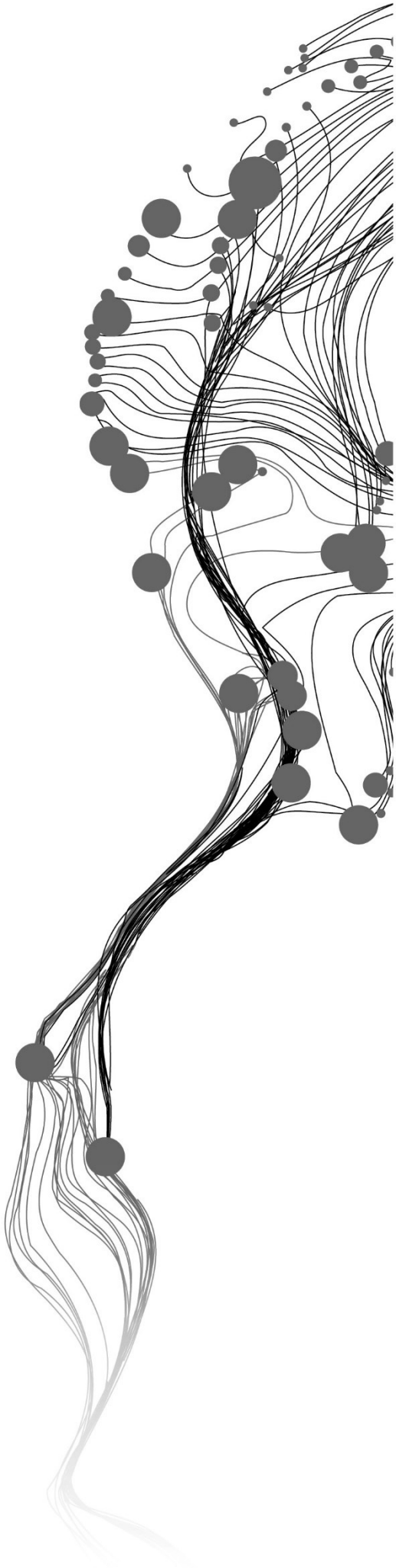
AUTONOMOUS EXPLORATION AND 3D RECONSTRUCTION USING COLLABORATIVE DRONES

AHMED HANIF

June, 2025

SUPERVISORS:
Dr. Eng., B.S.A., Alsadik
Prof. dr. ir. F.C., Nex



AUTONOMOUS EXPLORATION AND 3D RECONSTRUCTION USING COLLABORATIVE DRONES

AHMED HANIF
Enschede, The Netherlands, June 2025

Thesis submitted to the Faculty of Geo-Information Science and Earth Observation of the University of Twente in partial fulfilment of the requirements for the degree of Master of Science in Geo-information Science and Earth Observation.
Specialization: Geoinformatics

SUPERVISORS:
Dr. Eng., B.S.A., Alsadik
prof. dr. ir. F.C., Nex

THESIS ASSESSMENT BOARD:
Prof. dr. ir. M.G. Vosselman (Chair)
drs. Wan Bakx (Procedural advisor)
Dr Sofia Tilon (External Examiner)

ABSTRACT

The use of aerial robotics in the domain of civil engineering and infrastructure assessment has unlocked a very powerful domain in the Architecture, Engineering, and Construction Industry. As these infrastructures start to age, their structural integrity starts to deteriorate. Traditionally, the inspection of infrastructures is carried out manually, which can take days for normal infrastructures like a bridge and weeks for larger infrastructures like high-voltage powerlines. These applications demand methods that are scalable, rapid, and cost-effective solutions that can automate this process by understanding the information available in the scene. The cost effectiveness factor is defined to be the type of sensor and the expense of operation of an Unmanned Aerial Vehicle (UAV)-based inspection task. In comparison, LiDAR and depth sensors are readily available but have operational complexities like the payload and cost of the sensor. Current levels of autonomy involve complete control over the dynamics of an infrastructure inspection mission. Another implication is that traditional path planning methods require detailed information and insights of the infrastructure to be inspected. In time critical tasks like disaster response, such requirements can be problematic as the requirement of a predefined model is not guaranteed.

This research provides a solution to these problems by autonomously exploring a large 3D infrastructure object using UAVs equipped with an RGB Sensor. The autonomous exploration task is labelled as the Next Best View (NBV) problem. The autonomous exploration workflow estimates the NBVs required to cover an infrastructure object. A state-of-the-art NBV called MACARONS is selected in this research. The MACARONS-NBV is a self-supervised NBV prediction method that uses a monocular RGB camera to establish a 3D occupancy of an environment with infrastructure objects. The MACARONS-NBV architecture is designed in such a way to select a set of candidate camera views (NBVs) that maximize the surface coverage of an object. This trajectory is then assigned to a collaborative system of UAVs that coordinate with one another in a centralized communication channel to optimize their trajectories and avoid obstacles if encountered. The goal of the collaborative multi-UAV is to collect the infrastructure image data in an optimized manner. We utilize a set of methodologies to dynamically assign the workload of a UAV with awareness of their surroundings and any other UAV agents. The outcome of this is a network of UAVs that can simultaneously collect data for a large infrastructure object and can balance one another's workload depending on the remaining energy level. We then use an offline image-based 3D Reconstruction workflow of the infrastructure object and assess the completion of the object using Cloud-to-mesh (C2M) and Cloud-to-Cloud (C2C) distances.

We were able to optimize the path of UAVs to efficiently traverse large scenes as well as demonstrate dynamic Task Reassignment for the collaborative aspect of UAVs. Additionally, we also conduct a detailed comparison of an autonomous UAV mission using LiDAR-based inspection compared to a monocular RGB-based with reference to the ground truth.

Keywords: Unmanned Aerial Vehicle, Next Best View, LiDAR, MACARONS, Collaborative UAVs, Autonomous.

ACKNOWLEDGEMENTS

The journey of my Master's in Geoinformatics at ITC has been nothing short of a rollercoaster of emotions and surprises. From exhilarating highs to disheartening lows, the past two years have flown by in an instant. Arriving at ITC with high hopes and ambitions would be an understatement.

For my time during my MSc Research, I have nothing but the highest level of respect and gratitude towards my thesis supervisor, Dr. Bashar Alsadik. Your commitment and attention to detail are one of the driving factors that have left a long-lasting impact on me and my future. Your constant feedback, support, and guidance paved the way for me to accomplish this task. Your domain knowledge and trust in my capabilities were a major motivating factor for me to conduct my research in the domain of 3D Computer Vision and Photogrammetry.

I would also like to thank my co-supervisor, Prof. Francesco Nex, for his support and constructive criticism of my work. I would like to thank both my supervisors for their patience, support, and empathy during the final stages of my thesis research.

I would also like to emphasize the importance of having the unwavering support of my loving family, especially my parents and all six of my siblings (yes, six!) who have always been a beacon of positivity and hope. I will forever be grateful to my mother and father for placing their utmost trust in me. The time spent away from family has been challenging for all of us. I also wish to express my deepest gratitude to my dearest eldest brother, Kashif, who has stood by me through every high and low and has been a constant source of emotional and moral strength.

I would also like to thank Dr. Shayan Nikoohemat for providing me with the internship opportunity. It was a rewarding experience and gave me a deeper understanding of the importance of smaller things that can cause a really big impact.

I am thankful to the Chair of the Thesis Assessment Board, Prof. George Vosselman, for investing their time to review and assess the quality of my research and provide constructive feedback and recommendations.

Finally, I would like to thank all my Friends who have been there with me through thick and thin. Special gratitude goes to Ibrahim and Talha, who have always been supportive of my decisions regardless of the outcomes. Mujtaba and Taimoor, Muhtasham, and all the people who have had an impact on my life in one way or another.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1. Background Context and Motivation

The need for rapid, accurate, and cost-effective 3D reconstruction of infrastructure has grown significantly in domains such as construction, inspection, disaster response, and urban planning. Traditional methods involving manual surveying or ground-based laser scanning are often expensive, time-consuming, and inaccessible, particularly in hazardous and hard-to-reach environments. Unmanned Aerial Vehicles (UAVs) offer a compelling alternative, enabling remote data collection with minimal human endangerment and physical involvement. UAVs are used across various industries, including post-disaster structural assessment, overhead infrastructure inspection, archaeology, heritage site preservation, and traffic monitoring. (Fernandez Galarreta et al., 2015; Mohsan et al., 2022; Nex & Remondino, 2014).

As UAVs become integral to diverse industries for applications ranging from infrastructure inspection to earth observation, their capacity for efficient and precise data acquisition continues to expand. While traditional photogrammetric missions still rely heavily on pre-planned flight paths and manual input. Autonomous systems become time-efficient and critical. Autonomy, therefore, is not only an operational advantage but a strategic evolution in UAV technology, allowing for a new level of efficiency and flexibility in mission planning and execution. Developing UAV systems with high levels of autonomy can significantly reduce human intervention and offer more insights into the environment by efficient planning and collecting high-quality mission-critical data autonomously. However, most state-of-the-art UAV-based mapping systems rely on expensive hardware, such as LiDAR or stereo camera setups, which increases system complexity and cost. In contrast, this study explores a low-cost and agile solution using collaborative UAVs equipped solely with monocular RGB cameras to perform autonomous exploration and 3D reconstruction of infrastructure objects to document and develop digital twins for further inspection.

1.2. Research Gap

Autonomous exploration refers to exploring an unknown environment by gathering information and building a representation of the search space (Lee, 2025). In the context of UAV systems, Next Best View (NBV) planning is defined as selecting the optimal viewpoint or camera poses that maximize the information gained about the environment (Connolly, 1985). Several methods are used to represent an environment in 3D to quantify the gain in information. Established NBV methods rely on using LiDAR to generate depth maps that are used to create volumetric, frontier and surface- based representations to estimate the information gain (Bircher, Kamel, Alexis, Burri, et al., 2016; Dhami et al., 2023a; Guédon et al., 2022; Mendoza et al., 2020a; Vasquez-Gomez et al., 2021; Wang et al., 2024). Although LiDAR sensors create a high-quality representation of a 3D scene, they generally have much higher operational costs than monocular RGB cameras. When exploring large, complex scenes, monocular RGB-based depth estimation techniques are adaptable to the complexity by learning from a sequential set of frames.

There are limited methods that utilize only a monocular RGB camera to estimate the depth of a scene and create an occupancy field to estimate NBVs and reconstruct the scenes in 3D (Guédon et al., 2023). The research builds on top of the RGB-based NBV strategy to develop an end-to-end workflow to explore large, complex scenes with infrastructural objects autonomously and to assign data collection tasks collaboratively and offer per-UAV strategies to optimize their flight trajectories and handle and avert collisions with scene objects in an automated way. Current collaborative methods use a dedicated depth

sensor like LiDAR to establish the volume occupancy field (Dhami et al., 2023a; Hardouin et al., 2020; Nagasawa et al., 2021; B. Zhou et al., 2022)

1.3. Summary of methodology and main contributions

This MSc research focuses on autonomous UAV-based collaborative exploration and 3D reconstruction of infrastructure objects, building on the learning-based Next Best View method of MACARONS proposed by Guédon et al. (2023). The methodology integrates an end-to-end pipeline where NBV predictions guide the exploration of 3D scenes, followed by task assignment and coordination among multiple UAVs. Once the optimal views are selected, UAVs optimize flight trajectory with built-in collision avoidance mechanisms. A battery-aware planning system ensures dynamic task reassignment during missions. The pipeline includes data acquisition using monocular RGB sensors and concludes with dense 3D reconstruction from the collected imagery. This MSc. research introduces an integrated framework extending the MACARONS-NBV method to a collaborative multi-UAV setting. It features centralized task coordination, battery-aware scheduling for dynamic task reassignment, and a full pipeline from NBV prediction using a monocular RGB camera to dense 3D reconstruction using photogrammetric principles. The system is evaluated on complex infrastructure scenes, demonstrating coverage, reconstruction quality, and operational efficiency improvements.

1.4. Problem statement

Inspection of infrastructure objects is one of the most critical applications of autonomous UAV systems. Exploring large-scale and complex scenes is inherently time-consuming and operationally demanding, often requiring extensive pre-planning, manual navigation, and continuous human oversight. While UAV-based inspection has the potential to reduce cost, time, and risk significantly, most advanced systems currently operate at autonomy Level 3, where the operator retains complete control and intervention is often required (Alsadik & Nex, 2021). This limits scalability and efficiency, particularly in large or geometrically complex infrastructure scenarios.

Furthermore, current approaches often rely on single-agent UAV systems, which are suboptimal for covering large areas and cannot dynamically coordinate task execution or redistribute workload in real time. As a result, the inspection process becomes inefficient and difficult to scale.

This research addresses these limitations by proposing a collaborative consumer-grade multi-UAV framework in which a collection of UAVs, each equipped solely with a monocular RGB camera, autonomously navigate and explore an infrastructure environment. These UAVs maintain situational awareness through communication, possess onboard capabilities for obstacle avoidance, and collaboratively assign and balance tasks to ensure complete and high-quality data acquisition. The proposed methodology introduces a pipeline for autonomous mission planning, path optimization, and data collection, followed by a photogrammetric 3D reconstruction process. A centralized communication channel supports coordination, enabling UAVs to adaptively plan their trajectories while maintaining collision avoidance and efficient area coverage.

The outcome is a scalable and cost-effective system that advances UAV operations toward autonomy Level 4. It enables infrastructure inspection tasks to be executed collaboratively, with minimal human intervention and optimized data quality for applications such as site preservation, asset monitoring, and high-fidelity digital twin creation.

1.5. Objective of the research

Main Objective

To develop a solution to autonomously plan UAV paths to explore an infrastructure object and reconstruct it in 3D using a collaborative system of UAVs equipped with a monocular RGB camera.

Sub-objectives and Research Questions

1. To estimate depth from a UAV equipped with a monocular RGB sensor using learning-based depth estimation techniques
    - *Is the learning-based RGB depth prediction a reliable alternative to an onboard depth sensor or LiDAR?*
    - *How can monocular RGB cameras assist in the NBV process?*
2. To assign optimized flight paths to multiple UAVs collecting image data to 3D reconstruct an infrastructure object efficiently.
    - *What algorithms can be developed to enable UAVs to autonomously coordinate their flight paths for optimal coverage in a 3D reconstruction project?*
3. To assess the scalability of the collaborative multi-UAV workflow
    - *How efficient is the data collection process in a collaborative multi-UAV setting compared to a single UAV exploring a 3D Infrastructure environment?*

1.6. Outline of the research

This research consists of six chapters. Chapter 1 consists of the introduction, research gaps, problem statement, and methodology summary. Chapter 2 is an elaborate literature review of flight planning methods, NBV algorithms (learning based and traditional methods), and depth sensing techniques, followed by a review of existing multi-UAV collaborative systems. Chapter 3 explains the setup of the simulation environment and underlying assumptions, followed by Chapter 4, which explains the methodological workflow and conceptual models for the end-to-end system. Chapter 5 will later highlight the Results of Reconstruction and accuracy assessment metrics, along with the discussions, and Chapter 6 will conclude the research outcome and provide recommendations.

2. LITERATURE REVIEW

This chapter provides a comprehensive literature review about different methods for path planning, NBV strategies, and depth sensing technologies. Furthermore, it reviews literature on the collaborative strategies for multi-UAV task assignment and collaborative systems.

Unmanned Aerial Vehicles, due to their agility and maneuverability, can be equipped with a variety of sensors, including LIDAR, RGB-D, Depth Sensors, Monocular and Multiview stereo cameras. These are used to carry out a variety of reconstruction and inspection tasks. Ellenberg et al., (2015) emphasized using an RGB camera in various building inspection tasks, like crack detection and performing infrastructure deformation measurements. Similarly, RGB cameras are widely used for surveying, localization, and mapping for indoor navigation (González de Santos et al., 2021).

2.1. Model-guided Autonomous UAV Path Planning

Autonomous exploration of a scene can be deemed as a view planning problem. In many cases, this process is based on prior information of a scene, which depends on the orientation network of images acquired through a photogrammetric mission. These are called model-based view planning (Maboudi et al. 2023). Which relies heavily on an already existing coarse model as a baseline. Gao et al. (2023) Use an existing object's location as a reference point for collecting RGB data for a scene based on Next Best View. The goal is to obtain the fewest points for maximum data acquisition based on a distortion metric optimized on uniformly sampled candidate views. Model-based exploration and exploitation techniques are generally used to improve an existing coarse model, also termed proxy models, and are used as a reference to optimize camera orientations to gather detailed information about scene techniques. Koch et al. (2019) used semantic segmentation on buildings to make the flight path safe by filtering out buildings using an initial flight. They proposed a semantically aware flight path that maximizes the detail of the generated 3D model. Y.-L. Wu et al. (2023) improved the level of detail (LOD)2 models by proposing a NADIR centric viewpoint free of obstructions and using a genetic algorithm that solves the flight path for a minimal occlusion-based 3D photogrammetric model. Roberts et al. (2017) utilized an explore-and-exploit strategy to gather a coarse model of the scene through an initial flight and then exploit it to maximize the detailed exactness of the 3D reconstruction. Similarly, Zhang et al. (2024) used a similar explore and then exploit strategy, but also considers completeness, precision, and semantic information of the initial model and refines the same model using photogrammetric guided viewpoints that increase the overall precision of the model, later combined for completeness. They also utilized visibility and curvature information to detect defects from a semantic perspective. Gao et al. (2023) used multiple UAVs fusion for exploration and reconstruction with guaranteed safety, with centimeter positional accuracy based on Simultaneous Localization and Mapping (SLAM). Smith et al. (2018) introduced heuristic-based quality assessment of 3D reconstructions and path planning using an initial NADIR capture as a proxy 3D model. Nagasawa et al. (2021) further advanced in model-based path planning by applying clustering and optimization to improve multi-UAV coverage that resulted in faster, more accurate 3D reconstructions of disaster sites by using existing satellite data and utilizing it to create 3D models as a base reference.

2.2. NBV path planning

As defined earlier, Next Best View planning is selecting the next most informative viewpoint in a scene to acquire additional sensor measurements that maximize the improvement of the current observation. This selection can be guided by prior measurements, a known scene model, or adaptive strategies during exploration. (Connolly, 1985; Mendoza et al., 2020b).

2.2.1. Conventional NBV Methods

Model-free path planning relies on NBV to guide the UAV in covering an unexplored region. NBV planning can be broadly categorized into three subcategories, namely, Frontier-based, volumetric-based, and surface-based planning (Maboudi et al., 2023).

I) Frontier-based NBV algorithms

Frontier-based NBV exploration was first introduced by Yamauchi (1997) to explore unknown scenes autonomously. Frontiers are the boundary between explored and unexplored regions of an environment. The system will iteratively keep exploring new frontiers until all areas have been explored. Figure 2.1 represents a frontier-based exploration strategy. Cieslewski et al. (2017) extended the frontier-based method to maintain the maximum speed of a mobile robot. They determined an NBV by selecting target frontier volumes inside the current field of view. This method can prevent a mobile robot from slowing down due to heading rotation; therefore, the robots can explore unknown regions at high speed. Such frontier-based techniques were mainly used for 2D exploration strategies.

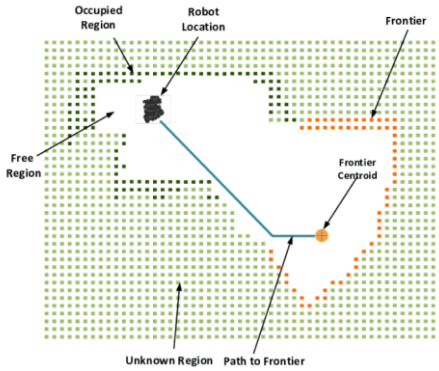


Figure 2.1 Frontiers, occupied and free regions (Tran et al., 2022)

II) Volumetric-based NBV algorithms

Volumetric view planners assume the study region to be represented as continuous volumes, as represented on the Octo Map (Hornung et al., 2013), which is a volumetric representation of space developed for robotic applications (Zhu et al., 2015). Fundamentally, these algorithms are based on an occupancy grid concept, which is a voxel-based discrete representation of an environment that divides the region into either occupied, free, or unknown (Wong et al., 1999). Volumetric environment representation includes NBV planning implemented by Bircher et al. (2016) which creates perspectives with Rapidly Exploring Random Trees (RRT) and chooses the branch that provides maximum information gain. Here, the information gained is the overall amount of unknown or unexplored voxels, modulated by the distance cost of nodes. The NBV is then chosen as the first node of the selected branch. This method originates in exploration paths with the lowest computational complexity, even in large areas. Papachristos et al. (2019) extended the receding horizon planning approach by adding a way to handle localization uncertainty during exploration using a two-step volumetric exploration and uncertainty optimization procedure. In the exploration step, they expand branches of a random tree and choose the NBV from the first node of the branch that gives the most information, considering unknown areas and occupancy likelihood. The

uncertainty optimization step then finds the best path to the NBV, reducing uncertainty for the overall mapping system. Similarly, Song & Jo (2017) also built on Bircher et al. (2016) by looking at online inspection planning based on an unknown or a partial model, exploring the large unexplored areas, and developing during the online exploration. It locally inspects continuously between NBVs. Figure 2.2 shows a volumetric representation of a 3D environment using voxels.

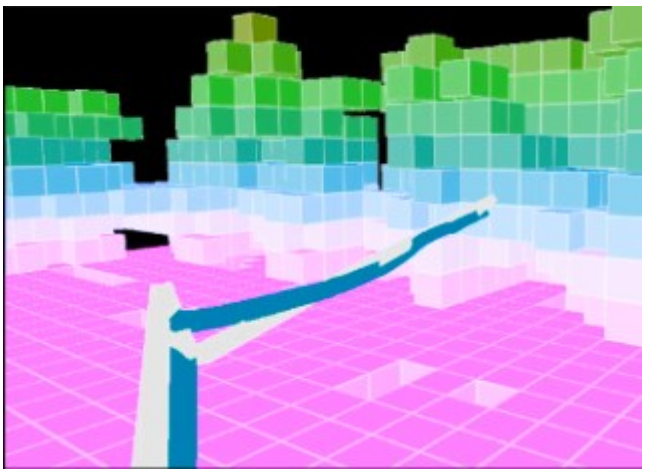


Figure 2.2 Volumetric representation of a 3D scene (Bircher et al., 2016)

III) Surface-based NBV algorithms

Song et al. (2022) proposed a surface-based exploration strategy that used a truncated distance field as a confidence criterion on top of a volumetric-based frontier exploration to plan the NBV (S. Wu et al., 2014). A point cloud is used to generate a Poisson surface, and the point-based surface is used to assess the information gained based on the quality of the initial scan. Based on this, Huang et al. (2018) introduce a multi-view stereo method, which was used first to generate a coarse 3D model of triangular meshes, which they smoothed using Poisson surface modeling, giving surface gain to identify the NBVs to maximize the coverage iteratively. The actual dense 3D reconstruction is done offline. Schmid et al. (2012) also use an information-based function based on the Truncated Signed Distance Function (TSDF) surface, where they extend the RRT-based tree implementation for surface inspection and coverage path planning. Kriegel et al. (2011) devised a direct surface-based NBV estimator that detects boundary patches, generates view candidates perpendicular to those patches with overlap to prior scans, and selects the candidate that best exposes occluded areas. Figure 2.3 shows a mesh-to-surface conversion to quantify the mesh.

Figure 2.4 summarizes the different surface representations of a 3D object.

