VP4D: View Planning for 3D and 4D Scene Understanding

Aditya Rauniyar

CMU-RI-TR-24-36

July 12, 2024

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:
Prof. Sebastian Scherer, *chair*
Prof. Katia Sycara, *co-chair*
Prof. Jiaoyang Li
Cherie Ho

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Robotics.

To my parents, Pratima and Pradeep Rauniyar.

Abstract

View planning plays a critical role in gathering views that optimize scene reconstruction, which is essential in virtual production and computer animation. A 3D map of the film set and motion capture of actors lead to an immersive experience. Current methods use uncertainty estimation in the neural rendering of view candidates to avoid updating a 3D map, but these are limited to smaller scenes. Moreover, existing techniques for 4D motion capture struggle with obstacles, occlusions, and multiple actors. VP4D addresses these challenges by considering practical circumstances for gathering views of larger 3D scenes and accounting for occlusions and obstacles while filming multiple actors in 4D scenes. It presents (i) Data curation methods to form image clusters that enable uncertainty estimation-based view planning for large-scale outdoor unknown scenes. This includes a clustering method for outdoor scenes using a similarity matrix computed with Structure from Motion (SfM) for stable training performance; (ii) Multi-view planning methods that address obstacles and occlusions using drone-mounted cameras for multiple moving actors in a known scene. This involves sequential and coordinated multi-view planning methods that capture the moving actors in the 4D scene, ensuring view diversity and pixel coverage. Together, these techniques provide novel view planning methods that enhance the quality of views gathered for both 3D and 4D outdoor scene reconstruction applications.

Acknowledgments

First and foremost, I would like to thank my advisor, Sebastian Scherer, whose enthusiasm for robotics and making a real-world impact has been the most influential and inspiring part of my Master's journey. His broader perspective on choosing problem statements has shaped my decision-making process. I am grateful for the opportunity to grow alongside Airlab and collaborate with exceptional people who share a common vision for robotics and AI.

I also extend my gratitude to my co-advisor, Katia Sycara, for welcoming me into the AART Labs mid-way through my Master's program, allowing me to explore, learn, and contribute. Her approach to deep diving into research problems and solving them from the core has been truly admirable.

I would like to thank my committee member, Jiaoyang Li, whose lectures have influenced several methods discussed in this thesis. Additionally, I am grateful to Cherie Ho for her guidance on thesis writing and helping me stay on track.

I would like to express my sincere gratitude to Micah Corah, my mentor throughout my Master's program, for his invaluable time discussing ideas and concepts, and for pushing me to grow as a person. He was also the first person to introduce me to bouldering.

I am deeply appreciative of the valuable lectures from several faculty members at CMU who have shaped my concepts over the years: Michael Erdmann, Henry Chai, Matt Gormley, Deva Ramanan, Deepak Pathak, Jun-Yan Zhu, Shubham Tulsiani, and Wennie Tabib.

I have been incredibly fortunate to build robots and algorithms with the talented individuals at Airlab and AART Lab: Marcelo, Yuechuan Hou, Krishna Suresh, Micah Nye, Silong Yong, Yaqi Xie, Simon Benjamin Stepputtis, and Dana Hughes. Additionally, course projects and presentations with Omar Alama, Yu Quan Chong, Akash Agrawal, Kaushik Balasundar, and Mukul have greatly influenced this thesis.

I am grateful to CMU's clubs for making my Master's program a fun experience. The CMU Poker Club hosted events where I met friends like

Tobias Schindler, Rakshith Srinivasa Murthy, Rutika Moharir, Piyush Mishra, and Achleshwar Luthra. GSA hosted summer leagues where I made soccer friends like Shaolin Kataria, Sayan Mondal, and Fausto Vega. Additionally, I want to thank my cycling buddies, Ryan Aponte, Arthur Bucker, and Anujraaj Goyal, for exploring Pittsburgh with me. I am thankful to my friends Sakar Adhikari and Prashant Bhandari for keeping home food alive, and to Bishu Giri for being like a brother.

I would like to thank Andrew for being an incredible lab buddy and our discussions on life in and outside the lab, which have helped me refine my principles and values. I am also grateful to Yves Georgy Daoud, Christian Berger, and Oliver Wang for being welcoming friends when I first joined CMU, and to Jay Kharade, Aman Mehra, Bharat Raj, and Prachi Garg for creating lasting memories.

The friends I have made during my Master's journey—Emma Cullo, Viekash Vinoth Kumar, Haochen Zhang, Helen Wang, Yifei Liu, James Emilian, Nikhil Keetha, Akshay Venkatesh, Zelin Ye, Ian Higgins, and Christopher Klammer—have made this experience exciting and fun with our travels, camping, hiking, bouldering, playing pool, and get-togethers. I also cherish my friends back home, Prabhat Kumar Shantanu, for climbing Mt. Annapurna Base Camp with me during the summer of 2023, and Sanjoli Joshi, for trying ice skating for the first time.

I am thankful to CMU's Swartz Center and Project Olympus for teaching me about entrepreneurship. Special thanks to Dave Mawhinney, Melanie Simko, and Kit Needham for their guidance, and to Frank V. Taylor for being a mentor. I also appreciate the entrepreneurship lessons learned and pursued alongside Andrew Jong, Pratika Dahal, Timothy Watts, Ruben Antonio Quesada, Conner Pulling, and Tyler Harp, as well as insights on digital twins from Varun Kasireddy.

I extend my gratitude to Basheer Mohamed, Debanik Roy, and Prabhu Sethuramalingam for supporting my journey to CMU, and to Diksha Gohlyan for her unwavering support from the application process through to this thesis.

Finally, my deepest gratitude goes to my family. To my parents, Pratima and Pradeep Rauniyar, for their endless love and support, and to my sibling Surbhi Rauniyar, for always being supportive. Thank you for believing in me and encouraging me to pursue my dreams.

Funding

This work was supported by the National Science Foundation (NSF) under Grant No. 2024173, ONR award No.N000142212548, DSTA award No. DST000EC1240000205.

Contents

When this dissertation is viewed as a PDF, the page header is a link to this Table of Contents.

List of Figures

List of Tables

Chapter 1

Introduction

In this thesis, we explore the critical role of view planning in optimizing 4D scene reconstruction. To address the challenges of capturing high-quality views, we present novel techniques and frameworks. These approaches leverage neural rendering, image clustering, and coordinated view planning to enhance the robustness and quality of view capture systems, ultimately creating more realistic and immersive scenic visuals. This chapter provides an overview of the motivation, problem statement, assumptions, and contributions of our research, setting the stage for the detailed discussions in the subsequent chapters.

1.1 Motivation

View planning plays a critical role in optimizing scene reconstruction by gathering the most informative views. Scene reconstruction is vital in various applications, including inspection, gaming, computer-aided geometric design, and robotics. The production of immersive virtual worlds has recently gained significant attention, driven by advancements in virtual reality and augmented reality within the metaverse. These advancements have also contributed to the success of sci-fi movies like "Black Panther" and "The Avengers," as well as animated films like "Inside Out" and "Spiderman: Into the Spider-Verse," which have grossed over $1 billion at the box office. This heightened interest has increased the demand for developing immersive and realistic scenic visuals. Scene reconstruction enhances downstream applications, such as

providing data for learning-based methods in robotics for autonomous navigation and bridging the sim-to-real gap by reconstructing simulations from real images.

These applications require gathering views of static 3D worlds and capturing actors' motion, particularly in outdoor settings. Ground-based view collection is often inefficient and fails to achieve high-quality results, necessitating aerial view-gathering systems capable of covering large scenes. Recent developments in aerial systems focus on planning the next best view at each timestep. However, these methods face constraints due to the computational demands of continuously updating an explicit 3D map [58] from collected images. Alternatively, methods like NeU-NBV [23] use uncertainty estimation in neural rendering to evaluate possible view candidates without needing to update a map. Despite their innovative approach, these methods struggle with larger scenes due to the extensive data curation required. To address these challenges in large-scale outdoor environments, we present techniques that cluster images from outdoor datasets, enabling stable training performance for such neural rendering methods.

Moreover, as these applications heavily depend on accurately reflecting actors' motions or adding CGI while they perform in sets with environmental occlusions, there must be a robust system to address these challenges. Aerial systems, with advancements in drone technology, have shown promise in providing acceptable views for motion capture of actors in outdoor settings. These can be used to create a robust multi-view capture system for actors performing in scenes with significant environmental occlusions. However, current methods like [19, 49], which develops multi-camera (drone) aerial systems, lack multi-view reasoning on scenarios with high environmental occlusions, especially when there is more than one actor in the scene. As groups of actors can have different kinds of unstructured motions [21]. This necessitates an aerial system that gathers reconstructable views of a group of actors performing unstructured motion in scenarios with high environmental occlusions. Moreover, as the occlusion and obstacle density of the environment increase, there is also a requirement to find non-egocentric views in the multi-view system. Hence, we present a sequential and coordinated planner that uses 3D renderings to reason about viewing positioning in a variety of environmental conditions and actor behaviors.

These developments in view planning methods are applied to unknown 3D and known 4D scenes involving groups of actors. Our approach captures the most

informative views, leading to visually immersive scene reconstructions and accurate motion capture, even in environments with high occlusion and unstructured actor motions. By leveraging techniques like image clustering from outdoor datasets and employing a sequential and coordinated planner with 3D renderings, we enhance the robustness and quality of view capture systems. This enables more effective and efficient data collection for CGI and motion capture, meeting the increasing demand for realistic and immersive scenic visuals in various applications.

1.2 Problem Statement

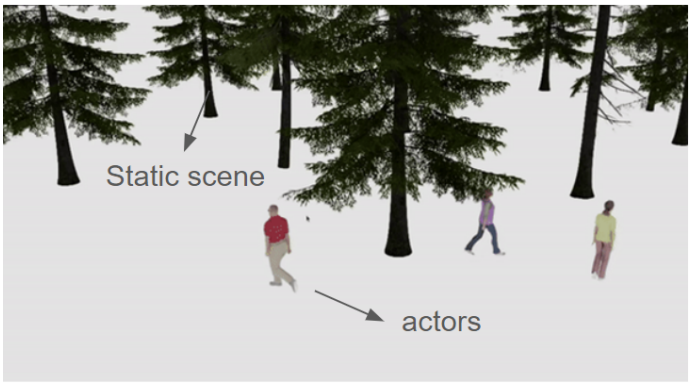


Figure 1.1: Overview of a 4D scene with fixed structures and obstacles, containing multiple moving actors. The goal is to gather views that best capture both static elements and dynamic actors for accurate 4D reconstruction.

The view planning for the 4D scene reconstruction problem highlighted in Figure 1.1 involves gathering a sequence of camera views that account for view diversity and coverage within a time and computational resource budget. This problem is broken down into two sub-problems:

- Unknown 3D Scene: Gather views of an unknown large-scale outdoor 3D static scene so that the view capture system (e.g., a drone) can efficiently cover large areas within a limited capture time.
- Known 4D scene: Capture views of multiple actors with known behaviour in a known outdoor setting containing obstacles and occlusions to provide informative and diverse perspectives of the actors in the scene.

High-quality views have been shown to provide more realistic scene understanding

and reconstruction. With post-processing, these views aim to create an immersive 4D virtual world.

1.3 Assumptions and Simplifications

As part of this thesis, we make the following assumptions and simplifications to focus on optimizing view quality and diversity:

- **Unknown 3D Scene**: The view gathering process aims to capture the next best view that provides the most information that optimizes scene reconstruction. We assume that the drone can move to these locations, avoiding collisions and following trajectories to maximize flight time.
- **Known 4D Scene**: For scenes with moving actors, we assume that the semantic world, including both static components and actors, is known. Actor behavior is assumed to be predictable within a fixed time horizon. Additionally, the starting positions of the drones are assumed to be known. We further assume that actors do not self-occlude with their own body parts and are only occluded by environmental factors. However, occlusions caused by interactions between actors are accounted for. Additionally, we address the collision avoidance problem among multiple drones operating in the 4D scene.

  For simplifications, we discretize time and space into discrete time steps and grid spaces. The drone-held camera operates with a fixed height, pitch, and roll, and its translation along the fixed plane and yaw adjustments occur in discrete steps. The 3D environment is represented using a 2.5D height map and is further simplified by assuming that actors move within a flat plane, meaning the environment does not contain hills or significant elevation changes.

1.4 Contributions

The following are the contributions:

- Developed a data curation method using Structure from Motion (SfM) for image clustering, leading to stable training performance of neural rendering methods

used by Next Best View (NBV) systems in unknown 3D scenes highlighted in Chapter 2.

- Designed a sequential view planning method that optimally plans view positions in unstructured actor group behavior settings, addressing high environmental occlusion and the need for collision avoidance highlighted in Section 3.6.
- Formulated a coordinated view planning approach for group actor behavior, generating non-egocentric views to enhance the system's view robustness. More in Section 3.7.

The work in Chapter 2 was carried out in collaboration with Omar Alama. The work in Chapter 3, specifically Section 3.6 was carried out in collaboration with Krishna Suresh and Micah Corah. More details on the contributions are mentioned in the respective chapters.

1.5 Outline

The thesis is organized as follows:

Chapter 2 focuses on developing a view planning framework specifically for unknown 3D scenes using a neural rendering approach. It addresses challenges encountered with existing methods when applied to large-scale outdoor scenes and proposes novel methods to overcome these challenges.

Chapter 3 focuses on developing a view planning framework for multiple moving actors (humans) in a known 3D world with known actor trajectories. It employs a 3D rendering technique that uses pixel-level metrics as a proxy for reconstruction metrics. It also considers obstacles and environmental occlusions. The framework utilizes multiple unmanned aerial vehicles (UAVs) equipped with cameras to capture diverse viewpoints of actor groups. It introduces two methods of view planning for capturing multiple actors: (1) a sequential approach discussed in Section 3.6, and a coordinated approach described Section 3.7.

Chapter 4 presents the overall conclusion of the thesis, and Chapter 5 outlines the shortcomings of the current approaches and details some future improvements.

In this thesis, the terms cameras, agents, robots, and drones will be used interchangeably.

Chapter 2

View Planning for Unknown 3D Scenes

2.1 Overview

When collecting views for 3D reconstruction and scene understanding, two significant hurdles arise. First, view collection is often constrained by limited resources, such as the operating time of a drone. Second, downsampling to identify the most informative views reduces redundancy and optimizes memory allocation, enabling the coverage of larger areas. This chapter considers a Next Best View (NBV) planning approach, similar to PC-NBV [58] and NeU-NBV [23], for selecting RGB camera positions to explore an unknown 3D scene. With this approach, the view planning horizon is 1, guided by an information gain metric.

Numerous existing methods generate an explicit 3D map of the scene and select the next best view (NBV) to maximize information gain of the map [57, 58]. However, maintaining this 3D map incurs additional computational costs and does not directly compare reconstruction quality. Recently, Neural Radiance Fields (NeRFs) [51] have demonstrated high-fidelity reconstructions compared to other methods like Lidar point clouds [26], meshes [53], or signed distance functions [37]. NeRFs capture the scene as a continuous, differentiable signal with a neural network, allowing for rendering novel views. However, NeRFs struggle to generalize from limited input

images. To address this, PixelNeRF [56] uses a CNN encoder as a prior, enabling better generalization to new, unseen scenes with limited images without test-time optimization.

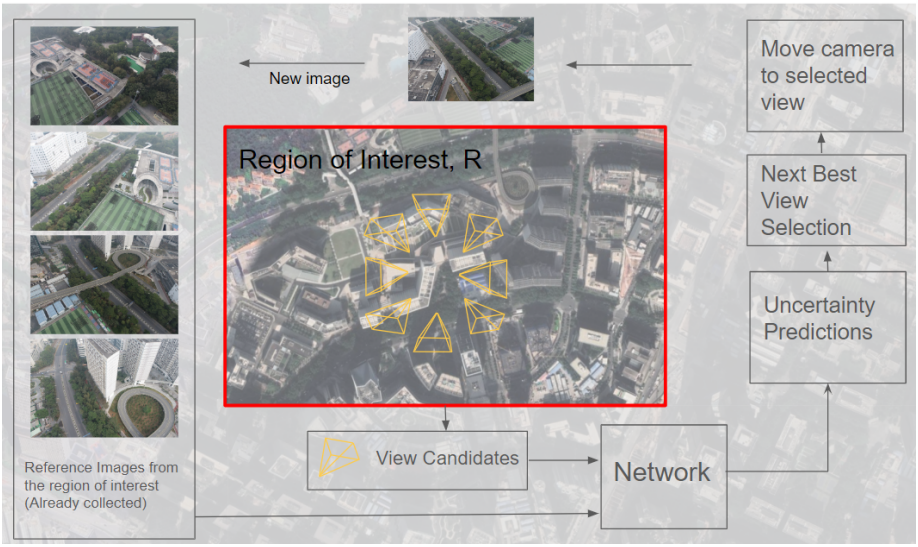


Figure 2.1: Overview of the Nest Best View planning, where the reference images already collected from the scene are used to guide the next view positions. View candidates are sampled from the action space of the camera (drone), and uncertainty predictions are computed for each candidate. The view with the highest uncertainty is selected as the next camera pose to collect samples from and added to the list of reference images.

When predicting an RGB pixel with a neural network, it is crucial to estimate its uncertainty, particularly epistemic uncertainty, which can be managed through test-time dropout [17, 18] or deep ensembles [25]. The network, whether predicting uncertainty or RGB values, as illustrated in Figure 2.1, is highly influenced by the correlation of common features among input images during both training and testing. While PixelNeRF [56] uses small-scale scenes like the DTU dataset, we consider larger-scale scenes from the UrbanScene3D dataset, commonly used in large-scale reconstruction works like MegaNeRF [52] and VastGaussians [29].

In this chapter, we present a clustering method using a Similarity Matrix computed from shared tiepoints through Structure from Motion (SfM). We demonstrate that clustering input images in this manner helps stabilize RGB predictions using PixelNeRF [56] on large-scale scenes, such as the campus scene of the UrbanScene3D [39] dataset.

