958). A method for solving traveling-salesman problems. *Operations Research*, 6(6):791–812.

Detterer, P., Nabi, M., Jiao, H., and Basten, T. (2021). Receiver-sensitivity control for energy-efficient iot networks. *IEEE Communications Letters*, 25(4):1383–1386.

DJI (2016). *Matrice 100 User Manual*. DJI, v1.6 edition. Available at `https://dl.djicdn.com/downloads/m100/M100_User_Manual_EN.pdf`.

DJI (2019). *DJI Manifold 2 Series User Guide*. DJI. Available at `https://dl.djicdn.com/downloads/manifold-2/20190528/Manifold_2_User_Guide_v1.0_EN.pdf`.

Dorigo, M. and Gambardella, L. (1997). Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1(1):53–66.

Engelbrecht, A. (2012). Particle swarm optimization: velocity initialization. In *2012 IEEE Congress on Evolutionary Computation*, pages 1–8.

Evers, L., Barros, A. I., Monsuur, H., and Wagelmans, A. (2014a). Online stochastic uav mission planning with time windows and time-sensitive targets. *European Journal of Operational Research*, 238(1):348–362.

Evers, L., Dollevoet, T., Barros, A. I., and Monsuur, H. (2014b). Robust uav mission planning. *Annals of Operations Research*, 222:293–315.

Faigl, J. (2019). Data collection path planning with spatially correlated measurements using growing self-organizing array. *Applied Soft Computing*, 75:130–147.

Feng, T., Xie, L., Yao, J., and Xu, J. (2022). Uav-enabled data collection for wireless sensor networks with distributed beamforming. *IEEE Transactions on Wireless Communications*, 21(2):1347–1361.

Fischetti, M., González, J. J. S., and Toth, P. (1998). Solving the orienteering problem through branch-and-cut. *INFORMS Journal on Computing*, 10(2):133–148.

Forthofer, J. M., Butler, B. W., and Wagenbrenner, N. S. (2014). A comparison of three approaches for simulating fine-scale surface winds in support of wildland fire management. part i. model formulation and comparison against measurements. *International Journal of Wildland Fire*, 23(7):969.

Friebe, A., Marković, F., Papadopoulos, A. V., and Nolte, T. (2021). Adaptive runtime estimate of task execution times using Bayesian modeling. In *2021 IEEE 27th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, pages 1–10.

Furnon, V. and Perron, L. (2024). OR-Tools routing library.

Gao, C. (2020). *On Routing Unmanned Aerial Vehicles for Surveillance and Reconnaissance Activities*. PhD thesis, State University of New York, Buffalo.

Gendreau, M., Laporte, G., and Semet, F. (1998). A branch-and-cut algorithm for the undirected selective traveling salesman problem. *Networks*, 32(4):263–273.

Ghorbel, M. B., Rodríguez-Duarte, D., Ghazzai, H., Hossain, M. J., and Menouar, H. (2019). Joint position and travel path optimization for energy efficient wireless data gathering using unmanned aerial vehicles. *IEEE Transactions on Vehicular Technology*, 68(3):2165–2175.

Golden, B. L., Levy, L., and Vohra, R. (1987). The orienteering problem. *Naval Research Logistics (NRL)*, 34(3):307–318.

Gonzales, R. (1962). Solution to the traveling salesman problem by dynamic programming on the hypercube. *Operations Research Center Technical Report*, (18):63.

Gulczynski, D. J., Heath, J. W., and Price, C. C. (2006). The close enough traveling salesman problem: a discussion of several heuristics. *Perspectives in Operations Research: Papers in Honor of Saul Gass' 80 th Birthday*, pages 271–283.

Guo, J., Walk, P., and Jafarkhani, H. (2020). Optimal deployments of uavs with directional antennas for a power-efficient coverage. *IEEE Transactions on Communications*, 68(8):5159–5174.

Gurobi Optimization, LLC (2024). Gurobi optimizer reference manual.

Hà, M. H., Bostel, N., Langevin, A., and Rousseau, L. (2012). An exact algorithm for the close enough traveling salesman problem with arc covering constraints. In *ICORES 2012 - Proceedings of the 1st International Conference on Operations Research and Enterprise Systems, Vilamoura, Algarve, Portugal, 4-6 February, 2012*, pages 233–238. SciTePress.

