Greedy Perspectives: Dynamic Multi-Drone View Planning for Collaborative Coverage

Krishna Suresh[1], Aditya Rauniyar[2], Micah Corah[2], and Sebastian Scherer[2]

***Abstract*—Teams of aerial robots allow for large-scale filming of dynamic groups of people in complex environments for novel applications in cinematography, sports, and exploration. Toward this end, methods for submodular maximization can be used for scalable optimization of camera views across teams of robots. However, multi-robot planning systems that use submodular maximization such as SGA (Sequential Greedy Assignment) face challenges with efficient and optimal coordination in highly cluttered environments. Dense obstacles increase inter-robot collision and environment view occlusions which can violate the partition matroid requirements for SGA's 50% optimality guarantee. To coordinate teams of aerial robots in filming groups of people in dense environments, a more general view-planning approach is required. In this paper, we explore how collision and occlusion impact optimality and performance in filming applications through the development of a multi-agent dynamic-multi-target view planner with an occlusion-aware objective for filming groups of people and compare with a naive fixed formation planner. To evaluate performance, we plan in three high occlusion/collision test environments with complex multi-target behaviors and measure the average target coverage. Compared with a fixed formation planner, our sequential planner generates 40% more target coverage with the same number of agents and similar performance with fewer agents. Even without a strict bound on suboptimality, we observe efficient and collaborative behaviors which demonstrate the capabilities of sequential greedy planning for real-world multi-agent view planning. Overall, through improving multi-agent filming, effective coordination of teams of aerial robots can enable novel higher systems system behaviors that are otherwise infeasible.**

I. Introduction

The capture of significant events via photos and video has become universal, and Unmanned aerial vehicles (UAVs) extend the capabilities of cameras by allowing for view placement in otherwise hard-to-reach places and by tracking intricate trajectories. Multiple aerial cameras can be used to not only view a target from multiple angles simultaneously but perform higher functions such as cinematic filming [1], efficient environment exploration [2], and outdoor human pose motion capture [3]. These applications rely on effective collaboration between groups of UAVs whereas manual control may result in poor shot selection and view duplication while requiring many coordinated operators. Therefore, autonomous coordination of UAV teams is needed to achieve reliable multi-robot filming. However, directly maximizing domain-specific metrics, such as reconstruction accuracy, can be difficult to estimate online which motivates proxy objectives such as coverage. For example, in [1], cinematic filming is attained through a joint objective combining collision and occlusion avoidance, shot diversity, and artistic principles in filming a single target.

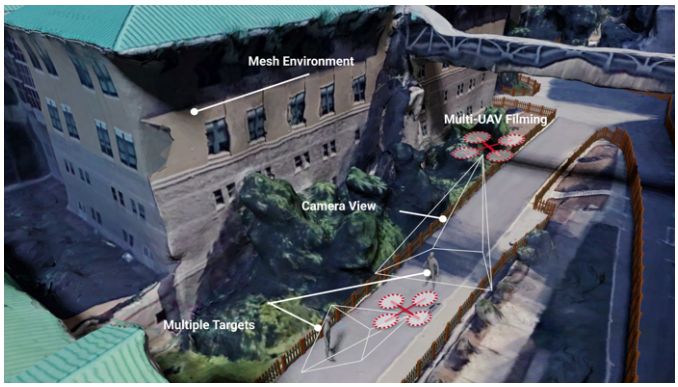

Fig. 1. **Multi-Target Coverage Scenario:** Known target and environment geometries as well as target trajectories with agent start locations are input into the view-planning system. The planner aims to maximize coverage over all of the target throughout the planning horizon.

While defining an objective can be difficult, planning for multi-agent aerial systems also presents a significant challenge due to the vast joint state space and non-linear objectives causing optimal planning to be often intractable. Many applications exploit problem-specific structures to reduce the overall search space or alter the search procedure to generate single-agent trajectories sequentially. [3] uses a preconfigured target-centric formation to coordinate views of a single target from multiple angles. This reduces the search space to a lower dimensional search over formations rather than a search over each agent's motions jointly. A key limitation of this approach is the agent-target assignment which requires agents to focus observation on a specific target and may fail to exploit the robots' capacity to observe multiple targets at once.

Sequential methods can also obtain sub-optimally guarantees when the problem is formulated as submodular maximization. [4] demonstrates dynamic target coverage as a notable objective that submodular maximization can exploit. [5] employs a submodular planning framework for single-UAV aerial mapping and [6] presents static multi-agent view planning for reconstruction with arm-mounted sensors.