2.2 Related Works

2.2.1 Next Best View Planning

View planning of RGB Cameras has shown to provide a minimal number of images that can provide all the details of the captured scene. In this section, we present existing methods that are used to capture the Next Best View (NBV) for a scene. Numerous existing approaches address these challenges by updating a 3D map of the scene. Zaenker. et. al. [57] updates a voxel map of the scene and selects the NBV targeted region of interest sampling and frontier-based exploration sampling. These approaches necessitates updating the 3D map (RGBD to Voxel) as more views are collected and additional computations to determine which views provide the most information. However, these views do not semantically reason about what they capture (e.g., buildings or trees), instead focusing solely on the overall scene content.

Recently, deep-learning methods have emerged that utilize 3D structure priors from large-scale data. For example, point cloud-based methods like PC-NBV [58] generate 3D representations such as point clouds. However, these methods also require the generation of 3D representations. All these approaches require explicit discretized 3D map representations to maintain current information about the scene, which limits their scalability and representation ability. One mapless direction is the NeRF approach, as seen in NeU-NBV [23], which employs a PixelNeRF [56] architecture, a generalizable novel view synthesis model, to compute neural uncertainty on view candidates and select the next best view with the highest uncertainty. These can also offer a direct comparison of reconstruction quality as NeRF [34] approaches have been shown to provide high-fidelity reconstruction.

2.2.2 Large scale scene reconstruction

Large city-scale reconstruction has been a long-standing field of research. Many works attempt to reconstruct large scenes using traditional methods such as Lidar point clouds [26], meshes [53], or signed distance functions [37]. However, there is an increased interest in using neural volumetric representations for their high-fidelity reconstructions. Some works [50, 52, 60] recognize NeRF's capacity limitations and

propose forms of spatial decomposition and train many NeRF's to represent different parts of the large scene. Mega-NeRF [52] and BirdNeRF [60] focus on bird view reconstruction, while Block-NeRF [50] focuses on street view. BungeeNeRF [54] takes a different approach focusing on satellite view reconstruction, recognizes the need for multi-scale reconstruction, and progressively trains from big to small scales while increasing network capacity. Urban Radiance Fields [39] presents a multi-modal approach of combining lidar information with RGB signals to address exposure differences in outdoor scenes. VastGaussian [28] introduces spatial decomposition approaches to 3D Gaussian splatting for large-scale bird view scene reconstruction.

Neural representation has been shown to provide photorealistic reconstruction compared to other map-based approaches. We employ an existing neural representation method, PixelNeRF, that has proven to capture the scene information with limited input images, through its CNN encoder-led architecture. Using such a model, however, has only been trained and evaluated on smaller scenes. As we consider, planning views for unknown large-scale 3D scenes, this would require a preprocessing methodology.

2.2.3 Implicit Neural Representation and Uncertainty Estimation

Implicit neural representations parameterize a continuous differentiable signal with a neural network [51]. For instance, NeRFs [61] lean a density and radiance field from the collection of input 2D images, and to render a novel view, it samples points discreetly along the ray and predicts the radiance and density from the position and view direction at each sample point. The final RGB and depth output at a pixel is computed using the differentiable volume rendering. As the scene information is encoded in the network parameters, NeRFs overfit to a single scene and require significant training time. Alternatively, PixelNeRF [56] leverages a CNN encoder to map the input images into a latent feature space. After aggregating features from reference images, a multilayer perception (MLP) is training to interpret the aggregated feature into appreance and geometry information at a novel view. This allows the model to generalize well to new scenes without requiring test-time optimization. Such methods can be used to predict uncertainty at novel views to guide the NBV process

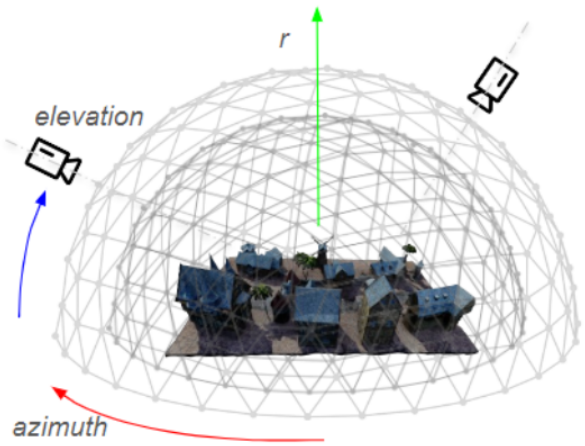


Figure 2.2: Existing datasets for training Generalizable Novel View Synthesis (GNVS) models are often captured in an object-centric manner, where images within a dataset share high feature similarity, as seen in datasets like DTU [1], which focus on small scenes and objects. However, this chapter addresses large-scale outdoor scene datasets, such as UrbanScene3D [31], which are captured using drones and do not follow an object-centric approach. Consequently, these datasets cannot be directly used for training GNVS models because image sequences may not consistently share high feature similarity (as noted during experiments). We present and compare four clustering methods designed to preprocess large-scale outdoor scene datasets, making them better suited for stable training.

as presented in NeU-NBV [23].

Despite the innovative approach of NeU-NBV [23], this method has limitations, particularly its reliance on the DTU dataset for training, which only includes smaller scenes. This chapter addresses problems in large outdoor scenes by exploring other datasets that can be used to pre-train networks for novel view prediction, notably UrbanScene3D [30]. UrbanScene3D has been extensively used as a large-scale outdoor scene dataset [29, 50, 52]. Here, we present the preprocessing procedure of large scenes for training generalizable novel view synthesis models, which serve as a backbone for uncertainty-guided next-best view prediction.

2.3 Preprocessing Real Datasets for GNVS Training.

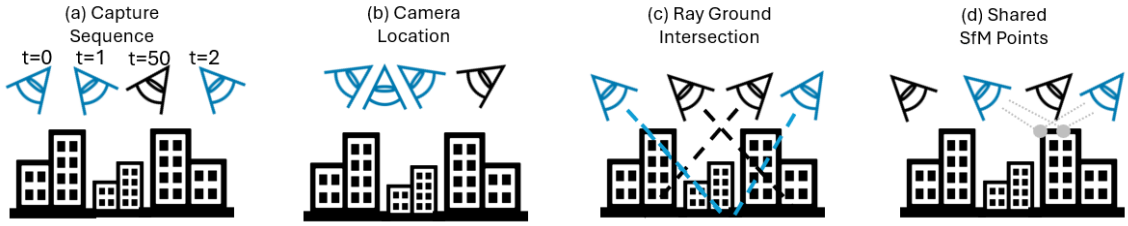


Figure 2.3: Approaches to cluster images from a capture. Colored cameras refer to cameras in the same cluster/group.

Large-scale urban scene data is not readily amenable for generalizable NVS as the data covers a huge baseline. For example, UrbanScene3D [31] real datasets can cover more than $1km^2$ areas spanning multiple high rise and low rise buildings. Hence, an image in the scan may not necessarily contribute meaningfully to the reconstruction of another view. Thus, clustering images meaningfully is critical. We test different algorithms as shown in Figure 2.3 to achieve that targeting the following criteria: First, it is pivotal to cluster images in the scan that are related to each other (ie. looking more or less at the same structures in the scene). Second, the selection of the group size is crucial as too small of a group size will give very little information to the model whilst a very big group size would give confusing and unrelated information to the model. Third, the group size should be constant to allow efficient batching when training. We show a qualitative output of clustering images at Figure 2.10.

Capture Sequence grouping: Using the capture sequence to cluster images is the simplest approach we can use. However sudden turns in the capture sequence can leave images in the same group looking at very different areas of the scene. Furthermore, this method fails to include valuable images from later time steps that are looking at the same area.

Grid-based grouping: In another approach, we can superimpose a grid on the ground plane clustering cameras that are the K nearest neighbors to the grid cell center. However, this approach may cluster cameras that are close by in Euclidean space yet very far apart in terms of their viewing frustums. We further experimented

by adding an angle constraint to ensure that cameras are not only close but roughly looking at the same direction. Still, this approach fails to capture images looking at the same area from different angles.

Ray intersection with ground plane: In an attempt to capture both Euclidean distance and viewing distance, we tried unprojecting the center pixel of each image such that it intersects with the ground plane. We then use the distances between the intersection point as our clustering metric. A drawback of this approach is that you need to estimate the distance of the ground plane to each camera. To do this we use Agisoft Metashape [2] to run SfM on small hand-picked images and calculate the height of the cameras relative to the ground plane then use that to calculate all other camera distances to the ground plane with the assumption that the ground is flat. This approach improved clustering performance but still failed in many cases near high rise buildings as cameras could be looking at different areas even though their rays intersect close to each other at the ground plane level.

SfM shared points: To enforce that images in the cluster are looking at the same structures, we resort to performing full SfM for each scene then using the number of shared points across different cameras as our distance metric for clustering. This approach yielded the best results as shown in Figure 2.10 and avoided the edge cases of all of the aforementioned approaches. Nevertheless, the approach like all others, requires careful tuning of the cluster size.

2.4 Experimental Setup

Dataset: For our experimental analysis, we utilize the comprehensive UrbanScene3D [31] dataset, which encompasses a variety of real (Campus, Hospital, Residence, SciArt, and Polytech) and virtual environments (School, Town, Bridge, and Castle). These scenes provide thousands of high-resolution images, capturing the intricate details and diverse structures. We also work on real scenes from the Mill-19 [52] dataset, which has two environments: building and rubble.

Metric: In evaluating our model, we will apply a combination of quantitative metrics and qualitative assessments to ensure a robust analysis. Quantitatively, we utilize the Peak Signal-to-Noise Ratio (PSNR) to measure the fidelity of the reconstructions against the corrupting noise. Qualitatively, our approach includes

visual inspections to assess the realistic rendering of the scenes. Together, these methods provide a comprehensive evaluation of our model's performance from both statistical accuracy and user experience perspectives. The presented methods are trained on UrbanScene3d's Campus environment.

Comparison: Since there is no existing work that has surveyed clustering algorithms for training G-NVS. We train PixelNeRF [55] on real Campus environments from UrbanScene3D for the 4 methods described in the section. As clustering based on SfM points shows to give the best results, we then analyze 10 and 20-view clusters. We conducted three experiments on Campus with 20-view clusters, using input images with 3, 6, and 9 views. This is done to analyze what cluster size is most suitable and the dependency of input images over the novel views.

Compute Setup: We run PixelNeRF [55] with 256 hidden layers with fixed encoder weights as to its default configuration to fit low computational requirements. We use 2 32Gb Tesla V100 GPUs to train and evaluate original campus, original campus plus plane experiment - training took nearly 23 hours each. All other mentioned experiments use 10Gb NVIDIA RTX3080.

2.5 Results and Discussions

Our experimental analysis focuses on evaluating the performance of our approach on using the 4 methods described in section 2.3. using the campus UrbanScene3D dataset, we first perform experiments using input views (3, 6, and 9). And select the best method to perform tests on cluster sizes (10 and 20) on reconstruction quality.

Comparing different clustering methods: During training, we evaluate different clustering methods for the campus dataset containing 1347 images, and collect results till 25 iterations, with input images as 3 views.

The evaluation of various clustering methods reveals distinct performance characteristics as shown in Table 2.1. Sequence Grouping exhibited the most unstable training curve, with a best PSNR of 9.7, the lowest PSNR of 0.0, and an average PSNR of 3.5. In contrast, Grid-based Grouping demonstrated a more stable training curve, achieving a best PSNR of 12.2, the lowest PSNR of 0.0, and an average PSNR of 4.6. The Ray Intersection with Ground Plane method provided a higher best PSNR of 13.6, with the lowest PSNR at 0.0 and an average PSNR of 9.9. The SfM

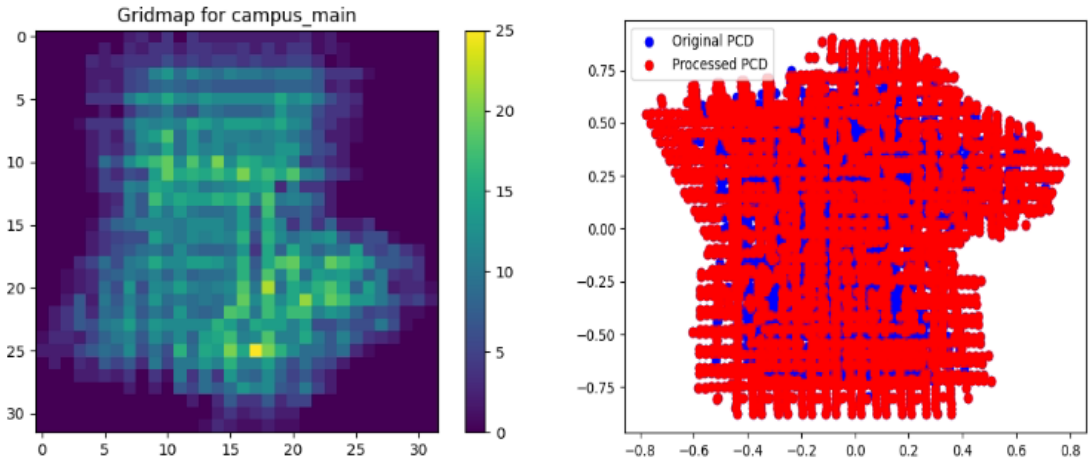


Figure 2.4: **Grid-based grouping:** The left plot shows the frequency of repetition across different clusters on a grid using the K-Medoids clustering algorithm. The right plot displays all the camera poses of the campus scene images from the UrbanScene3D dataset in blue, with the clustered images represented by their camera poses in red.

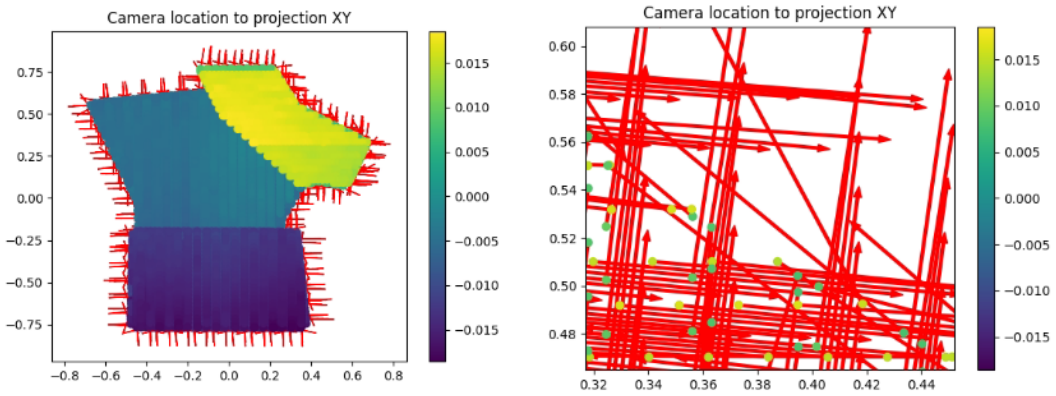
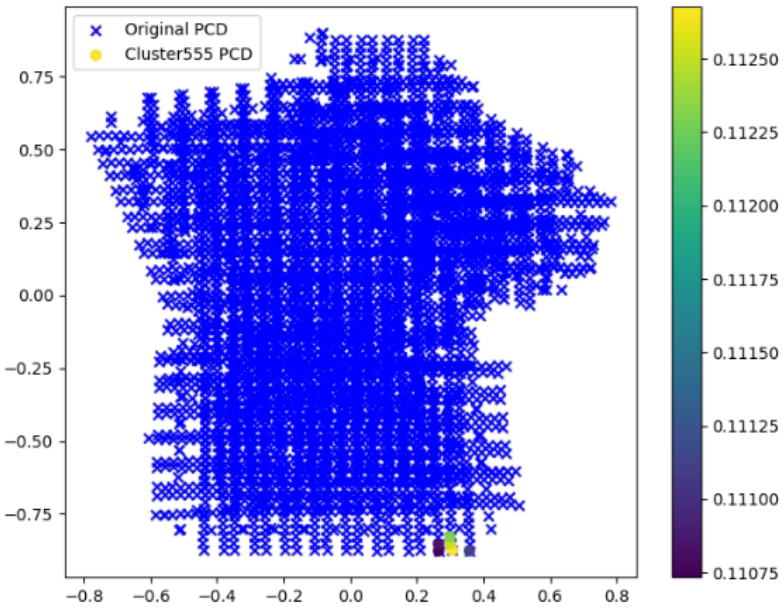


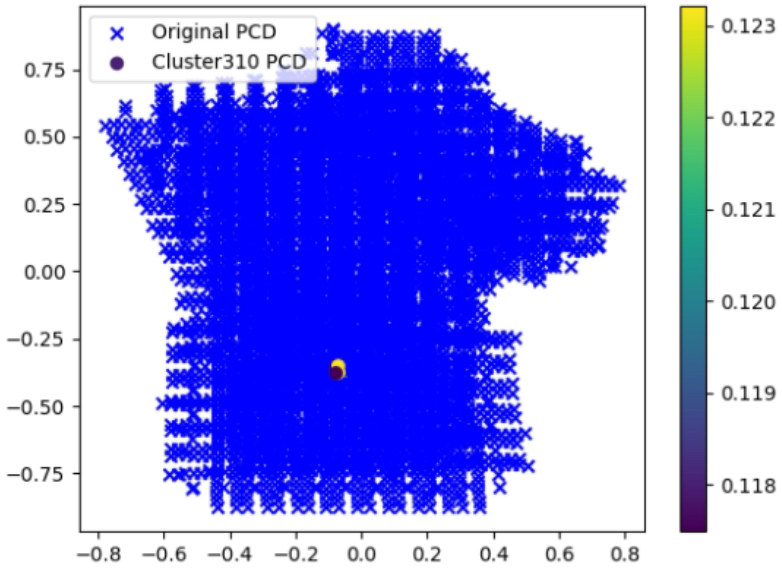
Figure 2.5: **Ray intersection with ground plane grouping:** The left plot shows point projections from all the camera poses from the campus scene of the UrbanScene3D dataset. The red arrows represent the rays being projected from the camera poses to the ground plane. The right plot displays different camera poses with its associated ray projection, where the z-scale (color-coded) indicates the altitude of the camera poses with respect to the ground plane. After the ray has been projected we run the K-Mediods on the projected points for clustering the images based on its projected points.