Hatamlou, A. (2018). Solving traveling salesman problem using black hole algorithm. *Soft Computing*, 22(24):8167–8175.

Henry-Labordere, A. L. (1969). The record balancing problem: A dynamic programming solution of a generalized traveling salesman problem. *Revue Francaise D Informatique De Recherche Operationnelle*, 3(NB 2):43.

Jünger, M., Reinelt, G., and Rinaldi, G. (1995). Chapter 4 the traveling salesman problem. In *Network Models*, volume 7 of *Handbooks in Operations Research and Management Science*, pages 225–330. Elsevier.

Kawamura, E. and Azimov, D. (2020). Integrated extremal control and explicit guidance for quadcopters. *Journal of Intelligent & Robotic Systems*, 100:1583–1613.

Kennedy, J. and Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of ICNN'95 - International Conference on Neural Networks*, volume 4, pages 1942–1948 vol.4.

Kim, H., Lim, D., and Yee, K. (2020). Flight control simulation and battery performance analysis of a quadrotor under wind gust. In *2020 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1782–1791.

Kobeaga, G., Merino, M., and Lozano, J. A. (2018). An efficient evolutionary algorithm for the orienteering problem. *Computers & Operations Research*, 90:42–59.

Kool, W., van Hoof, H., and Welling, M. (2019). Attention, learn to solve routing problems! In *International Conference on Learning Representations*.

Lau, H. C., Yeoh, W., Varakantham, P., Nguyen, D. T., and Chen, H. (2012). Dynamic stochastic orienteering problems for risk-aware applications. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence*, page 448–458, Arlington, Virginia, USA.

Leifer, A. C. and Rosenwein, M. B. (1994). Strong linear programming relaxations for the orienteering problem. *European Journal of Operational Research*, 73(3):517–523.

Li, Y., Liang, W., Xu, W., Xu, Z., Jia, X., Xu, Y., and Kan, H. (2023). Data collection maximization in iot-sensor networks via an energy-constrained uav. *IEEE Transactions on Mobile Computing*, 22(1):159–174.

Liang, Y.-C. and Smith, A. E. (2006). An ant colony approach to the orienteering problem. *Journal of the Chinese Institute of Industrial Engineers*, 23(5):403–414.

Lin, C., Yang, W., Dai, H., Li, T., Wang, Y., Wang, L., Wu, G., and Zhang, Q. (2023). Near optimal charging schedule for 3-d wireless rechargeable sensor networks. *IEEE Transactions on Mobile Computing*, 22(6):3525–3540.

Lin, S. (1965). Computer solutions of the traveling salesman problem. *The Bell System Technical Journal*, 44(10):2245–2269.

Little, J. D. C., Murty, K. G., Sweeney, D. W., and Karel, C. (1963). An algorithm for the traveling salesman problem. *Operations Research*, 11(6):972–989.

Liu, N., Zhang, J., Luo, C., Cao, J., Hong, Y., Chen, Z., and Chen, T. (2024). Dynamic charging strategy optimization for uav-assisted wireless rechargeable sensor networks based on deep Q-Network. *IEEE Internet of Things Journal*, 11(12):21125–21134.

Liu, Y., Pan, H., Sun, G., Wang, A., Li, J., and Liang, S. (2022). Joint scheduling and trajectory optimization of charging uav in wireless rechargeable sensor networks. *IEEE Internet of Things Journal*, 9(14):11796–11813.

Liu, Z., Kurzhanskiy, A., and Sengupta, R. (2017a). An energy-based optimal control problem for unmanned aircraft systems flight planning. In *2017 56th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE)*, pages 1320–1325.

Liu, Z., Sengupta, R., and Kurzhanskiy, A. (2017b). A power consumption model for multi-rotor small unmanned aircraft systems. In *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 310–315.

Mandal, A. K. and Dehuri, S. (2020). A survey on ant colony optimization for solving some of

the selected np-hard problem. In *Biologically Inspired Techniques in Many-Criteria Decision Making*, pages 85–100, Cham. Springer International Publishing.

Marini, F. and Walczak, B. (2015). Particle swarm optimization (PSO). a tutorial. *Chemometrics and Intelligent Laboratory Systems*, 149:153–165.

Mennell, W. K. (2009). *Heuristics for solving three routing problems: Close-enough traveling salesman problem, close-enough vehicle routing problem, and sequence-dependent team orienteering problem*. PhD thesis, University of Maryland, College Park.