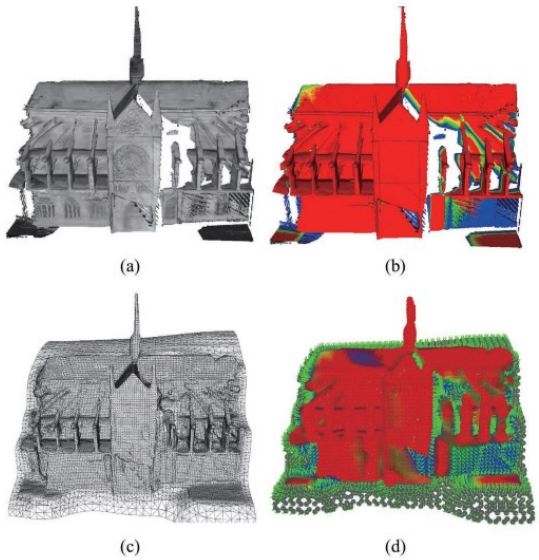


Figure 2.3 Surface point extraction: (a) Incomplete surface; (b) Confidence weights; (c) Reconstructed surface; (d) Target points (Song & Jo, 2017)

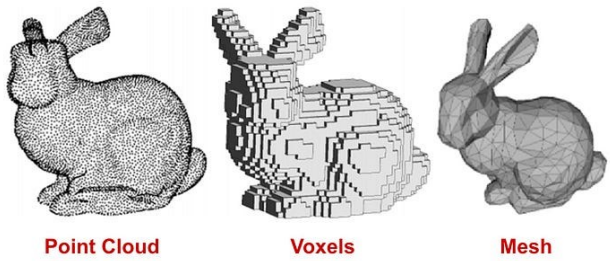


Figure 2.4 Different representations of a 3D object (Hoang et al., 2019)

2.2.2. Learning-based NBV algorithms

In contrast to algorithmic implementations using information theory, there is a vast collection of literature on prediction-based NBV solutions, where Mendoza et al. (2020a) modelled the NBV problem as a supervised classification-based solution called NBV-Net, using a partial model to predict the sensor pose. This is done using a model trained on a set of sensor poses with a training sample and a prediction label. The author uses a voxel-based representation of the 3D scene obtained by a depth camera. On the other hand, a 3D convolutional neural network was implemented as a regressor. They also use a voxel-based representation of the partial 3D model. They investigated why this method performs better than NBV-NET proposed by Mendoza et al. (2020a), which solves the NBV problem by predicting the NBV using a partially defined model as a supervised classification problem using a convolutional neural network (CNN), in terms of accuracy and generalization. Zeng et al. (2020) introduced PC-NBV, a novel point cloud-based deep learning NBV planner that uses a point cloud directly to infer the NBV. The use of a point cloud directly skips the voxelization process used in the majority of the literature Bircher et al. (2018); Mendoza et al. (2020a); Song et al. (2022) and Vasquez-Gomez et al. (2021) to speed up the prediction of the NBV for maximizing the coverage of viewpoints. Reinforcement learning has also been applied to solve the NBV problem. Wang et al. (2024) uses Reinforcement learning based on the Q-learning framework, which uses features extracted from the point cloud using pointNet++ introduced by Qi et al. (2017). Peralta et al. (2020) also implemented an NBV policy for reinforcement learning called scanRL to develop and create an NBV path plan to improve the quality of houses based on the house3K dataset. Dhami et al. (2023b) proposed Pred-NBV, a prediction-based Next best view planner using the

point cloud completion algorithm poinTr-C developed by Yu et al. (2021). The point cloud completion algorithm uses the completed point cloud to measure information gain and decide the NBV for trajectory planning. Later, Dhami et al. (2023a) introduced MAP-NBV, an extension of pred-NBV adopted for multiple agents in a decentralized control environment. Chen et al. (2024) developed a Generalizable NBV (GenNBV) policy that generalizes across datasets using a 5D action space and a novel multi-source state embedding combining geometric, semantic, and action features for efficient 3D scene scanning.
Guédon et al. (2022) Introduced SCONE, a supervised deep learning-based approach for surface coverage that converts surface coverage into a visibility score to scale to large areas. This uses a depth sensor to calculate the volume occupancy of the surface using a grid of local point clouds. This method requires 3D supervision and then uses the signals to expand to large scenes, scaling the point cloud-derived volume occupancy surface. Guédon et al. (2023) further enhanced SCONE with the MACARONS, the current state-of-the-art in self-supervised Exploration and reconstruction using an RGB camera. It works in a self-supervised manner to autonomously explore and predict the next best view so that it can reconstruct a scene simultaneously using only an RGB sensor.

2.3. Depth sensing technologies

The interpretation of depth is crucial to any autonomous exploration and 3D reconstruction as it gives insight into a robot's surrounding environment. Different sensors and systems provide different depth measurements. Multi-view stereo is extensively used in 3D depth estimation using the interior and exterior orientation of images (Scharstein et al., 2001). Stereo vision-based depth estimation algorithms can be categorized into local, global, and semi-global methods (K. Zhou et al., 2020). These methods are heavy in computing features with cost functions. Song et al. (2022) implement precise 3D model reconstruction using CasMVSNet by Gu et al. (2020), a deep-learning-based multi-view stereo (MVS) approach for depth estimation. CasMVSNet calculates depth maps with multiple small-cost volumes, progressively narrowing the depth hypothesis range to produce refined depth maps. These maps are combined into a single 3D model using surface point cloud mapping. In contrast, Respall et al. (2020) use LiDAR and depth cameras for a path-planning algorithm that explores unknown 3D spaces while generating an online occupancy map. This algorithm, adaptable for indoor or outdoor use, employs a depth camera and 360° LiDAR and includes an improved RRT sampling method and enhanced configuration orientation. Their approach is faster and more accurate, as confirmed by simulations and real-world tests with multi-rotors in indoor and outdoor environments. Madhuanand et al. (2020) do a comparison of deep learning based on a dual Convolutional neural network (CNN) and Generative adversarial networks (GANs) to generate pixel-wise disparity maps through explicit training. Upon comparison, they observed that CNN performs better than GANs. It was extended for further development by implementing self-supervised learning to estimate depth from oblique UAV videos (Madhuanand et al., 2021). Nex et al. (2024) introduced a global benchmark dataset aimed at different 3D reconstruction and depth estimation tasks, upon which Hermann et al. (2024) tested various 3D reconstruction and Depth estimation methods, including deep learning-based multi-view stereo, offline, and online depth estimation. They also evaluated various Learning-based (CNN and self-supervised) Depth estimation methods. They also evaluated the dataset for 3D reconstruction using Neural Radiance Fields (NeRFs) (Mildenhall et al., 2020). Figure 2.5 shows a Monocular and a stereo camera

Figure 2.5 Monocular (top right) and Stereo cameras Zed X One

2.4. Multi-UAV coverage planning

Hardouin et al. (2020) introduced a centralized multi-UAV architecture with weighted directed graphs is proposed to assign clusters of viewpoints and content sharing among the UAVs, which is assumed to always be present. They assign a cluster of viewpoints to each robot using a greedy strategy. However, they also considered a centralized architecture where communication is assumed to be perfect. The figure 2.6 shows how Hardouin et al. (2020) assign a collaborative exploration strategy to explore and reconstruct a 3D scene.

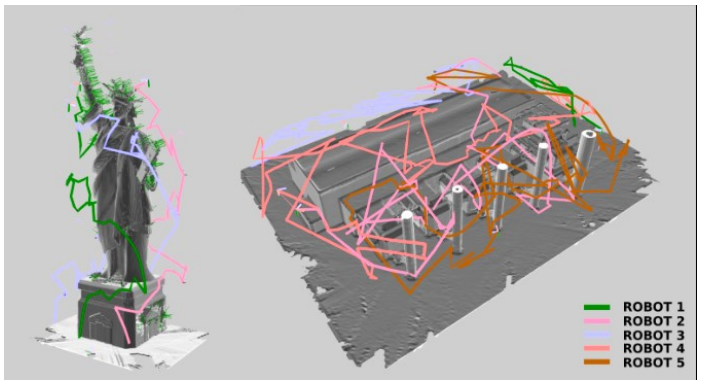


Figure 2.6 Collaborative exploration strategy for 3D Reconstruction (Hardouin et al., 2020).

Dhami et al. (2023a) developed a multi-agent, decentralized, prediction-based NBV planner with a greedy assignment strategy for agent selection based on information gained for each viewpoint. B. Zhou et al. (2022) proposed RACER, a decentralized multi-UAV exploration strategy that leverages a hierarchical grid (HGRID) for efficient, pairwise task allocation, is shown in Figure 2.7. It allows quadrotors to explore distinct regions with minimal, asynchronous communication. By framing task allocation as a Capacitated Vehicle Routing Problem (CVRP), RACER optimizes path lengths and balances the workload among UAVs. A hierarchical planner, guided by global coverage paths, further boosts exploration efficiency. This approach has been validated in real-world tests, demonstrating robust multi-robot state estimation in challenging environments. They use an occupancy grid-based exploration strategy developed using a depth camera and represent the information of a scene using voxels.

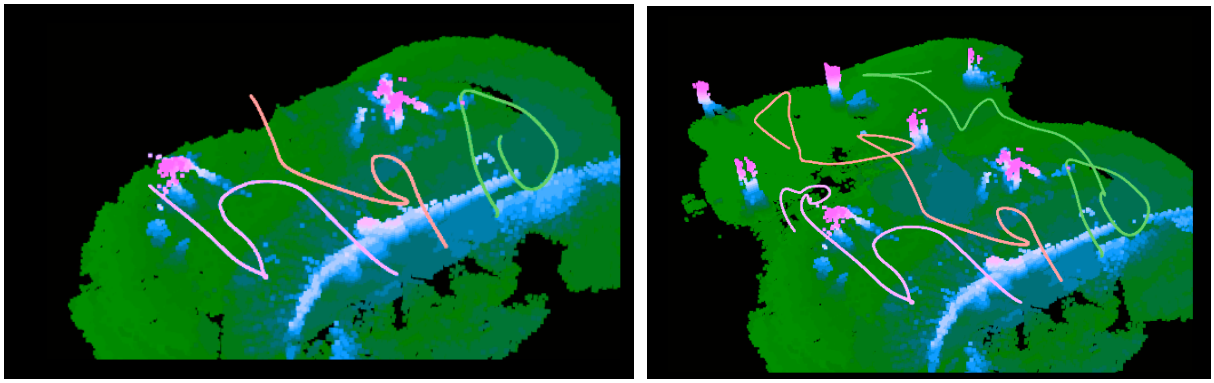


Figure 2.7 Decentralized collaborative exploration: Initial map (left) and final map (right)

Ribeiro & Basiri (2024) use a system of multiple UAVs equipped with LiDAR sensors coordinating through a decentralized communication system. Each robot has its own frontier-based local Octo map, merging them based on distance if they get close (Hornung et al., 2013). The LiDAR is used to sense nearby surroundings for the UAV and keep track of the explored frontiers. Anastasiou et al. (2024) proposed a novel approach for inspecting complex 3D infrastructure in indoor and outdoor environments, keeping the time and communications constraints in check. They use heterogeneous sets of decentralized UAVs named explorer, i.e., equipped with a LiDAR sensor and a data acquisition UAV called photographer. The conversion of the study area into a voxel-based representation and bounding of the infrastructure of interest is subsequently explored and merged for each UAV, and the inspection path planning is performed using the voxel-based representation as a gain function. Figure 2.8 shows a comparison of a centralized and decentralized communication system for collaborative UAVs.

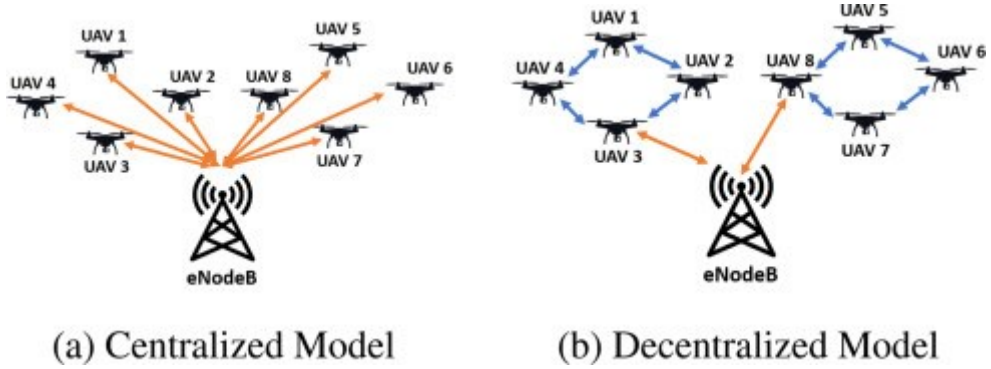


Figure 2.8 Centralized and Decentralized Communication Systems in UAVs (Mishra et al., 2022)

Most Collaborative exploration and reconstruction strategies in literature use volumetric representations of an environment to assess information gain. This approach makes them less likely to scale to large areas due to the computationally expensive nature of a voxel-based representation (Guédon et al., 2023).

3. SIMULATION ENVIRONMENT AND DATASETS

Due to various practical limitations, this research used a simulated environment rather than real UAV data collection. Simulation can provide an inexpensive, safe, and repeatable method to develop and test autonomous exploration strategies. Not only can environmental parameters be controlled, but also sensor configurations, trajectory planning, etc., and much rigorous experimentation can go without the variability and dangers associated with flying a physical drone. So, by simulating realistic scenes and UAV camera behaviour, we are confident that we are sufficiently testing, validating, and refining the core algorithms associated with 3D reconstruction and exploration in controlled conditions before any physical implementation (Gschwandtner et al., 2011).

This chapter provides an overview of how the 3D simulation environment is set up using the Blender tool (Blender Online Community, 2024) and how the infrastructure object is established. This chapter also elaborates on the Blensor simulator. This research utilizes this to emulate a LiDAR sensor along the same NBV camera trajectory.

3.1. 3D Simulation Environment Using Blender

Autonomous Exploration of an object in 3D space is often facilitated by developing a proof of concept of a system in a controlled environment. Advanced robotic simulation systems replicate many real-world characteristics and have evolved to enhance control, planning, and execution of outcomes. Many simulation software tools support generating realistic 3D environments like Air Sim, Gazebo, and Unreal Engine, which not only model and build virtual scenes, but also integrate with robot operating systems to handle control states of robots (Koenig & Howard, 2004; Shah et al., 2017). While these simulation environments are compelling and offer a variety of modelling capabilities, autonomous exploration tasks can be simulated in high-performance graphics applications like Blender and Autodesk Maya. Blender is a versatile open-source computer graphics software for 3D animation, modeling, and interactive and parametric 3D applications. It provides extensive support for customizable camera objects and configures various applications, including Monocular, multi-view stereo, trajectory-based mapping, and key-frame rendering tasks.

Blender workflows support many 3D computer graphics formats, including OBJ, PLY, GLTF, etc. These are used to represent an environment in a 3D system. In most applications, 3D meshes form the foundational structure for representing geometry in a virtual scene. A mesh is composed of vertices, edges, and faces that define the shape and surface of a 3D object. It supports importing and editing meshes, allowing precise control over topology and surface details. This is particularly useful in simulation-based autonomous exploration, where accurate and detailed mesh representations of environments are essential. The quality of 3D meshes directly impacts the rendering results for the objects we typically aim to reconstruct, as we aim to facilitate a simulation of a real-world infrastructure object.

The application domain considered is infrastructure objects, including, but not limited to, powerlines, bridges, buildings, and monuments. These datasets are a 3D mesh-based representation to simulate how an autonomous robot envisions the environment with complex objects.

To achieve our objectives, a few Underlying assumptions are defined below

1. **GNSS connectivity:** We assume that the region of interest has GNSS connectivity. This means that we have an accurate sense of location, so the UAVs do not need to simultaneously localize themselves with respect to one another. In addition, the 3D bounding box of the study area, or the ROI, is assumed to be known beforehand.
2. **Presence of light source:** We assume that the Exploration and reconstruction of the scene is taking place in the presence of a light source, i.e., either the sun or any other light source that illuminates the entire scene. This is because we are relying on using an RGB camera, which is a passive sensor.
3. **Communication between UAVs:** The communication channel between UAVs is assumed to be unhindered, i.e., it is assumed that the communication between the UAV and the ground station is always active.
4. **Coordination among the agents:** We assume that the UAVs are aware of their surroundings and can avoid each other's paths to prevent collision.
5. **Number of UAVs in a collaboration:** To test the working and proof of concept of the methodology and prevent computational complexity, we are going to be working with at least 2 UAVs. Furthermore, the UAV type is assumed to be multi-copters.
6. **Fault tolerance:** It is assumed that the UAVs are fault tolerant. I.e., all the UAVs are active all the time. Neglecting any faults.
7. **Take-off and Landing stations:** It is assumed that the take-off and landing positions are decided based on the type of infrastructure being considered for exploration.
8. **Weather:** The weather is assumed not to influence the performance of the UAVs.

3.2. Camera simulation parameters

To ensure high-quality renders and realistic simulated RGB images for data collection and 3D reconstruction, we replicate the camera configuration of the DJI Mini 4 Pro drone (*DJI Mini 4 Pro - Specs - DJI*, n.d.). The simulated camera in Blender is configured to closely match the physical properties of the drone's onboard camera system. This includes using a sensor size of 9.6 by 7.2 millimetres (mm) and a focal length of 20 mm to achieve a field of view of approximately 84 degrees. A focus distance of one meter is used, which can be adjusted by targeting objects within the scene. Depending on the use case, two output resolutions are applied: 1920 by 1080 pixels for Full HD output and 3840 by 2160 pixels for 4K Ultra HD. Some scenes require high-resolution images for better reconstruction due to lower detailed textures and redundant objects. The Table 3.1 summarizes the two camera intrinsic configurations for the simulation environment.

The Blender camera object is the fundamental component to render views generated by the coordinated collection of each UAV's data. There are two different lens options to represent a 3D object in a 2D image.

1. Perspective camera
2. Orthographic camera

We use a perspective camera to deal with the intrinsics (interior orientation) and extrinsics (exterior orientation) of the camera object for each instance of the camera. As defined in Camera simulation parameters, we consider the Focal length, field of view, and the principal point to resemble the Mini 4 Pro camera.

Camera Characteristic	Specifications
Camera Type	Perspective
Field of View (84°)	Sensor: 9.6×7.2 mm
Lens Focal Length	20 mm
Aperture (f/1.7)	f-stop: 1.7
Focus Distance	1 m
Resolution	1920 × 1080 (Full HD) and 3840 x 2160 (4K Ultra HD)

Table 3.1 Different configuration parameters for the Blender camera

3.3. Dataset characteristics and scene complexity

We use different 3D textured mesh datasets sourced from SketchFab (*Explore 3D Models*, n.d.). All the models we use to represent our scenes are granted under the Creative Commons (CC) license for public use and are publicly available to download. The data sets used in the simulation environment are listed in Table 3.2.

Name	NumVertices	Length	Width	Height
Alhambra	4531089	50.14	5.45	21.05
Bridge	4214934	62.18	9.93	22.39
Liberty	1251900	43.56	9.48	51.09
Powerlines	2466225	62.50	4.05	62.50

Table 3.2 Infrastructure objects used in the research

The scene's complexity can be defined by the number of vertices the respective meshes have. The units of measurement in our simulation setup are meters (m).

3.4. Coordinate system management

Pytorch3D is a deep learning library for research in 3D Computer Vision. It is a collection of implicit rendering methods for handling and processing 3D mesh and point cloud data for 3D deep learning tasks (Ravi et al., 2020). In PyTorch3D, camera orientations can be specified using spherical coordinates: distance. r, elevation θ, and azimuth ϕ.

These values benefit orbit-style rendering around a target object, often assumed to be the scene's origin.

To compute the camera's position in 3D space, the spherical coordinates are first converted to Cartesian coordinates using the following set of equations: 3.1 (LibreTexts, n.d.):

$$\begin{aligned} x &= r \cdot \cos(\theta) \cdot \sin(\phi) \\ y &= r \cdot \sin(\theta) \\ z &= r \cdot \cos(\theta) \cdot \cos(\phi) \end{aligned} \tag{3.1}$$

Where:

- r Is the radial distance from the origin,
- θ is the elevation angle (from the horizontal plane),
- ϕ is the azimuth angle (from the X-axis in the XY-plane),
- Angles are in radians.

This gives the camera position given by equation 3.2:

$$C = (x, y, z) \tag{3.2}$$

After determining the camera position, the rotation matrix R is computed using a look-at transformation. This aligns the camera so that it points toward the origin. The camera's forward direction is:

$$f = \frac{-C}{\| C \|} \tag{3.3}$$

With the up vector $u = (0,1,0)$, the right and true up vectors are:

$$r = \frac{u \times f}{\| u \times f \|} \tag{3.4}$$

$$t = f \times r$$

Then, the rotation matrix R is formed by stacking the t vectors as rows or columns. Subsequently, the translation vector T is computed as:

$$T = -R \cdot C \tag{3.5}$$

Together, (R, T) describe the complete transformation that places the camera at a spherical position and orients it to look at the target.

PyTorch3D and many 3D Graphics libraries use a right-handed coordinate system with X right, Y up, Z forward (XYZ). Blender, on the other hand, uses X right, Z up, Y backward, requiring a coordinate remapping where (x, y, z) are transformed to (x, −z, y) for compatibility (Pytorch3d.Transforms— PyTorch3D Documentation, n.d.).

3.5. Blensor LiDAR Sensor Simulation

To evaluate the performance of monocular depth estimation against real LiDAR data, we simulate a Velodyne VLP-16 sensor using the Blensor plugin within Blender (VLP 16 | Ouster, n.d.). The MACARONS-generated trajectory is first imported into Blender as a sequence of camera poses, representing the path followed by a UAV equipped with a monocular RGB sensor. Using Blensor's LiDAR simulation tools, we configure a custom 16-channel rotating sensor that closely mimics the VLP-16's vertical and horizontal angular resolution, field of view, and rotation rate. The simulated scans are recorded at each keyframe along the trajectory to generate synthetic point cloud data.

This simulated LiDAR dataset allows for a direct visual and quantitative comparison between the depth predicted by the monocular RGB model and the ground truth-like depth captured by the virtual VLP-16, as shown in Figure 3.1. We assess depth consistency, structural completeness, and range accuracy across both modalities by aligning the viewpoints and matching keyframes. This setup highlights the strengths and limitations of learning-based monocular depth estimation in autonomous exploration tasks.

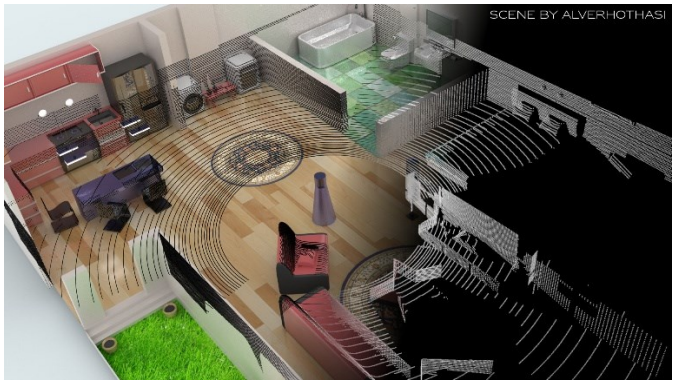


Figure 3.1 Velodyne VLP-16 sensor simulation in Blensor (Blensor.Org, n.d)

4. METHODOLOGY

The chapter provides a detailed description of the detailed methodology behind the collaborative UAV systems, which provide descriptions for the Task Assignment and path planning algorithms used. The second section provides the methodology from the initialization to the 3D reconstruction of the infrastructure objects and accuracy assessment metrics.