(a) Grid-based grouping: Blue cross marks indicates the camera poses of the images under the campus scene of UrbanScene3D [31] dataset in the ground plane. The color bar indicates the elevation value of the camera poses in a specific cluster. The colored circle marks indicate the camera poses corresponding to the images clustered for ID: 555.

(b) Images corresponding to the cluster shown in Section 2.4

Figure 2.6: (a) and (b) show the grid-based grouping camera pose and its associated images for a cluster. This presents an *acceptable* scenario as the images share *high* features amongst themselves (seen qualitatively).



(a) Grid-based grouping: Blue cross marks indicates the camera poses of the images under the campus scene of UrbanScene3D [31] dataset in the ground plane. The color bar indicates the camera elevation values from the ground specific to a cluster. The colored circle marks indicate the camera poses corresponding to the images clustered for ID: 310.

(b) Images corresponding to the cluster shown in Section 2.4

Figure 2.7: (a) and (b) show the grid-based grouping camera pose and its associated images for a cluster. This presents an *unacceptable* scenario as the images share *low* features amongst themselves (seen qualitatively).

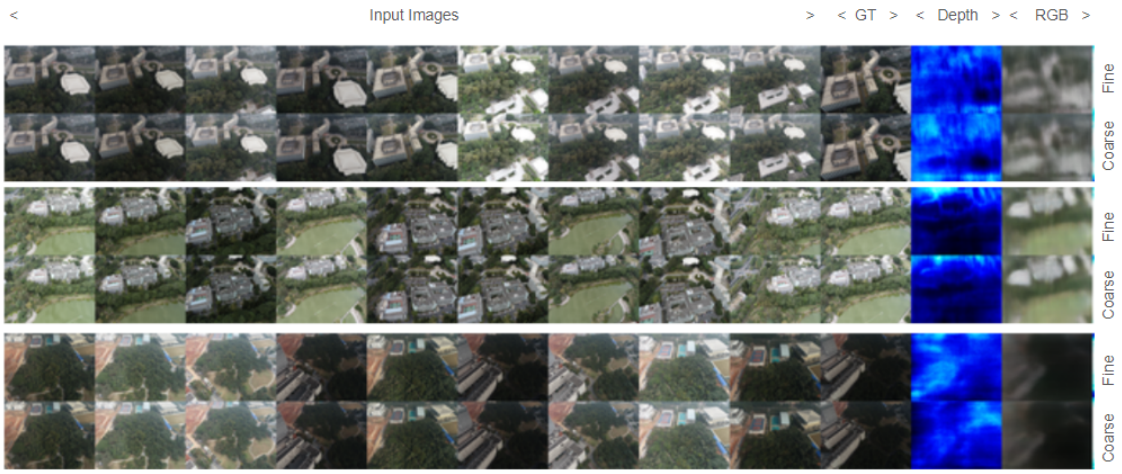


Figure 2.8: Qualitative results of RGB prediction shown in the last column versus the Ground Truth (GT) for 9 input images for clustering using SfM shared points. The results shown are for 3 different clusters where 1st row shows fine predictions and the second row makes coarse predictions.

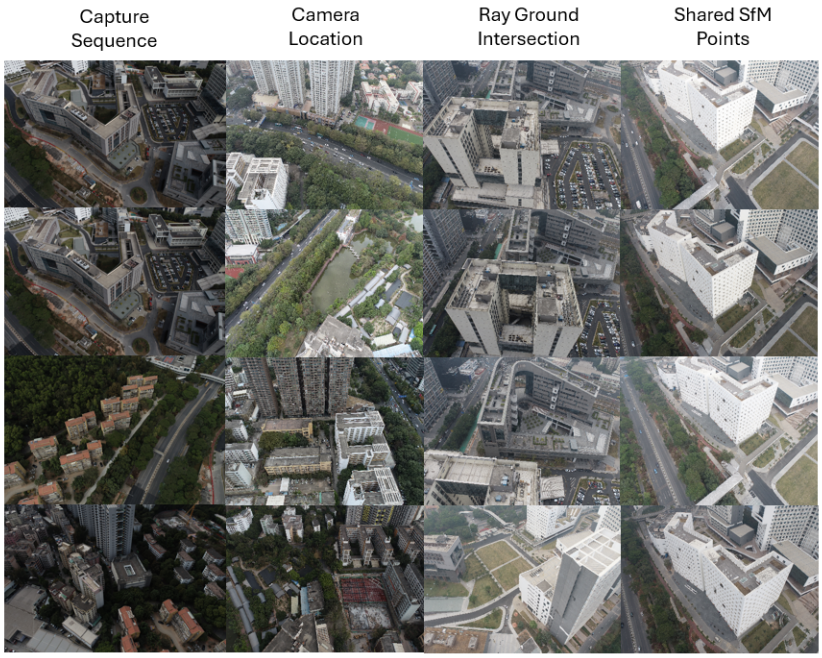


Figure 2.9: **Qualitative Results:** of different clustering algorithms. Images in a column belong to the same cluster

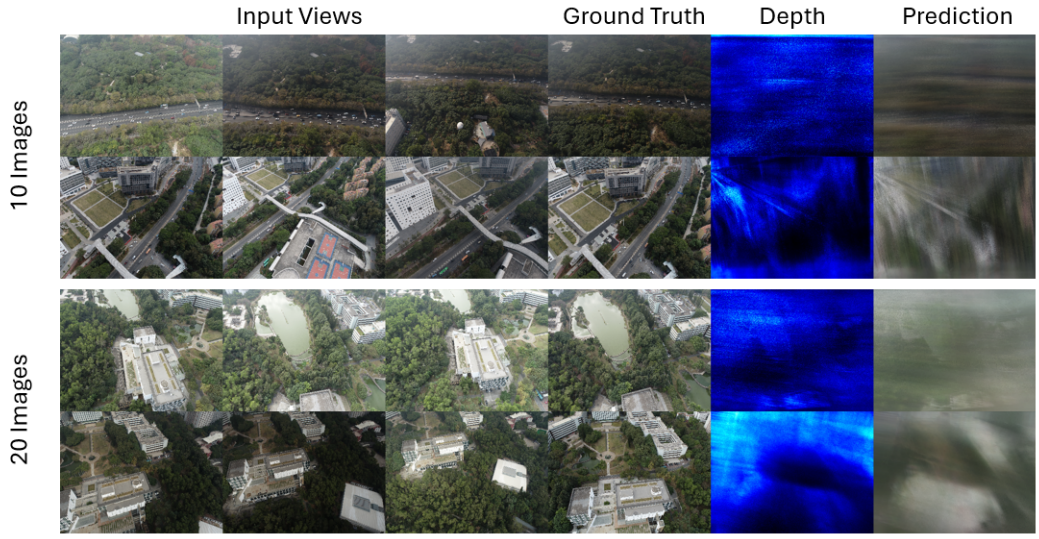


Figure 2.10: Qualitative results on the campus scene of models trained using 20 images per cluster vs 10 images per cluster using the SfM shared points method using 3 input images.

Shared Grouping method showed the most stable training curve, reaching a best PSNR of 20.03, a lowest PSNR of 10.9, and an average PSNR of 14.6.

Table 2.1: Performance of Different Clustering Methods

Method	**Best PSNR**	**Lowest PSNR**	**Average PSNR**
Sequence Grouping	9.7	0.0	3.5
Grid-based Grouping	12.2	0.0	4.6
Ray Intersection with Ground Plane	13.6	0.0	9.9
SfM Shared Grouping	**20.03**	**10.9**	**14.6**

Varying input images: Fig. 2.11, as we vary the number of input images for the campus real dataset clustered with 20 views using clustering based on similarity matrix, we observe that the best PSNR values for input views 3, 6, and 9, after 35k iterations, are 20.03, 19.95, and 19.59, respectively. Surprisingly, this suggests that PixelNeRF performs better with fewer input views, contrary to the typical behavior expected from GNVS models, where increasing the number of input images usually leads to better RGB predictions for novel viewpoints. Our analysis reveals that adding

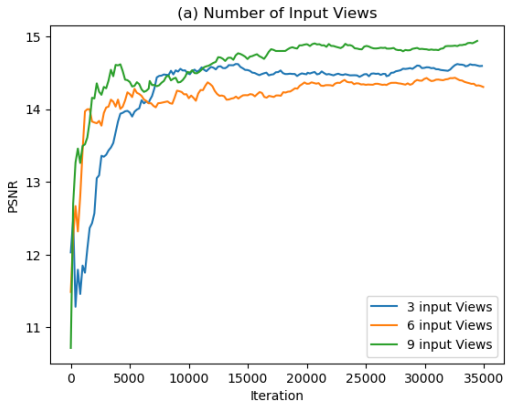


Figure 2.11: Varying the number of input views on campus real data with 20 images per cluster.

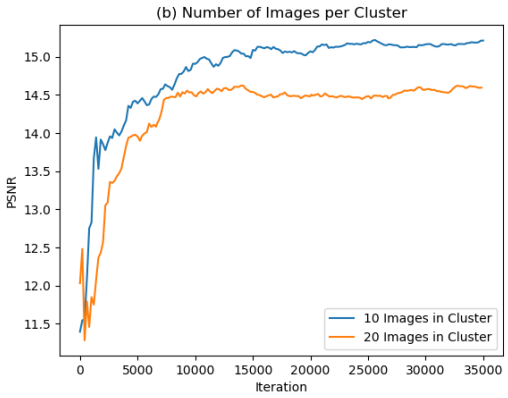


Figure 2.12: Varying the number of images in a cluster, aka cluster size.

Number of Images per Cluster	Best PSNR
10	**22.94**
20	20.03

Table 2.2: Best PSNR values for different numbers of images per cluster.

more input images introduces significant variance and new features among them, posing challenges for feature tracking. This key finding underscores PixelNeRF's sensitivity to novel features and high pose fluctuations among input images. This leads us to perform experiments by varying the number in images in the cluster.

Varying cluster size: This leads to another experiment 2.12. where we conduct experiment to check the size of cluster we form for training PixelNeRF. As we form the clusters based SfM shared points, and find the N images closest to each other that shares the most tie points. As we set a high N-view clustering it would contain images that are farther and make the cluster contain views that share lesser tie points. Best PSNR for 10 images per cluster: 22.94. Best PSNR for 20 images per cluster: 20.03.

2.6 Conclusion and Limitations

To conclude, we refer to a method that uses uncertainty estimation through implicit neural representation for the Next Best View (NBV) planning process in large-scale outdoor scenes, due to its ability to bypass the explicit map-building process. We find that the datasets usually referred to for these scenes are mostly used by large-scale reconstruction communities UrbanScene3D and Mill-19. However, these datasets cannot be readily used to compute uncertainty estimation as existing methods use datasets like DTU and Shapenet with object-centric representation and require a data curation strategy. We evaluate directly on PixelNeRF, which inspired the uncertainty estimation techniques in GNVS, to facilitate faster comparison. We present various clustering methods, out of which computing similarity matrix using Structure from Motion (SfM) technique gives the most stable training seen from Table 2.1. We also find that having a smaller cluster gives better metrics of reconstructibility. However, we also see that inceasing the number of input images to predict a novel view, decreases the PSNR score of the novel view, which is counter-intuitive. This can be so as the clustered images are still not object centric, but just share the features, and also that as we increase the number of input images, we add additional ambiguity amongst images that makes PixelNeRF like architecture unable to map its corresponding feature relation. Moreover, as planning for the next best view would require computing uncertainty over number of view candidates, this would lead to computationally inefficient system. Moreover, such NBV methods are greedily selecting views and doesnt consider a longer horizon view optimization, which if energy inefficient when deployed to a realtime system like a drone with camera.

Acknowledgment and contribution

A portion of this work was also presented as part of the project course requirement of the 16-825 Learning for 3D Vision course at Carnegie Mellon University.

Aditya Rauniyar: He is responsible for the idea, connecting GNVS with NBV and 3 initial methods of clustering to set up a baseline. He is also responsible for conducting experiments and reviewing the text.

Omar Alama: He is responsible with the similarity matrix method using SfM,

writing sections of 2.3, and finalizing figures. he is also responsible for running experiments for evaluating and finalizing graphs.

Aditya Rauniyar and Omar Alama, share equal contributions to writing and code.

The authors would also like to thank Silong Yong, Yaqi Xie, and Simon Stepputtis for their assistance, advice, and feedback on the work.

Chapter 3

View Planning for Known 4D Scenes

3.1 Overview

Figure 3.1: Motion capture setup for virtual production with two actors in a controlled and confined environment. The cameras are placed optimally on a height facing towards the actors such that they optimally cover the room with their view space. **Figure referenced from optitrack*

Chapter 2 discussed about the view planning process of an unknown scene, so to collect the most informative views of a unknown 3D scene towards large scale outdoor scenes. This chapter now assumes that the 3D static scene is known, additionally,

the moving dynamics of the 4D scene.

The capture of significant events via photos and video has become universal, and Unmanned Aerial Vehicles (UAVs) extend camera capabilities by enabling view placement in hard-to-reach places and tracking intricate trajectories. Multiple aerial cameras can simultaneously view an actor from different angles and perform advanced functions such as localization, tracking, environment exploration, mapping, cinematic filming, outdoor human pose reconstruction [3, 8, 9, 11, 13, 19, 35, 38, 41, 42] and motion capture for animation as shown in Figure 3.1. This approach also addresses occlusion issues, where parts of the subject may be hidden from a single camera's view [24]. The broader combined view space allows for greater actor movement freedom and compensates for misalignment between scripted and actual behavior. It also increases robustness against individual camera failures or obstructions and performs well in challenging outdoor scenes with moving backgrounds and varying illumination [15]. Multiple cameras are crucial for capturing scenes with multiple moving subjects [47]. These applications benefit from the effective collaboration of UAV teams, as manual control may result in poor shot selection and view duplication, requiring many coordinated operators. Autonomous coordination of UAVs is thus essential for tasks such as multi-robot filming or reconstruction.

However, directly maximizing domain-specific metrics, such as reconstruction accuracy, can be difficult to perform online—this motivates development of proxy objectives that quantify coverage and detail for multiple views. For example, Bucker et al. [8] demonstrate cinematic filming through a joint objective combining collision and occlusion avoidance, shot diversity, and artistic principles in filming a *single actor*. We are interested in similar settings but where robots collaborate to obtain diverse views of *multiple actors*—in a cluttered environment, with occlusions, where robots may observe more than one actor at once.

While defining an objective can be difficult, planning for multi-robot aerial systems also presents a challenge; the vast joint state space, non-convex environment, and non-linear view rewards make optimal planning intractable. Many applications exploit problem-specific structures to reduce the overall such as by optimizing an actor-centric formation [19, 48] or by altering alter the search procedure to generate single-robot trajectories sequentially [8, 9, 11, 13, 42]. In this work, we apply a planning approach

Figure 3.2: Ground motion capture setup in outdoor settings with no environmental occlusions using multiple tripod camera systems. **Figure referenced from optitrack*

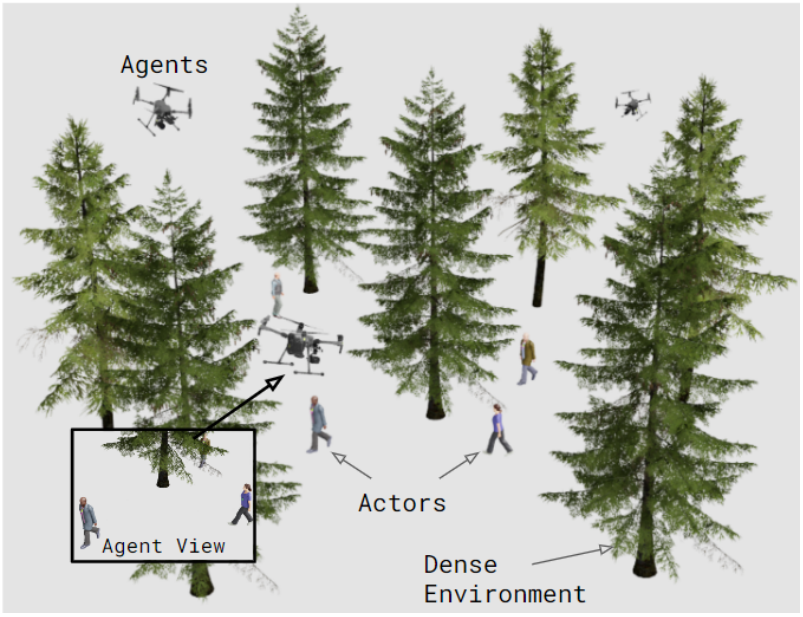


Figure 3.3: **View planning in a known 4D scene**: Overview of aerial motion capture system in an occlusion-prone and unconfined environment. View planning in a known scene, where the 3D static map and the motion behavior of human actors are predetermined. To capture the movements of multiple actors in the scene, this chapter deploys multiple drones equipped with high-definition cameras. As the actors move, the camera positions need to be recalculated for certain horizons based on the drones' action space to maximize pixel coverage and view diversity.