Miller, C. E., Tucker, A. W., and Zemlin, R. A. (1960a). Integer programming formulation of traveling salesman problems. *J. ACM*, 7(4):326–329.

Miller, C. E., Tucker, A. W., and Zemlin, R. A. (1960b). Integer programming formulation of traveling salesman problems. *J. ACM*, 7(4):326–329.

Murphy, K. P. (2007). Conjugate bayesian analysis of the gaussian distribution. *def*, 1(2σ2):16.

Niculescu, V., Palossi, D., Magno, M., and Benini, L. (2023). Energy-efficient, precise uwb-based 3-d localization of sensor nodes with a nano-uav. *IEEE Internet of Things Journal*, 10(7):5760–5777.

Pandiyan, A. Y. S., Boyle, D. E., Kiziroglou, M. E., Wright, S. W., and Yeatman, E. M. (2021). Optimal dynamic recharge scheduling for two-stage wireless power transfer. *IEEE Transactions on Industrial Informatics*, 17(8):5719–5729.

Pasha, J., Elmi, Z., Purkayastha, S., Fathollahi-Fard, A. M., Ge, Y.-E., Lau, Y.-Y., and Dulebenets, M. A. (2022a). The drone scheduling problem: A systematic state-of-the-art review. *IEEE Transactions on Intelligent Transportation Systems*, 23(9):14224–14247.

Pasha, J., Elmi, Z., Purkayastha, S., Fathollahi-Fard, A. M., Ge, Y.-E., Lau, Y.-Y., and Dulebenets, M. A. (2022b). The drone scheduling problem: A systematic state-of-the-art review. *IEEE Transactions on Intelligent Transportation Systems*, 23(9):14224–14247.

Placido, A. D., Archetti, C., and Cerrone, C. (2022). A genetic algorithm for the close-enough traveling salesman problem with application to solar panels diagnostic reconnaissance. *Computers & Operations Research*, 145:105831.

Placido, A. D., Archetti, C., Cerrone, C., and Golden, B. (2023). The generalized close enough traveling salesman problem. *European Journal of Operational Research*, 310(3):974–991.

Polo, J., Hornero, G., Duijneveld, C., García, A., and Casas, O. (2015). Design of a low-cost wireless sensor network with uav mobile node for agricultural applications. *Computers and Electronics in Agriculture*, 119:19–32.

Polonelli, T., Qin, Y., Yeatman, E. M., Benini, L., and Boyle, D. (2020). A flexible, low-power platform for uav-based data collection from remote sensors. *IEEE Access*, 8:164775–164785.

Polonelli, T., Schläpfer, S., and Magno, M. (2022). Performance comparison between decawave dw1000 and dw3000 in low-power double side ranging applications. In *2022 IEEE Sensors Applications Symposium (SAS)*, pages 1–6.

Potvin, J.-Y. (1996). Genetic algorithms for the traveling salesman problem. *Annals of Operations Research*, 63:337–370.

Powercast (2024). *TX91501B 915 MHz Powercaster Transmitter User Manual*. Available at `https://www.powercastco.com/rf-power/`.

Pěnička, R., Faigl, J., and Saska, M. (2019). Variable neighborhood search for the set orienteering problem and its application to other orienteering problem variants. *European Journal of Operational Research*, 276(3):816–825.

Pěnička, R., Faigl, J., Váňa, P., and Saska, M. (2017). Dubins orienteering problem. *IEEE Robotics and Automation Letters*, 2(2):1210–1217.

Qian, Q., O'Keeffe, J., Wang, Y., and Boyle, D. (2023). Practical mission planning for optimized uav-sensor wireless recharging. *arXiv preprint arXiv:2203.04595*.

Qian, Q., Wang, Y., and Boyle, D. (2024). On solving close enough orienteering problems with overlapped neighborhoods. *European Journal of Operational Research*, 318(2):369–387.

Qorvo (2016a). *DWM1000 IEEE 802.15.4-2011 UWB Transceiver Module*, vision 1.8 edition. Available at `https://www.qorvo.com/products/p/DWM1000#documents`.

Qorvo (2016b). *EVK1000 USER MANUAL*, Version 1.13 edition. Available at `https://www.qorvo.com/products/p/EVK1000#documents`.

Qorvo (2024). *Maximising range in DW1000 based systems*, v1.6 edition. Available at `https://www.qorvo.com/products/d/da008450`.