Methodological Workflow

This section outlines the methodology for autonomous exploration and 3D Reconstruction of an infrastructure object using a collaborative UAV strategy. It details how multiple UAVs are coordinated to collect, process, and manage data for 3D reconstruction. The workflow begins with preparing a 3D infrastructure environment that aligns with the Next Best View (NBV) method, followed by task assignment and multi-UAV coordination, data collection through Blender simulations, the 3D reconstruction pipeline, and finally, the evaluation of reconstruction quality. Figure 4.1 provides and overview for the different steps of the methodology. We start with setting up the 3D scene in Blender, then moving towards setting up the MACARONS-NBV method alongside the hyperparameters for the model. Then we follow-up with exporting the predicted camera poses to a collaborative setting using task assignment and then simulating data collection in Blender and evaluating the 3D reconstruction using Metashape and Cloud-to-Mesh and Cloud-to-Cloud metrics using the ground-truth.

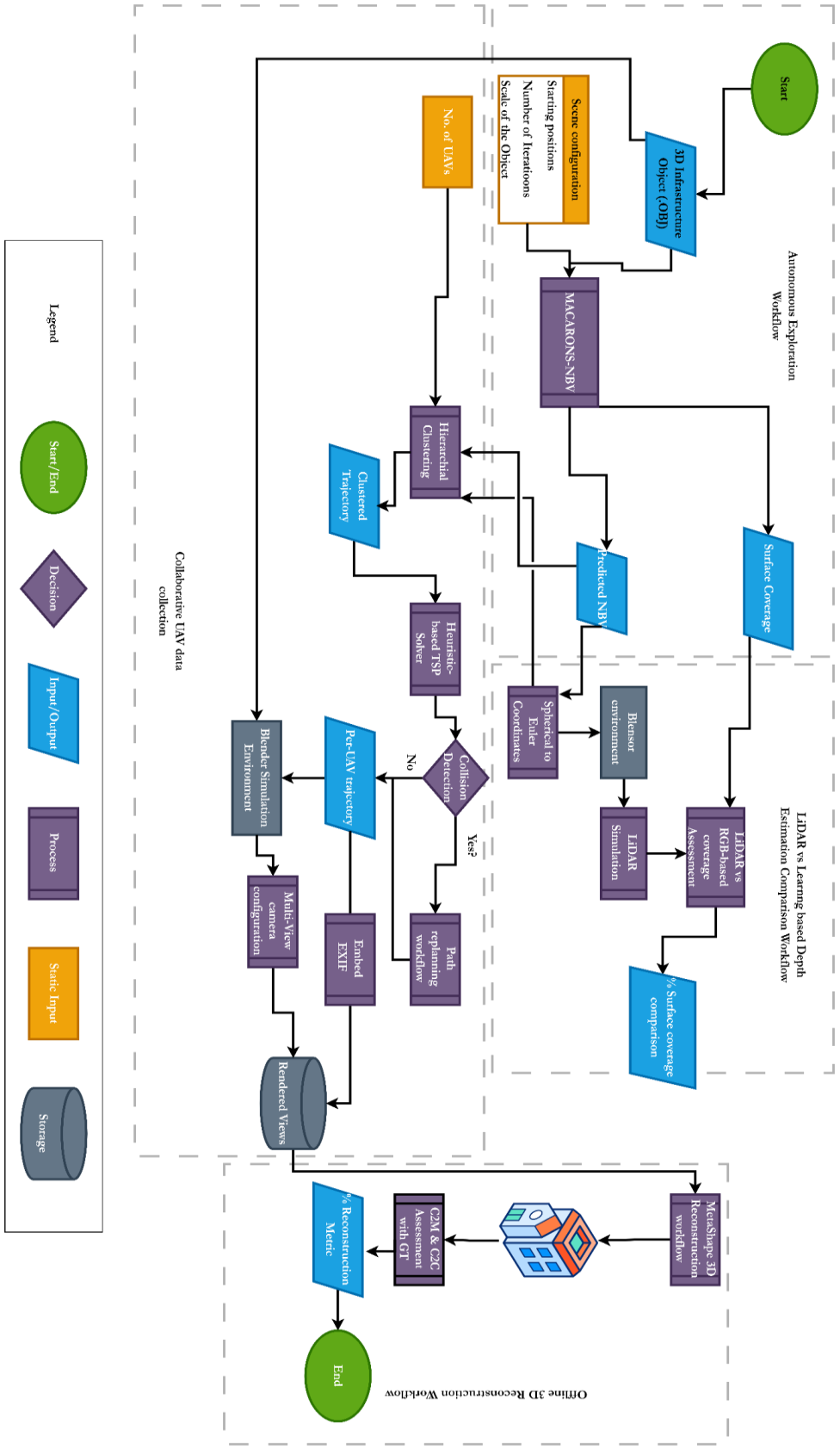


Figure 4.1 Overview of the end-to-end Autonomous Exploration and 3D Reconstruction Methodology

4.1. Scene Setup

We prepare the infrastructure scenes using the following steps in blender. Figure shows an overview of the inputs and outputs.

4.1.1. Preparing Scenes in Blender

The 3D models for a diverse set of scenes used are acquired from the SketchFab website, where they were created in Blender. Custom scenes include objects like powerlines, buildings, and bridge scenes, such as Blender scenes in (*. blend). The units of measurement are meters. This is the initial step for exporting our scene as input for our NBV algorithm. The blend scenes consist of textures and objects that make up the complex representation of a real-world 3D scene.

4.1.2. Defining object bounds

The bounding box (XMin, XMax) is one of the inputs for our 3D exploration pipeline, defining where the NBV exploration should focus. It restricts the camera movement (x, y, z, θ, φ) to this specific bound, preventing exploration of non-important infrastructure and regions of the entire scene to generate the next-best views. The bounds are helpful for the following reasons.

1. Providing a time-effective solution, and the camera does not wander into unknown regions of the Scene.
2. Prioritizing objects based on importance, the initial settings of the camera and scene object are computed based on the bounding box provided.

When defining the novel scenes, the NBV workflow considers the entire scene as the exploration range by default. However, by creating an object representing the area's XMin and XMax, the scenes' parameters and camera configurations are essential for a good quality and optimal exploration strategy for the NBVs.

4.1.3. Exporting the 3D scene and Bounding Box

Once the 3D scene is finalized and a bounding box has been created, we export all the mesh components alongside the textures into a Wavefront (.obj) format widely used in computer vision. It is crucial to group the objects and materials of the scene so that they are exported as one. It is also essential to get rid of the absolute path of texture files when exporting the object, so that the material template file (mtl) does not have absolute paths to the texture files. The structure of a working folder of the exported scene looks like as shown as Figure 4.2.

```
.
└── powerline/
    ├── powerline_rgb.obj          # Main geometry file (vertices, faces, normals, UVs)
    ├── powerline_rgb.mtl          # Material file referenced by the OBJ file
    ├── texture_diffuse.jpg
    ├── texture_normal.jpg
    ├── texture_specular.jpg
    └── bbox/
        └── powerline_bbox_bounds.obj
```

Figure 4.2 Configuration of a 3D infrastructure file folder

The scene's bounding box is also exported as a Wavefront (OBJ) file. Figure 4.3 shows the bounds of a 3D scene and the bounds of the infrastructure object within those bounds.

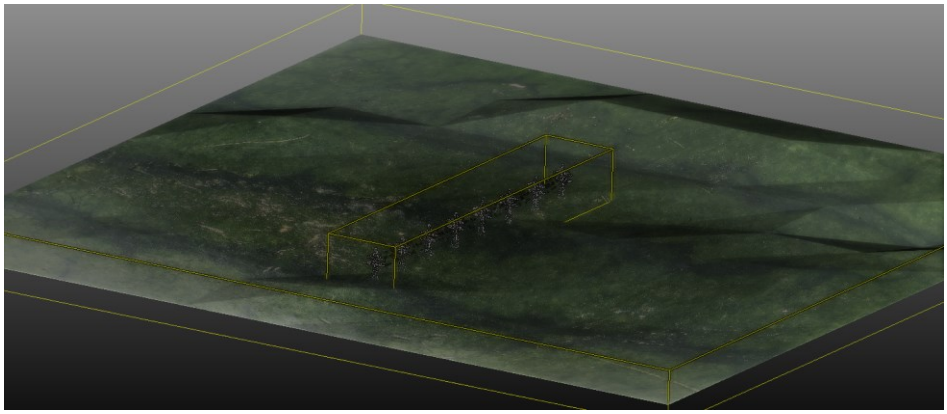


Figure 4.3 3D scene bounds and exploration bounds

. Figure 4.4 shows the data preparation steps for the NBV exploration method.

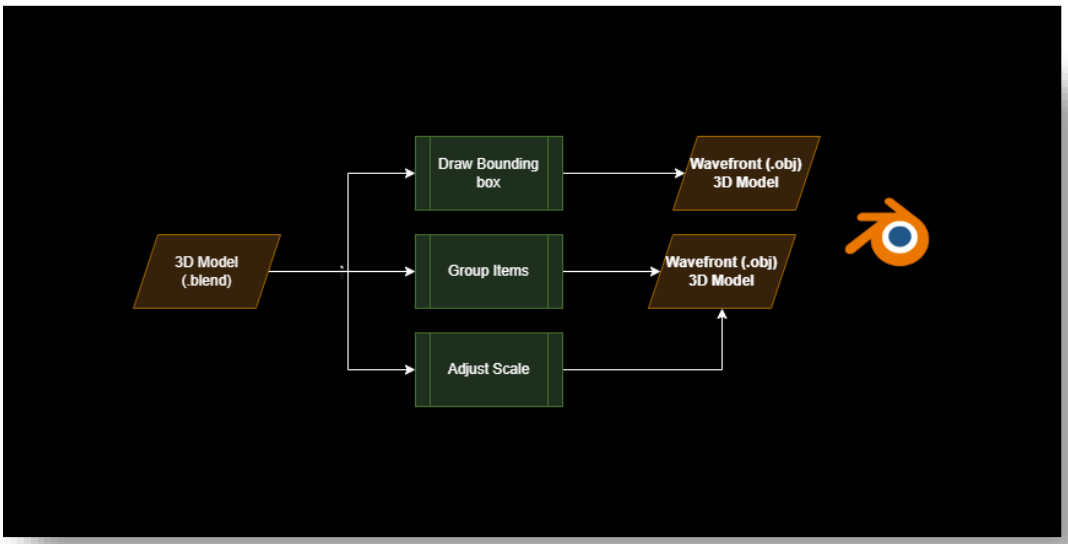


Figure 4.4 Overview of the data preparation step for the NBV algorithm

4.2. Autonomous Exploration with MACARONS

We use MACARONS as our NBV method. It is a self-supervised NBV method that trains and explores in an online fashion from a joint learning strategy (Guédon et al., 2023). It is based on the depth estimation from a sequence of RGB images acquired with a monocular RGB camera, by creating a cost-volume representation, starting with a sampled set of camera poses. These depth maps and camera poses are then used to calculate a volume occupancy field used by surface coverage gain and create a self-supervised training pipeline to create pseudo-ground truths and predict the NBVs with the highest coverage gains from an already sampled set of camera poses. A camera pose is encoded with five parameters (x, y, z, θ, φ), where x, y, and z represent the 3D coordinates. In contrast, θ and φ represent the camera's orientation parameters (Azimuth, Elevation).

4.2.1. Scene Pre-processing configuration.

Before an exploration pipeline starts, a few key configurations are required for the inference. These include

- Occupied poses
- Scaling factor for the scene to match MACARONS-NBV.
- Starting trajectory positions, or multiple starting positions in spherical coordinates (x, y, z, θ, φ).
- Camera bounds ($xCam_{min}$, $xCam_{max}$).
- Scene bounds (X_{min}, X_{max}).
- Contrast factor, by default set to 1.0.

By default, the settings.json parameters are computed from the 3D Mesh. However, that has a few drawbacks, such as considering the entirety of the scene as the scene bounds, and providing camera poses to explore the complete scene. This is resolved by parameterizing the scene using a custom data loader to read and overwrite the settings parameters.
The reference bounding box exported alongside the wavefront (OBJ) file is part of a custom script, provided in the appendix, that reads the bounding object and computes the scene and camera bounds. Furthermore, the grid parameters are also calculated by replicating the 3D scene and camera parameters defined in the MACARONS pre-processing.
Once the image and camera configurations are generated and we define the bounding boxes (X_{min}, X_{max}, $XCam_{min}$, $XCam_{max}$), the starting position (x_0, y_0, z_0, θ_0, φ_0), and the number of NBV iterations we want to explore our 3D scene. The NBV workflow is ready to run on a GPU-enabled system.

We use pretrained weights for all three neural modules: Depth, Volume Occupancy, and Surface Coverage Gain. The Volume Occupancy and Surface Coverage Gain modules are both pretrained on the ShapeNetCore v1 3D dataset (Chang et al., 2015). The objects used for pretraining include those structurally comparable to the infrastructure objects, like buildings, towers, and bridges. For this reason, we adopt the MACARONS configuration that has already been pretrained on this dataset. This saves valuable time and avoids the need for retraining on large-scale 3D scenes. The self-supervised architecture of the method scales to large 3D scenes using a monocular RGB camera for its input instead of a dedicated depth sensor. The models are implemented in PyTorch, and PyTorch3D is used for camera management, ray tracing to generate synthetic renders, and general 3D data processing.

4.2.2. Adjust scene scale

We typically use Blender's 1:1 scale environment model to reflect real-world dimensions. However, for NBV exploration, we uniformly scale down the scene (by ½, ¼, or ⅛) to match the expected input range of the pipeline. This scaling improves grid consistency and camera sampling without affecting mesh quality, as geometry and UV mapping remain intact. The scale factor is reapplied to the exploration task to restore real-world measurements. This is an essential step because we choose the pre-trained model configuration, which is trained on ShapeNet objects of size (1-5m). We scale our scene by a factor to match the bounding box, making it more comparable, and the estimation of surface coverage becomes more consistent. For exploring novel complex scenes, the autonomous exploration pipeline expects a Settings configuration (settings.json) file to define the key hyperparameters defined above, regarding the 3D Scene, including the resolution of the 3D grid for sampling camera poses, the bounds for the exploration, and the bounds for the camera exploration. According to the authors, the camera bounds are 1.1x of the scene bounds. Figure 4.5 provides an overview of these steps

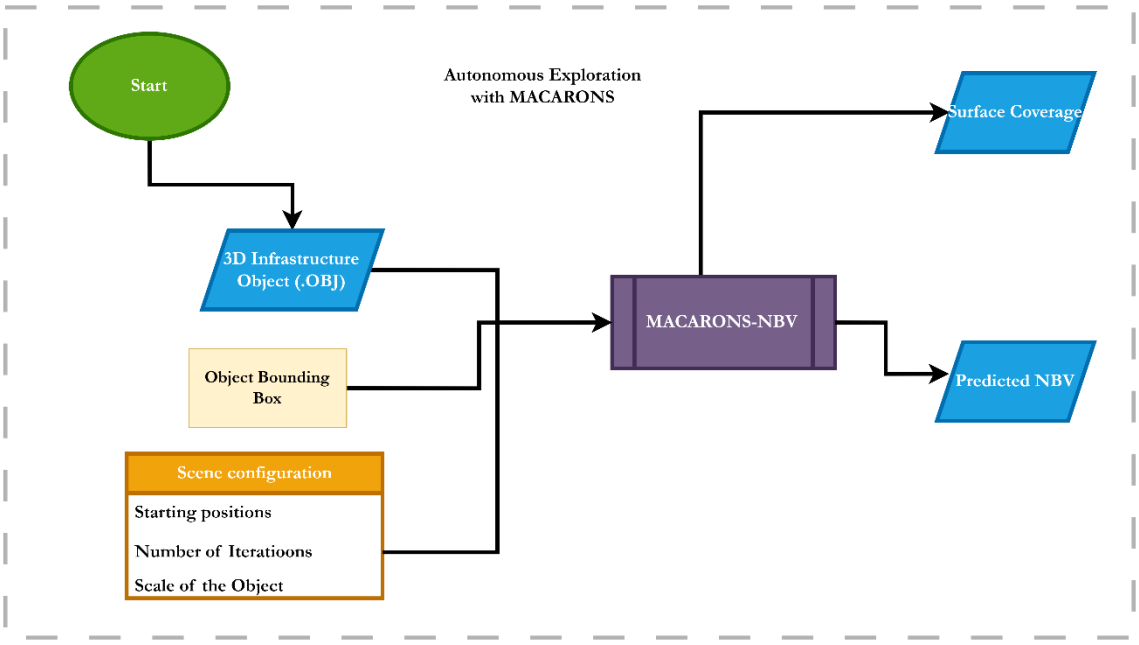


Figure 4.5 Overview of autonomous exploration with MACARONS-NBV

4.2.3. MACARONS Architecture

The MACARONS NBV method has three distinct neural components that work in synergy to estimate depth from monocular RGB images and predict the NBVs. These modules are

1. **Depth module** predicts depth from a sequence of images, It, It−1…, It−m, as well as the corresponding camera poses ct, ct−1, …, ct−m, with adaptive bounds based on the scene's bounding box defined in the input settings.
2. **The Volume occupancy** module is a Transformer-based neural network that uses the predicted depth maps to construct a probabilistic occupancy field, estimating for each 3D point whether it is likely to be occupied or empty.
3. **Surface coverage gain** module estimates how much new information a camera pose can gain. It uses the predicted volume occupancy field and past camera views, while accounting for occlusion. This helps choose the next best view in large scenes.

Figure 4.6 shows the workflow and general architecture of the three neural modules.

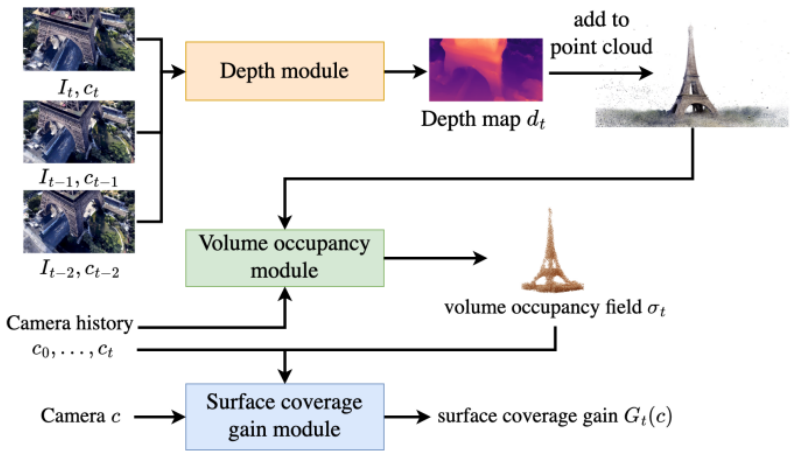


Figure 4.6 Overview of the three neural modules of the MACARONS-NBV method (Guédon et al., 2023)

4.3. Analysis and Export of Generated Camera Trajectories

The MACARONS-NBV exports camera poses depending on the initial NBV iterations defined when an exploration starts. During each iteration of NBV, the model performs evaluations for four camera poses. N_{iter} = number of iterations (varies from 100 to 150),

- N_{pose} = 4 number of camera poses selected per iteration,

Then the total number of NBVs is:

$$\mathrm{N_{NBV}} = \mathrm{N_{iter}} \times \mathrm{N_{pose}} \tag{4.1}$$

The number of N_{NBV} is computed as the product of the number of exploration iterations N_{iter} And the number of camera poses selected per iteration. N_{pose}. In our setup, $N_{\text{pose}} = 4$, while $N_{\text{iter}} \in [100,150]$ Depending on the scene and coverage, convergence. We typically skip the first four iterations as they are the starting poses of the exploration and are not as significant as the remaining NBVs. The output trajectory is a series of NBVs that cover the entire scene. Precisely, the output consists of

- Surface coverage percentage
- X_Cam_history (Viewpoint locations)
- V_Cam_History (View direction, Azimuth, and Elevation)

Alongside the NBVs, we also have the surface point cloud generated while exploring the scene. The density of the point cloud is comparatively less because of subsampling of the already lower resolution images (456x256) pixels used for estimating depth and volume occupancy. This encouraged us to adopt an explore-and-exploit strategy for a high-quality and dense 3D reconstruction of the Scene.
The camera management module generates virtual camera poses around the 3D object and renders views using ray casting in PyTorch3D. Each camera pose is defined in spherical coordinates, which are then converted into a 3D transformation matrix (rotation + translation) to position the camera in the scene.

4.3.1. Coordinate System Conversion

The NBV trajectory provides the camera poses in a spherical coordinate system. The MACARONS codebase provides utility functions that transform camera positions (X_cam_history) and spherical view angles (V_cam_history) into Blender-compatible curves explained in section 3.2. It first converts spherical coordinates into Cartesian viewing direction vectors (rays) and then computes look-at points by projecting them from each camera position. The positions and targets are adjusted to Blender's coordinate system, optionally mirrored, and scaled for visualization.

4.3.2. Adapt coordinates to Blender's coordinate system

We manage the camera coordinates as a direction vector from each camera position (Camera_X_List) to its corresponding look-at point (Camera_Look_At). This vector is normalized and used to compute the camera's rotation so that its Z-axis points toward the target, ensuring each camera is oriented correctly in 3D space. When importing these NBVs, this step defines the 3D location and orientation of the cameras in Blender by converting the two corresponding points into a unit direction vector, which is converted to a rotation that aligns the camera's Z-axis toward the target, and this rotation is applied to the camera using Euler angles conversions defined in section 3.4. We also set the focal length to 20 mm. The other intrinsic parameters are kept as default. Figure 4.7 shows the direction vector between the camera and the target point.

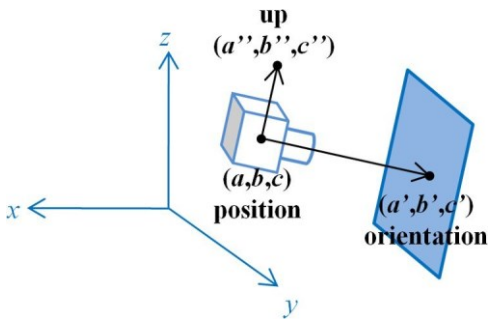


Figure 4.7 Direction vector from the camera to the target (Liu et al., 2010)

4.4. Multi-UAV Task Management

The collaborative pipeline of the Autonomous Exploration and 3D Reconstruction workflow builds on top of the NBV trajectory obtained in the previous step. The collaborative aspect of the research handles how multiple agents, i.e., UAVs, are assigned to different regions to exploit and gather information. Figure 4.8 provides an overview of the multi-UAV Task Management module.