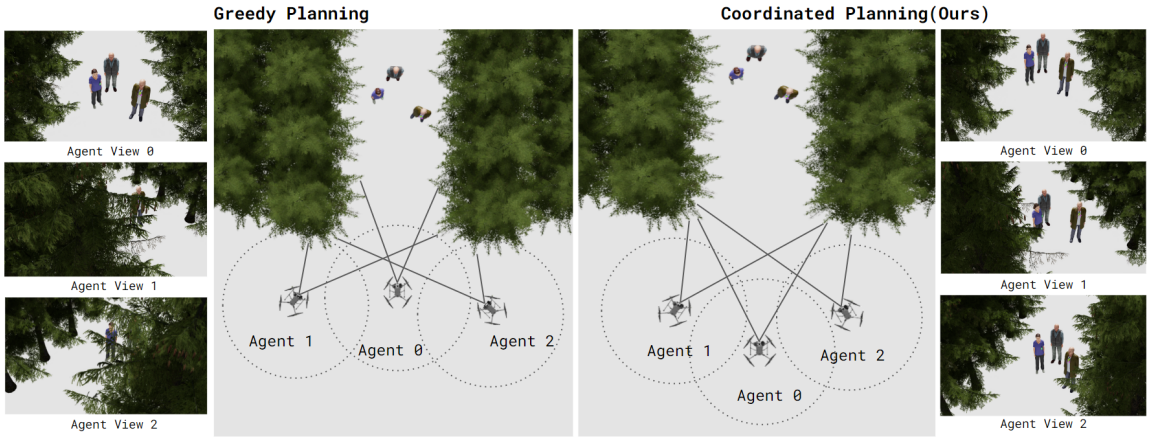


Figure 3.4: **Sequential (Greedy) View Planning**: On the left, there is the sequential view planning of multiple camera positions, where there are egocentric behaviors across multiple viewpoints as seen in the three camera outputs on the left under greedy planning. **Coordinated planning**, on right: we propose a coordinated view planning approach where there is pixel-level negotiation amongst view positions to allow non-egocentric behaviors as seen in the three camera outputs on the right under coordinated planning.

much like Bucker et al. [8], and develop a system design and view rewards that enable the application to a multi-actor setting.

We first present a sequential approach towards planning for multiple cameras then present a coordindated method inspired from [44], where camera motions are computed with overall coverage considerations. This ensures complementary angles, enhances overall capture quality, and significantly improves robustness to failures and errors in the scene as shown by a scenario in Figure 3.4.

Filming multiple actors presents unique challenges as discussed by Hughes. et. al [20]. When actors move in a structured way, like in clusters, they can be represented as a single entity and filmed by UAVs in formation [19]. Assignment schemes [41] can also divide actors into sub-groups for easier filming. However, in scenarios like team sports, dance, and group parkour, actor movements are often unstructured as illustrated in Figure 3.5, making it difficult to maintain formations . This necessitates systems that can optimize collective camera views more directly.

Problem: The dynamic multi-actor view planning problem consists of generating sequences of camera views over a fixed planning horizon to maximize a collective

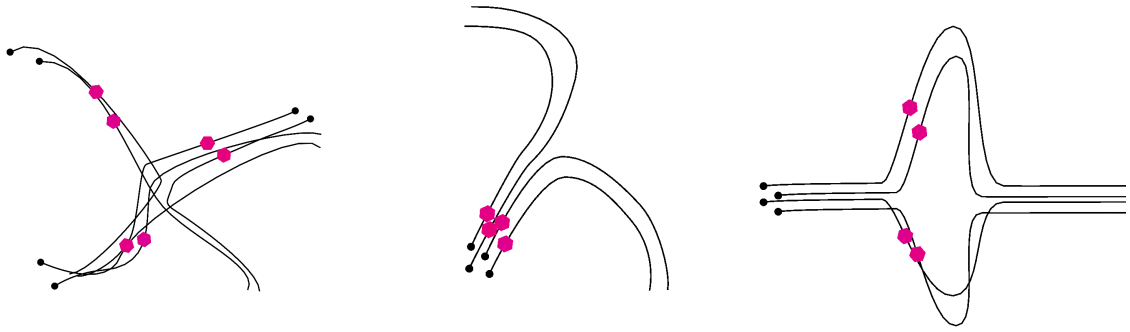


Figure 3.5: Unstructured Actor Group behaviors from Hughes. et. al. [20]. Each scenario is illustrated from a top-down perspective, with actors depicted as red hexagons. The paths of the actors are indicated by black lines, and the starting positions are marked with dots.

view reward which is a function of pixel densities over the surfaces of the actors. The primary assumptions for our approach are: *known static environment*, *known actor trajectories* (e.g. scripted scenarios), and *known robot start state*. An illustration of the problem setup is depicted in Figure 3.3. These assumptions are fairly strict for the purpose of evaluating the planning approach. In practice, our approach could be applied in a receding-horizon setting based on scripted or predicted actor trajectories [8].

Contributions: The main contributions of this work are summarized as:

- An occlusion-aware objective based on rendered camera views for filming groups of people.
- Implementation of a collision-aware multi-robot, multi-actor view planner.
- Evaluation of the view planner in scenarios with challenging obstacles, occlusions, and group behaviors.
- Coordinated view planning that negotiates the camera positions at a timestep for non-egocentric views and optimizes the overall pixel coverage over actors.
- Single-agent view search that is perceptually guided in its heuristics, which is highly computationally efficient and holds real-time applications.

Our contributions build on prior work developing perception objectives based on pixel densities by Jiang and Isler [22] and work by Hughes et al. [20] developing objectives

for submodular multi-robot settings. This work presents a variant of such objectives that is occlusion-aware, and we present results for scenarios with a wide variety of obstacles and occlusions. These contributions toward objective design also enable us employ a planning approach similar to Bucker et al. [8] but for filming multiple actors.

3.2 Related Works

3.2.1 Motion Capture and Dynamic Scenes:

Aerial motion capture [19, 41, 48, 49] such as to reconstruct the motion of a moving person is closely related to filming. So far, these systems consist of groups of robots that observe a single moving subject with various degrees of awareness of obstacles or occlusions. However, all specify either an actor-centric policy [49] or a formation [19, 41, 48]. Although actor-centric planning can reduce the search space to orient the formation versus planning for robots individually, this choice limits the relevance to multi-actor settings. The view planning approach we present could enable robot teams to reconstruct motions of complex group behaviors in environments with varied obstacles and occlusions.

3.2.2 Planning for Perception and Reconstruction

The challenging task of perception planning for UAVs in dynamic obstacle and occlusion environments has been addressed by several researchers. Notably, looking into the Urbanscene3D [31] dataset that was introduced in Chapter 2, the dataset was collected using perception planning methods that capture views to reconstruct a 3D scene. There is a naive solution where the views can be simply collected with oblique photography. However, [46], presents an approach that tries to maximize the *reconstructibility* by adjusting the position and orientation of each viewpoint. [62] uses a similar metric and approach, except it reduces useless viewpoints. [59] further optimizes the viewpoints through smoothness trajectory for energy-efficient behaviors.

Planning for 4D reconstruction is however more complex as the scene is contin-

uously changing with moving objects/subjects and the views have to be computed for each time step. To achieve the objective of maximizing coverage over dynamic targets, a face segmentation and pixel formulation model is required. One such technique is presented in [22], where a target, such as a person, is represented as a polygonal cylinder, and pixel coverage from a specified viewpoint is computed. This technique has been utilized in the formation planning of multiple UAVs for 3D pose reconstruction of moving individuals [19].

The view planning approach also bears similarity to methods for reconstruction of static scenes [14, 22, 40]. Like these works, our approach emphasizes the design of a view reward and optimizing paths to maximize that reward. In particular, we draw on the approach by Jiang and Isler [22] which reasons about pixel densities.

3.2.3 Sequential and Submodular Multi-Robot Planning:

Typically multi-robot perception planning and information gathering problems, cannot tractably be solved optimally due to their combinatorial nature, but greedy methods for submodular optimization can often promise information gain or perception quality no worse than half of optimal in polynomial time [16, 36]. Submodular optimization and sequential greedy planning has been applied extensively to such multi-robot coordination problems [8, 9, 11, 13, 27, 33, 42, 45]. However, questions of occlusions and camera views have been explored primarily in the setting of mapping and exploration [11, 42]. Lauri et al. [27] present an exception in which eye-in-hand cameras inspect and reconstruct static scenes. Unlike exploration and mapping, applications involving filming moving actors can force persistent interaction between robots over the duration of a planning horizon or experiment, and our early work on this topic indicates that sequential planning is important for effective cooperation in settings involving observing moving subjects [10].

Multi-robot Perception Planning with moving actors is challenging due to limited adaptability in cluttered environments, unknown destinations, and inter-robot coordination. One of the approaches towards this issue is addressed by [20]. This work formulates the problem as a perception planning task, achieving suboptimal guarantees through value iteration and sequential submodular maximization. Results demonstrate performance and robustness in challenging scenarios.

In this thesis we would employ such methods towards high obstacles and occlusion environment with 3D perceptual reasoning.

3.2.4 Multi-Agent Path Finding

Finding solutions for multiple agents (robots) and resolving conflicts is a significant challenge in designing dense multi-robot systems, particularly when multiple agents attempt to occupy the same position. One of the simplistic way to do would be sequentially planning for each robot as also described in Section 3.6, which is also known as Prioritized Planning in the Multi-Agent Path Finding (MAPF) world. However, such methods directly takes the order of agents to plan and assumes that sequentially solving would lead to optimal scenario. Consequently, [32] presents a high level solver that determines the correct order of planning amongst multiple agents that leads to optimal value. However both of these methods fail to make the agents move in a coordinated manner.

Recent research in this field has focused on addressing this issue, with the Conflict-Based Search (CBS) algorithm [44] emerging as one of the prominent solutions in the domain of Multi Agent Path Finding(MAPF). Yet, in perception-based planning scenarios, reasoning with perceptual heuristics is a much needed cabability. Since, CBS is orginally build to resolve conflicts which is collisions in a MAPF problem, it doesn't provide much details on use cases where agents have perceptual abilities, which would be laid out in Section 3.7.

3.3 Preliminaries

We will begin with some background regarding submodular and greedy planning:

3.3.1 Submodularity and Monotonicity

The view reward we employ satisfies monotonicity properties that are useful in developing our approach to planning and coordination. Informally, submodularity expresses the principle of diminishing returns and monotonicity requires functions to be always increasing. Given a set of actions Ω, a set function g maps subsets of

actions (robots' plans) to a real value (the view reward). A set function is monotonic if adding elements to a set does not decrease its value; that is for any $A \subseteq B \subseteq \Omega$ then $g(A) \leq g(B)$. A set function is submodular if marginal gains decrease monotonically; specifically, given $A \subseteq B \subseteq \Omega$ and $C \subseteq \Omega \backslash B$, then $g(A \cup C) - g(A) \geq g(B \cup C) - g(B)$. Objectives related to perception planning [12, 27] and information gathering [45] are often submodular, and this corresponds to how marginal view rewards may diminish given by repeated views of the same actor from the same perspective.

3.3.2 Partition Matroids

A partition matroid can be used to represent product-spaces of actions or trajectories that arise in multi-robot planning problems [43, Sec. 39.4]. Consider a view planning problem involving a team of robots $\mathcal{R} = \{1, \ldots, N\}$ where each robot $i \in \mathcal{R}$ has access to a set of actions $\mathscr{U}_i$. These actions can take many forms such as assignments, trajectories, or paths. The set of all actions for a robot is the ground set $\Omega = \bigcup_{i \in \mathcal{R}} \mathscr{U}_i$. Each robot is assigned one action from its corresponding set $\mathscr{U}_i$. If there are no collisions between robots the set of valid and partial assignments forms a *partition matroid*: $\mathscr{I} = \{X \subseteq \Omega \mid 1 \geq |X \cap \mathscr{U}_i| \ \forall i \in \mathcal{R}\}$. To satisfy this structure each robots' actions must be interchangeable to satisfy the *exchange property* of a matroid. Inter-robot collisions violate this property because swapping actions can cause conflicts with other robots.

3.4 Problem Formulation

We aim to coordinate a team of UAVs to maximize coverage-like view rewards for observing a group of actors moving through an obstacle-dense environment. Consider a set of actors $\mathcal{A} = \{1, \ldots, N^{\mathrm{a}}\}$ each with a set of faces $\mathcal{F}_a = \{1, \ldots, N_a^{\mathrm{f}}\}$ where $a \in \mathcal{A}$ and a set of robots $\mathcal{R} = \{1, \ldots, N\}$. Also, let all the sets of faces across all the actors be $\mathcal{F} = \{\mathcal{F}_1, ..., \mathcal{F}_{N_t}\}$. Each of the faces, $\mathcal{F}_j : j \in N_t$, are associated with a set of pixel coverage by robots, $\mathcal{P}\mathbf{x} = \{\mathcal{P}\mathbf{x}_{111}, ..., \mathcal{P}\mathbf{x}_{ijk}\}$, where $i \in N_r$, $j \in N_t$, and $k \in N_{j,f}$. All the robots move in a workspace represented as a finite graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, and share a global clock that start at $t = 0$. Vertex set $\mathcal{V}$ represents the set of all the possible locations of i^{th} robot at time t as $v_t^i \in \mathcal{V}$ and the edge set $\mathcal{E}$ as actions

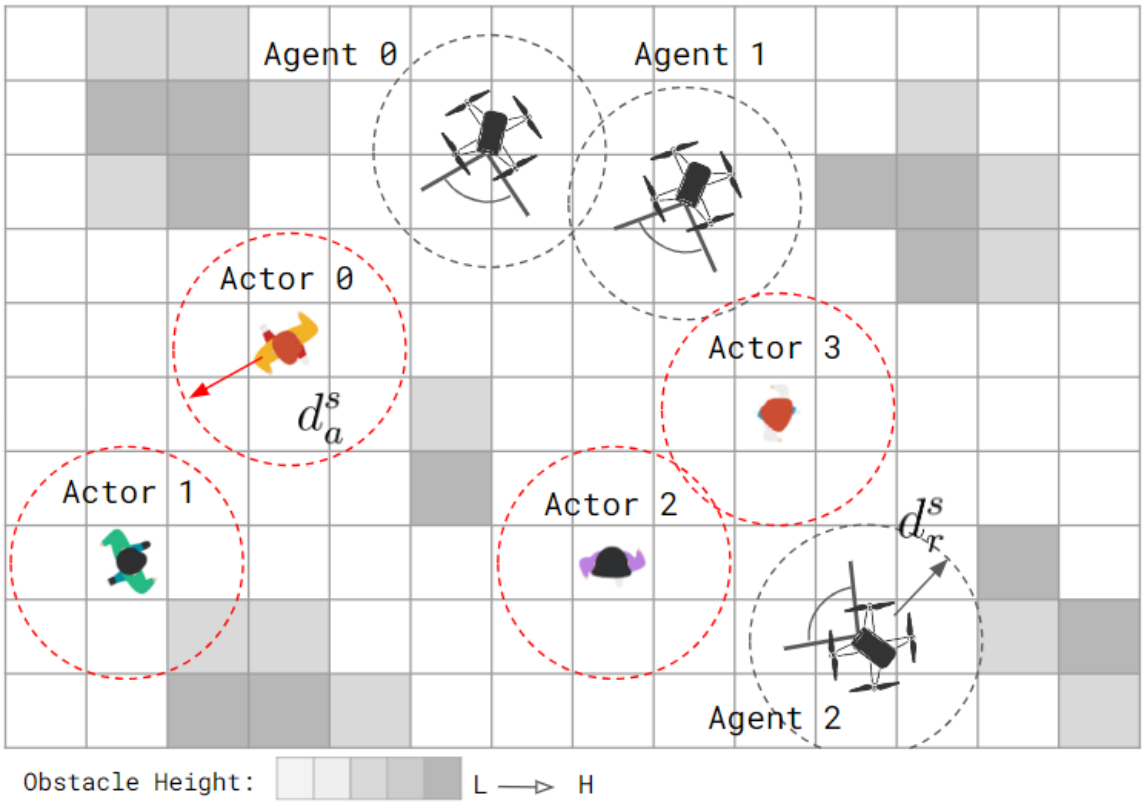


Figure 3.6: Problem representation of the gimbaled camera (also formulated as a robot, $\mathcal{R}$) with projection matrix facilitating coverage on dynamic targets with occlusion and obstacles.

$e_t^i \in \mathcal{E}$ of i^{th} robot at time $t \in \{0, \ldots, T\}$ where $i \in \mathcal{R}$. The reward for each edge is a M-dimensional nonnegative vector reward$(e_t^i) \in \mathbb{R}^{+M} \setminus \{0\}$.

Each robot $i \in \mathcal{R}$ can execute a control action $u_{i,t} \in \mathcal{U}_i \subseteq SE(2)$ at time $t \in \{0, \ldots, T\}$. The robots go on to select a finite-horizon sequence of viable control actions which form their plans. Additionally, robots have associated states $x_{i,t} \in \mathcal{X}$ which is a subset of $SE(2)$. Sequences of states form the robots' trajectories $\xi_i = [x_{i,0}, \ldots, x_{i,T}]$—we will occasionally index trajectories to obtain $\xi_{i,t} = x_{i,t}$. Each trajectory, once fixed, produces non-collision constraints for all other robots.

Let $\xi^i(v_1^i, v_l^i)$ be a path that connects the vertices v_1^i and v_l^i via a sequence of vertices $(v_1^i, v_2^i, ..., v_l^i) \in \mathcal{G}$. Let $g^i(\xi^i(v_1^i, v_l^i))$ denote the M-dimensional reward vector associated with the path $\xi^i(v_1^i, v_l^i)$ as the sum of all the reward vectors of all the edges present in the path, *i.e.*, $g^i(\xi^i(v_1^i, v_l^i)) = \sum_{j=1,\ldots,l-1}(v_j^i, v_{j+1}^i)$.