Reiter, S. and Sherman, G. (1965). Discrete optimizing. *Journal of the Society for Industrial and Applied Mathematics*, 13(3):864–889.

Rodrigues, T. A., Patrikar, J., Choudhry, A., Feldgoise, J., Arcot, V., Gahlaut, A., Lau, S., Moon, B., Wagner, B., Matthews, H. S., et al. (2021). In-flight positional and energy use data set of a dji matrice 100 quadcopter for small package delivery. *Scientific Data*, 8(1):155.

Rodrigues, T. A., Patrikar, J., Oliveira, N. L., Matthews, H. S., Scherer, S., and Samaras, C. (2022). Drone flight data reveal energy and greenhouse gas emissions savings for very small package delivery. *Patterns*, 3(8):100569.

Rosenkrantz, D. J., Stearns, R. E., and Lewis, P. M. (2009). *An analysis of several heuristics for the traveling salesman problem*, pages 45–69. Springer Netherlands, Dordrecht.

Samir, M., Sharafeddine, S., Assi, C. M., Nguyen, T. M., and Ghrayeb, A. (2020). Uav trajectory planning for data collection from time-constrained iot devices. *IEEE Transactions on Wireless Communications*, 19(1):34–46.

Savchuk, V. and Tsokos, C. P. (2011). *Bayesian theory and methods with applications*. Springer Science & Business Media.

Schrijver, A. (2005). On the history of combinatorial optimization (till 1960). In Aardal, K., Nemhauser, G., and Weismantel, R., editors, *Discrete Optimization*, volume 12 of *Handbooks in Operations Research and Management Science*, pages 1–68. Elsevier.

She, X. T. P., Lin, X., and Lang, H. (2020). A data-driven power consumption model for electric UAVs. In *2020 American Control Conference (ACC)*, pages 4957–4962.

Shi, J., Cong, P., Zhao, L., Wang, X., Wan, S., and Guizani, M. (2024). A two-stage strategy for uav-enabled wireless power transfer in unknown environments. *IEEE Transactions on Mobile Computing*, 23(2):1785–1802.

Silva, C., Sousa, J., and Runkler, T. (2008). Rescheduling and optimization of logistic processes using ga and aco. *Engineering Applications of Artificial Intelligence*, 21(3):343–352.

Stutzle, T. and Dorigo, M. (2002). A short convergence proof for a class of ant colony optimization algorithms. *IEEE Transactions on Evolutionary Computation*, 6(4):358–365.

Suman, S., Kumar, S., and De, S. (2019). Uav-assisted rfet: A novel framework for sustainable wsn. *IEEE Transactions on Green Communications and Networking*, 3(4):1117–1131.

Sun, X., Zhang, L., Song, D., and Wu, Q. J. (2023). A novel path planning method for multiple usvs to collect seabed-based data. *Ocean Engineering*, 269:113510.

Tasgetiren, M. F. and Smith, A. E. (2000). A genetic algorithm for the orienteering problem. In *Proceedings of the 2000 Congress on Evolutionary Computation. CEC00 (Cat. No. 00TH8512)*, volume 2, pages 910–915. IEEE.

Thayer, T. C. and Carpin, S. (2021). An adaptive method for the stochastic orienteering problem. *IEEE Robotics and Automation Letters*, 6(2):4185–4192.

The MathWorks Inc. (2024). Communications toolbox version: R2024b.

Thibbotuwawa, A., Bocewicz, G., Nielsen, P., and Zbigniew, B. (2019). Planning deliveries with uav routing under weather forecast and energy consumption constraints. *IFAC-PapersOnLine*, 52(13):820–825. 9th IFAC Conference on Manufacturing Modelling, Management and Control MIM 2019.

Toaza, B. and Esztergár-Kiss, D. (2023). A review of metaheuristic algorithms for solving tsp-based scheduling optimization problems. *Applied Soft Computing*, 148:110908.

Tsiligirides, T. (1984). Heuristic methods applied to orienteering. *The Journal of the Operational Research Society*, 35(9):797–809.

Turlapaty, A. C. (2020). Variational bayesian estimation of statistical properties of composite gamma log-normal distribution. *IEEE Transactions on Signal Processing*, 68:6481–6492.