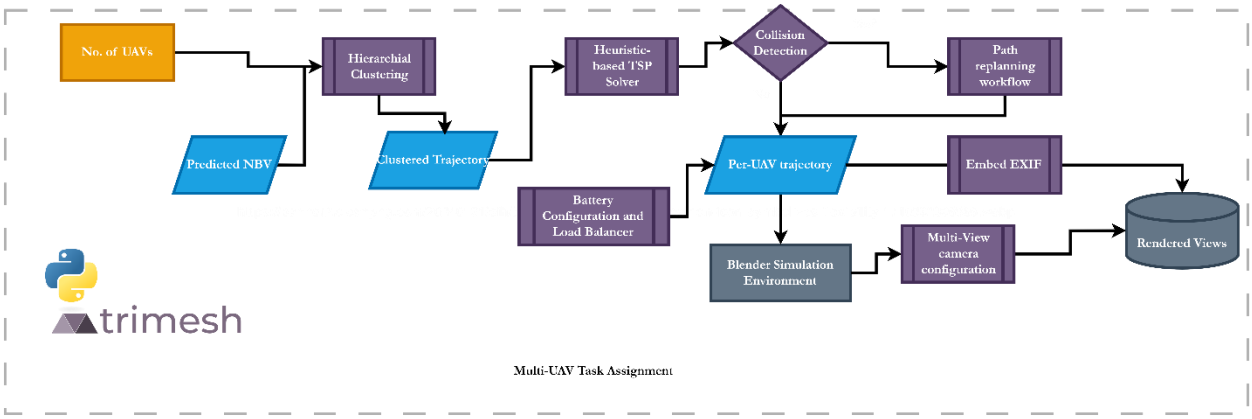


Figure 4.8 Overview of multi-UAV task assignment

4.4.1. Trajectory Clustering

As a first step, we apply hierarchical clustering on the 3D positions of NBV camera poses to partition the task space into spatially coherent clusters. This clustering helps distribute the workload across multiple UAVs, ensuring that each UAV is assigned to a localized region of operation.

Instead of randomly assigning the found NBV viewpoints to collaborative UAVs, we use an unsupervised clustering method called hierarchical clustering from the SciPy clustering module. We proceed with an agglomeration technique with the following characteristics

1. Use Euclidean distance as a pairwise distance metric
2. The linkage method we use is Ward's method for minimizing within-cluster variance (Murtagh & Legendre, 2014).
3. Use a max_cluster cut-off to assign a fixed number of clusters

The result is a clustered output. Each 3D camera pose, defined by its position (x, y, z) and orientation (look_x, look_y, look_z), was assigned a cluster label representing a spatial region or bound intended for UAV-specific exploitation during the mission. Figure 4.9 shows a clustered trajectory for a 3D trajectory of a camera object divided into three clusters

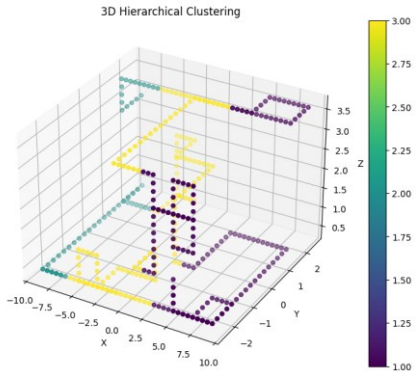


Figure 4.9 Clustered trajectory using Hierarchical clustering

The hierarchical clustering method is mathematically modelled as:

Let each cluster $C_i \subseteq \mathcal{N}$ Correspond to the NBV set allocated to the UAV u_i. The number of clusters is set equal to the number of available UAVs M, and the clustering is based on minimizing intra-cluster spatial variance in equation 4.2 as:

$$\textbf{minimize} \sum_{i=1}^{M} \sum\{n_j \in C_i\} \|p_j - \mu_i\|^2 \tag{4.2}$$

where $\mathbf{p}j \in \mathbb{R}^3$ is the position of NBV nj, and $\boldsymbol{\mu}i$ is the centroid of the cluster C_i.

This yields:

$$\mathcal{C} = \{C_1, C_2, \dots, C_M\}, \quad \bigcup_{i=1}^{M} C_i = \mathcal{N}, \quad C_i \cap C_j = \emptyset \quad \text{for } i \neq j \tag{4.3}$$

Each cluster is then assigned to a corresponding UAV. The significant advantage of calculating hierarchical clusters is that they are adaptable to expansion as the number of UAVs increases. We use the agglomerative strategy to cluster our points in 3D, which works from bottom to top. It starts with individual points as clusters and merges the closest pair of clusters at different levels until one cluster exists. This unsupervised clustering technique uses the XYZ locations of the NBVs as features. Using a dendrogram helps illustrate how the clusters are arranged in space. The Figure 4.10 gives an overview of the hierarchical clustering method

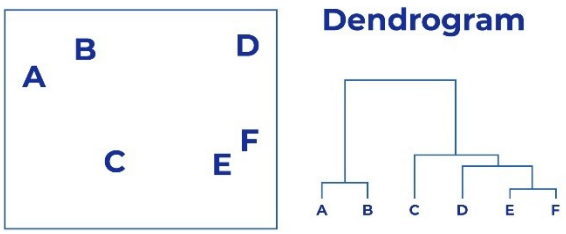


Figure 4.10 Dendrogram for viewing a hierarchical cluster (Understanding Hierarchical Clustering & Its Use Cases, 2022)

4.4.2. Assign Tasks using TSP-based clustering

The clustered trajectory for each UAV is then processed using a Traveling Salesman Problem (TSP) solver from the networkx Python module. This solver computes an approximate solution to the TSP using heuristics that aim to minimize the overall travel cost. In our implementation, we evaluate two heuristic strategies.

1. Distance based Heuristic
2. Elevation and distance-based custom weightage

The classical Travelling Salesperson Problem (TSP) aims to find the shortest possible route that visits a set of points once and returns to the starting point, minimizing Euclidean distance. However, for UAVs, this problem is scene dependent. Hence, energy consumption depends on horizontal distance and vertical displacement due to altitude change, which impacts battery usage.

To account for horizontal travel and energy consumed due to elevation, we define a modified cost between two points. i and j As:

$$\text{Cost}(i,j) = \alpha \cdot d_{ij} + \beta \cdot \Delta z_{ij} \tag{4.4}$$

Where:

- $d_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}$ gives horizontal (2D) Euclidean distance
- $\Delta z_{ij} = |z_i - z_j|$ accounts for absolute elevation change
- α = weight for horizontal distance (default: 1.0)
- β = weight for elevation penalty (default: 0.5)

As a result, for the 3D coordinates (x_i, y_i, z_i), the custom cost encourages minimizing distance and altitude change, with the following tunable priorities.

- $\alpha = 1.0$ Adds full weight to Euclidean distance, reflects standard TSP.
- $\beta = 0.5$ assigns a moderate penalty for climbing or descending, which adds cost when elevation changes occur.
- Larger β results in more aggressive battery-aware routing (e.g., to avoid steep climbs).

The first strategy focuses on a distance-based criterion, where a path that minimizes the total 3D Euclidean distance travelled between sequential camera poses within each cluster. This approach ensures efficient spatial coverage by reducing redundant movement between viewpoints.

The second strategy incorporates a composite cost function that balances 2D distance and altitude change, the latter being a proxy for UAV battery consumption. The output paths are then evaluated based on a total_path_length method to visualize the optimized flight path for each UAV.

4.4.3. Collision Detection

Collision avoidance is significant when a trajectory is assigned to each UAV. Whenever a trajectory is assigned, we must check that the path does not pass through or intersect with any part of the 3D object. This is done using the trimesh module in python. The selected triangles from the line-mesh intersection (explained in Appendix A) are an essential part of this, such that when a set of adjacent points $S = \{P_1, P_2\}$, we need to create a 3D bounding box from a set of selected triangles, you need to compute the axis-aligned bounding box (AABB) that encompasses all triangle vertices.

Given the bounds of selected triangles, A set of M Triangles, where each triangle has three vertices:

$$T_i = \{v_{i1}, v_{i2}, v_{i3}\}, \quad v_{ij} \in \mathbb{R}^3 \text{ for } i = 1, \dots, M;\ j = 1,2,3$$

We compute minimum and maximum bounds along each axis:

$$\begin{aligned} x_{\min} &= \min_{v \in V} v_x, \quad x_{\max} = \max_{v \in V} v_x \\ y_{\min} &= \min_{v \in V} v_y, \quad y_{\max} = \max_{v \in V} v_y \end{aligned} \tag{4.5}$$

$$z_{\min} = \min_{v \in V} v_z\,, \quad z_{\max} = \max_{v \in V} v_z$$

The two corner points define the resulting 3D bounding box:

- Minimum corner: $(x_{\min}, y_{\min}, z_{\min})$
- Maximum corner: $(x_{\max}, y_{\max}, z_{\max})$

Expanding the bounding box, by keeping the direction axis constant
Given the bounding box from the triangles, for each step of the trajectory, we consider a direction vector $\mathrm{d} \in \mathbb{R}^3$ representing from waypoint P_i to P_{i+1}.

$$\mathrm{d} = \frac{P_{i+1} - P_i}{\| P_{i+1} - P_i \|} \tag{4.6}$$

We keep the parallel component's (Along vector d) scale constant, and the orthogonal components scaled by a factor $\alpha > 1$, e.g., $\alpha = 1.5$. The result is a reference bounding box, an input for the clipping module.

4.4.4. Bounding Box Refinement

The reference expanded bounding box is part of a two-step refinement to identify potential collision zones. First, the bounding box defines a coarse 3D region of interest around a UAV path. In the second step, we intersect this bounding box with the scene mesh to extract only the parts of the geometry within the box.

This is done by clipping the mesh using the six planes forming the bounding box's sides. At each step, we retain only the part of the mesh inside the box, gradually narrowing down the geometry. The result is a part of the local mesh representing the actual geometry within the defined space.

We define a mesh $\mathcal{M}$ As a set of triangles in 3D space. We also define a 3D bounding box using two corner points:

- The minimum corner $\mathbf{b}_{\min} = (x_{\min}, y_{\min}, z_{\min})$
- The maximum corner $\mathbf{b}_{\max} = (x_{\max}, y_{\max}, z_{\max})$

This box contains all points $\mathbf{p} = (x, y, z)$ Such as:

$$x_{\min} \le x \le x_{\max}, \quad y_{\min} \le y \le y_{\max}, \quad z_{\min} \le z \le z_{\max} \tag{4.7}$$

To find the part of the mesh inside the box, we cut the mesh using the six sides (planes) of the box. Each plane removes anything outside one side of the box.

We do this step-by-step:

- For each face of the box, we slice the mesh and keep only the part that lies inside.
- After all, six slices, only the part of the mesh inside the bounding box remains.

This gives us the clipped mesh:

$$\mathcal{M}_{\text{clipped}} = \mathcal{M} \cap \text{Box} \tag{4.8}$$

The bounding box $B_{clipped}$ for the clipped mesh, $\mathcal{M}_{\text{clipped}}$ It is calculated as an input for the path re-planning algorithm using the same logic as the bounds of selected triangles.

4.4.5. Applying 3D RRT-based path Replanning

In some cases, the TSP-solver can connect two camera poses that intersect with the surface of the mesh, which is not possible as a UAV cannot go through a solid surface. To counter this, a bi-directional Rapidly Exploring Random Tree (bi-RRT*) was implemented, which checks for each trajectory segment if a ray collides with any face of the mesh. The path replanning workflow consists of the following steps.

1. Detection of mesh-line intersection faces
2. Creating an expanded bounding box perpendicular to the direction of flight.
3. Selection of mesh faces within the expanded bounding box
4. Creating a bounding box from the intersected faces.

As defined earlier, rapidly exploring random trees (RRTs) is a path planning algorithm to efficiently explore high-dimensional spaces by incrementally building a space-filling tree rooted at a start configuration (S. M. LaValle, 2006a). It's beneficial for motion planning problems where the environment is complex or partially known. In our case, scene S can be considered an environment as a search space, SO.

These dense trees cover the search space from the start point P (init) and the Endpoint P(goal). The Axis-aligned bounding box $B_{clipped}$ is considered an obstacle in the search space.

A variety of RRT-based algorithms have been implemented and are adaptable to N-dimensions based on the original work (S. LaValle, 1998). For our use case, we use the bidirectional 3D RRT* implementation. The algorithm works as follows.

Let T_a and T_b be two trees starting at q_I and q_G, respectively. During each iteration, the algorithm performs the following operations:

1. For $i = 1$ to K (maximum iterations):

   Sample a random configuration $\alpha(i) \in SO$

   Find the nearest node in T_a:

   $$q_n \leftarrow \text{nearest}(T_a, \alpha(i))$$

   Steer towards the sample:

   $$q_s \leftarrow \text{steer}(q_n, \alpha(i))$$

   If $q_s \neq q_n$, then:

   $$T_a.\text{add}_{\text{vertex}(q_s)}, \quad T_a.\text{add}_{\text{edge}(q_n, q_s)}$$

   Try to connect the second tree T_b toward q_s:

   $$q_{n'} \leftarrow \text{nearest}(T_b, q_s)$$
   $$q_{s'} \leftarrow \text{steer}(q_{n'}, q_s)$$

   If $q_{s'} \neq q_{n'}$, add:

   $$T_b.\text{add}_{\text{vertex}(q_{s'})}, \quad T_b.\text{add}_{\text{edge}(q_{n'}, q_{s'})}$$

   If $q_{s'} = q_s$Trees are connected, return path.

2. Balance Step: After each iteration, if:

$|T_b| > |T_a|$ Then swap T_a and T_b. This ensures more exploration by the smaller tree, improving coverage in harder regions. Figure 4.11 represents a collision detection between a node and an obstacle.

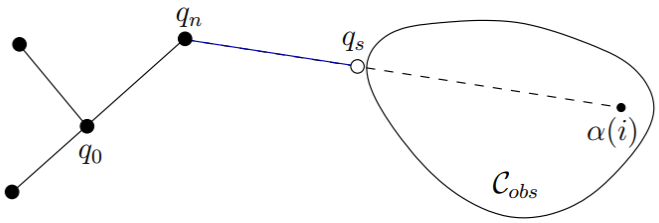


Figure 4.11 shows an intersection of an RRT node and a 2D obstacle (LaValle, 2006b)

The obstacle threshold is configurable based on the defined criterion by the use case (The nature of the scene we tend to explore).
Once a new trajectory between P_i i.e. the initial point and P_{i+1} i.e. the end point, is successfully replanned via the Bidirectional RRT* algorithm, the new points are added to each UAV's trajectory viewpoints.
We can call the result of the RRT Bi-directional path planning algorithm as

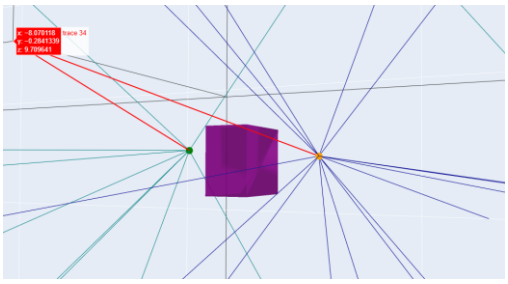
$$\mathcal{T}_{i,i+1} = \{Q_{i,1}, Q_{i,2}, \dots, Q_{i,k_i}\}, \quad \text{for each } i \in \{1, \dots, N-1\},$$

The new trajectory, hence, can be represented as

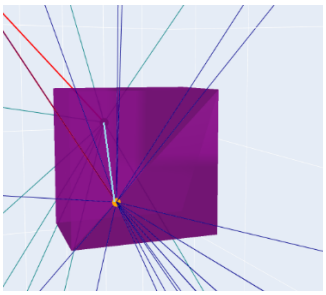
$$\mathcal{T} = \{P_1, Q_{1,1}, \dots, Q_{1,k_1}, \dots, P_N\} \tag{4.9}$$

As the new trajectory consists of updated waypoints, the additional points coexist with the same bounds defined by the input. The search space is defined as well.

The final bounding box is then used as an obstacle for the bi-RRT* planner, and the initial and goal points are the subsequent 3D points. Furthermore, the search space is defined as the scene bounds. The resulting replanned path lies within the search space and avoids the obstacle. The camera poses are ignored for this trajectory segment as it is not part of the NBV trajectory. Figure 4.12 shows a replanned path around an obstacle. The tree originates from both the init and goal point, the figure also illustrates the original path assigned to the UAV which passes through the obstacle.



a)



b)

Figure 4.12 a) Bi-RRT* obstacle aversion trajectory between goal (green) and yellow (goal) b) original path intersecting the obstacle.

4.4.6. UAV Energy Monitoring and Task Reassignment

The updated and replanned trajectory for each UAV is initially simulated based on a simple battery simulation which for every meter of movement, a battery depletion occurs in each UAV. When each trajectory is assigned to a UAV, it is estimated whether it can complete the trajectory or will fall short.
The battery module has a threshold of 20% remaining battery. When a UAV reaches the threshold, and another UAV has either completed its exploitation task or is near completion (based on covered percentage), the remaining points are assigned to the free UAV. This can be considered a simpler version of load balancing. Figure 4.13 shows how the consumption occurs during a given UAV's flight trajectory.

Algorithm 1 Battery Consumption Along a UAV Trajectory

Require: List of 3D points $\mathbf{P}$, battery max level $b_{\max}$, consumption rate r
Ensure: Lists D (cumulative distances), B (remaining battery)

$D \leftarrow [0]$ {Initialize distance list}
$B \leftarrow [b_{\max}]$ {Initialize battery list}
for $i = 1$ to $|\mathbf{P}| - 1$ **do**
  $\delta \leftarrow \|\mathbf{P}[i] - \mathbf{P}[i-1]\|$
  $d_{\text{tot}} \leftarrow D[i-1] + \delta$
  $b_{\text{rem}} \leftarrow \max(B[i-1] - \delta \cdot r,\ 0)$
  append d_{tot} to D
  append b_{rem} to B
end for
return D, B

Figure 4.13 shows the algorithm for battery consumption per UAV

Figure 4.14 shows the depletion of the battery with each waypoint once the new trajectory is assigned to each of the UAVs. The depletion of the battery is not uniform and varies with the distance covered between the way points, which directly impacts the consumption.

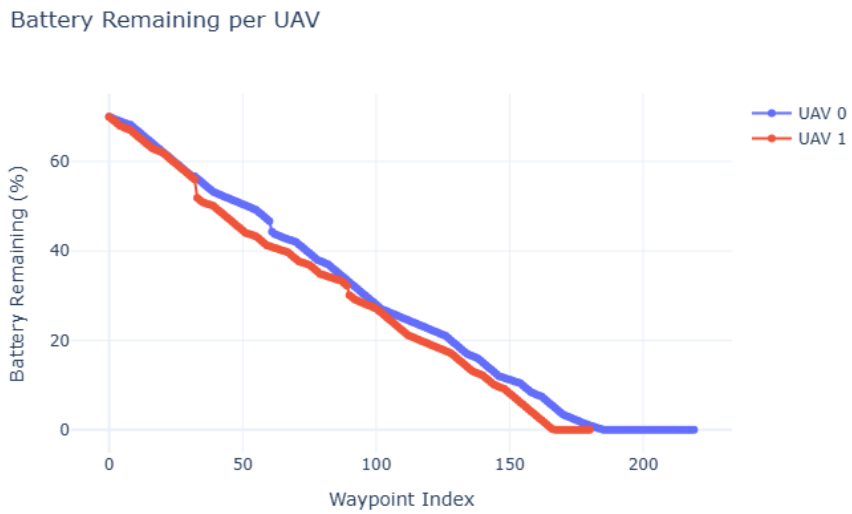


Figure 4.14 shows simultaneous battery depletion per waypoint

The reassigned points are added to the subsequent UAV based on a unique identifier to keep track of the original trajectory assigned and the reassigned section. To preserve the indices and sequence, the trajectories are written into a NumPy zip file (.npz).

4.5. Data Collection Simulation in Blender

This section explains the steps to configure Blender software for importing multi-UAV trajectories and set up the data collection pipeline.

4.5.1. Importing Updated Trajectories in Blender

The exported trajectory is loaded in Blender using a custom script that reads the camera poses (min_points) and (max_points). Each UAV is iterated to add the camera object using the same script explained in the coordinate conversion, but with some modifications. We create animation for each UAV and add the subsequent camera objects as key frames in Blender. This creates a continuous loop for each UAV to collect data simultaneously. We use an identifier column for each trajectory that manages the points reassignment. Each UAV trajectory has a cluster assigned to it. Ideally, it starts with one cluster per UAV, but if the UAV has been assigned additional camera views to exploit. This is equivalent to a simple version of task re-assignment and collaboration of tasks between two or more UAVs.

4.5.2. Multi-View Camera Configuration

Multi-view camera rendering is a feature in Blender that simulates simultaneous renders of a single object, a 3D scene, in our case, from multiple camera views. This resembles a real-time collaborative system that explores an object with two or more cameras, resembling an autonomous collaborative exploitation. The images rendered are saved separately but in parallel. It is important to configure the scenes when rendering data for each scene. Figure 4.15 shows a multi-UAV data collection system simulated in Blender. The stereoscopy function supports multiple views, under output properties > stereoscopy > multi-views, where two or more views can be configured. The render quality can be set to 1920 by 1080 (HD) or 3840 by 2160 (4K). It is designed to render the images with a naming convention as UAV_{id}_{pose_id}.jpg. The renders apply frame by frame simultaneously for all the UAVs.

Figure 4.15 Optimized trajectories for two UAVs in Blender

4.5.3. Velodyne Simulation

The LiDAR simulation was done using Blensor, it was used on the original trajectory that was acquired after the MACARONS-NBV method, the same camera trajectory as the NBV is used to compare the surface coverage with a VLP-16 LiDAR for a direct comparison of surface coverage. An overview of this method is provided in Figure 4.16.

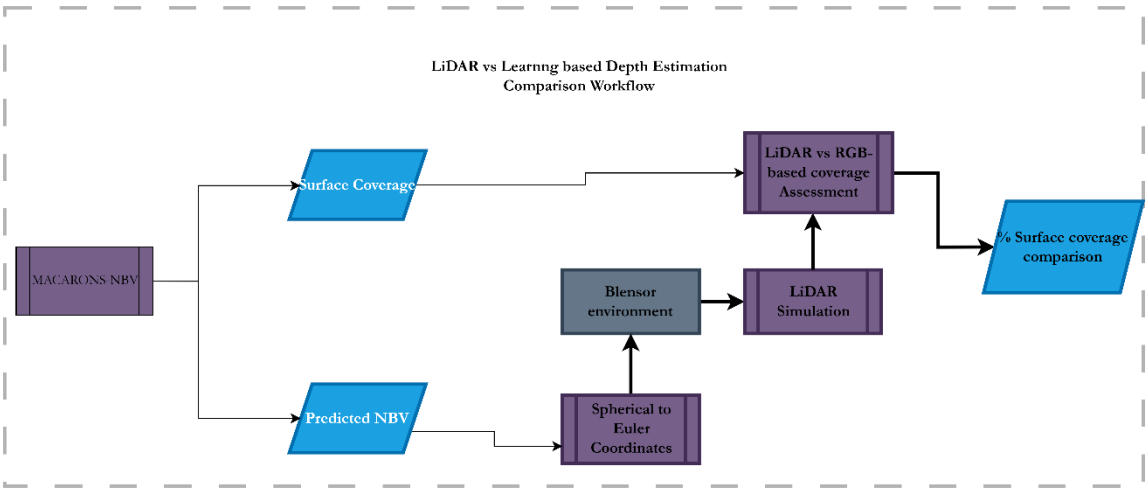


Figure 4.16 Overview of LiDAR simulation workflow

4.6. 3D Reconstruction Pipeline

This section gives an overview for the methodology to Reconstruct an object using the photogrammetric software Agisoft Metashape professional.