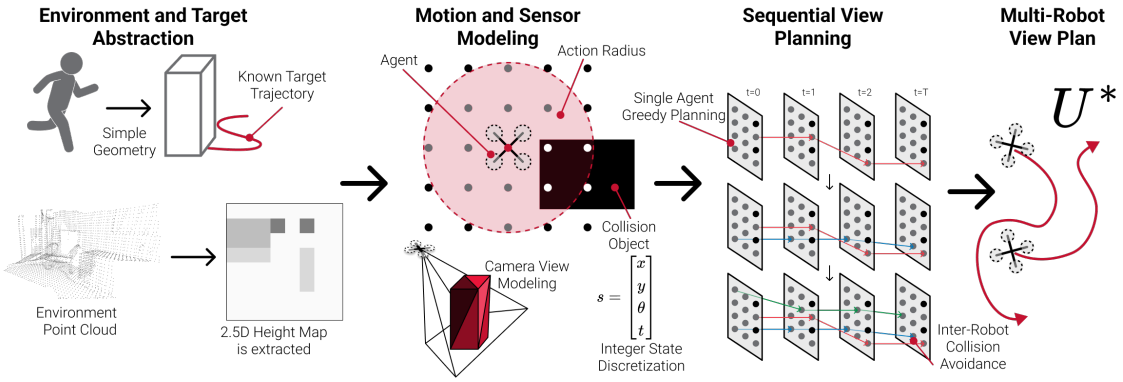


Figure 3.7: **View Planning System Overview** The multi-actor scenario is translated to an internal planner representation. The Markov Decision Process with DAG structure encodes collision constraints and view rewards. The Multi-Robot View Plan is produced through sequential greedy planning. Joint multi-drone trajectories are brought back to continuous 3D coordinates and output to the navigation stack.

As each robot i is associated with a safety boundary with radius s^{ia}, the set of all the vertices inside this boundary at time t be denoted as S_t^{ia}. Similarly, i^{th} actor set of vertices under its safety boundary is denoted as S_t^{it} where $i \in \mathcal{T}$, making the available states space for robots as $\mathcal{V} \setminus S_t^{it}$. A conflict between robots, $(i, j) \in \mathcal{T}$, called a "Breach" would be defined when $||(v_t^i - v_t^j)|| \leq s^{ia}$ where $v_t^i \in S_t^{ia}$ and $v_t^j \in \mathcal{V}_t^j : j \in \mathcal{R} \setminus \{i\}$.

Given the trajectories of all actors in $SE(3)$, start states $x_{i,0}$, and environment geometry, we aim to find joint collision-free control sequences $U^* = \bigcup_{i \in \mathcal{R}} [u_{i,0}, \ldots, u_{i,T}]$ that maximize our objective and fit our motion model.

[1]

3.4.1 Motion Model

State transitions for each robot are specified by the following motion model

$$x_{i,t+1} = f_i(x_{i,t}, u_{i,t}) \tag{3.1}$$

where f_i is defined to only allow collision-free motions to positions and orientations within a constant velocity constraint. Given the time step duration, maximum

[1]Please note that camera, agent, robot, and drone could be used interchangeably.

translational and rotational velocities are converted to bounds on rotation and Euclidean distance as illustrated by Figure 3.7.

3.4.2 Non-Collision Constraints

Robots are considered in collision with the environment when the discretized state location is occupied by an element of the environment map that exceeds the robot's height. Similarly, a pair of robots is in collision when both occupy the same discretized cell at the same time. We implicitly assume a conservative model of the environment (the heightmap and discretization of the state space) to ensure safety of states that satisfy the obstacle and inter-robot collision constraints.

3.4.3 Camera and View Reward Model

We use a coverage-like reward based on pixel densities as a proxy for effectively observing an actor. Inspired by [22], we compute rewards based on cumulative pixel densities ($\frac{px}{m^2}$) for observations of faces from polyhedral representations of each actor j. We define a function $\texttt{pixels}(x_{i,t}, t, j, f) \to \mathbb{R}$ which returns the pixel density for actor-j's face f when observed from a robot's state at time t.

In order to encourage robots to assume views that uniformly cover the actors and their corresponding faces, we apply a square root to introduce diminishing returns on increasing pixel densities from multiple views [20, 21]. Finally, given robot trajectories according to the dynamics equation 3.1 and selected control inputs, the robots obtain the following view reward for the given face and time:

$$R_{\mathrm{v}}(t, j, f) = \sqrt{\sum\nolimits_{i \in \mathcal{R}} \texttt{pixels}(x_{j,t}, t, j, f)}. \tag{3.2}$$

The formal statement of monotonicity and submodularity properties for rewards of this form and the relationship to coverage are the subject of [21]. Intuitively, equation 3.2 obtains these desirable properties because the square-root is one of many real functions that increases monotonically but at ever-decreasing rates.[2] If

[2]In fact, any other real function with similar monotonicity properties other than the square-root would satisfy the requirements of submodularity and monotonicity. However, we will not focus on the choice amongst such functions in this work.

not constrained through the selection of camera views, the submodularity of our objective based on equation 3.2 would ensure that rewards would be maximized by distributing all pixels approximately uniformly over the faces of the actor models. Likewise, submodularity due to the square-root encourages robots to distribute their views evenly across the actors and their surfaces. By contrast, summing pixel densities without the square-root would assign the same reward for distributing all pixels on one face or for distributing pixels uniformly. Our approach also allows for more variation in rewards than for simply thresholding on range or pixel density.

3.4.4 Objective

In addition to maximizing the view reward, we add a reward for stationary behavior $R_s(u_{i,t})$ to reduce unnecessary movement whereas

$$R_s(u) = \begin{cases} \epsilon & \text{if } u \text{ is stationary} \\ 0 & \text{otherwise.} \end{cases} \tag{3.3}$$

So, robot $i \in \mathcal{R}$ obtains $\sum_{t\in\{0,\dots,T\}} R_s(u_{i,t})$ reward for timesteps it remains stationary. The joint objective is then as follows:

$$\mathcal{J}_3(X_{\texttt{init}}, U) = \sum_{t\in\{0,\dots,T\}} \left(\sum_{i\in\mathcal{R}} R_{\text{s}}(u_{i,t}) + \sum_{j\in\mathcal{A}} \sum_{f\in\mathcal{F}_j} R_{\text{v}}(t,j,f)) \right) \tag{3.4}$$

where $X_{\texttt{init}} = [x_0, \dots, x_N]$ is an array of initial robot states and U represents the robots' sequences of control actions. Since we aim to find the control sequence that maximizes this objective, our optimal control sequence can be defined as:

$$U^* = \underset{U}{\operatorname{argmax}}\, \mathcal{J}(X_{\texttt{init}}, U) \tag{3.5}$$

3.5 Single Robot Multi-Actor View Planning

We now present our single robot view planning approach. The planner aims not only to produce sufficient coverage over the actors but also to exploit problem structure to efficiently find single-robot trajectories by greedy planning.

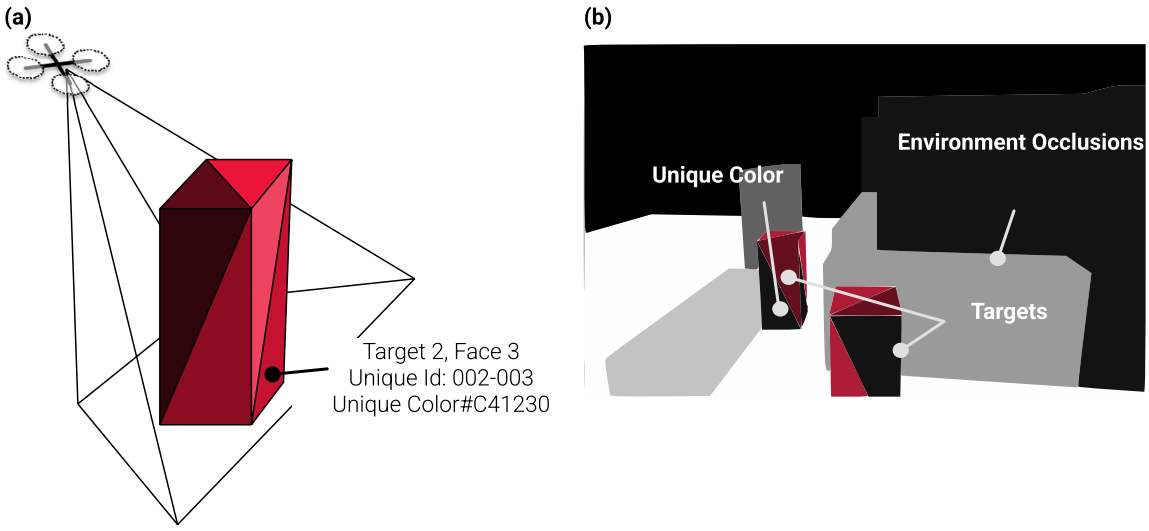


Figure 3.8: Actor Coverage **(a)** UAV camera model frustum observing a simplified actor geometry. Actor faces are colored slightly differently based on a face identification system to allow for pixel density computation. **(b)** Example camera output from OpenGL internal rendering system.

3.5.1 Evaluation of View Reward

To produce an occlusion-aware implementation of our view reward equation 3.2 we compute `pixels` by implementing an OpenGL rasterization renderer based on a 2.5D height map of the environment and simplified actor geometry—we use polygonal cylinders. We then use a perspective camera based on specified camera intrinsics to capture an occlusion-aware representation of the scene from a given robot state. The system renders the environment via the GPU with a geometry shader to draw the heightmap. To determine how many pixels the cameras observe for each face, we render the faces with unique colors associated with the actor and face IDs. Counting pixels of each color and dividing by the areas of the faces yields the corresponding pixel densities. Figure 3.8 illustrates this process and provides an example of a rendered view.

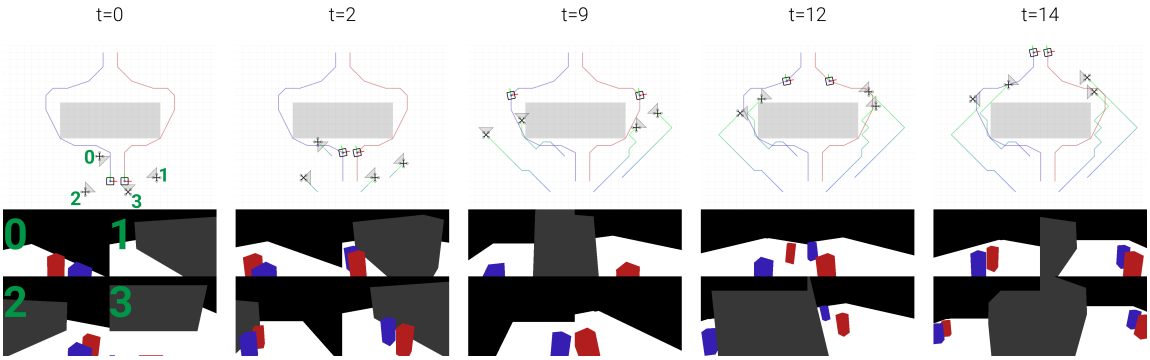


Figure 3.9: Example robot views and joint trajectories from sequential plan in `Split` test case. Robot first-person views at each time step display viewing of the actors over the planning horizon.

3.5.2 Single-Agent Value Iteration Planner

With the robot state in $SE(2)$ we aim to represent the single-robot planning problem as a Markov Decision Process (MDP) which has an underlying Directed Acyclic Graph (DAG) structure. We use the AI-ToolBox library to represent and solve the MDP [4]. The MDP state s is represented as an integer vector:

$$s = \begin{bmatrix} x & y & \theta & t \end{bmatrix}$$

Each MDP action a is in the same discrete space, encoding the next state and incrementing the time by 1. This forces the MDP structure to be directed since states can never go back in time. The MDP is constructed with a transition matrix associating each (s, a, s') tuple with a transition probability and a reward matrix associating each (s, a) pair with a reward according to equation 3.4 with knowledge other robots' actions (to be introduced in Section 3.6). We perform a breadth-first search over the state space by branching on feasible actions to populate the transition and reward matrices. As depicted in Figure 3.7, the set of available actions is pruned based on environmental and inter-robot collisions. This directed MDP can be solved with one backward pass of value iteration to find the optimal greedy policy,[3] similar to the approach by Bucker et al. [8]. Following this policy from the initial state produces

[3]Our current implementation converges in 5 passes without exploiting this structure.

the optimal single-robot control sequence. This method is mostly used for calculating the optimal view positions and is done offline due to computation requirement.

3.5.3 Single-Agent View Search

Algorithm 1 Single-Agent View Search

Data: ξ_0^i, $\mathcal{T}$ trajectories , envHeightMap(H), agentMaxMotion(m^i), $\mathcal{P}\mathbf{x}_i$, Ω_i
Result: ξ^i
Initialize $\xi^i \leftarrow \{\}$
TREE $\leftarrow \{\mathbf{R}_0, \xi_0^i, e_0^i, t\}$
while TREE.top().t is not T **do**
    $Q_k \leftarrow (\overline{\mathbf{R}_k}, \xi_k^i, e_{k-1}^i, k)$ ▷ TREE.pop()
    updateProcessedStates[ξ_k^i] = true
    updateStatesFromPrevAction(Q_k)
    $e_{k_1}^i, ..., e_{k_{ST}}^i \leftarrow availableActions(\xi_k^i, k, \Omega_i, H, m^i)$

  for all $e_{k_1}^i, ..., e_{k_{ST}}^i$ **do**
    $\overline{\mathbf{R}_{k+1}} \leftarrow (\mathbf{R}_{k+1} + \overline{\mathbf{R}_k})\gamma^{k-1}$
      ▷ // Cumulative reward x discount factor (< 1) for encouraging to reach reward states in lesser timesteps
    $Q_{k+1} \leftarrow \{(\overline{\mathbf{R}_{k+1}}, \xi_{k+1}^i, e_{k_n}^i, k+1\}$

    if $processedStates[\xi_{k+1}^i] == true \quad || \quad \overline{\mathbf{R}_{k+1}} < \text{TREE}[\xi_{k+1}^i].\mathbf{R}$ **then**
      continue
    end
    TREE $\leftarrow Q_{k+1}$
$\xi^i \leftarrow BackTracking(processedStates)$
return ξ^i

Here we will discuss a single-agent view search technique that is developed using $A*$, however, using perceptual metric as a heuristics to form a max heap that explores the next node in the max heap that contains the maximum perceptual reward as described in 1. The algorithm starts by exploring the start position of the camera and adds all the nodes the TREE that has a valid action to it. However, the reward is added multiplied with a discount factor to value near rewards more than later in the timestep, which also encourages finding max reward in shorter timesteps. Using this new reward, we create a new P_{k+1} and check if it's in the TREE, if its not in the TREE,

or its new reward is lesser than that from the one that is already present in the `TREE`, we skip adding the Pto the `TREE`, else we add. At the end the trajectory is computed through backtracking from the last state.

3.6 Sequential Multi-Agent View Planning

3.6.1 Algorithm

Algorithm 2 Sequential Greedy View Planning

Initialize $U_{\text{seq}} \leftarrow \{\}$
Initialize collisionMap $\leftarrow \{\}$
for each i in $\mathcal{R}$
    $S_i \leftarrow$ DiscretizeStateSpace(envHeightMap)
    $A_i \leftarrow$ DiscretizeActionSpace(robotMaxMotion)
    MDP $\leftarrow$ BreadthFirstSearch($x_{i,0}$, S_i, A_i)
      In BFS, R_{v} is computed at each explored state. Branching through `availableActions` removes actions that lead to collision states.
    $\pi_i \leftarrow$ ValueIteration(MDP)
    $\{u_{i,0}, \ldots, u_{i,T}\} \leftarrow$ ExtractTrajectory(π_i)
    Append $\{u_{i,0}, \ldots, u_{i,T}\}$ to U_{seq}
    $\xi_i \leftarrow$ applyActions($x_{i,0}$, $\{u_{i,0}, \ldots, u_{i,T}\}$)
    addCollisions(ξ_i)
return U_{seq}

Now, we are able to generate the joint view plans for the multi-robot team. We do so by sequentially planning greedy single-robot trajectories as is common for methods based on submodular optimization [8, 9, 11, 13, 27, 33, 42, 45]. With some abuse of notation, each robot maximizes the objective for itself with access to prior decisions in the sequence:

$$U_i = \underset{U}{\operatorname{argmax}}\, \mathcal{J}(X_{\texttt{init},1:i}, U_{1:i-1} \cup U) \tag{3.6}$$

This forms a series of single-robot subproblems that we solve with the value iteration planner (Sec. 3.5.2). Through the course of this process, we accumulate pixel densities per each face to evaluate the view reward equation 3.2 and filter out states that would produce collisions with other robots (Sec. 3.4.2).

3.6.2 Suboptimality guarantees and inter-robot collisions

If we ignore inter-robot collisions, equation 3.5 has the form of a submodular maximization problem with a partition matroid constraint. Thanks to the famous result by Fisher et al. [16], sequential greedy planning via equation 3.6 is guaranteed to produce solutions to equation 3.5 with objective values no worse than 50% of optimal. However, inter-robot collisions violate the form of a partition matroid [11] so solutions that incorporate inter-robot non-collision constraints will not satisfy this guarantee. However, if the operating environment is not congested with robots, these non-collision constraints may not significantly influence the view rewards in practice. Our simulation results in Section 3.9.3 support this conclusion that inter-robot constraints have negligible impacts on solution quality while *averting the serious consequences of collision.*

3.6.3 Time Scaling Analysis

- Value iteration runs in a single pass; once per robot; proportional to size of environment
- Computation time for rendering may be difficult to account for (depends on scene, number of actors)
- Collision checks
- Computing context: (quadratic term) We render prior robots' trajectories

This section seeks to clarify the computational cost of Algorithm 2. After instantiating the MDP, value iteration for a single robot runs (ideally) with a single backwards pass over the reachable states. For planar motion the number of reachable states at step t is $O(t^2)$, and the total number of states over a horizon of T steps is $O(T^3)$. Inter-robot collision checking involves computation for each prior robot at every state, and for a single robot requires $O(T^3N)$ time. The total time for the entire sequential planning process is then $O(T^3N^2)$. This addresses number of robots but not necessarily increasing problem scale in terms of the number of actors or environment complexity. Incorporating larger environments or more actors introduces more nuance. For example, evaluating the objective equation 3.4 is at least proportional to N^{a} but may be larger depending on the cost of rendering scenes

with different numbers of actors. So, we can also say that the computation time scales as $\Omega(T^3 N^2 N^{\mathrm{a}})$.