Problem: The dynamic multi-target coverage problem is defined as generating a set of camera sequences for a finite

[1]K. Suresh is with Olin College of Engineering, Needham, MA, USA ksuresh@olin.edu

[2] A. Rauniyar, M. Corah, and S. Scherer are with the Robotics Institute, School of Computer Science at Carnegie Mellon University, Pittsburgh, PA, USA {rauniyar, micahc, basti}@andrew.cmu.edu

This work is supported by the National Science Foundation under Grant No. 2024173.

horizon that maximize collective pixels per area (PPA) viewed by the cameras. The primary assumptions required for this problem are: known static environment, known actor trajectories (e.g. scripted scenarios), and known agent start state. An illustration of the problem setup is depicted in **fig. 1**. Generating unassigned view plans though the environment allows agents to increase target coverage and in some cases require less sensors to achieve similar performance.

Approach: Similar to other multi-agent planning systems, our planner employs Sequential Greedy Assignment (SGA) to sequentially plan optimal single-agent actions to generate a collective multi-agent plan. An overview of the view planning system is depicted in **fig. 2**. We first abstract our scene representation by converting target geometries to a simplified mesh, translating a 3D world representation (point cloud, mesh, etc.) to a 2.5D height map, and instantiating sensor/motion models for the agents. Next, we discretize the agent state space and action space as a time-dependent 2D grid with heading to allow us to formulate the single agent planning problem as a Markov Decision Process (MDP). Target coverage is computed through an occlusion-aware software rendering system which computes the pixel densities for each target face. By applying an objective that features decreasing reward for repeated views of the same face and monotonically increasing the reward for more coverage, submodular maximization methods can be applied to the problem.

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

- Implementation of a dynamic multi-target multi-agent view planner.
- Occlusion-aware objective for filming groups of people through software rendering
- Initial evaluation of such a planner with awareness of inter-agent collisions and comparison against a planner based on fixed formations.

II. Related Works

Aerial Filming: Aerial perception systems have grown to widespread use through their success in providing low-cost filming of conventionally challenging unscripted scenes. Consumer and commercial systems such as the Skydio S2+ [7] demonstrate single-drone filming capabilities and are starting to incorporate collaborative multi-drone behaviors for mapping. Viable autonomous aerial filming systems for cinematography have been demonstrated by [1] using a single actor tracking and filming system. As well as [8] through a multi-drone filming system with a variety of filming and coordination modes.

Submodular Multi-Agent Planning: Multi-agent submodular planning aims to efficiently generate bounded sub-optimal trajectories and is explored in many works including [6] with sensors fixed at the end of robot arms as well as with [2] for efficient exploration in unknown environments. [4] demonstrates that submodular techniques are specifically beneficial to target coverage problems.

III. Problem Formulation

We aim to coordinate a team of UAVs to maximize coverage (or observation) of multiple targets through an obstacle-dense environment. Application specific metrics are often difficult to directly measure online, therefore, we use a coverage-like objective as a proxy for effective target observation. To promote view diversity we integrate a diminishing square root reward for coverage of the same target face by multiple views.

Consider a set of targets $\mathcal{T} = \{1, \ldots, N_t\}$ each with a set of faces $\mathcal{F}_j = \{1, \ldots, N_{j,f}\}$ where $j \in \mathcal{T}$ and a set of robots $\mathcal{R} = \{1, \ldots, N_r\}$. Each robot $i \in \mathcal{R}$ can take action $u_{i,t} \in U_i \in SE(2)$ at time $t \in \{0, \ldots, T\}$. Each robot i will select its plan from its local set of finite-horizon sequences of viable control actions.[1] Additionally, robots have an associated state $x_{i,t} \in X$ where X is also a subset of $SE(2)$. X is a shared state space with all robots, however, each robot's trajectory $\xi_i = \{x_{i,0}, \ldots, x_{i,T}\}$ once fixed, produces non-collision constraints for all other robots. Given the trajectories of all targets in $SE(3)$, start states $x_{i,0}$ and environment geometry, we aim to find a joint collision-free control sequence $U^* = \bigcup_{i \in \mathcal{R}} \{u_{i,0}, \ldots, u_{i,T}\}$ which maximizes our objective and fits our motion model.

A. Motion Model

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

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

Where f_i is defined to only allow collision-free actions within the constant velocity constraints. In a time step length, the maximum translational and rotational velocity is converted to a maximum Euclidean radius as illustrated by fig. 2 motion model. The current state of the robot is used to find the set of available actions and changes as the robot navigates the state space.