Vansteenwegen, P., Souffriau, W., and Oudheusden, D. V. (2011). The orienteering problem: A survey. *European Journal of Operational Research*, 209(1):1–10.

Štefaníková, P., Váňa, P., and Faigl, J. (2020). Greedy randomized adaptive search procedure for close enough orienteering problem. In *Proceedings of the 35th Annual ACM Symposium on Applied Computing*, SAC '20, page 808–814, New York, NY, USA. Association for Computing Machinery.

Wang, J., Guo, J., Zheng, M., MuRong, Z., and Li, Z. (2019a). Research on a novel minimum-risk model for uncertain orienteering problem based on uncertainty theory. *Soft Computing*, 23:4573–4584.

Wang, X., Golden, B., and Wasil, E. (2019b). A steiner zone variable neighborhood search heuristic for the close-enough traveling salesman problem. *Computers & Operations Research*, 101:200–219.

Wang, Y. and Boyle, D. (2024). Constrained reinforcement learning using distributional representation for trustworthy quadrotor uav tracking control. *IEEE Transactions on Automation Science and Engineering*, pages 1–18.

Wang, Y. and Han, Z. (2021). Ant colony optimization for traveling salesman problem based on parameters optimization. *Applied Soft Computing*, 107:107439.

Wang, Y., O'Keeffe, J., Qian, Q., and Boyle, D. (2022). KinoJGM: A framework for efficient and accurate quadrotor trajectory generation and tracking in dynamic environments. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 11036–11043.

Xue, P., Li, X., Jiang, Z., Luo, B., Miao, Y., Liu, X., and Deng, R. H. (2023). A multi-cuav multi-uav electricity scheduling scheme: from charging location selection to electricity transaction. *IEEE Internet of Things Journal*, 10(23):20899–20913.

Yang, B., Taleb, T., Shen, Y., Jiang, X., and Yang, W. (2021). Performance, fairness, and tradeoff in uav swarm underlaid mmwave cellular networks with directional antennas. *IEEE Transactions on Wireless Communications*, 20(4):2383–2397.

Yang, H., Ruby, R., Pham, Q.-V., and Wu, K. (2022). Aiding a disaster spot via multi-uav-based iot networks: Energy and mission completion time-aware trajectory optimization. *IEEE Internet of Things Journal*, 9(8):5853–5867.

Yang, Z., Xiao, M.-Q., Ge, Y.-W., Feng, D.-L., Zhang, L., Song, H.-F., and Tang, X.-L. (2018). A double-loop hybrid algorithm for the traveling salesman problem with arbitrary neighbourhoods. *European Journal of Operational Research*, 265(1):65–80.

Zeng, Y., Xu, J., and Zhang, R. (2019). Energy minimization for wireless communication with rotary-wing uav. *IEEE Transactions on Wireless Communications*, 18(4):2329–2345.

Zhang, J., Campbell, J. F., Sweeney II, D. C., and Hupman, A. C. (2021). Energy consumption models for delivery drones: a comparison and assessment. *Transportation Research Part D: Transport and Environment*, 90:102668.