3.7 Coordinated Multi-Agent View Planning

3.7.1 Brief Overview on Conflict-Based Search

Conflict-Based Search [44] is a two-level search that creates a binary tree to resolve conflicts in the agent paths on a high level and runs an optimal low-level search algorithm for a MAPF problem.

Conflict Resolution: Consider a pair of agents, i and j, each following their respective paths ξ^i and ξ^j. To detect conflicts between these paths, we utilize the function $\Psi(\xi^i, \xi^j)$. This function returns either an empty set if no conflict is present, or it provides details about the first conflict encountered along the paths. A conflict occurs at time t when agent i is at position v_t^i and agent j is at position v_t^j. This conflict is denoted by (i, j, v_t^i, v_t^j, t). To prevent such conflicts, a corresponding constraint is added to the path of either agent i or agent j. This constraint is represented as $\omega^i = (i, u_a^i, u_b^i, t)$, where u_a^i and u_b^i are selected from the set of possible locations V. This constraint is associated with agent i and ensures that at time t, agent i follows the specified path, thus avoiding the potential conflict.

Given a set of constraints Ω, let $\Omega_i \subseteq \Omega$ represent the subset of all constraints in Ω that belong to agent i (i.e., $\Omega = \bigcup_{i \in N_r} \Omega_i$). A path ξ_i is consistent with respect to Ω if ξ_i satisfies every constraint in Ω_i. A joint path ξ is consistent with respect to Ω if every individual path $\xi_i \in \xi$ is consistent.

Two level Search: In the high-level while creating the constraint tree with each node containing $(\xi, \Omega, g(\xi))$, each of them are in a priority queue. Initially, parent node $\mathcal{P}\mathbf{x}_0 \in \texttt{TREE}$ is computed for all the agents such that the constraint set, $\Omega_o = \phi$, and $P_o = (\xi_0, \Omega_o, g_o)$ gets pushed to TREE. Let, total number of nodes created be denoted as N_G, and total number of nodes expanded as $N_E : N_E \leq N_G$.

As each node, k, in the TREE gets explored based on the least g value, $P_k = (\xi_k, \Omega_k, g_k)$, $\Psi(\xi_k^i, \xi_k^j)$ gets called for all the sequence pairs $(i, j) \in N_r : (i \neq j)$. If there is no conflict detected, the solution, ξ, is found and the algorithm terminates. If there is a conflict detected, (i, j, v_t^i, v_t^j, t), constraints to the agents are created in

the following way. Each condition is created where constraint is added to either i or j, such that $\omega^i = (i, u_a^i = v_t^i, u_b^i = v_t^j, t)$ and $\omega^j = (j, u_a^j = v_t^j, u_b^j = v_t^i, t)$. Each of these constraints leads to a new node in the TREE, where $P_{l^i} = (\xi_{l^i}, \Omega_{l^i}, g_{l^i})$, and $P_{l^j} = (\xi_{l^j}, \Omega_{l^j}, g_{l^j})$, such that $l^i \in N_G$ where new set of constraints is added to i or j. Here, $\Omega_{l^i} = \Omega_k \cup \omega^i$, and $\Omega_{l^j} = \Omega_k \cup \omega^j$. In each of these cases, for agent i, the algorithm updates the path ξ_k^i in ξ_{l^i} using the low-level search with a set of constraints Ω_{l^i}. A similar call is made for agent j leading to ξ_k^j. If the low-level search is unable to find a solution for any of these cases, respective P_{l^n} is discarded.

CBS solves a single objective cost function, g, optimally by iterative expansion of a node in the constraint tree based on the least g val, resolving first agent-agent conflict (if exists) and creating a new node in the tree with new Ω, or returning the solution $P = (\xi, \Omega, g)$.

Algorithm 3 Coordinated View Planner

Data: $X_{\texttt{init}}$, $\mathcal{T}$ trajectories , envHeightMap, agentMaxMotion
Result: $U^\dagger$
Initialization()
TREE $\leftarrow$ *GreedyPlanningWithoutConstraints*()
Initialize $U_{\text{seq}} \leftarrow \{\}$
while TREE not empty **do**
  $P_k \leftarrow (\xi_3 = \{\xi^1, \xi^{2*}\}, \Omega_3 = \ \omega_2\}, \mathcal{P}\mathtt{x}_3, \mathrm{g}_1)$ ▷ // TREE.pop()
  if no breach detected in ξ_k **then**
    return P_k
  $\Omega \leftarrow$ Split detected breach
  for all $\omega_i \in \Omega$ **do**
    $\Omega_{kj} \leftarrow \Omega_k \cup \{\omega_j\}$
    $\mathcal{P}\mathtt{x}_l \leftarrow \mathcal{P}\mathtt{x} \setminus \{\mathcal{P}\mathtt{x}_{itk}\}$
    $\xi^{i*} \leftarrow$ LowLevelViewSearch$(i, \Omega_l, \mathcal{P}\mathtt{x}_l)$
    $\mathcal{P}\mathtt{x}_l^* \leftarrow \mathcal{P}\mathtt{x}_l \cup \mathit{GetCoverage}(\xi^{i*})$
    $\xi_l^* \leftarrow \xi_l \setminus \{\xi^i\} \cup \{\xi^{i*}\}$
    $g_l \leftarrow \mathit{GetObjectiveValue}(\xi_l^*, \mathcal{P}\mathtt{x}_l^*)$
    $P_l \leftarrow \{\xi_l^*, \Omega_l, \mathcal{P}\mathtt{x}_l^*, \mathrm{g}_l\}$ TREE $\leftarrow P_l$
  return U_{seq}

3.7.2 Coordinated View Planner

Our proposed planner Algorithm 3 also works in a similar structure to CBS. With the following key differences:

- **Greedy planning**: Rather than initializing with solutions to single-agent path-finding problems we initialize with an approximate solution to the joint multi-agent view planning problem that relaxes constraints between robots.
- **Tree Node($\mathbf{P}_k$)**: The node contains additional information on the pixels coverage by the agents over the faces of corresponding targets. This is given by $P = (\xi, \Omega, \mathcal{P}\mathtt{x}, \mathrm{g})$.
- **Cost(g)**: The cost, g, consists of a objective cost function that is given by eq. 3.6, i.e.,
- **Termination criteria**: Taking computational expense into consideration, we return the first set of the solution P without any conflicts. If there is no valid solution, a failure is reported.
- **Replanning**: As a new conflict between any two agents (i, j) is detected using Ψ, the low level plans for agent i, with constraint set Ω_{l^i} and $\mathcal{P}\mathtt{x} \setminus \{\mathcal{P}\mathtt{x}_{ijk}\}$: $j \in N_t$, and $k \in N_{j,f}$. Similarly, for agent "j".
- **Low-level search**: We want a low-level solver that considers perceptual heuristics. Here, we use a value iteration solver at the low level, whose reward structure considers coverage over targets, and stationarity. All of which is an important criteria for a single-agent view planner. We further present a single agent view search that contains perceptual coverage as a heuristics as presented in Algorithm 1.

3.7.3 Constraint Tree Formation

Initialization: During this state, with $(\Omega_o = \phi, \mathcal{P}\mathtt{x}_o = \phi)$, as we plan for a series of agents, $i \in$, we plan each agent greedily towards maximizing its reward using the single-agent view planner to produce the set of trajectories ξ_o. The root node, $P_o = \{\xi_o, \Omega_o, \mathcal{P}\mathtt{x}_o, \mathrm{g}_o\}$ gets added to `TREE`. As each agent, j, gets planned, the pixel context as passed such that $\mathcal{P}\mathtt{x}_j = \{\mathcal{P}\mathtt{x}_{jkl}\} : j < i, k \in \mathcal{T}, l \in N_{k,f}$.

Finding a solution: As node is popped from the TREE, $P_k = \{\xi_k, \Omega_k, \mathcal{P}\mathtt{x}_k, \mathrm{g}_k\}$. Ψ checks for all the breaches amongst all the sequence pair of agents $(i, j) : i, j \in N_r$, and $i \neq j$. If there are no breaches found, such that $||(v_t^i - v_t^j)|| > s^{ia}$ where $v_t^i \in S_t^{ia}$ and $v_t^j \in \mathcal{V}_t^j : j \in \mathcal{R} \setminus \{i\}$, P is the solution, and the high-level search terminates. There are multiple conflict-free solutions to a set of valid trajectories maximizing g, however, we are interested in the first solution for computational performance and guide the tree expansion by the objective function g= $J(\Xi)$.

Resolving conflicts: If there exists a breach between any two agent, $(i, j) : i \neq j$, and $||(v_t^i - v_t^j)|| \leq s^{ia}$, then agent i is re-planned with constraint set $\Omega_{l^i} = \Omega_k \cup S_t^{ja}$ and $\mathcal{P}\mathtt{x} \setminus \{\mathcal{P}\mathtt{x}_{itk}\} : t \in \mathcal{T}$, and $k \in N_{t,f}$, and agent j is re-planned with constraint set $\Omega_{l^j} = \Omega_k \cup S_t^{ia}$ and $\mathcal{P}\mathtt{x} \setminus \{\mathcal{P}\mathtt{x}_{jtk}\} : t \in N_t$, and $k \in N_{t,f}$. The output trajectory of each agent "i" from the low-level solver is updated to $(\xi_{l^i}^i)$, and pixel coverage set to $\mathcal{P}\mathtt{x}_{l^i}$, such that $P_{l^i} = (\xi_{l^i}, \Omega_{l^i}, \mathcal{P}\mathtt{x}_{l^i}, g_{l^i})$, where g_{l^i} is calculated with the updated set of Ω_{l^i} and $\mathcal{P}\mathtt{x}_{l^i}$. Similarly, for agent "$j$", new constraints and pixel-set leads to $P_{l^j} = (\xi_{l^j}, \Omega_{l^j}, \mathcal{P}\mathtt{x}_{l^j}, g_{l^j})$. P_{l^i}, and P_{l^j} are added to the TREE with their corresponding g, and the tree is rearranged such that the root node has max(g= $J(\Xi)$).

3.8 Considerations for application to real systems

Now, let us discuss how the proposed approach could be adapted for implementation on physical robots. First, our approach relies on known or predicted actor trajectories. This is reasonable for a scripted sequence such as when filming a movie—in this case, offline planning may also be reasonable. However, to account for uncertainty in actor or robot motions, systems should employ receding-horizon planning; though practical application would require substantial improvement in planning time. Then, for outdoor operation, GPS is often sufficient for localizing robots [7, 19, 41], particularly with RTK systems. Although prior works have implemented visual tracking for a single actor [6], additional instrumentation on the actors (such as additional GPS units) may be preferable for multi-actor settings. Then, while long-horizon prediction may be challenging, a Kalman filter can provide predictions over a short horizon [6] based on velocity and orientation. Regarding the map, if robots operate frequently in the same location (like a sports arena), a pre-existing map along with local collision avoidance may be appropriate.

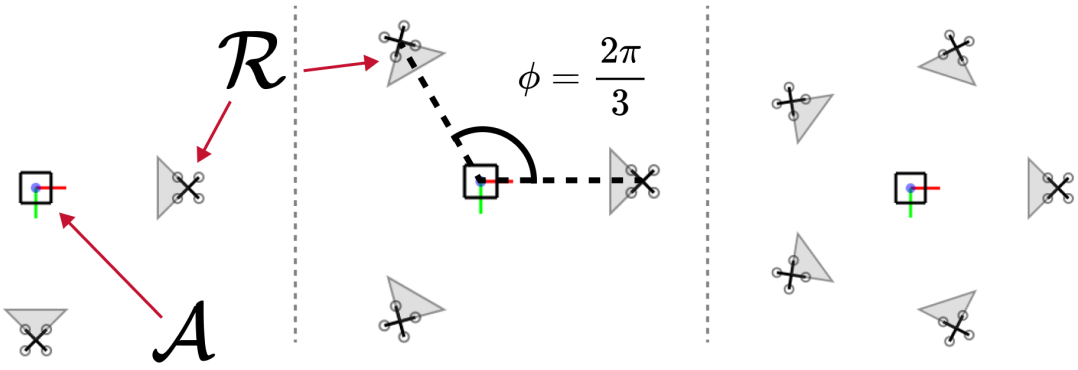


Figure 3.10: Multi-robot formations for $N = 2, 3, 5$

3.9 Experiments with Sequential MAPF

We evaluate the performance of the sequential view planner in five test scenarios that aim to demonstrate view planning under a variety of conditions, and we compare to a formation planning baseline.

3.9.1 Formation Planning

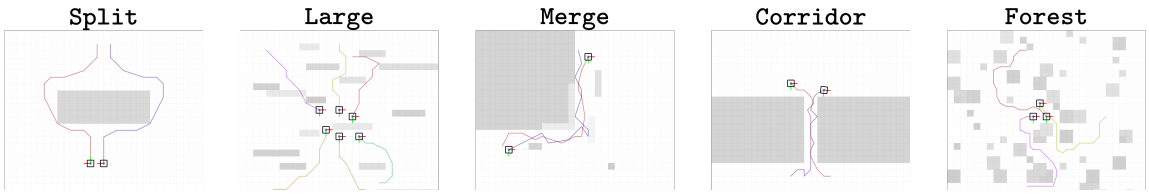


Figure 3.11: Scenarios to evaluate specific aspects of multi-actor view planning. Actors are illustrated as boxes with uniquely colored trajectories. The darkness of elements in the height map indicates their occupied height. `Large` is the only test case with no collision obstacles as all elements are below the robot operating plane.

We compare our sequential view planner against an assignment and formation based planner which we model off the multi-view formations applied in [19, 48] following analysis by Bishop et al. [5]. We assign equal numbers of robots to each actor, and groups assume formations as follows. The formation has a constant radius around an actor and a separation angle ϕ; for $N > 2$ then $\phi = \frac{2\pi}{N}$ and when $N = 2$

Test Name	# Robot	# Actors	Timesteps	Env Collision
`Merge`	4	2	17	Yes
`Corridor`	2	2	17	Yes
`Forest`	2*	3	20	Yes
`Large`	18	6	10	No
`Split`	4	2	15	Yes
`Bottleneck`	4	4	11	Yes

Table 3.1: Test scenarios and parameters. The formation planner obtains an extra robot in the `Forest` scenario "for free" to match the number of actors

then $\phi = \frac{\pi}{2}$ (see Figure 3.10). We directly orient the formation about the actor to maximize the view reward R_v (including all actors) at each timestep. The formation planner ignores the motion model (Sec. 3.4.1) and does not consider environment and robot collisions; so view rewards for formation planning are generally optimistic.

3.9.2 Test Scenarios

Test scenarios are detailed in Figure 3.11 and Table 3.1. We use robots with camera intrinsic parameters of 2500px, 4000px, and 3000px (focal length, image width, image height). All drones are placed at 5 meters high with a camera tilt of 10 degrees from the horizon. For each test scenario, we also specify 10 unique robot starting configurations to introduce further variation. The scenarios are as follows:

`Split`: This test case is a simple group split and merge of 2 actors around an obstacle. A full view sequence from an example trajectory is displayed in Figure 3.9.

`Large`: Focuses on scaling to larger teams and features 18 robots. Actors move through a series of short walls that produce occlusions but not collision constraints since they are below the navigation plane.

`Merge`: Contains actors moving around a corner in opposite directions. This test case investigates implicit actor assignment with actors being "handed off" at the corner.

`Corridor`: Tests robots moving through a narrow corridor. This focuses on the collision-aware aspect of the planner.

`Forest`: This is a dense occlusion/collision environment. For this scenario we limit sequential planning to two robots, fewer than the number of actors (three).

Test	Formation	Seq. w/o Inter-Robot		*Sequential*
`Split`	**1380**	1352 ± 34	$r_c = 0.4$	1351 ± 35
`Large`	**1413**	1390 ± 27	$r_c = 10.7$	1381 ± 27
`Merge`	1149	**1275 ± 26**	$r_c = 0.2$	1274 ± 28
`Corridor`	1612	1808 ± 86	$r_c = 0.8$	**1812 ± 85**
`Forest`	2114	**2534 ± 73**	$r_c = 0.2$	2505 ± 116

Table 3.2: Average and standard deviation of view reward (R_v) per robot for all test cases from 10 robot start configurations, comparing baselines to our approach (sequential). For sequential planning without inter-robot constraints, we also report the collision rate r_c (robots collided per trial).

`Bottleneck`: Tests robot in an environment where two narrow corridors merge to form a narrow corridor. This focuses on scenarios with high inter-robot collisions as the corridor merges.

This test aims to demonstrate the capacity to adapt to scenarios where assignments are not possible by evaluating if fewer robots can achieve similar or better coverage compared with the formation planner which requires 3 robots due to the minimum of 1 robot per formation. For the purpose of evaluation, we treat both planners as featuring only two robots when we report *per-robot* results.

3.9.3 Sequential Planner Performance

Figure 3.12 and Table 3.2 summarize planner performance across each of the scenarios in terms of the view reward for formation planning and sequential planning both with and without inter-robot collision constraints. We observe that sequential planning[4] outperforms formation planning in three of five scenarios—by an average of 13.9% in the `Merge`, `Corridor` and `Forest` scenarios. Notably, sequential planning also outperforms formation planning by 18% in the `Forest` scenario despite having one fewer robot. We also observe that inter-robot non-collision constraints do not significantly impair the performance of sequential planning. In the `Split` and `Large` test cases, all planners perform similarly. This may be because these scenarios provide more favorable conditions for the formation planner (and because the formation planner does respect starting positions or robot dynamics). `Split` provides ample space for

[4]Referring to the collision-free version, but both obtain similar performance.