B. Sensor Model

Inspired by [9], observation of faces of each target j is captured by the pixel density ($\frac{px}{m^2}$) measured from a linear camera model's image. We can define a function $\texttt{cov}(x_{i,t}, j, f) \to \mathbb{R}$ which returns the pixel density for a specific target's face when observed from a robot's state. And $\texttt{covsum}(j, f) \to \mathbb{R}$ which returns the summed pixel density from other robots' plans. We then apply a square root to introduce diminishing returns on increasing pixel density to produce the following:

$$\texttt{viewreward}(x_{i,t}) = \sum_{j \in \mathcal{T}} \sum_{f \in \mathcal{F}_j} \sqrt{\texttt{cov}(x_{i,t}, j, f) + \texttt{covsum}(j, f)}$$

[1] In the literature on suboptimal maximization, these local sets of actions form *blocks* in the structure known as a *partition matroid* which forms the structure of the joint multi-robot problem

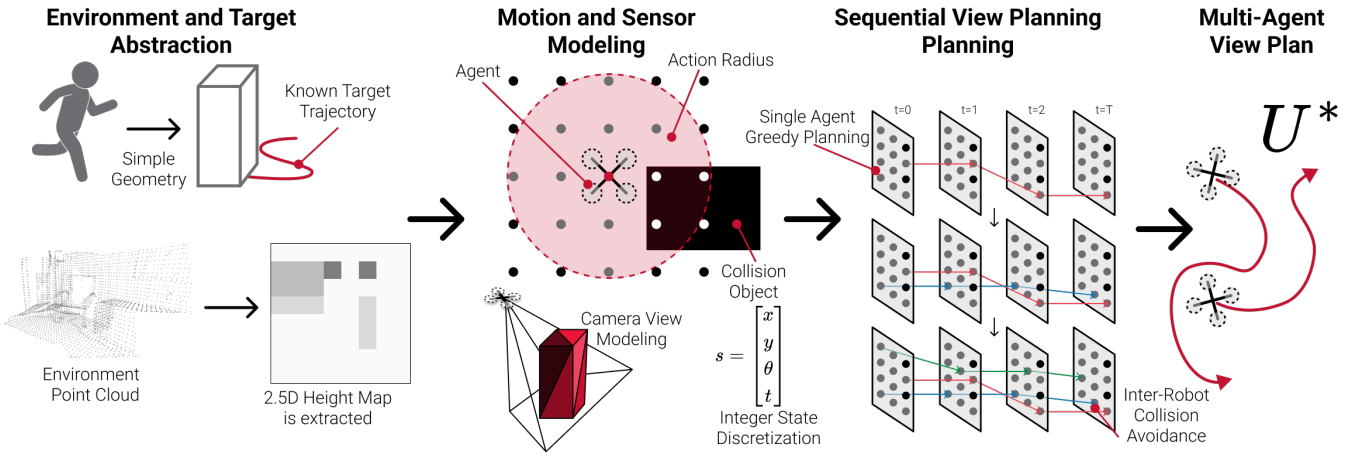

Fig. 2. **Coverage View Planner System Overview** Multi-target scenario is translated to internal planner representation. Markov Decision Process with DAG encodes collision constraints and coverage objective. Multi-Agent View Plan is produced through sequential greedy planning. Joint multi-drone trajectories are brought back to continuous 3D coordinates and outputted to navigation stack.

C. Objective

In addition to maximizing coverage, we add a reward for stationary behavior $R_s(u_{i,t})$ to reduce unnecessary movement. We define the reward objective for a single agent as follows:

$$R(x_{i,0}, \{u_{i,0}, \ldots, u_{i,T}\}) = \sum_{t \in \{0,\ldots,T\}} R_s(u_{i,t}) + \texttt{viewreward}(f(x_{i,t-1}, u_{i,t}))$$

And the joint objective as follows:

$$Q(X_{\texttt{init}}, U) = \sum_{i \in \mathcal{R}} R(x_{i,0}, \{u_{i,0}, \ldots, u_{i,T}\})$$

where $X_{\texttt{init}} = \{x_{0,0}, \ldots, x_{N_r,0}\}$ is the set of initial robot states.

Since we aim to find the control sequence which maximizes this objective our optimal control sequence can be defined as:

$$U^* = \arg\max_U Q(X_{\texttt{init}}, U)$$

IV. Multi-Drone Multi-Target Coverage View Planner

We now present our multi-UAV coverage view planning system. This planner aims to not only produce sufficient target coverage but also exploit problem structure to efficiently find single-agent greedy trajectories.