Appendices

Appendix A

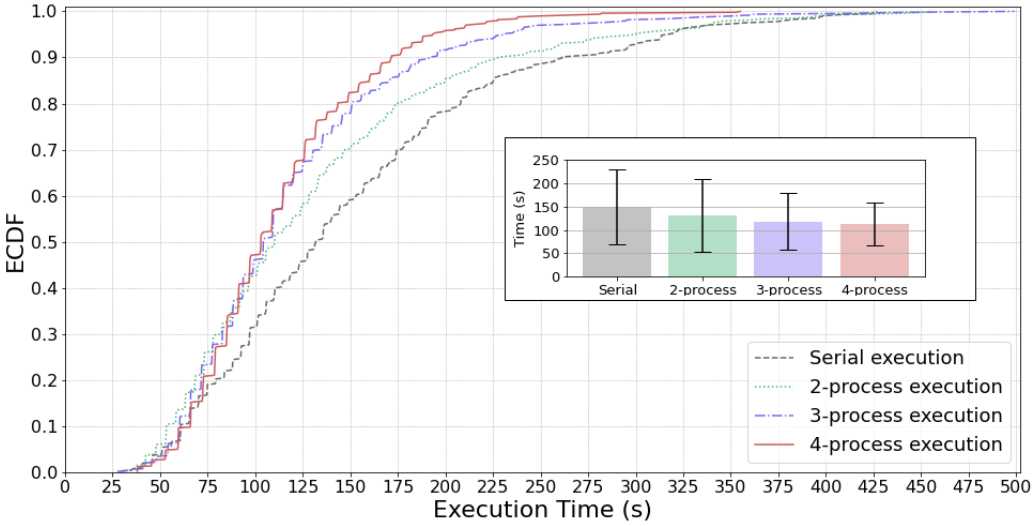
Parallel Search Efficiency of CBHA

Parallelization performance analysis focuses on determining the probability that the optimization solver could find a route with superior fitness value compared to GLS across varying numbers of distributed computation processes. We fix the generation number 100 for these experiments while continuously increasing population sizes over 100 executions. The results are fitted to a logistic function curve to illustrate convergence trends (see Figure (A.1a)). Our findings revealed that serial execution could not achieve 100% probability of finding better solutions even with 10000 populations, while four-process parallel execution achieved 100% convergence at 5000 populations. For smaller population sizes (i.e., $N_{pop} \leq 500$), parallel execution demonstrated approximately linear improvement in solution probability, indicating a linear reduction of the optimization solver's execution time with increased process allocation.

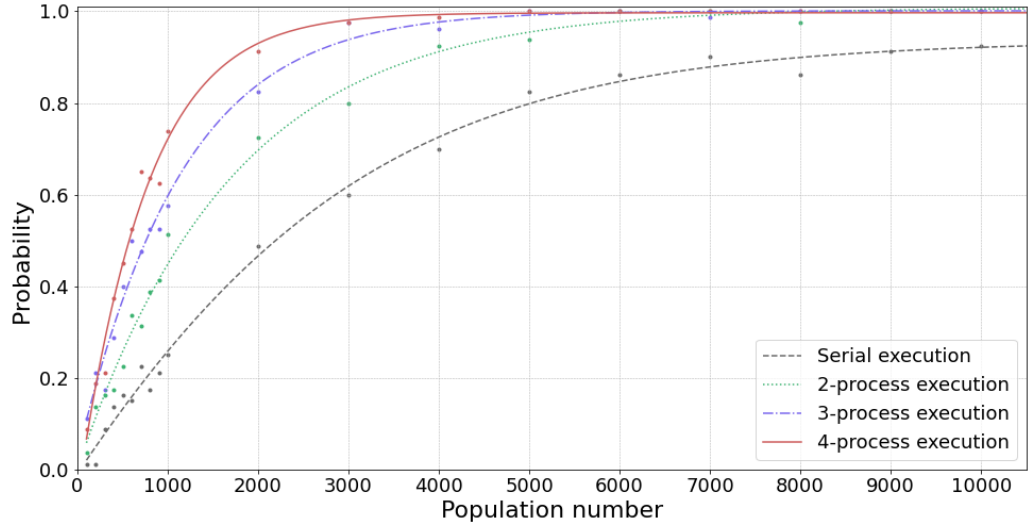
To determine the minimum execution time required to surpass GLS performance, we conducted additional experiments using 5000 populations and 100 generations (see Figure (A.1b)), parameters at which four-process execution achieved 100% convergence. Empirical cumulative distribution (ECDF) analysis of execution times showed that CBHA required at least 5 minutes with four allocated processes to guarantee superior performance compared to the GLS.

Notably, four-process execution reduced computation time by approximately 30% compared to serial execution. In cases where CBHA achieved better solutions, four-process execution demonstrated improved stability with lower mean execution time (112.04 s vs. 149.76 s) and standard deviation (45.71 s vs. 81.29 s) compared to serial execution.

Despite our straightforward parallelization strategy, these results demonstrate that parallelization enhances solver stability and convergence speed. Furthermore, the experiments validate ROMP's suitability for online re-planning applications on standard flight computers.



(a) The probability is the percentage that the CBHA obtains a solution better than GLS over all executions. The 4-process execution can achieve 100% convergence at 5000 populations, while the serial execution fails to converge even with a population size of 10000.



(b) Empirical cumulative distribution function for finding better solutions against time using ROMP. The results span 500 executions using $N_{pop} = 5000$ and $N_{gen} = 100$. The 4-process execution can save ∼30% execution time over serial while achieving better solution quality.

Figure A.1: Parallel performance summary.