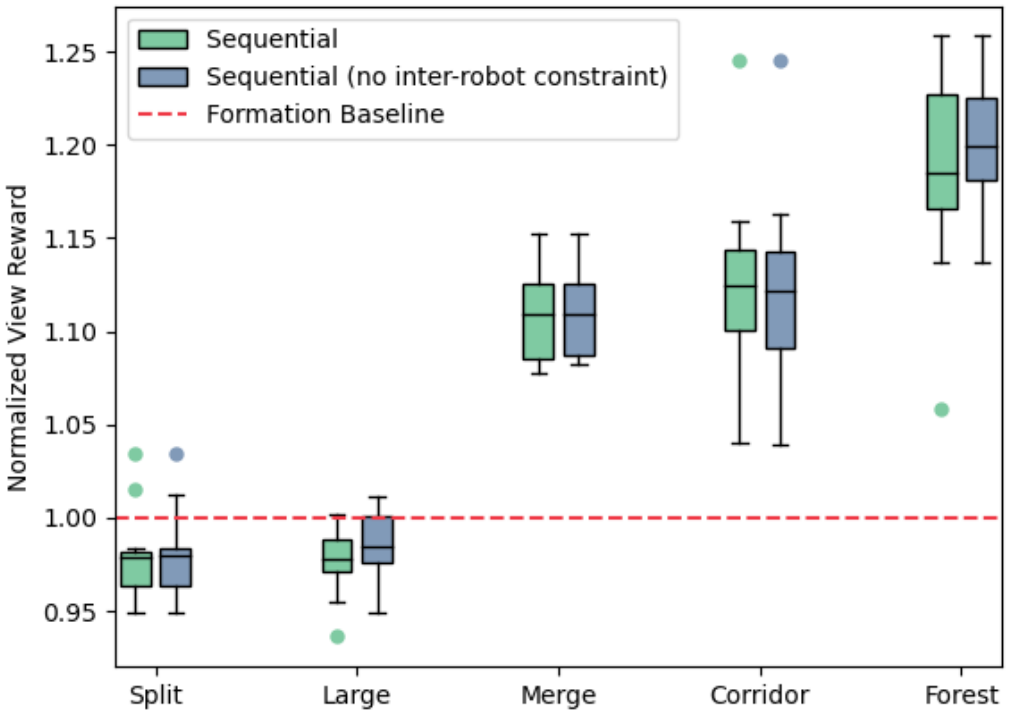


Figure 3.12: Average view reward normalized by the baseline formation planner performance. 10 unique robot start configurations were specified for the sequential planners and are compared with the unique output from the formation planner. Both sequential planners outperform the formation planner baseline in the `Merge`, `Corridor`, and `Forest` scenarios, or else perform similarly. Both versions of the sequential planner perform similarly in all scenarios; view reward for the collision-aware planner is not impaired, though that planner no longer satisfies guarantees on solution quality.

formations of two robots to view the actors, and `Large` has shorter obstacles that produce occlusions but not collisions.

The sequential planner that ignores inter-robot collisions guarantees solutions within half of optimal but may allow robots to collide. Since both versions of the sequential planner obtain similar solution quality (Fig. 3.12), we conclude that inter-robot collision constraints do not significantly impair performance in this setting. We also report collision rates r_c in terms of *robots collided per trial* for the sequential planner that ignores collisions Table 3.2. In most scenarios, we observe less than one collision per trial except for `Large` where ten robots (more than half) collide on average, but we still do not see a significant difference in objective values. The `Corridor` scenario is also highly-constrained, and we see similar objective values despite increased collisions.

Figure 3.9 displays the joint view plans and internal view planner renderings for the `Split` test case. This figure illustrates the capacity of sequential planning to optimize views of one or more actors and to implicitly reconfigure or "hand-off" assignments over the course of a trial. The robots all view both actors at $t = 2$; the pairs split off and transition to each viewing a single actor by $t = 9$; and they go back to jointly viewing actors by $t = 14$. This behavior is not manually specified and arises only from optimizing trajectories and views.

3.10 Experiments with Coordinated MAPF

3.10.1 Ego-Centric Test

In this test, we evaluated scenarios where the agents exhibited non-egoic behavior, such as one robot getting out of the way to allow another robot to pass. The objective was to assess the system's ability to handle interactions and coordination among agents. We also evaluate the constraints developed in the system when comparing the no-inter-robot constraints, while adding inter-robot constraints, and while doing a conflict-based constraint assignment.

Figure 3.13 presents the scaled rewards of three view planning methods in a corridor environment with 8 timesteps, 2 robots, and 2 actors. The robots are randomly initialized outside the corridor while the actors move through it in the

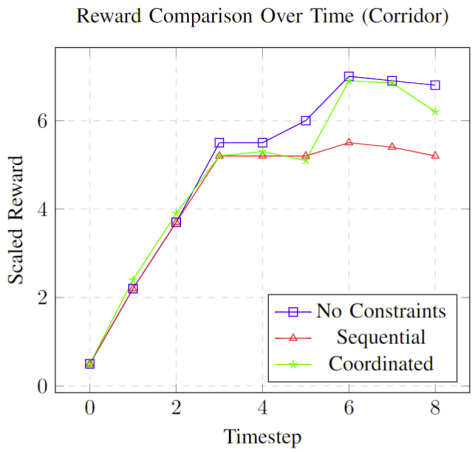


Figure 3.13: Reward comparison amongst different methods in corridor environment.

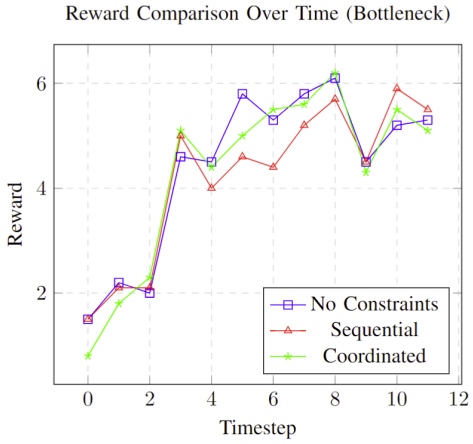


Figure 3.14: Reward comparison amongst different methods in bottleneck environment.

Figure 3.15: Scale rewards using multiple cameras performing view planning using No inter-robot Constraint, Sequential Constraint, and Conflict Based MDP Value Iteration, over total planning horizon for the environment.

same direction. From timesteps 3 to 8, when the robots are inside the corridor, a clear gap emerges between the performance of the proposed sequential planning and planning without system constraints. The proposed Algorithm 3 aims to narrow this gap, bringing performance closer to the unconstrained system.

Similarly, Figure 3.15 compares the three methods in a bottleneck environment over 11 timesteps with 4 robots and 4 actors, where actors from different corridors merge in the bottleneck area. From timesteps 3 to 9, the gap between the sequential planning view reward and planning without inter-robot constraints highlights the system's reward decrease as constraints are introduced. This indicates the need for an intelligent approach to adding constraints without significantly reducing the view reward. The proposed coordinated capture effectively bridges this gap, achieving performance comparable to the unconstrained method but with added inter-robot constraints.

Together, these two scenarios with high obstacles and occlusions demonstrate the effectiveness of the proposed methods in maintaining high-view rewards despite added constraints.

Method	Total Reward (scaled)	Compute Time (s)
Sequential w/ Value Itr.	4127	482
Coordinated w/ Value Itr.	**4662**	3981
Coordinated w/ View Search	3922	**2.7**

Table 3.3: Comparison of Total Reward and Compute Time across different methods for corridor.

3.10.2 Single-Agent Value Iteration vs Search

For offline planning approaches, the value iteration solver can compute sub-optimal view positions to achieve high view rewards. However, in more online settings, there is a need for more efficient algorithms. The proposed single-agent view search, described in Algorithm 1, aims to address these use cases.

The Table 3.3 compares the total reward (scaled) and compute time (in seconds) for different view planning methods in a corridor environment. The Sequential method with Value Iteration achieved a total reward of 4127 and a compute time of 482 seconds. The Coordinated method with Value Iteration yielded the highest total reward of 4662 but required significantly more compute time, at 3981 seconds. In contrast, the Coordinated method with View Search had a lower total reward of 3922 but was the fastest, with a compute time of only 2.7 seconds. This highlights a trade-off between achieving higher rewards and compute efficiency, showcasing the effectiveness of using single-agent view search in online settings.

3.11 Conclusion and limitation

In conclusion, this chapter addresses the intricate challenge of 4D reconstruction in complex, obstructed environments using a fleet of UAVs. By treating targets as dynamic groups, we emphasize the importance of comprehensive coverage through collision avoidance and perceptual redundancy. In this work, we presented a novel system for multi-robot view planning based on sequential greedy planning Section 3.6 and coordinated view planning Section 3.7, inspired from Conflict-Based Search (CBS) with an occlusion-aware objective. Through the evaluation in five different scenarios, for sequential planning, we observe sequential planning outperforming formation-

based planning and specifically excels in obstacle-dense environments. Additionally, we observe similar perception performance for sequential planning with and without inter-robot collision constraints. This demonstrates that sequential planning can find good solutions when accounting for possible collisions between robots (despite no longer satisfying requirements for bounded suboptimality). Moreover, we also test coordinated view planning in two environments that requires constraint reasoning and compared with sequential greedy planning and planning with no constraints. we see that coordinated view planning performs close to case where there is no collision constraints to the system. We further present a single-agent view search approach as compared to the single-agent value iteration solver which shows the computational performance boost.

The system for view planning of cameras over moving actors, while foundational, has several limitations. The world is constructed from a 2.5D height map, lacking the ability to reason about overhanging structures like cantilevers and manage coverage when subjects are at doors or windows. The geometric representation of actors as cuboids fails to capture intricate details adequately. The system has not been validated through deployment on an actual multi-UAV system, where communication challenges with the centralized planning approach could arise. Furthermore, the coordinated view planning requires additional testing in scenarios with increased obstacle and occlusion density. The search approach prioritizes the nearest reward and does not account for exploration in its heuristics, necessitating more experiments, particularly in scenarios with high rewards later in the view planning process. Finally, for deployment on drones, communication reliability must be ensured due to the system's centralized approach.

Acknowledgment and contribution

The Sequential and Single-Agent Value Iteration Planner, entitled "Greedy Perspectives: Multi-Drone View Planning for Collaborative Perception in Cluttered Environments" has been accepted to IROS 2024. Following are the author contributions in the order of authorship:

Krishna Suresh: He has written major parts of the paper, conducted experiments, and finalized figures. He is also solely responsible for writing the OpenGL code that

uses 3D renderings to compute pixel-level metrics, major parts of planning code, and helper functions.

Aditya Rauniyar: He is responsible for the ideation of the 3D rendering approach and lower-level world representation in 2.5D height map. He is majorly responsible for code architecture, and reviewing code for the planner and reward function. He and Krishna have actively been in co-working with the code performance, and initial draft on the introduction section.

Micah Corah: He is responsible for the overall method used, providing background knowledge and feedback on the types of experiments to run. He is also responsible for reviewing and completing parts of the writing.

Sebastian Scherer: He pushed the team, reviewed the paper, and provided feedback on figures and experiments. He also provided with necessary compute requirements for performing the experiments.

Moreover the work with coordinated view planning and view search has following authors involved:

Aditya Rauniyar: He is responsible for the ideation and code for new approach built on top of greedy perspectives. He is also responsible for writing, test cases and simulation setup for the coordinated approach.

Yuechaun Hou: He has co-developed Figure 3.3, Figure 3.4, and Figure 3.6 with Aditya.

Chapter 4

Conclusion

In conclusion, this thesis argues that view planning is a critical component for efficiently and effectively reconstructing scenes, with a primary focus on outdoor environments and applications in virtual production and computer animation. The work is divided into two main components: (i) performing view planning in unknown 3D scenes as described in Chapter 2, and (ii) view planning using multiple cameras for known 4D scenes as detailed in Chapter 3.

Chapter 2 introduces uncertainty estimation-based next-best view planning, which plans for the next view with a single fixed horizon. This method is important because it bypasses the need to update the 3D map to determine the next view that will increase the map's information, instead using the collected reference images directly. However, this approach has not been developed for large-scale outdoor scenes due to the limited availability of suitable datasets. We utilize existing datasets like UrbanScene3D, commonly used by the large-scale scene reconstruction community, and curate them according to the requirements of these models. For experiments, we break down the uncertainty estimation model to its inspiring model, the Generalizable Novel View Synthesis (GNVS) model like PixelNeRF. We present four data curation methods: sequential grouping, grouping based on camera poses, clustering based on ray-to-ground intersection, and clustering based on a similarity matrix computed using Structure from Motion (SfM). Using a similarity matrix showed the most stable training time on PixelNeRF, making it suitable for uncertainty estimation-based next-best view planning processes for large-scale outdoor scenes.

Chapter 3 starts with the assumption that the static 3D scene and the trajectories of the moving actors are known. It notes that existing methods are not developed for multi-view planning of multiple actors, especially in environments with obstacles and occlusions. The section introduces a reconstruction proxy called *Pixel-Per-Area (PPA)* and a simplistic geometric representation of the actors in the scene. It creates a reward structure that maximizes PPA, ensures view diversity, and penalizes stationarity of the multi-camera system. The method uses 3D renderings in OpenGL to account for obstacles and occlusions, reflecting such attributes by creating a 3D semantic world using a 2.5D height map of the environment, actor geometry, and trajectory. It sequentially plans for multiple drone-mounted cameras, ensuring that collision and view constraints are passed to subsequent drones. Additionally, it presents a coordinated approach that intelligently forms these constraints through minimization. The search algorithm uses discounted rewards as heuristics to guide the view planning process, demonstrating computational efficiency over solving through the formation of a Markov Decision Process (MDP) world and using a value iteration solver.

Together, these methods present novel approaches that enable existing view planning techniques to be applied to large-scale outdoor 3D scene reconstruction and motion capture of actors in environments with significant obstacles and occlusions, requiring multi-view reasoning.

Chapter 5

Future work

As we progress towards developing a comprehensive view planning framework for 4D scene understanding, encompassing both static and moving attributes as discussed in Chapter 2 and Chapter 3, we recognize several requirements. The framework needs a task assignment formulation where multiple camera-equipped drones capture the unknown 3D scene, which is then used to reconstruct a static semantic representation. This representation serves as a foundation for capturing the moving attributes of the scene. These approaches have not yet been integrated such that a camera captures all aspects (static and moving) of the scenes for a complete representation.

In Chapter 2, view planning for unknown 3D scene, we narrow down a related work that uses uncertanity estimation using implicit neural representation select the next best view of a scene from a given collected reference images to Generalizable Nove View Synthesis (GNVS) models to render novel views using references images in large scale outdoor unkown scenes. This chapter extends such methods towards view selection in large-scale outdoor scenes using such scene datasets like UrbanScene3D to fit training pipeline for the GNVS model using various clustering algorithms, out of which using similarity matrix where the distance is calculated using shared tiepoints amongst images performs the best. However, even though, we freeze the encoder weights while training, we also notice that the PixelNeRF architecture contains the CNN encoder with ResNet34 architecture, which possibly can limit the encoder to extract latent feature space of large outdoor scenes, which could be replaced with deeper networks, or quantized versions of vision foundational models like DINOv2.

One another observation was that computing the RGB for a view candidate takes a high amount of MLP calls. As for Next Best View (NBV) planning, we would be required to make predictions for multiple view candidates, which can exponentially increase MLP calls. This can further be optimized by using Gausian Splatting-based techniques that are computationally efficient in this process through rasterization, which would even allow us to make predictions for the higher horizon. Using such would also require an alternative way of uncertainty estimation from the ones discussed under Section 2.1.

With Chapter 3, several aspects need further exploration and development. As the current system is built on a 2.5D height map, it requires a 3D representation to have better perceptual reasoning in the world, this would be a post processing 3D reconstruction using Structure from Motion (SfM) or other such reconstruction technique to generate a .obj such that it can be directly imported in OpenGL for semantic and perceptual reasoning. Improving the geometric representation of actors, from 4 faces to 6 faces has shown to provide better results can should be employed. Moreover, as the trajectory of the actors are considered to be known, there are always cases where an actor has to improvise position from the scripted scenario, in such cases adding human motion prediction algorithm as described in Section 3.8 should be employed. The system must be tested with multiple scenarios with high obstacle and occlusions. Coordinated view planning requires further testing in scenarios with increased obstacle and occlusion density. The search approach needs enhancement to balance between prioritizing the nearest reward and accounting for exploration in its heuristics. This can be done by adding an exploration component in its heuristics.

Moreover, the system should address deployment challenges related to energy efficiency by generating smoother trajectories that account for wind dynamics. Communication among the multi-robot system is critical, requiring high reliability inherent in a centralized planning system. The system also needs to be robust in handling unscripted scenes, adapting to unknown dynamics, and incorporating adaptive horizon uncertainty reasoning in view planning.