A. Coverage Representation

To incorporate an occlusion-aware coverage representation we define `cov` by implementing an OpenGL rasterization renderer which draws a 2.5D height map of our environment and simplified geometries of each target. We then use a perspective camera based on our specified camera intrinsics to capture an occlusion-aware representation of what the sensor would expect to see at a given robot state. To render environment occlusions we use a geometry shader to draw the heightmap directly on the GPU. To determine how many pixels we are observing from each face, we render each face with a unique color that corresponds to an encoding of the target ID and face ID. When this unique color appears in the rendered image, we can count the pixel frequency to measure our observed pixels and divide by the associated face surface area to measure the corresponding final pixel density. Figure 3 (b) illustrates an example rendered view produced by the OpenGL rendering system.

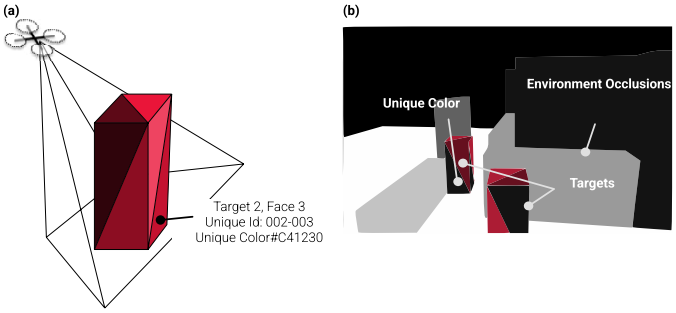

Fig. 3. Target Coverage **(a)** UAV camera model frustrum observing a simplified target geometry. Target faces are colored slightly differently based on a face identification system to allow for pixel density computation. **(b)** Example camera output from OpenGL internal rendering system.

B. Single Agent Planning

With the robot state in $SE(2)$ we aim to represent the single agent planning problem as an MDP which has an underlying DAG structure. The MDP state s is represented as an integer vector:

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

and each MDP action a is in the same discrete space with a time increment of 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 (s, a, s_1) pair with a transition probability and a reward matrix associating each (s, a) pair with a reward. We then perform a Breadth

First Search over the state space by branch on feasible actions to populate our transition and reward matrices. As depicted in fig. 2 the set of available actions is pruned based on environment and inter-robot collisions. This directed MDP can be solved with one pass of value iteration to find the optimal greedy policy (however, our current implementation converges in 5 passes without exploiting this structure). Finally, we can follow this policy from our initial state to produce the optimal single-agent control sequence. We use the AI-ToolBox library to represent and solve the MDP [10]

C. Sequential Planning

Finally, we are able to generate the joint view plan for all of the agents by sequentially planning greedy single-agent trajectories. This formulation is close to submodular maximization which would yield sub-optimally bounded trajectories, however, our consideration of inter-robot collisions violates the formulation.

V. Experiments

We evaluate the performance of the sequential view planner in three test scenarios which aim to demonstrate the dynamic view planning capabilities.

A. Naive Fixed Formation Planning

To compare our coverage planner with an assignment-based view planner, we implement a naive fixed formation planner modeled off of the multi-view formations described in [3]. We define the formation with a constant radius around a target and a separation angle ϕ. As described by [3] for $N_r > 2$ $\phi = \frac{2\pi}{N_r}$ and $\phi = \frac{\pi}{2}$ when $N_r = 2$. A key consideration with our implementation is that formations do not consider environment and robot collisions since robots are fixed to their orientation throughout the horizon.

B. Test Scenarios

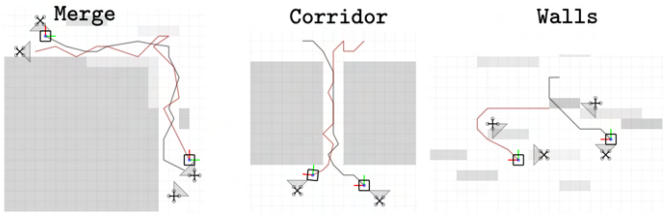

Fig. 4. Preliminary test cases to evaluate specific aspects of multi-target view planning.

Test scenarios in fig. 4 use agents with a 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.

`Merge`: Contains 2 targets/4 agents moving around a corner in opposite directions for 17 timesteps. This test case demonstrates dynamic target assignment with targets being "handed off" at the corner shown by fig. 5.

`Corridor`: Contains 2 targets/2 agents moving through a narrow corridor in 17 timesteps. This test focuses on the collision-aware aspect of the planner.

`Walls`: Contains 2 targets/4 agents moving through a sequence of occlusion walls in 10 timesteps. This test aims to demonstrate the occlusion-aware objective which promotes target views that are obstruction free.