Once the rendering is completed, the camera intrinsics like focal length, principal points, and distortion parameters (we assume no distortion) are constant and the same for all the UAVs. The extrinsic coordinates are also exported for each image that is rendered. The location information is embedded for the dataset using location extrinsic and a script to read and write it for all the renders generated. The Blender (Euler) coordinates are converted to Yaw, roll, and pitch, which Agisoft Metashape uses. We set the coordinate system to our local coordinate system as the dataset is not geographically referenced.

Metashape Professional (Evaluation version) is used to process the complete captured renders from the UAV trajectories. We follow the standard 3D Reconstruction workflow in Metashape. The images are first aligned to estimate the poses, and the reference poses from the camera extrinsic are added to the workflow, too. The camera is calibrated with the same intrinsic setup as Blender. (Focal length, principal point, and no distortion). Once a sparse point cloud is generated with the correct orientations of the images, we reconstruct a high-quality dense point cloud from the image-based multi-view stereo pipeline of Agisoft Metashape.

The point cloud saved as a (.ply) file with colors and dense points is further processed using Cloud Compare (CC) to align and evaluate the quality of 3D Reconstruction using the ground truth mesh as a reference. Figure 4.17 shows the overall Reconstruction process.

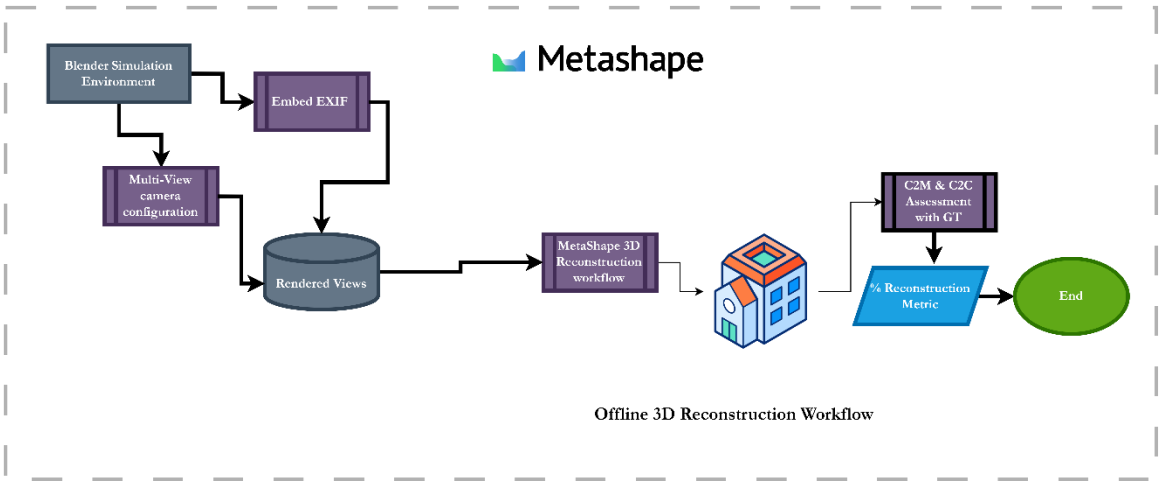


Figure 4.17 Workflow for 3D reconstruction using Agisoft Metashape

4.7. Evaluation of 3D Reconstruction Accuracy Assessment

The 3D reconstruction is evaluated using Cloud-to-Cloud (C2C) and Cloud-to-Mesh (C2M) between the ground truth and the 3D reconstruction.

4.7.1. Align the reconstructed point cloud and the ground truth mesh

The reference axis for Metashape export and Cloud Compare CC requires a rotation along the Z axis; the cloud is roughly aligned to the mesh using Translate/Rotate in CC. The mesh and point cloud are aligned using cloud registration for exact alignment. The alignment is a crucial step in assessing the quality of the 3D reconstruction.

4.7.2. Compute cloud-to-mesh distances

Cloud-to-mesh distances between a point cloud and a mesh are used to quantify the presence of points on the surface of the mesh. When the point cloud and mesh are co-registered, ideally, the 3D points should be on the surface of the Mesh. This tool calculates the distribution of points based on the C2M distance from the reference (Ground truth) surface. The figure 4-18 depicts variations in the point cloud to surface mesh distance, assessing the alignment of the cloud with reference to the ground truth.

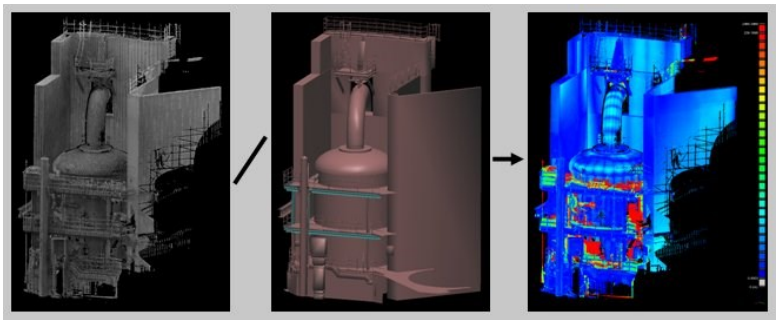


Figure 4.18 shows a cloud-to-mesh distance example (*Cloud-to-Mesh Distance - CloudCompareWiki*, n.d.)

The result of the C2M distance histogram is exported. Using a simple computation of inliers based on the bins, the mean error, RMS error, and inlier percentage are computed using the following formulas. The threshold distance can be set from 0 to 5 cm, depending on the size and complexity of the exploration area.

Inlier Percentage:

$$\text{Inlier Percent} = \frac{\sum \text{Value in range}[-0.02, 0.02]}{\sum \text{Total Value}} \times 100 \tag{4.10}$$

Mean Error:

$$\text{Mean Error} = \frac{\sum(\text{Value} \times |\text{Bin Center}|)}{\sum \text{Value}} \tag{4.11}$$

RMS Error (Root Mean Square Error):

$$\text{RMS Error} = \sqrt{\frac{\sum(\text{Value} \times \text{Bin Center}^2)}{\sum \text{Value}}} \tag{4.12}$$

4.7.3. C2C Distance for Reconstruction Accuracy Assessment

Cloud-to-cloud distances are used as a metric to assess the completion of a 3D Reconstruction. C2C uses the nearest neighbour approach to check corresponding points to a reference cloud. We use a threshold of 0.2 to 0.5m; all points within 0.2m of the ground truth sampled point cloud are considered "completed," and the rest are considered "non-reconstructed" points. This gives a metric to assess the completeness of a 3D Reconstruction process. The bins are used similar to C2M metric for estimating the percentage non-reconstruction.

5. RESULTS AND DISCUSSIONS

This chapter provides an overview of the results of the autonomous UAV exploration and 3D reconstruction with a focus on different infrastructure objects; this starts from the exploration of the 3D Scene using the NBV to the 3D reconstruction and evaluation of the quality of the reconstruction. The second section of the chapter defines the reconstruction method. We evaluate our methodology using three different forms of infrastructure objects. We use bridges, buildings, and high transmission powerline objects. The purpose of choosing a diverse set of infrastructure objects is to evaluate the performance of autonomous exploration and the completeness of the 3D reconstruction in different environments. Figure 5.1 shows an overview of the three objects. The exploration is centered on the main infrastructures.

Figure 5.1 shows the selected objects used for the evaluation of the methodology

5.1. LiDAR and RGB-Based Coverage Comparison

MACARONS-NBV estimates the surface coverage using the generated depth maps from the depth prediction module. To assess the completeness of the 3D reconstruction obtained via self-supervised monocular depth estimation, we evaluate the surface coverage of the generated point clouds by comparing it to the ground truth mesh. The analysis is conducted in two parts: one using depth maps derived from an NBV-planned RGB image sequence, and the other using a simulated LiDAR scan along the same NBV trajectory. We use a 3D statue of Liberty Mesh from the SketchFab dataset to achieve this task.

The cloud-to-cloud (C2C) distance explained in the previous chapter allows a direct comparison between the surface coverage of the self-supervised depth-based point cloud and the LiDAR point cloud. The coverage percentage provides an estimate of 3D reconstruction completeness under the same trajectory, highlighting the limitations and advantages of each sensing method. Figure 5.2 shows a comparison of ground truth, RGB-based coverage and LiDAR coverage

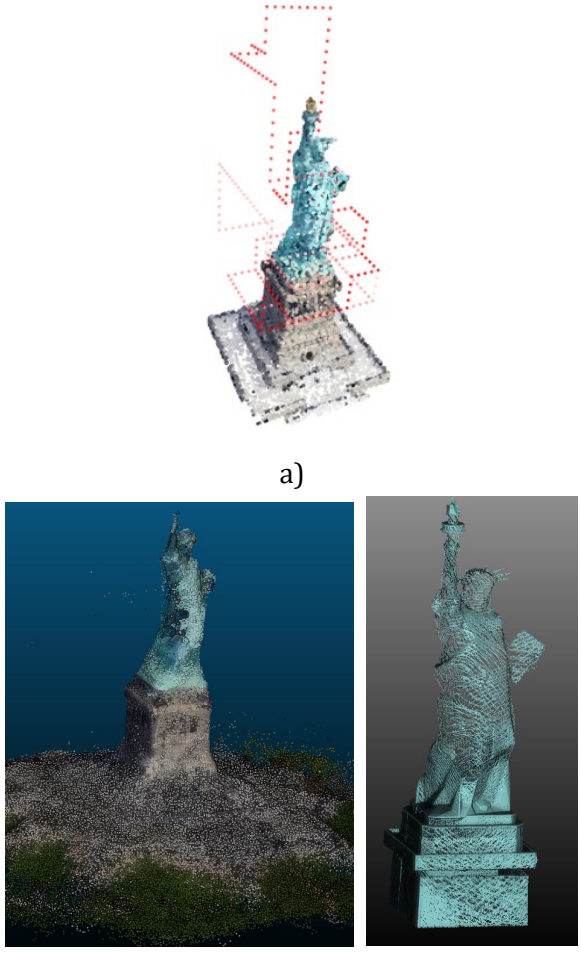


Figure 5.2 (a) Autonomous path around the object (b) Image-based point cloud (c) VLP-16 LiDAR point cloud

Figure 5.3 shows the progression of surface coverage with each NBV iteration. MACARONS-NBV achieves 81.1 percent surface coverage at the end of the trajectory, visualized in Figure 5.2. The Velodyne VLP-16 LiDAR, on the other hand, achieves a surface coverage score of 86.77 percent. We take the threshold of cloud to cloud with direct distances below 0.2 m and the distribution of the points are visualized in figure 5.4.

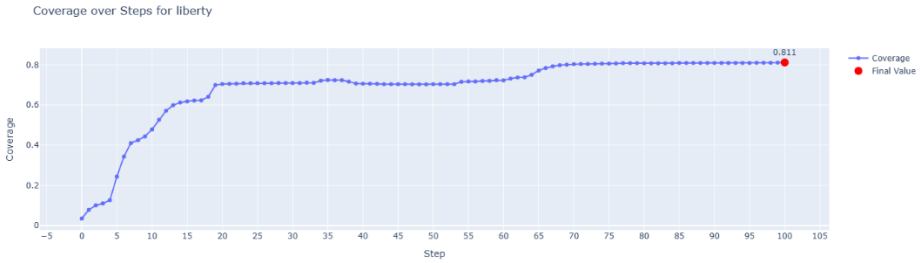


Figure 5.3 Surface coverage of the Liberty object

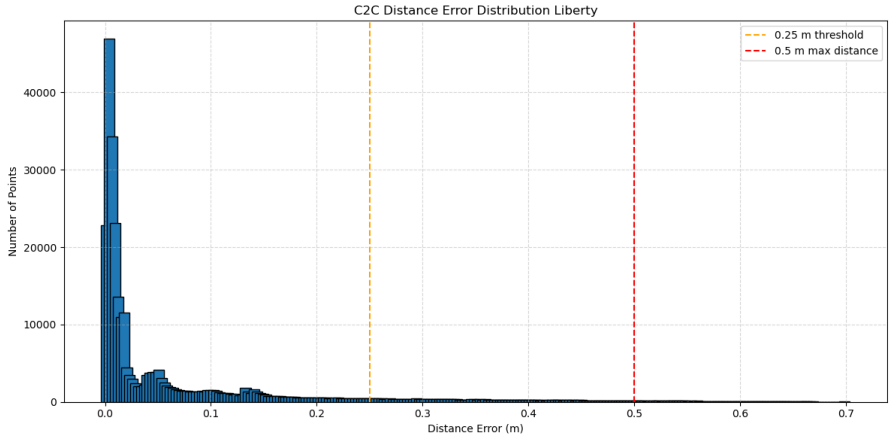
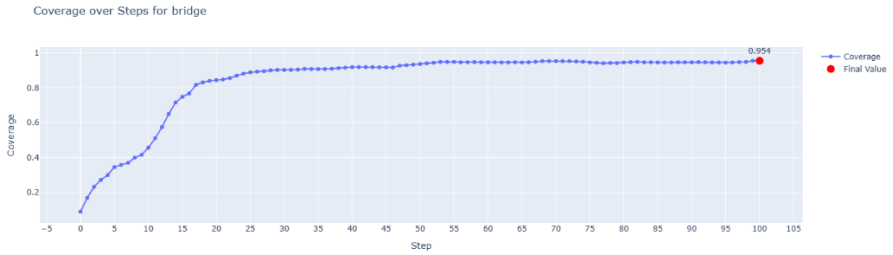


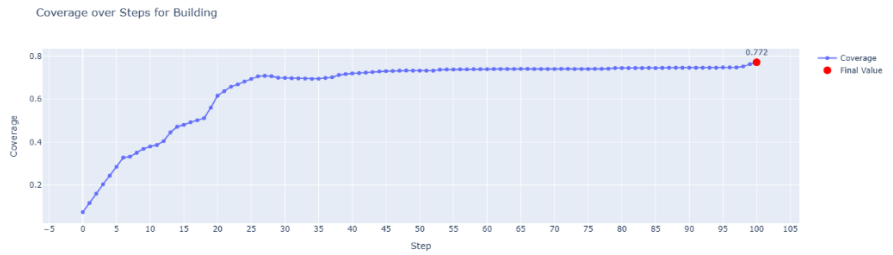
Figure 5.4 C2C threshold for surface coverage estimation for VLP-16

5.2. Generation of NBVs

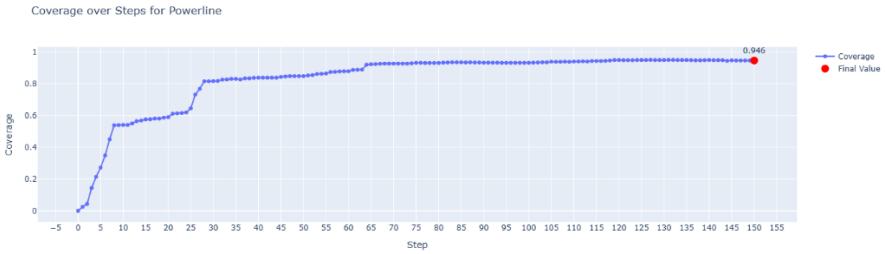
MACARONS-NBV pipeline generates camera positions which provides maximum surface coverage increase. It uses the 3D object and bounds of the infrastructure object that we intend to explore. The autonomous exploration process starts with this and using the self-supervised method explores the scene to provide NBVs by predicting the information-gain for each camera pose. The outcome for this step is a set of 3D trajectory with Spherical (azimuth, elevation) coordinates and the surface coverage at each step. These camera poses are then converted to blender compatible camera objects. The 3D reconstructed point cloud for the building object is given in Figure 5.6.



a) Surface Coverage per Iteration for Bridge Infrastructure



b) Surface Coverage per Iteration for Building Infrastructure



c) Surface Coverage per Iteration for Powerline Infrastructure

Figure 5.5 Surface Coverage Gains for Infrastructure Objects a) Bridge, b) Building and c) Powerline Infrastructure

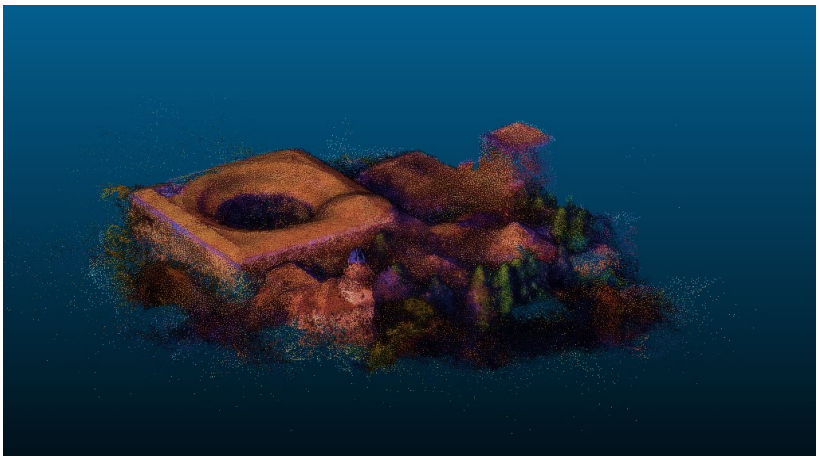


Figure 5.6 Point cloud reconstructed for the building case study

5.3. Collaborative UAV Strategy and Task Management Module

The converted camera objects are handled by the Task Management and collaborative path planning module, which runs identically for all the infrastructure objects. It takes input of the camera objects from the coordinate conversion method and first performs hierarchical clustering based on the 3D coordinates (x, y, z) and defines user-specific clusters elaborated in the methodology. For our methodology we define 2 clusters following our underlying assumptions in section 3.1, which equate to the number of UAVs we use to demonstrate the collaborative multi-UAV workflow. Figure 5.7 shows the number of assigned points to each UAV and Figure 5.8 shows the initial cluster of the trajectories.

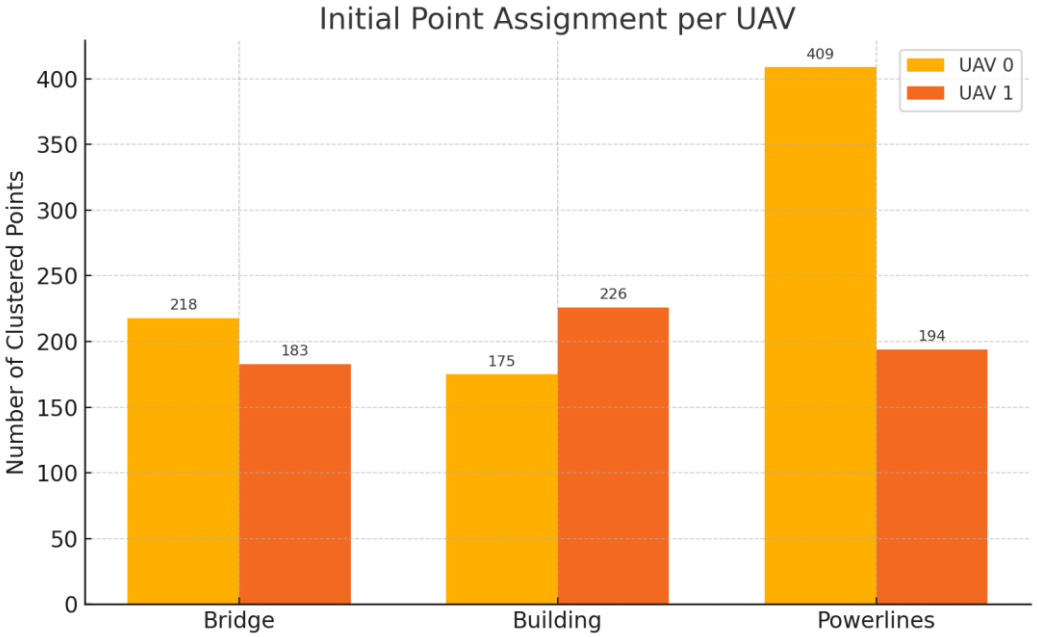


Figure 5.7 Bar plot representing initial points per UAV to capture images

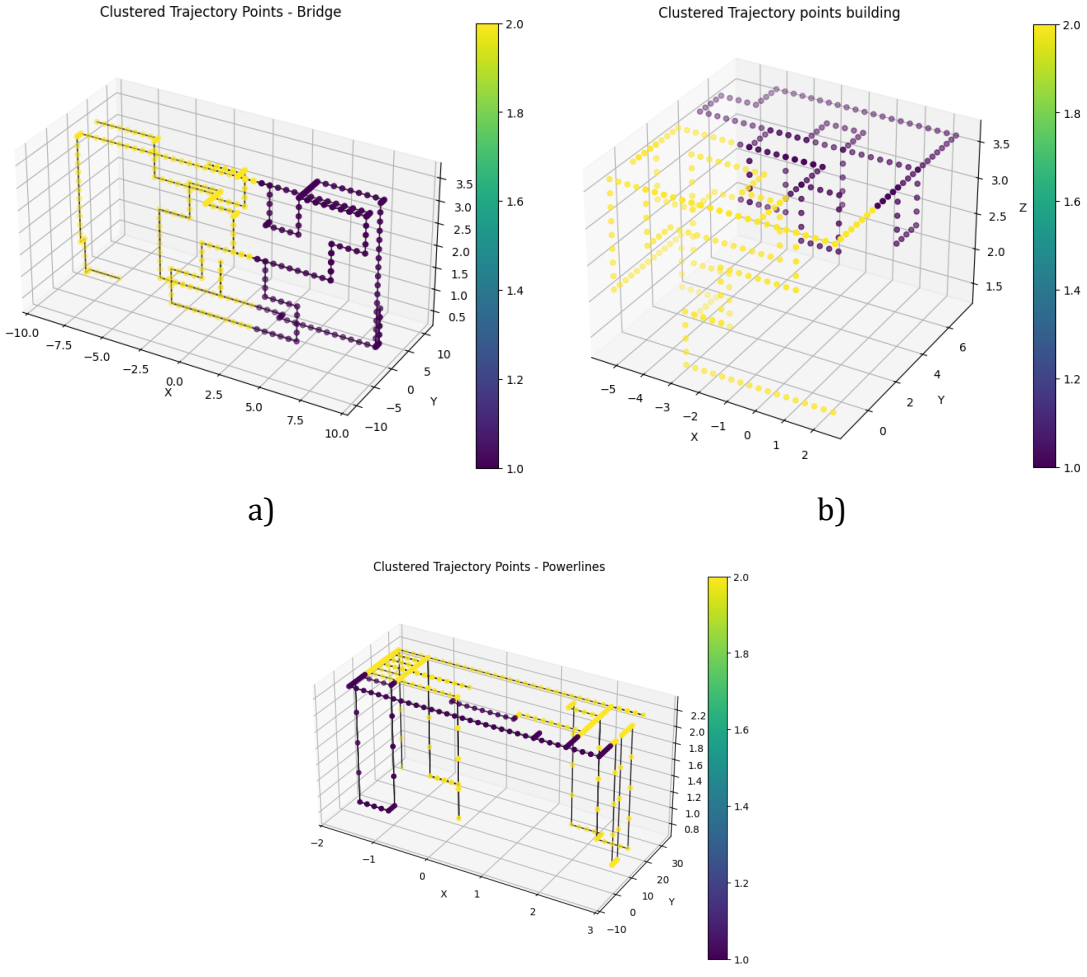


Figure 5.8 Initial clusters of trajectories for the infrastructure objects a) bridge b) building c) powerlines

The clusters are then optimized based on distance, and the modified distance + elevation heuristic is enabled. Both are used for experiments with different objects. To show variations in task assignment for multiple UAVs, different configurations for the battery consumption model are used to simulate how the data collection tasks are reassigned to the UAVs dynamically based on remaining battery. Figure 5.9 provides an overview of the optimized distances of the trajectories after running the TSP algorithm.