Appendix B

Sensitivity analysis of the minimum safety belief

Solutions generated by different $\Theta_{\min}$ under various scenarios are presented in Table (B.1). In summary, all settings of $\Theta_{\min}$ can have a high mission success rate of over 50 executions under most scenarios. The setting of $\Theta \in [45, 99]\%$ can frequently find higher-quality solutions compared to others because it allows more search space. However, the balance between safety beliefs and prize collection is challenging to maintain, resulting in risky solutions that pursue high prize collection (see *California20* with $\Delta\mu_{\bar{\mathbf{P}}} = 20\%$). In conclusion, the prize advancement achieved by setting a low $\Theta_{\min}$ is insufficient to compensate for mission safety within the CSP context. The adaptive setting of weights to the safety belief and prize collection may allow a lower $\Theta_{\min}$ to achieve a higher mission success rate.

Table B.1: Sensitivity analysis of safety belief Θ.

	$\Delta\mu_{\bar{\mathbf{P}}^*}$ (%)	$\Delta\sigma_{\bar{\mathbf{P}}^*}$ (%)	$\Theta \in [45,99]\%$			$\Theta \in [55,99]\%$			$\Theta \in [65,99]\%$			$\Theta \in [75,99]\%$			$\Theta \in [85,99]\%$		
			MSR(%)	$\mathcal{P}^*$(kJ)	$\mathcal{C}^*$(kJ)	MSR(%)	$\mathcal{P}^*$(kJ)	$\mathcal{C}^*$(kJ)	MSR(%)	$\mathcal{P}^*$(kJ)	$\mathcal{C}^*$(kJ)	MSR(%)	$\mathcal{P}^*$(kJ)	$\mathcal{C}^*$(kJ)	MSR(%)	$\mathcal{P}^*$(kJ)	$\mathcal{C}^*$(kJ)
California20	-10	-10	100	**50.95**	348.93	100	50.85	349.75	100	50.62	350.45	100	50.35	351.49	100	50.28	350.97
	-10	0	100	50.84	349.51	100	**50.84**	349.28	100	50.69	350.56	100	50.29	352.07	100	50.24	351.42
	-10	10	100	**50.77**	349.99	100	50.73	349.63	100	50.68	350.27	100	50.30	352.48	100	50.25	351.24
	-10	20	98	**50.94**	350.29	100	50.82	349.98	100	50.52	350.51	100	50.34	351.56	100	50.29	350.79
	0	-10	100	47.18	350.37	100	47.48	349.27	100	**47.63**	349.54	100	47.61	349.32	100	47.58	350.14
	0	0	100	47.16	349.86	100	47.49	348.66	100	**47.71**	350.21	100	47.63	349.62	100	47.53	350.23
	0	10	100	47.55	350.30	100	**47.66**	350.47	100	47.56	350.51	100	47.57	349.17	100	47.60	350.90
	0	20	100	47.461	351.02	100	47.63	350.28	100	47.44	349.73	100	**47.63**	349.64	100	47.60	350.65
	10	-10	94	**44.67**	350.36	96	44.190	350.25	100	44.62	350.39	100	44.37	350.23	100	44.42	350.03
	10	0	92	**44.70**	350.96	100	44.33	349.76	100	44.44	350.33	100	44.36	349.98	100	44.28	349.82
	10	10	92	**44.67**	351.19	100	44.54	351.29	100	44.39	350.83	100	44.43	350.01	100	44.31	349.56
	10	20	86	**44.63**	350.60	98	44.50	351.06	96	44.34	350.13	100	44.47	350.49	100	44.49	350.16
	20	-10	32	40.99	347.31	68	40.52	341.24	94	40.32	342.79	100	41.14	351.54	100	**41.17**	350.78
	20	0	48	40.64	344.36	74	40.74	343.23	90	40.78	344.84	100	**41.10**	349.92	100	41.07	350.34
	20	10	36	40.55	346.40	88	40.13	341.63	92	40.56	344.34	100	41.07	348.26	100	**41.08**	350.77
	20	20	57	41.08	348.03	90	40.24	342.96	92	40.85	344.43	100	**41.10**	350.43	100	41.08	350.00