C. Sequential Planner Performance

TABLE I
Average Pixel Density($\frac{Mpx}{m^2}$) in Test Cases

	Merge	**Corridor**	**Walls**
Formation	1.70	0.68	1.77
Sequential	**2.40**	**1.68**	**3.11**
Sequential $N_r - 1$	1.91		

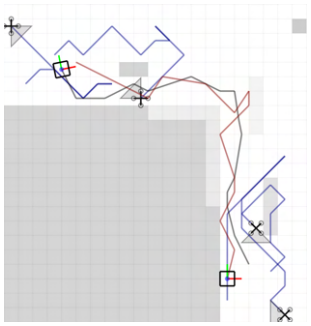

Fig. 5. Generated joint trajectory for `Merge` test case in blue. Notably, agents remain on their side of the corner and perform a "handoff" of the targets at the corner.

In table I we track the average coverage observed by all agents. All of our test cases demonstrate an increase in megapixels per meter squared pixel density with a sequential planner over a formation planner. Additionally, in the `Merge` test case, sequential planning achieves similar results with fewer agents. One reason why the sequential planner observes targets more than the fixed formation planner may be due to poor hand-chosen parameters for the formation planner. The formation radius and camera tilt for the agent views can be optimized to best evaluate the performance increase with sequential planning.

VI. Conclusion and Future Work

Through preliminary evaluation in three test cases, we observe collaborative behaviors which suggest effective solutions that demonstrate submodular maximization may still perform when considering inter-robot collisions even without bounded-suboptimality. A key challenge with the current sequential planner is computational efficiency, many of the processes are unparallelized limiting online planning capabilities. In future work, we aim to extend the view planning system to consider sequence order, deploy view plans to a 3D human pose reconstruction task, and optimize computation to run at real-time rates.

Acknowledgment

This work is supported by the National Science Foundation under Grant No. 2024173. and sponsored by the AirLab at the Robotics Institute, Carnegie Mellon University, as a part of the Robotics Institute Summer Scholars (RISS) program.

REFERENCES

[1] A. Bucker, R. Bonatti, and S. Scherer, "Do You See What I See? Coordinating Multiple Aerial Cameras for Robot Cinematography," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, May 2021, pp. 7972–7979, iSSN: 2577-087X.

[2] M. Corah, "Sensor Planning for Large Numbers of Robots," PhD Thesis, Carnegie Mellon University, Pittsburgh, PA, Sep. 2020, issue: CMU-RI-TR-20-53.

[3] C. Ho, A. Jong, H. Freeman, R. Rao, R. Bonatti, and S. Scherer, "3D Human Reconstruction in the Wild with Collaborative Aerial Cameras," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2021, pp. 5263–5269, iSSN: 2153-0866.

[4] M. Corah, "On Performance Impacts of Coordination via Submodular Maximization for Multi-Robot Perception Planning and the Dynamics of Target Coverage and Cinematography," in *RSS 2022 Workshop on Envisioning an Infrastructure for Multi-Robot and Collaborative Autonomy Testing and Evaluation*, 2022.

[5] M. Roberts, S. Shah, D. Dey, A. Truong, S. Sinha, A. Kapoor, P. Hanrahan, and N. Joshi, "Submodular Trajectory Optimization for Aerial 3D Scanning." IEEE Computer Society, Oct. 2017, pp. 5334–5343, iSSN: 2380-7504. [Online]. Available: https://www.computer.org/csdl/proceedings-article/iccv/2017/1032f334/12OmNzA6GPq

[6] M. Lauri, J. Pajarinen, J. Peters, and S. Frintrop, "Multi-Sensor Next-Best-View Planning as Matroid-Constrained Submodular Maximization," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 5323–5330, Oct. 2020, conference Name: IEEE Robotics and Automation Letters.

[7] "Drone That Follows You - Skydio 2+ | Skydio." [Online]. Available: https://www.skydio.com/skydio-2-plus/

[8] A. Alcántara, J. Capitán, A. Torres-González, R. Cunha, and A. Ollero, "Autonomous Execution of Cinematographic Shots With Multiple Drones," *IEEE Access*, vol. 8, pp. 201 300–201 316, 2020, conference Name: IEEE Access.

[9] Q. Jiang and V. Isler, "Onboard View Planning of a Flying Camera for High Fidelity 3D Reconstruction of a Moving Actor," Jul. 2023, arXiv:2308.00134 [cs]. [Online]. Available: http://arxiv.org/abs/2308.00134

[10] "Svalorzen/AI-Toolbox: A C++ framework for MDPs and POMDPs with Python bindings." [Online]. Available: https://github.com/Svalorzen/AI-Toolbox/tree/master