Bibliography

[1] Henrik Aanæs, Rasmus Ramsbøl Jensen, George Vogiatzis, Engin Tola, and Anders Bjorholm Dahl. Large-scale data for multiple-view stereopsis. *International Journal of Computer Vision*, pages 1–16, 2016. (document), 2.2

[2] Agisoft LLC. Agisoft Metashape, 2021. URL https://www.agisoft.com/. 2.3

[3] Alfonso Alcántara, Jesús Capitán, Arturo Torres-González, Rita Cunha, and An´Ollero. Autonomous execution of cinematographic shots with multiple drones. *IEEE Access*, 8:201300–201316, 2020. 3.1

[4] Eugenio Bargiacchi, Diederik M. Roijers, and Ann Nowé. AI-Toolbox: A C++ library for reinforcement learning and planning (with Python bindings). *Journal of Machine Learning Research*, 21(102):1–12, 2020. 3.5.2

[5] Adrian N Bishop, Barış Fidan, Brian DO Anderson, Kutluyıl Doğançay, and Pubudu N Pathirana. Optimality analysis of sensor-target localization geometries. *Automatica*, 46(3):479–492, 2010. 3.9.1

[6] Rogerio Bonatti, Cherie Ho, Wenshan Wang, Sanjiban Choudhury, and Sebastian Scherer. Towards a robust aerial cinematography platform: Localizing and tracking moving targets in unstructured environments. Macau, China, November 2019. 3.8

[7] Rogerio Bonatti, Wenshan Wang, Cherie Ho, Aayush Ahuja, Mirko Gschwindt, Efe Camci, Erdal Kayacan, Sanjiban Choudhury, and Sebastian Scherer. Autonomous aerial cinematography in unstructured environments with learned artistic decision-making. 37(4):606–641, 2020. 3.8

[8] Arthur Bucker, Rogerio Bonatti, and Sebastian Scherer. Do you see what I see? Coordinating multiple aerial cameras for robot cinematography. Xi'an, China, May 2021. 3.1, 3.1, 3.1, 3.2.3, 3.5.2, 3.6.1

[9] Xiaoyi Cai, Brent Schlotfeldt, Kasra Khosoussi, Nikolay Atanasov, George J Pappas, and Jonathan P How. Energy-aware, collision-free information gathering for heterogeneous robot teams. 39:2585–2602, 2023. 3.1, 3.1, 3.2.3, 3.6.1

[10] Micah Corah. On performance impacts of coordination via submodular maxi-

mization for multi-robot perception planning and the dynamics of target coverage and cinematography. In *RSS Workshop on Envisioning an Infrastructure for Multi-robot and Collaborative Autonomy Testing and Evaluation*, 2022. 3.2.3

[11] Micah Corah and Nathan Michael. Distributed matroid-constrained submodular maximization for multi-robot exploration: theory and practice. 43(2):485–501, 2019. 3.1, 3.1, 3.2.3, 3.6.1, 3.6.2

[12] Micah Corah and Nathan Michael. Volumetric objectives for multi-robot exploration of three-dimensional environments. Xi'an, China, May 2021. 3.3.1

[13] Micah Corah and Nathan Michael. Scalable distributed planning for multi-robot, multi-target tracking. Prague, Czech Republic, September 2021. 3.1, 3.1, 3.2.3, 3.6.1

[14] Jeffrey Delmerico, Stefan Isler, Reza Sabzevari, and Davide Scaramuzza. A comparison of volumetric information gain metrics for active 3D object reconstruction. 42(2):197–208, 2018. 3.2.2

[15] A. Elhayek, C. Stoll, K. I. Kim, and C. Theobalt. Outdoor Human Motion Capture by Simultaneous Optimization of Pose and Camera Parameters. *Computer Graphics Forum*, 34(6):86–98, September 2015. ISSN 0167-7055, 1467-8659. doi: 10.1111/cgf.12519. URL https://onlinelibrary.wiley.com/doi/10.1111/cgf.12519. 3.1

[16] M. L. Fisher, G. L. Nemhauser, and L. A. Wolsey. An analysis of approximations for maximizing submodular set functions-II. *Polyhedral Combinatorics*, 8:73–87, 1978. 3.2.3, 3.6.2

[17] Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. 2.1

[18] Sergio Hernández, Diego Vergara, Matías Valdenegro-Toro, and Felipe Jorquera. Improving predictive uncertainty estimation using Dropout–Hamiltonian Monte Carlo. *Soft Computing*, 24(6):4307–4322, March 2020. ISSN 1432-7643, 1433-7479. doi: 10.1007/s00500-019-04195-w. URL http://link.springer.com/10.1007/s00500-019-04195-w. 2.1

[19] Cherie Ho, Andrew Jong, Harry Freeman, Rohan Rao, Rogerio Bonatti, and Sebastian Scherer. 3D human reconstruction in the wild with collaborative aerial cameras. September 2021. 1.1, 3.1, 3.1, 3.1, 3.2.1, 3.2.2, 3.8, 3.9.1

[20] Skyler Hughes, Micah Corah, and Sebastian Scherer. Towards informed multi-robot planners for group reconstruction. *Robotics Institute Summer Scholar' Working Papers Journals*, 2022. (document), 3.1, 3.5, 3.2.3, 3.4.3

[21] Skyler Hughes, Rebecca Martin, Micah Corah, and Sebastian Scherer. Multi-robot planning for filming groups of moving actors leveraging submodularity and

pixel density. 2024. in preparation for submission to CDC 2024. 1.1, 3.4.3, 3.4.3

[22] Qingyuan Jiang and Volkan Isler. Onboard view planning of a flying camera for high fidelity 3D reconstruction of a moving actor, July 2023. URL http://arxiv.org/abs/2308.00134. 3.1, 3.2.2, 3.4.3

[23] Liren Jin, Xieyuanli Chen, Julius Rückin, and Marija Popović. NeU-NBV: Next Best View Planning Using Uncertainty Estimation in Image-Based Neural Rendering, July 2023. URL http://arxiv.org/abs/2303.01284. arXiv:2303.01284 [cs]. 1.1, 2.1, 2.2.1, 2.2.3

[24] Hanbyul Joo, Hyun Soo Park, and Yaser Sheikh. MAP Visibility Estimation for Large-Scale Dynamic 3D Reconstruction. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1122–1129, Columbus, OH, USA, June 2014. IEEE. ISBN 978-1-4799-5118-5. doi: 10.1109/CVPR.2014.147. URL https://ieeexplore.ieee.org/document/6909543. 3.1

[25] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://papers.nips.cc/paper_files/paper/2017/hash/9ef2ed4b7fd2c810847ffa5fa85bce38-Abstract.html. 2.1

[26] Ziquan Lan, Zi Jian Yew, and Gim Hee Lee. Robust Point Cloud Based Reconstruction of Large-Scale Outdoor Scenes. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9682–9690, Long Beach, CA, USA, June 2019. IEEE. ISBN 978-1-72813-293-8. doi: 10.1109/CVPR.2019.00992. URL https://ieeexplore.ieee.org/document/8953959/. 2.1, 2.2.2

[27] Mikko Lauri, Joni Pajarinen, Jan Peters, and Simone Frintrop. Multi-sensor next-best-view planning as matroid-constrained submodular maximization. 5(4): 5323–5330, 2020. 3.2.3, 3.3.1, 3.6.1

[28] Jiaqi Lin, Zhihao Li, Xiao Tang, Jianzhuang Liu, Shiyong Liu, Jiayue Liu, Yangdi Lu, Xiaofei Wu, Songcen Xu, Youliang Yan, and Wenming Yang. VastGaussian: Vast 3D Gaussians for Large Scene Reconstruction. 2.2.2

[29] Jiaqi Lin, Zhihao Li, Xiao Tang, Jianzhuang Liu, Shiyong Liu, Jiayue Liu, Yangdi Lu, Xiaofei Wu, Songcen Xu, Youliang Yan, and Wenming Yang. VastGaussian: Vast 3D Gaussians for Large Scene Reconstruction, February 2024. URL http://arxiv.org/abs/2402.17427. arXiv:2402.17427 [cs]. 2.1, 2.2.3

[30] Liqiang Lin, Yilin Liu, Yue Hu, Xingguang Yan, Ke Xie, and Hui Huang. Capturing, Reconstructing, and Simulating: The UrbanScene3D Dataset. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision – ECCV 2022*, volume 13668, pages

93–109. Springer Nature Switzerland, Cham, 2022. ISBN 978-3-031-20073-1 978-3-031-20074-8. doi: 10.1007/978-3-031-20074-8_6. URL https://link.springer.com/10.1007/978-3-031-20074-8_6. Series Title: Lecture Notes in Computer Science. 2.2.3

[31] Liqiang Lin, Yilin Liu, Yue Hu, Xingguang Yan, Ke Xie, and Hui Huang. Capturing, Reconstructing, and Simulating: the UrbanScene3D Dataset, July 2022. URL http://arxiv.org/abs/2107.04286. arXiv:2107.04286 [cs]. (document), 2.2, 2.3, 2.4, ??, ??, 3.2.2

[32] Hang Ma, Daniel Harabor, Peter J. Stuckey, Jiaoyang Li, and Sven Koenig. Searching with consistent prioritization for multi-agent path finding. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):7643–7650, 2019. doi: 10.1609/aaai.v33i01.33017643. 3.2.4

[33] Seth McCammon, Gilberto Marcon dos Santos, Matthew Frantz, Timothy P Welch, Graeme Best, R Kipp Shearman, Jonathan D Nash, John A Barth, Julie A Adams, and Geoffrey A Hollinger. Ocean front detection and tracking using a team of heterogeneous marine vehicles. 38(6):854–881, 2021. 3.2.3, 3.6.1

[34] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, August 2020. URL http://arxiv.org/abs/2003.08934. arXiv:2003.08934 [cs]. 2.2.1

[35] Tobias Nägeli, Javier Alonso-Mora, Alexander Domahidi, Daniela Rus, and Otmar Hilliges. Real-time motion planning for aerial videography with dynamic obstacle avoidance and viewpoint optimization. 2(3):1696–1703, 2017. 3.1

[36] G. L. Nemhauser, L. A. Wolsey, and M. L. Fisher. An analysis of approximations for maximizing submodular set functions-I. *Math. Program.*, 14(1):265–294, 1978. 3.2.3

[37] Helen Oleynikova, Alexander Millane, Zachary Taylor, Enric Galceran, Juan Nieto, and Roland Siegwart. Signed Distance Fields: A Natural Representation for Both Mapping and Planning. page 6 p., 2016. doi: 10.3929/ETHZ-A-010820134. URL http://hdl.handle.net/20.500.11850/128029. Artwork Size: 6 p. Medium: application/pdf Publisher: [object Object]. 2.1, 2.2.2

[38] Pablo Pueyo, Juan Dendarieta, Eduardo Montijano, Ana C Murillo, and Mac Schwager. CineMPC: A fully autonomous drone cinematography system incorporating zoom, focus, pose, and scene composition. 40:1740–1757, 2024. 3.1

[39] Konstantinos Rematas, Andrew Liu, Pratul Srinivasan, Jonathan Barron, Andrea Tagliasacchi, Thomas Funkhouser, and Vittorio Ferrari. Urban Radiance Fields. In *2022 IEEE/CVF Conference on Computer Vision and Pattern*

Recognition (CVPR), pages 12922–12932, New Orleans, LA, USA, June 2022. IEEE. ISBN 978-1-66546-946-3. doi: 10.1109/CVPR52688.2022.01259. URL https://ieeexplore.ieee.org/document/9879805/. 2.1, 2.2.2

[40] Mike Roberts, Shital Shah, Debadeepta Dey, Anh Truong, Sudipta Sinha, Ashish Kapoor, Pat Hanrahan, and Neel Joshi. Submodular trajectory optimization for aerial 3D scanning. pages 5334–5343, Venice, Italy, October 2017. 3.2.2

[41] Nitin Saini, Eric Price, Rahul Tallamraju, Raffi Enficiaud, Roman Ludwig, Igor Martinovic, Aamir Ahmad, and Michael J Black. Markerless outdoor human motion capture using multiple autonomous micro aerial vehicles. Seoul, South Korea, 2019. 3.1, 3.1, 3.2.1, 3.8

[42] Brent Schlotfeldt, Vasileios Tzoumas, and George J Pappas. Resilient active information acquisition with teams of robots. 38(1):244–261, 2021. 3.1, 3.1, 3.2.3, 3.6.1

[43] Alexander Schrijver. *Combinatorial optimization: polyhedra and efficiency*, volume 24. Springer Science & Business Media, 2003. 3.3.2

[44] Guni Sharon, Roni Stern, Ariel Felner, and Nathan R. Sturtevant. Conflict-based search for optimal multi-agent pathfinding. *Artificial Intelligence*, 219: 40–66, February 2015. ISSN 00043702. doi: 10.1016/j.artint.2014.11.006. URL https://linkinghub.elsevier.com/retrieve/pii/S0004370214001386. 3.1, 3.2.4, 3.7.1

[45] Amarjeet Singh, Andreas Krause, Carlos Guestrin, and William J. Kaiser. Efficient informative sensing using multiple robots. *J. Artif. Intell. Res.*, 34:707–755, 2009. 3.2.3, 3.3.1, 3.6.1

[46] Neil Smith, Nils Moehrle, Michael Goesele, and Wolfgang Heidrich. Aerial path planning for urban scene reconstruction: a continuous optimization method and benchmark. *ACM Transactions on Graphics*, 37(6):1–15, December 2018. ISSN 0730-0301, 1557-7368. doi: 10.1145/3272127.3275010. URL https://dl.acm.org/doi/10.1145/3272127.3275010. 3.2.2

[47] Krishna Suresh, Aditya Rauniyar, Micah Corah, and Sebastian Scherer. Greedy Perspectives: Multi-Drone View Planning for Collaborative Perception in Cluttered Environments, March 2024. URL http://arxiv.org/abs/2310.10863. arXiv:2310.10863 [cs]. 3.1

[48] Rahul Tallamraju, Eric Price, Roman Ludwig, Kamalakar Karlapalem, Heinrich H Bülthoff, Michael J Black, and Aamir Ahmad. Active perception based formation control for multiple aerial vehicles. 4(4):4491–4498, 2019. 3.1, 3.2.1, 3.9.1

[49] Rahul Tallamraju, Nitin Saini, Elia Bonetto, Michael Pabst, Yu Liu, Michael Black, and Aamir Ahmad. AirCapRL: Autonomous aerial human motion capture

using deep reinforcement learning. 5(4):6678–6685, 2020. 1.1, 3.2.1

[50] Matthew Tancik, Vincent Casser, Xinchen Yan, Sabeek Pradhan, Ben Mildenhall, Pratul P. Srinivasan, Jonathan T. Barron, and Henrik Kretzschmar. Block-NeRF: Scalable Large Scene Neural View Synthesis, February 2022. URL http://arxiv.org/abs/2202.05263. arXiv:2202.05263 [cs]. 2.2.2, 2.2.3

[51] A. Tewari, J. Thies, B. Mildenhall, P. Srinivasan, E. Tretschk, W. Yifan, C. Lassner, V. Sitzmann, R. Martin-Brualla, S. Lombardi, T. Simon, C. Theobalt, M. Nießner, J. T. Barron, G. Wetzstein, M. Zollhöfer, and V. Golyanik. Advances in Neural Rendering. *Computer Graphics Forum*, 41(2):703–735, 2022. ISSN 1467-8659. doi: 10.1111/cgf.14507. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.14507. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.14507. 2.1, 2.2.3

[52] Haithem Turki, Deva Ramanan, and Mahadev Satyanarayanan. Mega-NeRF: Scalable Construction of Large-Scale NeRFs for Virtual Fly- Throughs. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12912–12921, New Orleans, LA, USA, June 2022. IEEE. ISBN 978-1-66546-946-3. doi: 10.1109/CVPR52688.2022.01258. URL https://ieeexplore.ieee.org/document/9878491/. 2.1, 2.2.2, 2.2.3, 2.4

[53] Julien P.C. Valentin, Sunando Sengupta, Jonathan Warrell, Ali Shahrokni, and Philip H.S. Torr. Mesh Based Semantic Modelling for Indoor and Outdoor Scenes. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2067–2074, Portland, OR, USA, June 2013. IEEE. ISBN 978-0-7695-4989-7. doi: 10.1109/CVPR.2013.269. URL http://ieeexplore.ieee.org/document/6619113/. 2.1, 2.2.2

[54] Yuanbo Xiangli, Linning Xu, Xingang Pan, Nanxuan Zhao, Anyi Rao, Christian Theobalt, Bo Dai, and Dahua Lin. BungeeNeRF: Progressive Neural Radiance Field for Extreme Multi-scale Scene Rendering. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision – ECCV 2022*, volume 13692, pages 106–122. Springer Nature Switzerland, Cham, 2022. ISBN 978-3-031-19823-6 978-3-031-19824-3. doi: 10.1007/978-3-031-19824-3_7. URL https://link.springer.com/10.1007/978-3-031-19824-3_7. Series Title: Lecture Notes in Computer Science. 2.2.2

[55] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural Radiance Fields From One or Few Images. 2.4

[56] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural Radiance Fields from One or Few Images, May 2021. URL http://arxiv.org/abs/2012.02190. arXiv:2012.02190 [cs]. 2.1, 2.1, 2.2.1, 2.2.3

[57] Tobias Zaenker, Claus Smitt, Chris McCool, and Maren Bennewitz. Viewpoint Planning for Fruit Size and Position Estimation. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3271–3277, Prague, Czech Republic, September 2021. IEEE. ISBN 978-1-66541-714-3. doi: 10.1109/IROS51168.2021.9636701. URL https://ieeexplore.ieee.org/document/9636701/. 2.1, 2.2.1

[58] Rui Zeng, Wang Zhao, and Yong-Jin Liu. PC-NBV: A Point Cloud Based Deep Network for Efficient Next Best View Planning. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7050–7057, Las Vegas, NV, USA, October 2020. IEEE. ISBN 978-1-72816-212-6. doi: 10.1109/IROS45743.2020.9340916. URL https://ieeexplore.ieee.org/document/9340916/. 1.1, 2.1, 2.2.1

[59] Hongrui Zhang, Yuning Yao, Kai Xie, Chi-Wing Fu, Hui Zhang, and Hui Huang. Continuous aerial path planning for 3d urban scene reconstruction. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, 40(6):225:1–225:15, 2021. doi: 10.1145/3478513.3480508. 3.2.2

[60] Huiqing Zhang, Yifei Xue, Ming Liao, and Yizhen Lao. BirdNeRF: Fast Neural Reconstruction of Large-Scale Scenes From Aerial Imagery, February 2024. URL http://arxiv.org/abs/2402.04554. arXiv:2402.04554 [cs]. 2.2.2

[61] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. NeRF++: Analyzing and Improving Neural Radiance Fields, October 2020. URL http://arxiv.org/abs/2010.07492. arXiv:2010.07492 [cs]. 2.2.3

[62] Xiangyu Zhou, Kai Xie, Ke Huang, Yi Liu, Yufeng Zhou, Minghua Gong, and Hui Huang. Offsite aerial path planning for efficient urban scene reconstruction. *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*, 39(6):192:1–192:16, 2020. doi: 10.1145/3414685.3417828. 3.2.2