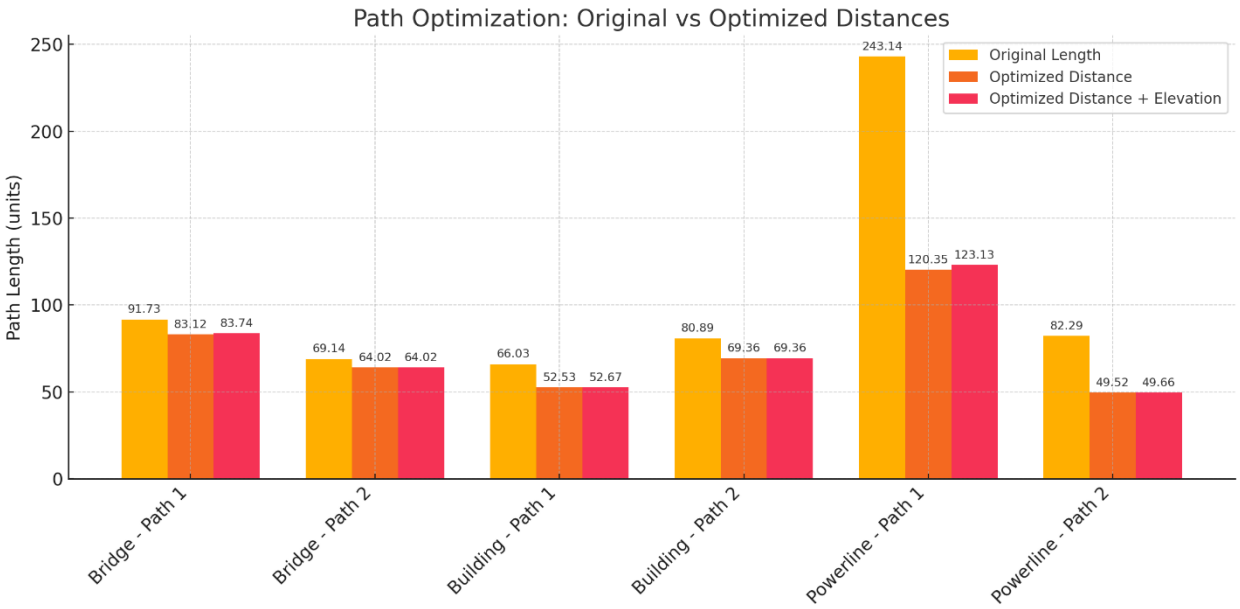


Figure 5.9 Plot represents original and optimized path distances per UAV

For the simulation of the battery consumption for the three infrastructures, they were evaluated using different heuristics and battery consumption models. Both the Bridge and Powerlines used a Euclidean distance-only heuristic, while the building object incorporated elevation into the planning. All objects had a maximum battery level of 100% and a 20% threshold, but the Powerlines had a slightly lower depletion rate of 0.85% per meter compared to 1.00% for the others. The UAV speed was fixed at 1.0 m/s in all cases.

A. Bridge Infrastructure

In both heuristics, UAV 0 reaches its battery threshold before completing its assigned path. Under distance-only heuristic, UAV 0 stops at index 210 with a final battery level of 19.88%, leaving 9 points to be reassigned to UAV 1. When incorporating elevation penalties, UAV 0 stops slightly earlier at index 208 with 19.86%, requiring 12 points to be reassigned. UAV 1 successfully completes its trajectory in both cases without battery constraints. Figure 5.10 provides a comparison of these heuristics. Figure 5.11 shows the collaborative planned trajectories for the bridge infrastructure.

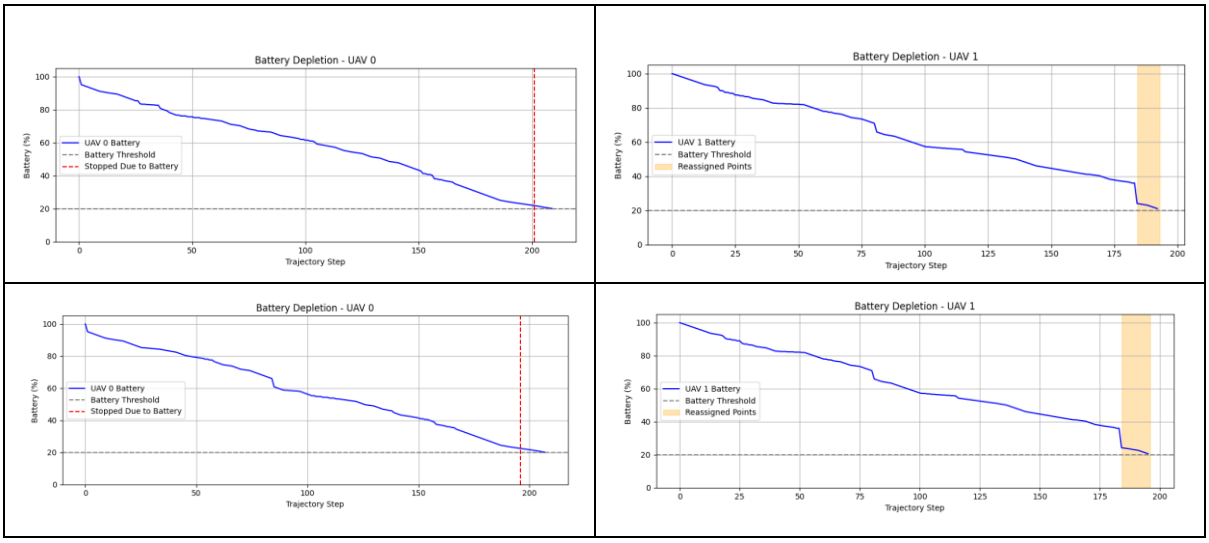


Figure 5.10 Battery consumption and task reassignment for bridge infrastructure on distance (top) and distance + elevation (bottom) heuristics.

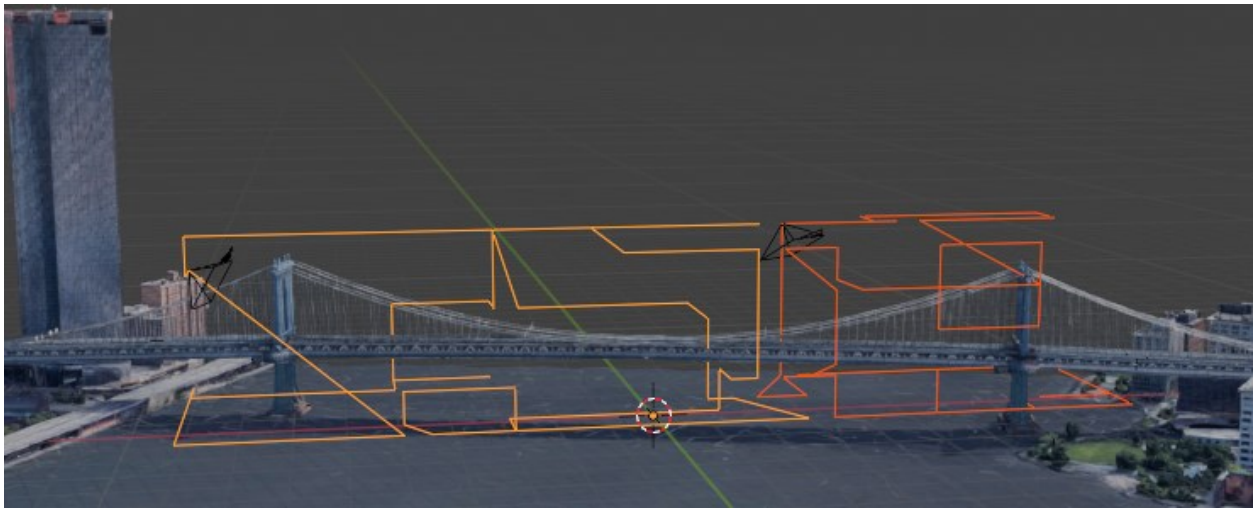


Figure 5.11 Trajectory of collaborative autonomous UAVs for the bridge 3d reconstruction

B. Building Infrastructure

Using the distance + elevation heuristic, both UAV 0 and UAV 1 completed their full assigned paths with sufficient battery remaining, requiring no point reassignment. Figure 5-12 shows the propagation of battery depletion for both UAVs based on the depletion rates in Table 5.1. Figure 5.13 provides an overview of the trajectories for the building infrastructure.

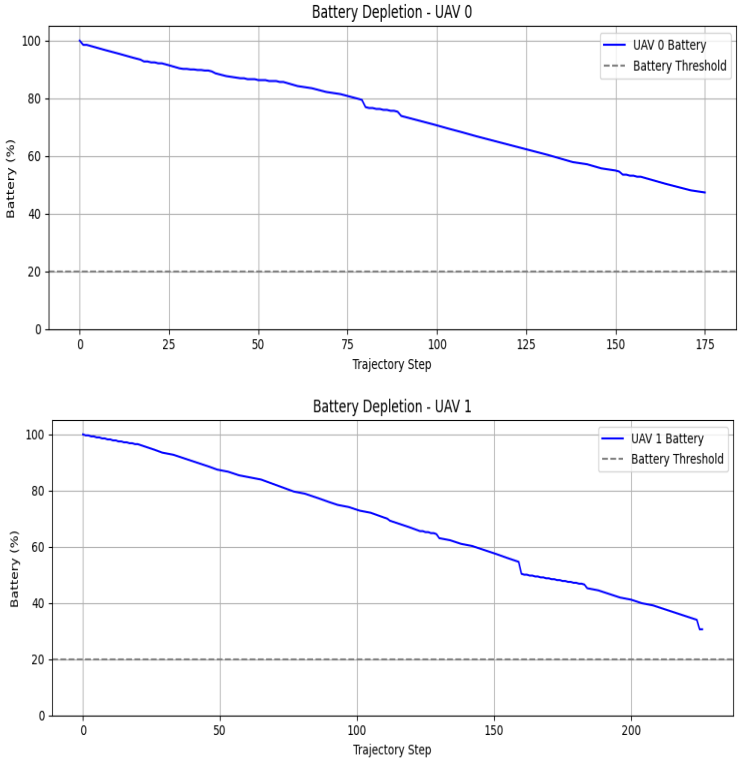


Figure 5.12 Battery consumption and task reassignment of UAVs for building scene

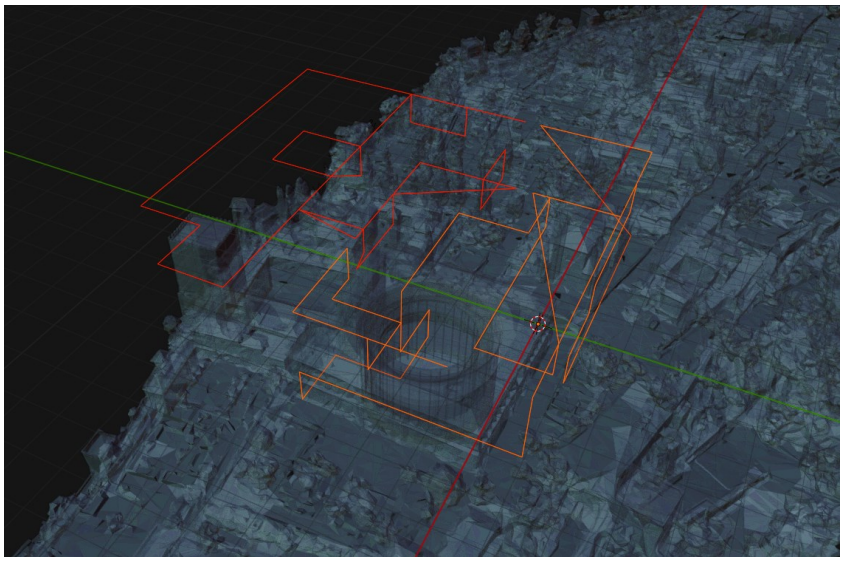


Figure 5.13 Trajectory of collaborative autonomous UAVs for building 3D reconstruction

C. Powerline infrastructure

A total of 603 trajectory points were shared between the UAVs initially with 401 and 194 tasked to UAV 0 and 1 respectively. Figure 5-14 shows the trajectory reassignment taking place for UAV 1.

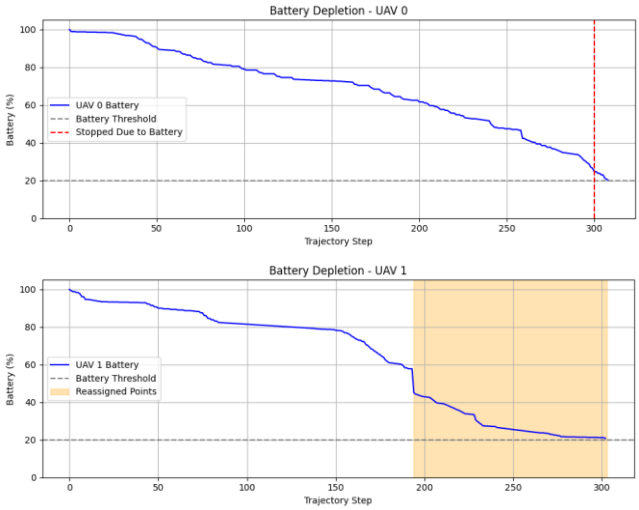


Figure 5.14 Battery consumption and task reassignment of UAVs for powerline infrastructure

In this scenario, UAV 0 stops at index 300 with 19.65% battery remaining, unable to complete its path. UAV 1 completed its assigned trajectory with sufficient battery and was tasked with covering the 109 unvisited points left by UAV 0. Figure 5.15 provides an overview of the newly assigned trajectory for both UAVs.

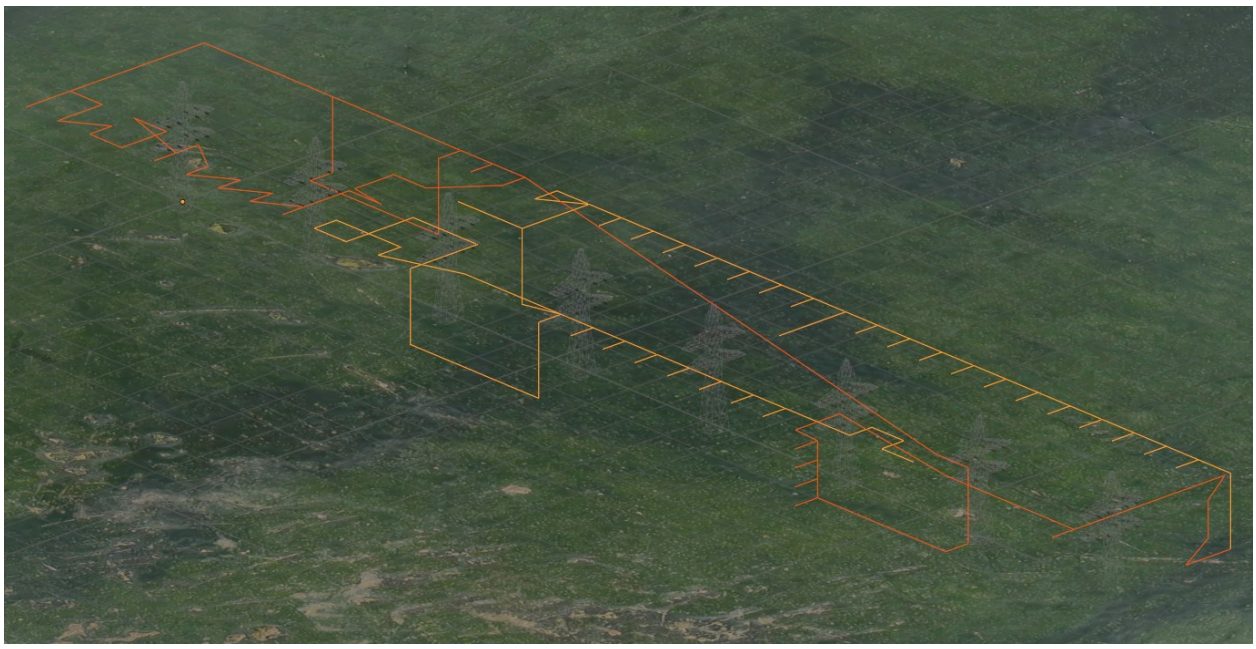


Figure 5.15 Trajectory of collaborative autonomous UAVs for powerline 3D inspection

5.4. Surface Coverage and Reconstruction Quality Evaluation

To evaluate the geometric completeness of the reconstructed point clouds, each reconstruction is compared against a uniformly sampled version of the ground truth (GT) mesh. The GT mesh is sampled

to produce a point distribution that matches the density of the reconstructed output, ensuring a fair and consistent basis for comparison.
A visual comparison is provided using side-by-side screenshots of the ground truth and the reconstructed geometry. Additionally, a quantitative assessment is performed using the Cloud-to-Cloud (C2C) distance metric, which measures the Euclidean distance between each reconstructed point and its closest counterpart on the sampled GT surface. "The confidence factor for each reconstructed object assesses the number of depth maps that contribute to the estimation of the 3D points. We also consider the confidence metric of the 3D reconstruction process. It is defined as: for the dense cloud points, the confidence value represents the number of depth maps involved in the point generation process. This value is an integer. We filter out the points that have 0-2 confidence values to refine the dense point cloud.

To identify low-confidence or missing regions, the C2C distance results are binned into a histogram. The range of 0.25 to 0.5 meters is used to define partially or non-reconstructed areas. The proportion of points falling within this range provides an estimate of the percentage of the surface that was not adequately covered during the reconstruction process.

A. Bridge infrastructure

The ground truth mesh and 3D Reconstruction using 4K renderings visualized side by side in Figure 5.16 gives an indication that regions where the NBV is focused on are much denser in the reconstruction as compared to the background regions. To assess the completion of the bridge infrastructure, Confidence Maps provide an insight into the reconstruction quality and regions of high and low confidence. Figure 5.17 shows confidence plots

Figure 5.16 Visual comparison of ground truth (left) and reconstructed point cloud (right) for the bridge

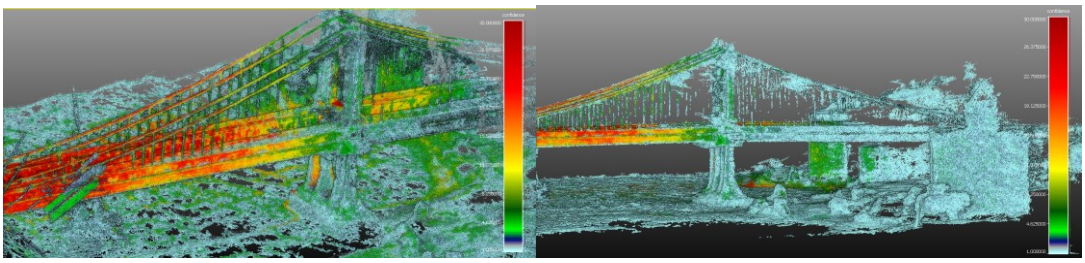


Figure 5.17 Confidence maps for the bridge object

We assess that around 15.94% of the bridge object is not reconstructed, or in other words, the 3D reconstruction workflow reaches a value of 84%. This is assessed using cloud to cloud (C2C) distance

between the ground truth Mesh's uniform sampled point cloud matching density of the bridge Reconstruction object. The non-reconstructed regions are highlighted in Figure 5.18 as red.

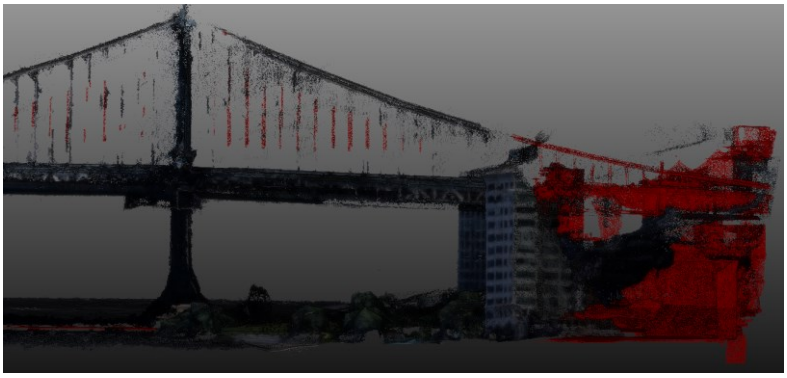


Figure 5.18 Non-Reconstructed Regions of bridge coloured in red

Figure 5.19 shows the distribution of points based on the distance threshold.

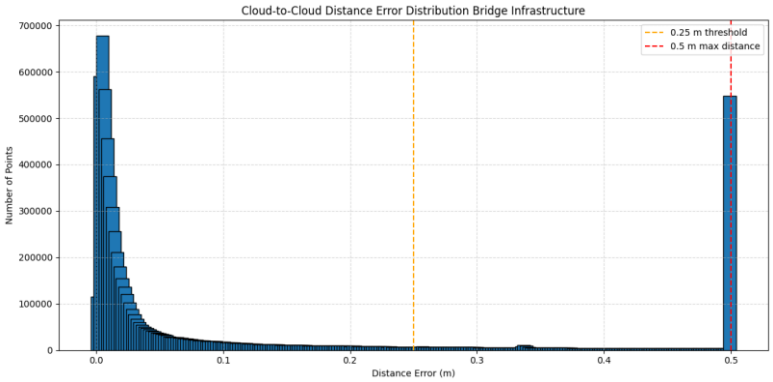


Figure 5.19 Histogram Distribution of C2C threshold points for bridge

The bridge showed moderate surface completeness, with about 16% of the GT surface outside the 25 cm threshold. Confidence maps correlate these gaps with smaller and non-well-defined components like the suspensions and areas which are outside the trajectory but were captured when collecting the data.

B. Building Infrastructure

Like for the bridge infrastructure, a visual inspection of the reconstruction of the building assisted with the confidence map giving an indication of some missing facades as viewed in Figures 5.20 and 5.21.

Figure 5.20 Ground truth (left) and the reconstructed point cloud (right) for the building

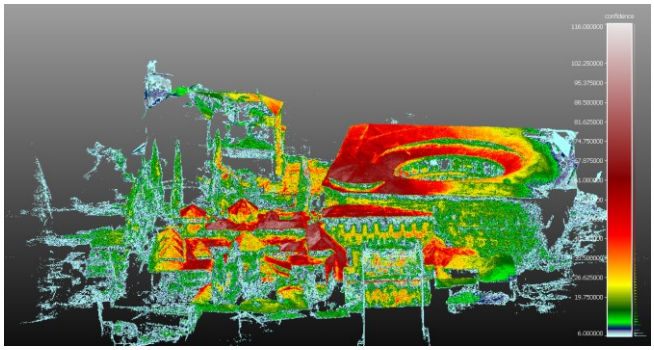


Figure 5.21 Confidence Map for building Reconstruction

The assessment highlights that 19.69% of building infrastructure is not covered by reconstruction, this is supported by the C2C distance comparison with the building ground truth mesh and the 3D reconstruction to assess that visual analysis quantitatively.