California30	-10	-10	100	45.52	350.69	100	**45.67**	350.70	100	45.62	350.22	100	45.62	350.44	100	45.46	351.03
	-10	0	100	45.46	349.63	100	**45.65**	350.04	100	45.58	350.39	100	45.63	349.66	100	45.48	349.69
	-10	10	100	**45.68**	350.64	100	45.64	350.56	100	45.64	349.78	100	45.52	349.76	100	45.50	350.47
	-10	20	100	**45.78**	350.76	100	45.78	351.01	100	45.53	349.73	100	45.62	350.16	100	45.51	350.08
	0	-10	96	41.78	351.97	100	41.76	352.37	100	**42.00**	352.70	100	41.50	350.64	100	41.65	351.64
	0	0	98	41.67	351.17	100	41.69	351.58	100	**41.90**	352.67	100	41.80	352.25	100	41.56	350.78
	0	10	100	**41.72**	351.27	98	41.66	352.31	100	41.55	350.73	100	41.57	351.38	100	41.54	350.60
	0	20	100	**41.65**	351.73	100	41.55	351.15	100	41.49	350.45	100	41.57	351.25	100	41.48	350.48
	10	-10	98	38.31	354.59	100	**38.50**	355.19	100	38.26	354.21	100	37.66	353.27	100	37.39	352.62
	10	0	96	38.39	355.26	100	**38.52**	354.78	100	38.05	354.22	100	37.61	353.48	100	37.63	352.73
	10	10	100	**38.48**	355.07	100	38.37	355.02	100	38.31	354.99	100	37.66	353.25	100	37.53	353.05
	10	20	100	**38.44**	355.31	100	38.12	354.68	100	38.19	354.88	100	38.02	354.00	100	37.18	351.89
	20	-10	96	**34.64**	352.66	94	34.56	351.70	100	34.42	349.67	100	34.07	346.50	100	33.69	345.08
	20	0	100	**34.73**	352.76	100	34.47	352.58	96	34.41	348.69	100	34.14	346.53	100	33.96	346.85
	20	10	100	**34.79**	352.44	96	34.30	351.12	96	34.27	350.31	100	33.85	345.24	100	34.04	346.40
	20	20	98	34.53	352.74	92	34.41	351.61	100	34.41	349.92	100	33.68	346.71	100	**34.53**	346.61
California40	-10	-10	100	53.89	344.31	100	53.88	342.00	100	53.89	342.04	100	**54.28**	343.97	100	53.89	341.97
	-10	0	100	53.83	345.23	100	53.91	342.04	100	53.86	341.94	100	**54.14**	343.18	100	53.93	342.06
	-10	10	100	53.93	343.50	100	53.89	342.24	100	54.00	342.68	100	**54.09**	342.94	100	53.92	342.07
	-10	20	100	53.83	343.90	100	53.84	341.69	100	53.91	342.02	100	**54.19**	343.27	100	53.85	341.92
	0	-10	100	51.72	353.04	100	**51.77**	352.32	100	51.68	352.47	100	51.60	349.82	100	51.61	346.46
	0	0	100	51.74	351.80	100	**51.77**	352.44	100	51.69	352.80	100	51.71	351.09	100	51.58	346.06
	0	10	100	51.57	353.03	100	**51.77**	351.51	100	51.76	352.17	100	51.76	349.45	100	51.59	347.14
	0	20	100	51.76	353.10	100	51.60	353.36	100	51.66	351.42	100	**51.78**	351.29	100	51.58	346.51
	10	-10	100	49.72	355.49	100	49.69	355.45	100	**49.72**	355.49	100	49.19	354.08	100	48.84	352.95
	10	0	100	**49.72**	355.44	100	49.69	355.44	100	49.72	355.56	100	49.17	354.04	100	48.50	351.94
	10	10	100	**49.72**	355.46	100	49.72	355.44	100	49.72	355.45	100	49.15	353.81	100	48.73	352.68
	10	20	100	**49.72**	355.51	100	49.72	355.56	100	49.72	355.47	100	49.18	353.89	100	48.73	352.65
	20	-10	100	**47.23**	353.80	100	47.13	353.59	100	47.04	353.31	100	46.76	352.60	100	45.43	349.14
	20	0	100	**47.23**	353.81	100	47.13	353.63	100	47.10	353.48	100	46.91	353.00	100	45.30	348.62
	20	10	100	**47.23**	353.76	100	47.11	353.56	100	47.15	353.61	100	46.75	352.63	100	45.15	348.42
	20	20	100	**47.23**	353.71	100	47.16	353.73	100	47.13	353.55	100	46.70	352.39	100	45.43	349.12