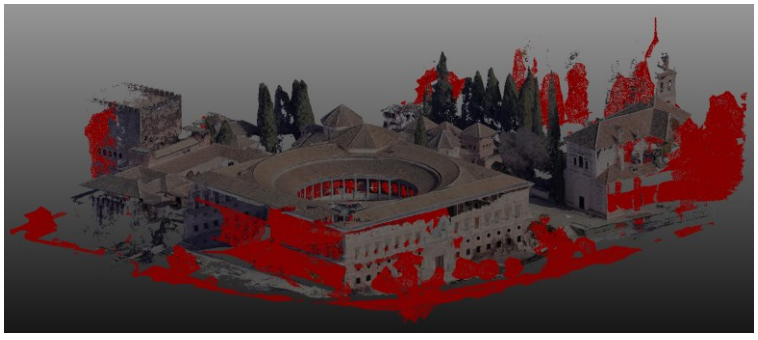


Figure 5.22 non-reconstructed regions based on C2C metric for building

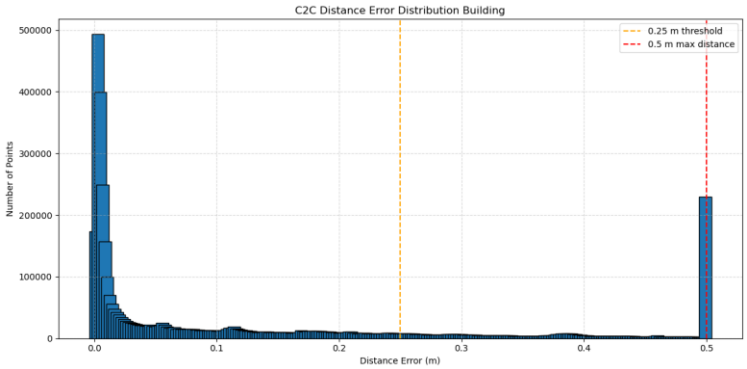


Figure 5.23 Histogram Distribution of C2C threshold points for building

Most of the missing regions, according to the confidence plots in Figure 5.21 and the C2C distance maps shown in Figure 5.22 also highlights regions that were not initially covered with the NBV exploration and due to initial lower coverage, these regions are subsequently not well covered when rendering data for the NBV trajectory. Furthermore, the C2C histogram in Figure 5.23 highlights the distribution of reconstructed 3D points according to the C2C distance threshold.

C. Powerlines Infrastructure

Only the 4K (3840×2160) images were used in the 3D reconstruction process due to the repetitiveness and of the powerline infrastructure. The powerlines are thin and often get confused with the background as compared to the main towers. The confidence of reconstruction maps the high confidence for the regions near the ground and the changes are inversely proportional to the elevation. Ground truth and Reconstruction comparison is provided in Figure 5.24

(a) Ground truth (b) 3D Reconstruction

Figure 5.24 Comparison of ground truth and the reconstructed point cloud for the powerline

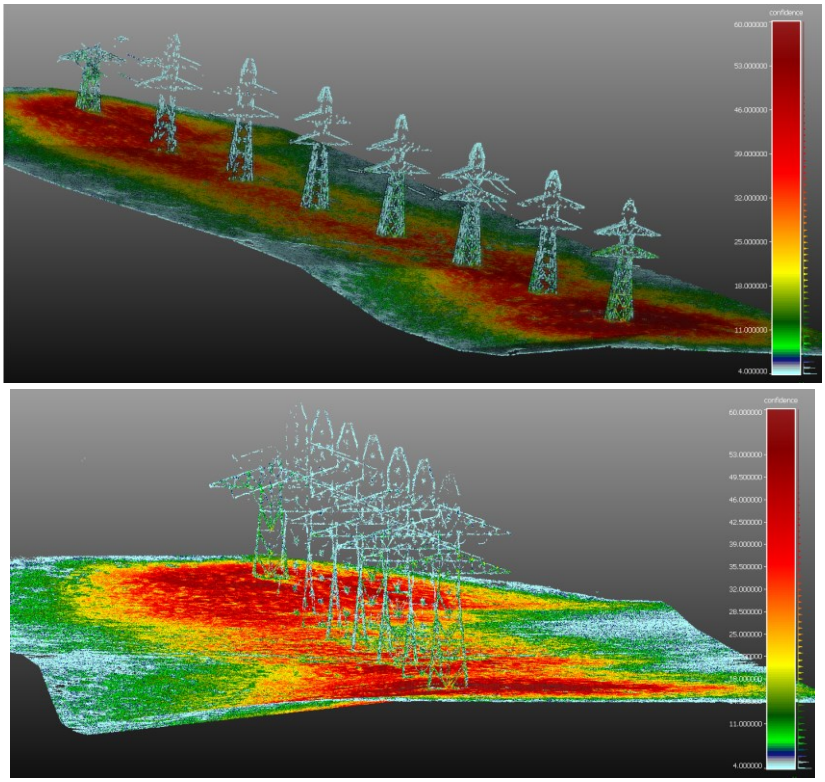


Figure 5.25 Reconstruction confidence maps for the powerline infrastructure

A value of 11.20% is for non-reconstruction observed with the C2C evaluation metric. The missing components are mainly the voltage lines and some part of a tower. The missing parts of the tower are due to not being not being visible in multiple images, as the first and the last tower are similar in appearance and there is very low variation within the powerlines. This hinders the creation of depth maps by the software. Similarly, the high voltage lines offer a very low number of points in contrast to the other objects and ground in the overall powerline infrastructure scene. The high voltage lines account for only a small percentage of the number of points in the point cloud. This is prominent in Figure 5.26, where the red colour represents the parts of the ground truth that were not reconstructed.

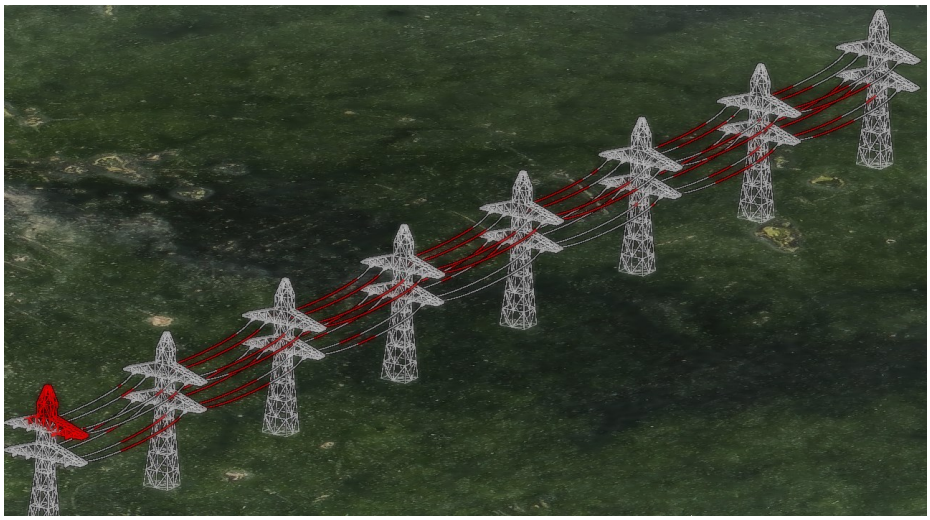


Figure 5.26 Non-reconstructed regions based on C2C metric for powerline infrastructure (red points)

The C2C histogram in Figure 5.27 also gives the frequency of the non-reconstructed points that confirm the low number of points representing the high voltage powerlines.

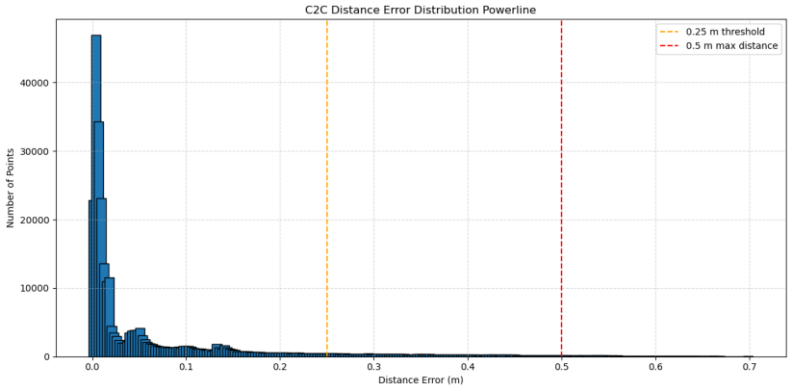


Figure 5.27 Histogram Distribution of C2C threshold points for powerline infrastructure

The reconstruction of the powerline scene demonstrates high completeness and confidence, with 11.20% non-reconstructed regions primarily associated with the thin high-voltage lines. When testing the different rendered resolutions and the quality of 3D reconstruction, the thin powerlines significantly improved with 4K images. The core structural elements were captured accurately, validating the effectiveness of using high-resolution images for such a fine-grained infrastructure.

5.5. Cloud Alignment Metrics

The same 3D Reconstruction methodology is applied to each of the objects using Metashape. Fixed parameters of the camera were defined with the assumption of no distortion, alongside the focal length and camera height and width. The photos were then aligned, and a dense point cloud was generated. Due to camera alignment and bundle adjustments, the final dense point clouds were translated and rotated to match Cloud Compare's coordinate alignment. Using the C2M metrics, the alignment was assessed according to the method described in the chapter methodology. Table 5.1 shows a summary of the metrics and the accuracy of alignment for each object.

Scene	Mean Error (m)	RMS Error (m)	Points within ±2 cm (%)
Bridge	0.013	0.024	90.96%
Building	0.004	0.005	99.83%
Powerlines	0.001	0.001	100.00%

Table 5.1 Summarizes the C2M alignment metrics for the infrastructure scenes

The bridge exhibited the largest alignment deviation, with a mean error of 1.29 cm and RMS error of 2.36 cm, likely due to occlusions or structural complexity resulting in noise. The building scene achieved sub-centimeter alignment accuracy, with 99.83% of points falling within the 2 cm tolerance, reflecting reliable reconstruction in structured environments. Powerlines showed the highest alignment precision with all points within the threshold and a negligible mean error of 0.8 mm.

5.6. Rendering of frames in 4K and HD Resolution

The camera frames were rendered from the camera intrinsics matching the DJI Mini 4 Pro to render the camera views in both HD (1920x1080) and 4K (3840x2160) resolutions. Furthermore, the photos were embedded with EXIF data to replicate a UAV capture, using the location of the cameras at each keyframe. Table 5.2 shows a comparison of times taken to render the images along the trajectories simultaneously for HD and 4K images for all the infrastructure.

Scene	Frames Rendered	Render Time (s) HD	Render Time (s) 4K
Bridge	401	358.51	966.88
Building	401	211.64	602.04
Powerlines	602	445.73	1094.99

Table 5.2 Rendering times image data

5.7. Summary of Interpretation of Results

The comparisons of reconstruction completeness with reference to the ground truth gives us a good indication of the discrepancy between the NBV-based surface coverage estimation and our dense 3D reconstruction of the object using the same trajectory to capture and reconstruct the infrastructure objects. We consider a direct comparison of the 3D Reconstructed extent to the ground truth. The Bridge object had a final coverage with MACARONS-NBV of 94%, while the reconstruction has an overall 3D Reconstruction of 85% using cloud-to-cloud (C2C) assessment metrics. This is an indication that there are several factors that influence the complete 3D reconstruction. We use a 4K captured data to generate the dense point cloud. Similarly, the building achieves a final surface 3D reconstruction of around 80% of the entire building search area, which is close to the 77% surface coverage predicted by the MACARONS-NBV. The powerline object achieved the highest surface coverage of 97% of the ground truth infrastructure object. The 3D reconstruction assessment reached 88.8% which indicates the efficiency of the NBV method to effectively cover the surface of an infrastructure.

For the bridge object, we observe that the missing parts that are near to the edge of the bridge as shown in Figure 5.18, which is supported by the confidence point cloud representing regions with low visibility as compared to the center of the bridge. This gives a very good indication that the edge-points were not well covered due to the trajectory mainly focused on the main parts of the bridge as shown in Figure 5.11. Smaller parts of the bridge, like the suspensions are also not hard to reconstruct due to low visibility, the quality of the 3D mesh and quality of the rendered camera images.

For the building infrastructure, the 3D points estimated with low confidence are around the edge of the scene as visible in Figure 5.21. These points are reconstructed mainly due to their visibility and the complexity of the environment for the building. It is also however important to consider the 77% surface coverage of the MACARONS-NBV method. The NBVs only offer points that offer a 77% coverage of the overall surface visible within the bounding box. This can be due to similar looking facades of the building and the adequate detail not being encapsulated by the 4K images. As a result, the edges of the 3D scene had lower 3D reconstruction confidence and non-reconstruction in general, as shown by Figures

5.21 and 5.22 respectively. This can be further improved with increasing the NBV iterations, but for each increase iteration, but consequently, the number of NBVs are increase by 4 with each iteration.

The 3D reconstruction of the powerline infrastructure relied solely on 4K (3840×2160) images to address the challenges posed by the repetitive and thin nature of voltage lines, which often blend with the background. A non-reconstructed region of 11.20% was observed using the C2C evaluation metric, primarily affecting the high-voltage lines and parts of one tower that lacked visibility across multiple views. The reconstruction confidence was higher near ground-level regions and decreased with elevation. Despite the thin lines contributing only a small number of points to the point cloud, the use of 4K imagery improved their visibility and reconstruction quality. The results confirm that high-resolution images are critical for accurate modeling of fine infrastructure elements. Additionally, we ran the powerline object for a higher NBV iteration (150), which resulted in a larger collection of images, improving the overall 3D reconstruction of the powerline.

6. CONCLUSIONS AND RECOMMENDATIONS

This chapter provides the conclusions and the impact of the overall research, followed by advantages, limitations, and recommendations. The answer to the research questions is also part of this chapter, followed by the ethical considerations of the MSc research.

6.1. Conclusions

This research presents a novel and cost-effective approach for autonomous infrastructure inspection and 3D reconstruction using a collaborative system of UAVs equipped solely with monocular RGB cameras. The proposed framework addresses key limitations in current UAV-based inspection workflows by integrating learning-based depth estimation, optimized multi-agent path planning, and centralized coordination for task allocation. The developed pipeline enables efficient data collection, adaptive trajectory planning, and high-quality reconstruction with minimal human input. The results demonstrate the scalability and practicality of the approach, highlighting its potential for applications such as asset monitoring, site preservation, and the creation of accurate digital twins in complex environments. The proposed method for autonomous exploration has been evaluated across three very different forms of infrastructure: bridge, building, and high-voltage powerline. The learning-based NBV method, MACARONS, uses a sequential monocular RGB camera to assess the NBVs to best explore a scene efficiently by being guided by a 3D bounding box surrounding the object. Despite relying solely on monocular RGB cameras and self-supervised depth estimation, the MACARONS-NBV achieved considerable surface coverage, i.e., 95% for the bridge object, 77 % for the building, and 88.8% for the powerlines covered by the autonomous exploration trajectory.

The first part of the research was to compare the surface coverage achieved with a learning-based depth estimation method, as the depth map, as opposed to a LiDAR sensor, along the same NBV trajectory. The results with only RGB images show that the MACARONS-NBV achieves 81.1% coverage for the liberty object compared to 86.8% coverage using LiDAR. There is only a 5% difference between the coverages achieved by both techniques, which are comparable as shown in Figure 5-2. The LiDAR-based point cloud coverage is computed using C2C distance with a similarly uniform sampled point cloud of the surface mesh. The threshold distance we used is 0.2 meters. All points within this distance are positive and contribute to the estimation of coverage for the LiDAR point cloud. This gives us a quantitative assessment of the completion of the Velodyne point cloud.

The second part of the research builds on top of the autonomous exploration pipeline of MACARONS-NBV to scale the NBV trajectories to a collaborative UAV setting. The Task Assignment module consists of trajectory clustering, path optimization based on distance, and distance + elevation-based cost functions to optimize the assignment of waypoints of the trajectory, considering the complications of the collision of trajectories with the 3D object. These tasks reduce the distance needed for the corresponding UAVs to cover the same object by optimizing the trajectory. This is evident for all the objects in Figure 5-8, irrespective of the cost-heuristic used. The path for each UAV is optimized by solving the Travelling Salesperson Problem in 3D. The system is also equipped with a Task Reassignment module, which looks for the completion of tasks for each UAV and then assigns (if available) the uncompleted sections of the scene to a UAV with remaining endurance to cover the remaining significant points. These components work together to autonomously explore an infrastructure object with no prior knowledge, and using a collaborative system of UAVs to optimize, monitor and enhance the communication of the UAVs to acquire high resolution data to 3D reconstruct an infrastructure object while only being equipped with a consumer-grade monocular RGB camera.

Advantages of RGB-based exploration and 3D Reconstruction using collaborative UAVs can be listed as follows:

- The RGB-based NBV method can autonomously explore and estimate the surface coverage achieved by the NBVs and scale to explore large, unknown infrastructure objects.
- This exploration method does not require a depth sensor to estimate the surface coverage; it is done with self-supervised learning-based depth estimation from sequential monocular RGB images.
- This method assigns data collection tasks to UAVs with awareness of the other UAVs, making the overall system fault-tolerant and energy-aware. This also makes the system more scalable.
- It supports dynamic trajectory replanning to avoid collisions with scene structures and adapt to environmental changes.
- The UAVs collectively ensure comprehensive data acquisition, reducing the data-collection time.

The limitations of the approach can be summarized as follows

- It is a simulator-based solution, hence using many assumptions and ignoring real-world complexities regarding infrastructure objects.
- A single camera achieves the exploration of infrastructure as compared to the collaborative data collection for the high-resolution 3D Reconstruction.
- The scale of the simulator environment must be uniform across all the different processing pipelines of the workflow, e.g., Estimating NBVs using MACARONS, Task-Assignment, collaborative data collection, and obstacle avoidance.
- The current workflow is sequential, i.e. the collaborative data collection and 3D Reconstruction steps are heavily dependent on the configuration of MACARONS, hence the system is very prone to error-propagation.

6.2. Recommendations for further studies

Based on the findings and limitations of this thesis, the following recommendations are proposed for future research and development:

- The MACARONS-NBV method can be implemented in a physics-based robotics simulator like ROS and AirSim to reconstruct the infrastructure scenes in real-time with high-resolution data while being acquired by each UAV. This can be achieved by integrating with robotics middleware such as ROS 2, and simulation environments like AirSim or Isaac Sim, allowing live testing of perception, planning, and control.
- Current exploration is geometric in nature. Future pipelines could explore semantic-aware NBV planning, where high-interest regions (e.g., cracks, windows, anomalies) are prioritized for dense scanning. Leveraging models such as Mask2Former or Segment Anything (SAM) for real-time instance segmentation could drive a smarter NBV exploration policy.
- The exploration relies on RGB-based depth estimation via the ManyDepth , with no simultaneous localization or loop-closure. Integrating RGB-based SLAM methods like ORB-SLAM3, DSO, or deep learning-based depth-SLAM could allow UAVs to map, localize, and select views adaptively. This would also enable better loop-closure detection, critical for high-fidelity mesh reconstruction.
- One of the key limitations is the mismatch in spatial scales across modules. Future work should incorporate geo-referenced scene models and real-world coordinate systems (e.g., UTM, WGS84)

into the simulation pipeline to maintain spatial consistency across exploration, planning, and reconstruction. This would also simplify integration with external datasets such as cadastral maps or BIM.

6.3. Answers to the research questions

Is the learning-based RGB depth prediction a reliable alternative to an onboard depth sensor or LiDAR?

Yes, learning-based monocular RGB depth estimation methods that specifically use self-supervised learning can be considered a reliable alternative to a LiDAR or a depth sensor when constrained by limitations of cost and payload. The first part of the experiment in section 5.1 compares the surface coverage of an RGB-based Surface coverage and LiDAR-based simulations to compare the coverage of an object during the exploration phase. The results are within 5% with LiDAR-based simulations representing higher coverage due to more consistent and uniform distribution of 3D points.

What algorithms can be developed to enable UAVs to autonomously coordinate their flight paths for optimal coverage in a 3D reconstruction project?

A series of different methods can enable UAVs to autonomously coordinate their flight paths with one another in a centralized setting. The exploration area can be clustered into regions. UAVs can be assigned to those regions with an optimized version of the trajectories working with active collision avoidance by using algorithms like RRT to be aware of the surrounding objects. This is essential as when optimizing the trajectory with a custom heuristic like Table 5-1, the overall travelling distance and energy consumption are reduced, but it does not guarantee a collision-free path. Furthermore, we also demonstrate in section 5.3 that some part of agents may or may not be able to complete their assigned data collection task, which can be resolved by assigning the remaining trajectory points to the agent with the lesser load, i.e., the UAV with an already completed trajectory or higher remaining battery.

How can monocular RGB cameras assist in the NBV process?

The majority of the NBV methods rely on the use of depth sensors and LiDAR to assess the 3D representation of an object. These representations vary between surface, volumetric, and sampling-based techniques to evaluate the information gain with each iteration. RGB cameras, when used sequentially, i.e., a series of images, can be used to estimate the depth aspect of the environment they represent. The depth estimates are initially at an unknown scale, but the monocular camera defines the range of depth from the sequence of images and the bounds of exploration. This extrinsic method learns the scale from the scene data using self-supervised learning and adjusts the scale according to the environment. Therefore, monocular cameras contribute to estimating the depth of any infrastructure or 3D object in an environment. The depth maps are used to assess the occupancy field, which is consequently used to calculate the NBV trajectory.

How efficient is the data collection process in a collaborative multi-UAV setting compared to a single UAV exploring a 3D Infrastructure environment?

The data collected in a collaborative multi-UAV system is quicker than a single UAV exploration in two significant ways. The viewpoints are assigned to each UAV after following a series of optimization steps. Firstly, the NBV points are assessed, and clusters are assigned according to the input. The path optimization methods and dynamic task reassignment increase the efficiency of each UAV. The second aspect is the simultaneous collection of data according to the newly informed trajectories and the system's scalability to more than two UAVs. The region-specific clusters are already considered when the hierarchical clustering is performed. This keeps the upscaling to additional UAVs open-ended, and more UAVs can be needed depending on the extent and scale of the infrastructure object.

6.4. Ethical Considerations

Using simulated data in robotics offers significant ethical advantages, particularly in enhancing safety and privacy. Simulated environments allow developers to test systems without exposing living beings or costly equipment to real-world risks, which is especially beneficial in critical applications like autonomous UAV navigation. Simulations also alleviate some privacy concerns by reducing the need to collect proprietary sensitive data. On the contrary, however, simulations can introduce biases if the virtual environment fails to adequately reflect real-world complexity and diversity, leading to systems that may not work as expected in actual situations. Additionally, simulations based on deep learning are resource-intensive due to their computationally expensive nature, making them vulnerable and highly prone to simulation factors, forcing us to introduce assumptions as we explained for our experimental setup.

Use of AI Support Tools in Writing

To support the development of this thesis, the author made use of various academic tools to enhance the understanding of complex topics. ChatGPT Plus was occasionally consulted to refine writing ideas; however, all information obtained was cross verified with academic literature and reputable resources. Grammarly was employed to assist with grammar and spelling corrections. The entire content was independently written by the author and thoroughly reviewed to ensure it meets academic standards.

LIST OF REFERENCES

Anastasiou, A., Zacharia, A., Papaioannou, S., Kolios, P., Panayiotou, C. G., & Polycarpou, M. M. (2024). Automated Real-Time Inspection in Indoor and Outdoor 3D Environments with Cooperative Aerial Robots. *2024 International Conference on Unmanned Aircraft Systems (ICUAS)*, 496–504. https://doi.org/10.1109/ICUAS60882.2024.10557006

Bircher, A., Kamel, M., Alexis, K., Oleynikova, H., & Siegwart, R. (2016). Receding Horizon "Next-Best-View" Planner for 3D Exploration. *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 1462–1468. https://doi.org/10.1109/ICRA.2016.7487281

Bircher, A., Kamel, M., Alexis, K., Oleynikova, H., & Siegwart, R. (2018). Receding horizon path planning for 3D exploration and surface inspection. *Autonomous Robots*, *42*(2), 291–306. https://doi.org/10.1007/s10514-016-9610-0

Bircher, A., Kamel, M. S., Alexis, K., Burri, M., Oettershagen, P., Omari, S., Mantel, T., & Siegwart, R. (2016). Three-dimensional coverage path planning via viewpoint resampling and tour optimization for aerial robots. *Autonomous Robots*, *40*. https://doi.org/10.1007/s10514-015-9517-1

Blensor.org. (n.d.). Retrieved June 24, 2025, from https://www.blensor.org/

Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., & Yu, F. (2015). *ShapeNet: An Information-Rich 3D Model Repository* (No. arXiv:1512.03012). arXiv. https://doi.org/10.48550/arXiv.1512.03012

Chen, X., Li, Q., Wang, T., Xue, T., & Pang, J. (2024). *GenNBV: Generalizable Next-Best-View Policy for Active 3D Reconstruction* (No. arXiv:2402.16174; Version 1). arXiv. https://doi.org/10.48550/arXiv.2402.16174

Cieslewski, T., Kaufmann, E., & Scaramuzza, D. (2017). Rapid exploration with multi-rotors: A frontier selection method for high speed flight. *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2135–2142. https://doi.org/10.1109/IROS.2017.8206030

Cloud-to-Mesh Distance—CloudCompareWiki. (n.d.). Retrieved June 25, 2025, from https://www.cloudcompare.org/doc/wiki/index.php/Cloud-to-Mesh_Distance

Connolly, C. (1985). The determination of next best views. *1985 IEEE International Conference on Robotics and Automation Proceedings*, *2*, 432–435. https://doi.org/10.1109/ROBOT.1985.1087372

Dhami, H., Sharma, V. D., & Tokekar, P. (2023a). *MAP-NBV: Multi-agent Prediction-guided Next-Best-View Planning for Active 3D Object Reconstruction* (Version 3). arXiv. https://doi.org/10.48550/ARXIV.2307.04004

Dhami, H., Sharma, V. D., & Tokekar, P. (2023b). *Pred-NBV: Prediction-guided Next-Best-View for 3D Object Reconstruction* (No. arXiv:2304.11465). arXiv. http://arxiv.org/abs/2304.11465

DJI Mini 4 Pro—Specs—DJI. (n.d.). DJI Official. Retrieved June 24, 2025, from https://www.dji.com/nl/mini-4-pro/specs

Ellenberg, A., Branco, L., Krick, A., Bartoli, I., & Kontsos, A. (2015). Use of Unmanned Aerial Vehicle for Quantitative Infrastructure Evaluation. *Journal of Infrastructure Systems*, *21*(3), 04014054. https://doi.org/10.1061/(ASCE)IS.1943-555X.0000246

Explore 3D Models. (n.d.). Sketchfab. Retrieved June 21, 2025, from https://sketchfab.com/3d-models

Fernandez Galarreta, J., Kerle, N., & Gerke, M. (2015). UAV-based urban structural damage assessment using object-based image analysis and semantic reasoning. *Natural Hazards and Earth System Sciences*, *15*(6), 1087–1101. https://doi.org/10.5194/nhess-15-1087-2015

Gao, C., Wang, X., Wang, R., Zhao, Z., Zhai, Y., Chen, X., & Chen, B. M. (2023). A UAV-based explore-then-exploit system for autonomous indoor facility inspection and scene reconstruction. *Automation in Construction*, *148*, 104753. https://doi.org/10.1016/j.autcon.2023.104753

González de Santos, L. M., Frías Nores, E., Martínez Sánchez, J., & González Jorge, H. (2021). Indoor Path-Planning Algorithm for UAV-Based Contact Inspection. *Sensors*, *21*(2), Article 2. https://doi.org/10.3390/s21020642

Gschwandtner, M., Kwitt, R., Uhl, A., & Pree, W. (2011). BlenSor: Blender Sensor Simulation Toolbox. In G. Bebis, R. Boyle, B. Parvin, D. Koracin, S. Wang, K. Kyungnam, B. Benes, K. Moreland, C. Borst, S. DiVerdi, C. Yi-Jen, & J. Ming (Eds.), *Advances in Visual Computing* (pp. 199–208). Springer. https://doi.org/10.1007/978-3-642-24031-7_20

Gu, X., Fan, Z., Dai, Z., Zhu, S., Tan, F., & Tan, P. (2020). *Cascade Cost Volume for High-Resolution Multi-View Stereo and Stereo Matching* (No. arXiv:1912.06378). arXiv. https://doi.org/10.48550/arXiv.1912.06378

Guédon, A., Monasse, P., & Lepetit, V. (2022). *SCONE: Surface Coverage Optimization in Unknown Environments by Volumetric Integration* (Version 2). arXiv. https://doi.org/10.48550/ARXIV.2208.10449

Guédon, A., Monnier, T., Monasse, P., & Lepetit, V. (2023). *MACARONS: Mapping And Coverage Anticipation with RGB Online Self-Supervision* (Version 2). arXiv. https://doi.org/10.48550/ARXIV.2303.03315

Hardouin, G., Moras, J., Morbidi, F., Marzat, J., & Mouaddib, E. M. (2020). Next-Best-View planning for surface reconstruction of large-scale 3D environments with multiple UAVs. *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 1567–1574. https://doi.org/10.1109/IROS45743.2020.9340897

Hermann, M., Weinmann, M., Nex, F., Stathopoulou, E. K., Remondino, F., Jutzi, B., & Ruf, B. (2024). Depth estimation and 3D reconstruction from UAV-borne imagery: Evaluation on the UseGeo dataset. *ISPRS Open Journal of Photogrammetry and Remote Sensing*, *13*, 100065. https://doi.org/10.1016/j.ophoto.2024.100065

Hoang, L., Lee, S.-H., Kwon, O.-H., & Kwon, K.-R. (2019). A Deep Learning Method for 3D Object Classification Using the Wave Kernel Signature and A Center Point of the 3D-Triangle Mesh. *Electronics*, *8*(10), Article 10. https://doi.org/10.3390/electronics8101196

Hornung, A., Wurm, K. M., Bennewitz, M., Stachniss, C., & Burgard, W. (2013). OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots*, *34*(3), 189–206. https://doi.org/10.1007/s10514-012-9321-0

Koch, T., Körner, M., & Fraundorfer, F. (2019). Automatic and Semantically-Aware 3D UAV Flight Planning for Image-Based 3D Reconstruction. *Remote Sensing*, *11*(13), 1550. https://doi.org/10.3390/rs11131550

Koenig, N., & Howard, A. (2004). Design and use paradigms for Gazebo, an open-source multi-robot simulator. *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, *3*, 2149–2154 vol.3. https://doi.org/10.1109/IROS.2004.1389727

Kriegel, S., Bodenmuller, T., Suppa, M., & Hirzinger, G. (2011). A surface-based Next-Best-View approach for automated 3D model completion of unknown objects. *2011 IEEE International Conference on Robotics and Automation*, 4869–4874. https://doi.org/10.1109/ICRA.2011.5979947

LaValle, S. (1998). Rapidly-exploring random trees: A new tool for path planning. *The Annual Research Report*. https://www.semanticscholar.org/paper/Rapidly-exploring-random-trees-%3A-a-new-tool-for-LaValle/d967d9550f831a8b3f5cb00f8835a4c866da60ad

LaValle, S. M. (2006a). *Planning Algorithms* (1st ed.). Cambridge University Press. https://doi.org/10.1017/CBO9780511546877

LaValle, S. M. (2006b). *Planning Algorithms* (1st ed.). Cambridge University Press. https://doi.org/10.1017/CBO9780511546877

Liu, T., Carlberg, M., Chen, G., Chen, J., Kua, J., & Zakhor, A. (2010). Indoor localization and visualization using a human-operated backpack system. *2010 International Conference on Indoor Positioning and Indoor Navigation*, 1–10. https://doi.org/10.1109/IPIN.2010.5646820

Maboudi, M., Homaei, M., Song, S., Malihi, S., Saadatseresht, M., & Gerke, M. (2023). A Review on Viewpoints and Path Planning for UAV-Based 3-D Reconstruction. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *16*, 5026–5048. https://doi.org/10.1109/JSTARS.2023.3276427

Madhuanand, L., Nex, F., & Yang, M. Y. (2020). DEEP LEARNING FOR MONOCULAR DEPTH ESTIMATION FROM UAV IMAGES. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *V-2–2020*, 451–458. https://doi.org/10.5194/isprs-annals-V-2-2020-451-2020

Madhuanand, L., Nex, F., & Yang, M. Y. (2021). Self-supervised monocular depth estimation from oblique UAV videos. *ISPRS Journal of Photogrammetry and Remote Sensing*, *176*, 1–14. https://doi.org/10.1016/j.isprsjprs.2021.03.024

Mendoza, M., Vasquez-Gomez, J. I., Taud, H., Sucar, L. E., & Reta, C. (2020a). Supervised learning of the next-best-view for 3d object reconstruction. *Pattern Recognition Letters*, *133*, 224–231. https://doi.org/10.1016/j.patrec.2020.02.024

Mendoza, M., Vasquez-Gomez, J. I., Taud, H., Sucar, L. E., & Reta, C. (2020b). Supervised learning of the next-best-view for 3d object reconstruction. *Pattern Recognition Letters*, *133*, 224–231. https://doi.org/10.1016/j.patrec.2020.02.024

Mildenhall, B., Srinivasan, P. P., Tancik, M., Barron, J. T., Ramamoorthi, R., & Ng, R. (2020). *NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis* (No. arXiv:2003.08934). arXiv. https://doi.org/10.48550/arXiv.2003.08934

Mishra, D., Trotta, A., Traversi, E., Felice, M. D., & Natalizio, E. (2022). Cooperative Cellular UAV-to-Everything (C-U2X) communication based on 5G sidelink for UAV swarms. *Computer Communications*, *192*, 173–184. https://doi.org/10.1016/j.comcom.2022.06.001

Mohsan, S. A. H., Khan, M. A., Noor, F., Ullah, I., & Alsharif, M. H. (2022). Towards the Unmanned Aerial Vehicles (UAVs): A Comprehensive Review. *Drones*, *6*(6), 147. https://doi.org/10.3390/drones6060147

Moller, T., & Trumbore, B. (n.d.). *Fast Minimum Storage Ray Triangle Intersection*.

Murtagh, F., & Legendre, P. (2014). Ward's Hierarchical Clustering Method: Clustering Criterion and Agglomerative Algorithm. *Journal of Classification*, *31*(3), 274–295. https://doi.org/10.1007/s00357-014-9161-z

Nagasawa, R., Mas, E., Moya, L., & Koshimura, S. (2021). Model-based analysis of multi-UAV path planning for surveying postdisaster building damage. *Scientific Reports*, *11*(1), 18588. https://doi.org/10.1038/s41598-021-97804-4

Nex, F., & Remondino, F. (2014). UAV for 3D mapping applications: A review. *Applied Geomatics*, *6*(1), 1–15. https://doi.org/10.1007/s12518-013-0120-x

Nex, F., Stathopoulou, E. K., Remondino, F., Yang, M. Y., Madhuanand, L., Yogender, Y., Alsadik, B., Weinmann, M., Jutzi, B., & Qin, R. (2024). UseGeo—A UAV-based multi-sensor dataset for geospatial research. *ISPRS Open Journal of Photogrammetry and Remote Sensing*, *13*, 100070. https://doi.org/10.1016/j.ophoto.2024.100070

Papachristos, C., Kamel, M., Popović, M., Khattak, S., Bircher, A., Oleynikova, H., Dang, T., Mascarich, F., Alexis, K., & Siegwart, R. (2019). Autonomous Exploration and Inspection Path Planning for Aerial Robots Using the Robot Operating System. In A. Koubaa (Ed.), *Robot Operating System (ROS): The Complete Reference (Volume 3)* (pp. 67–111). Springer International Publishing. https://doi.org/10.1007/978-3-319-91590-6_3

Peralta, D., Casimiro, J., Nilles, A. M., Aguilar, J. A., Atienza, R., & Cajote, R. (2020). *Next-Best View Policy for 3D Reconstruction* (No. arXiv:2008.12664). arXiv. http://arxiv.org/abs/2008.12664

pytorch3d.transforms—PyTorch3D documentation. (n.d.). Retrieved June 18, 2025, from https://pytorch3d.readthedocs.io/en/latest/modules/transforms.html

Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). *PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation* (No. arXiv:1612.00593). arXiv. http://arxiv.org/abs/1612.00593

Ravi, N., Reizenstein, J., Novotny, D., Gordon, T., Lo, W.-Y., Johnson, J., & Gkioxari, G. (2020). *Accelerating 3D Deep Learning with PyTorch3D* (No. arXiv:2007.08501). arXiv. https://doi.org/10.48550/arXiv.2007.08501

Respall, V. M., Devitt, D., & Fedorenko, R. (2020). Unmanned Aerial Vehicle Path Planning for Exploration Mapping. *2020 International Conference Nonlinearity, Information and Robotics (NIR)*, 1–6. https://doi.org/10.1109/NIR50484.2020.9290232

Ribeiro, A., & Basiri, M. (2024). Cooperative 3D Exploration and Mapping using Distributed Multi-Robot Teams. *2024 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*, 132–137. https://doi.org/10.1109/ICARSC61747.2024.10535925

Roberts, M., Dey, D., Truong, A., Sinha, S., Shah, S., Kapoor, A., Hanrahan, P., & Joshi, N. (2017). *Submodular Trajectory Optimization for Aerial 3D Scanning* (No. arXiv:1705.00703). arXiv. https://doi.org/10.48550/arXiv.1705.00703

Scharstein, D., Szeliski, R., & Zabih, R. (2001). A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *Proceedings IEEE Workshop on Stereo and Multi-Baseline Vision (SMBV 2001)*, 131–140. https://doi.org/10.1109/SMBV.2001.988771

Schmid, K., Hirschmüller, H., Dömel, A., Grixa, I., Suppa, M., & Hirzinger, G. (2012). View Planning for Multi-View Stereo 3D Reconstruction Using an Autonomous Multicopter. *Journal of Intelligent & Robotic Systems*, *65*(1), 309–323. https://doi.org/10.1007/s10846-011-9576-2

Shah, S., Dey, D., Lovett, C., & Kapoor, A. (2017). *AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles* (No. arXiv:1705.05065). arXiv. https://doi.org/10.48550/arXiv.1705.05065

Smith, N., Moehrle, N., Goesele, M., & Heidrich, W. (2018). Aerial path planning for urban scene reconstruction: A continuous optimization method and benchmark. *ACM Transactions on Graphics*, *37*(6), 1–15. https://doi.org/10.1145/3272127.3275010

Song, S., & Jo, S. (2017). Online inspection path planning for autonomous 3D modeling using a micro-aerial vehicle. *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 6217–6224. https://doi.org/10.1109/ICRA.2017.7989737

Song, S., Kim, D., & Choi, S. (2022). View Path Planning via Online Multiview Stereo for 3-D Modeling of Large-Scale Structures. *IEEE Transactions on Robotics*, *38*(1), 372–390. IEEE Transactions on Robotics. https://doi.org/10.1109/TRO.2021.3083197

The rise in UAV inspections for civil infrastructure. (n.d.). Retrieved June 24, 2025, from https://www.gim-international.com/content/article/the-rise-in-uav-inspections-for-civil-infrastructure

Tran, V. P., Garratt, M. A., Kasmarik, K., & Anavatti, S. G. (2022). *Frontier-led Swarming: Robust Multi-Robot Coverage of Unknown Environments* (No. arXiv:2111.14295). arXiv. https://doi.org/10.48550/arXiv.2111.14295

Understanding Hierarchical Clustering & Its Use Cases. (2022, March 28). Enterprise Search and Analytics. https://www.searchunify.com/su/blog/understanding-hierarchical-clustering-its-use-cases/

Vasquez-Gomez, J. I., Troncoso, D., Becerra, I., Sucar, E., & Murrieta-Cid, R. (2021). Next-best-view regression using a 3D convolutional neural network. *Machine Vision and Applications*, *32*(2), 42. https://doi.org/10.1007/s00138-020-01166-2

VLP 16 | Ouster. (n.d.). Retrieved June 21, 2025, from https://ouster.com/products/hardware/vlp-16

Wang, T., Xi, W., Cheng, Y., Han, H., & Yang, Y. (2024). RL-NBV: A deep reinforcement learning based next-best-view method for unknown object reconstruction. *Pattern Recognition Letters*, *184*, 1–6. https://doi.org/10.1016/j.patrec.2024.05.014

Wong, L. M., Dumont, C., & Abidi, M. A. (1999). Next best view system in a 3D object modeling task. *Proceedings 1999 IEEE International Symposium on Computational Intelligence in Robotics and Automation. CIRA'99 (Cat. No.99EX375)*, 306–311. https://doi.org/10.1109/CIRA.1999.810066

Wu, S., Sun, W., Long, P., Huang, H., Cohen-Or, D., Gong, M., Deussen, O., & Chen, B. (2014). Quality-driven poisson-guided autoscanning. *ACM Trans. Graph.*, *33*(6), 203:1-203:12. https://doi.org/10.1145/2661229.2661242

Wu, Y.-L., Alsadik, B., Oude Elberink, S., & Vosselman, G. (2023). EFFICIENT UAV FLIGHT PLANNING FOR LOD2 CITY MODEL IMPROVEMENT. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *X-1/W1-2023*, 1105–1112. https://doi.org/10.5194/isprs-annals-X-1-W1-2023-1105-2023

Yamauchi, B. (1997). A frontier-based approach for autonomous exploration. *Proceedings 1997 IEEE International Symposium on Computational Intelligence in Robotics and Automation CIRA'97. "Towards New Computational Principles for Robotics and Automation,"* 146–151. https://doi.org/10.1109/CIRA.1997.613851

Yu, X., Rao, Y., Wang, Z., Liu, Z., Lu, J., & Zhou, J. (2021). *PoinTr: Diverse Point Cloud Completion with Geometry-Aware Transformers* (Version 1). arXiv. https://doi.org/10.48550/ARXIV.2108.08839

Zeng, R., Zhao, W., & Liu, Y.-J. (2020). PC-NBV: A Point Cloud Based Deep Network for Efficient Next Best View Planning. *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 7050–7057. https://doi.org/10.1109/IROS45743.2020.9340916

Zhang, S., Liu, C., & Haala, N. (2024). Guided by model quality: UAV path planning for complete and precise 3D reconstruction of complex buildings. *International Journal of Applied Earth Observation and Geoinformation*, *127*, 103667. https://doi.org/10.1016/j.jag.2024.103667

Zhou, B., Xu, H., & Shen, S. (2022). *RACER: Rapid Collaborative Exploration with a Decentralized Multi-UAV System* (No. arXiv:2209.08533). arXiv. http://arxiv.org/abs/2209.08533

Zhou, K., Meng, X., & Cheng, B. (2020). Review of Stereo Matching Algorithms Based on Deep Learning. *Computational Intelligence and Neuroscience*, *2020*(1), 8562323. https://doi.org/10.1155/2020/8562323

Zhu, C., Ding, R., Lin, M., & Wu, Y. (2015). A 3D Frontier-Based Exploration Tool for MAVs. *2015 IEEE 27th International Conference on Tools with Artificial Intelligence (ICTAI)*, 348–352. https://doi.org/10.1109/ICTAI.2015.60

APPENDICES

Appendix A: Mathematical concept behind Triangle Line intersection for Collision detection

Mathematically, the ray–triangle intersection problem addresses whether a ray (an infinite or semi-infinite line) intersects a given triangle in 3D space. The solution proposed by Moller & Trumbore (n.d.) is a widely used algorithm due to its computational efficiency and simplicity.

It states that, given a triangle defined by its vertices $V_0, V_1, V_2 \in \mathbb{R}^3$, and A ray defined by an origin $\mathbf{O} \in \mathbb{R}^3$ and direction $\mathbf{D} \in \mathbb{R}^3$, where $\mathbf{D}$ is normalized,

The algorithm solves for parameters t, u, v Such that:

$$\mathbf{O} + t\mathbf{D} = (1 - u - v)V_0 + uV_1 + vV_2$$

Subject to the constraints:

$$t \geq 0, \quad u \geq 0, \quad v \geq 0, \quad u + v \leq 1$$

If a solution exists, the ray intersects the triangle at a point. $\mathbf{P} = \mathbf{O} + t\mathbf{D}$.

In our context, this method detects intersections between UAV paths and a 3D triangular mesh (e.g., obstacles, terrain, or building surfaces). Given a discrete set of **waypoints** $S = \{P_1, P_2, \ldots, P_N\}$ assigned to each UAV, we consider the linear segment between consecutive waypoints, e.g., $P_i \rightarrow P_{i+1}$, as a ray segment for collision testing.

For each such segment, A ray is cast from P_i In the direction $\mathbf{D} = P_{i+1} - P_i$, The segment is finite, so intersection tests are only considered valid if the computed $t \in [0, \| \mathbf{D} \|]$. The Möller–Trumbore intersection test is run against all mesh triangles fully enclosed within the spatial bounds defined by the line segment.

This approach ensures that any direct collision with any part of the geometry can be detected and handled during the data acquisition process for each UAV. As a result, it returns indices for the respective triangles that the vertices represent. The figure 4-2 shows an intersection of a triangle and a ray with the origin highlighted at the intersection point.

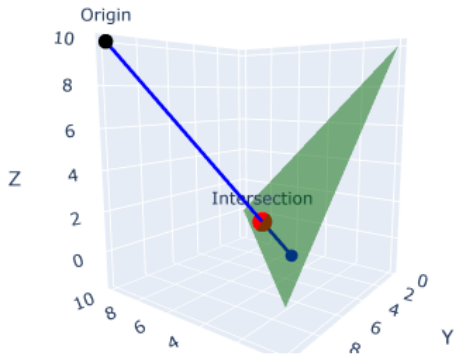


Figure 1 shows Ray-Triangle Intersection (Möller–Trumbore)

Appendix B: Scripts and Code

The codebase and various scripts are provided as part of the following repository.
https://github.com/ahmed991/3DRecon-Thesis

Overview of Scripts

1. Pre-processing datasets for MACARONS
2. MACARONS-NBV results to Blender Euler coordinates
3. Save MACARONS results to surface point cloud
4. Visualize scene parameters using a threeJS visualizer
5 Scene parameters (settings.json) visualizer
6. Trajectory clustering
7. Path optimization using Networkx (distance, distance+elevation)
8. Mesh Line Intersection
9. Collision Detection
10. 3D RRT Planning
11. Battery Assignment Module and endurance simulation
12. Accuracy Assessment of 3D Reconstruction

Blender Scripts

1. Load Trajectories
2. Load Trajectories lines
3. Render Frames
4. Export Camera Orientations
5. Embed EXIF data