2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2025)
October 19-25, 2025. Hangzhou, China

2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) | 979-8-3315-4393-8/25/$31.00 ©2025 IEEE | DOI: 10.1109/IROS60139.2025.11247464

Robots Calling the Shots: Using Multiple Ground Robots for Autonomous Tracking in Cluttered Environments

Weijian Zhang, Charlie Street, Masoumeh Mansouri

***Abstract*— A common task in cinematography is *tracking* a subject or character through a scene. For complex setups, multiple cameras must track the subject simultaneously to attain sufficient coverage. Recently, researchers have considered using multiple camera-mounted autonomous mobile robots for this task. Existing work is limited to UAVs, which may be unavailable due to cost, safety requirements, or flight restrictions. Therefore, in this paper we present a tracking approach for complex and unstructured environments using *differential-drive robots with gimbal-mounted cameras*. Differential-drive robots pose a challenge, as their movement is more restricted than UAVs. For this, we introduce a novel hierarchical planning framework which ensures safety and visibility while maximizing shot diversity. We begin by synthesizing a set of paths using sequential greedy viewpoint planning and conflict-based search under a set of optimal viewpoint constraints. These paths then form an initial guess for joint trajectory optimization, which synthesizes stable trajectories under the motion constraints of the robots and gimbals. Empirically, we show how our approach outperforms approaches aimed at UAVs, which may synthesize infeasible trajectories when applied to differential-drive robots.**

I. Introduction

Movable cameras are widely utilized in the entertainment industry for sports coverage, and for search and rescue tasks, where dynamic changes in perspective are essential [1]. A frequent and complex task in cinematography is tracking a moving human using multiple cameras. This allows large or complex events to be recorded efficiently while offering directors greater artistic flexibility [2]. One approach for this task is to use teams of camera-mounted autonomous mobile robots. Compared to traditional techniques such as rail-mounted cameras and camera cranes, autonomous mobile robots offer significant advantages in efficiency and safety [3]. To solve the tracking task, a team of robots must cover the human's surface continuously while ensuring each robot maintains visibility of the human [4] (see Fig. 1). Each trajectory must also avoid collisions with obstacles, other robots, and the human. Maintaining visibility is challenging, as obstacles may occlude the human within a robot's limited field of view (FOV) [5]. To efficiently cover the human, the robots should coordinate their viewpoints to maximize coverage while synthesizing smooth trajectories that avoid motion blur in their FOV.

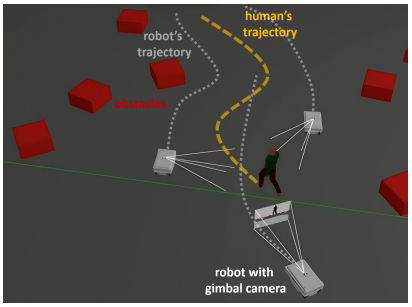

Fig. 1. Three differential-drive robots collaboratively film a human in an unstructured environment. Our formation planner maximizes visibility and shot diversity for the human over the entire planning horizon.

Existing work solves the tracking problem using teams of unmanned aerial vehicles (UAVs) [1], [3], [6]. Though UAVs are effective at tracking due to their high maneuverability, they may be unsuitable due to safety constraints or flight restrictions. For example, if the cameras must keep low to the ground, spinning rotors may pose danger to humans nearby. UAVs can also have a limited battery life, and be sensitive to the wind and weather. To mitigate these issues, in this paper we propose using teams of *differential-drive ground robots* equipped with cameras on gimbals for tracking. By attaching cameras to a gimbal, we overcome camera motion limitations imposed by the nonholonomic constraints of differential-drive robots, improving their suitability for cinematography. However, planning safe and visually optimal trajectories for differential-drive ground robots is challenging, as robots must navigate through confined 2D spaces with limited path options and frequent occlusions. For this, we introduce a novel hierarchical multi-robot trajectory planning framework which maximizes camera coverage in cluttered environments while avoiding collisions and maintaining visibility of the human. We begin by planning a sequence of viewpoints for each robot which maximize shot diversity using a ring-shaped representation of visibility around the human [7]. We then use enhanced conflict-based search (ECBS) [8] to synthesize collision-free paths between the viewpoints. These paths then form an initial guess for joint trajectory optimization, which synthesizes stable trajectories under the motion constraints of the robots and gimbals. Our trajectory planner aims to keep the robots in the same homotopy class as the human, minimizing the chance of occlusions.

To the best of our knowledge, ours is the first approach to consider teams of differential-drive ground robots equipped with movable cameras for multi-robot cinematography and

Weijian Zhang, Charlie Street and Masoumeh Mansouri are with the School of Computer Science at the University of Birmingham, wxz163@student.bham.ac.uk and {c.l.street, m.mansouri}@bham.ac.uk

Charlie Street and Masoumeh Mansouri are UK participants in Horizon Europe Project CONVINCE, and supported by UKRI grant number 10042096.

tracking. We evaluate our approach in simulation, demonstrating how we outperform existing works by explicitly considering the kinodynamic constraints of differential-drive robots and the topology of the environment. All corresponding software is released open source online[1].

II. Related Work

Collaborative multi-robot cinematography through tracking has received significant research attention in recent years [1], [3], [4], [6], [9]–[14]. For example, Zhou et al. [9] solve the tracking problem using a leader-follower formation of UAVs. There, when a robot observes the human they broadcast it to others, enhancing the team's ability to handle occlusions. Though robots are more aware of occlusions during execution, there is no proactive decision-making to minimize the risk of visibility loss. In comparison, Bucker et al. [3] proposed a greedy multi-UAV camera coordination framework which scores discrete cells on a hemispherical surface around the human based on how well they avoid occlusions. A centralized greedy search is then used to generate motion sequences for each UAV, minimizing visibility loss. The final UAV trajectories are synthesized to balance between smoothness, shot diversity, collision avoidance, and mutual visibility. Under this approach (and its extension in [6]), the relative angles of each robot to the human are computed sequentially. This produces suboptimal solutions, as the team is not considered jointly. Similar to [3], Xu et al. [10] proposed a controller based on hemispherical coverage. However, their approach does not consider domains with obstacles, limiting applicability. Nageli et al. [11] introduced a model predictive control (MPC) approach that calculates visible and occluded regions on the horizontal plane by modeling obstacles as ellipsoids. However, this reliance on ellipsoidal obstacle models limits the method's applicability in unstructured environments. A deep reinforcement learning approach is presented in [12] which attempts to learn optimal camera formation strategies for achieving ideal viewpoints. However, this approach struggles to handle obstacles and occlusions in the environment.

Autonomous multi-robot tracking can be formulated as a formation coordination problem, where robots must maintain a specific formation relative to the human while adjusting their configurations to avoid obstacles and occlusions [1], [4], [13], [14]. Tallamraju et al. [1] synthesize optimal local motion plans which correspond to optimal viewpoint configurations. MPC is then used for formation maintenance and collision avoidance. However, their approach does not explicitly consider occlusion avoidance, limiting visibility in cluttered environments. In [13], a multi-UAV trajectory optimization framework is presented for 3D aerial reconstruction. UAVs may rotate around the human, where sequences of UAV rotation angles are computed through dynamic programming. Formation rigidty is enforced as a hard constraint while synthesising the initial trajectory, which is overly restrictive and causes planning failures in complex environments. The framework we present in this paper does not constrain robots to a rigid formation. Instead, we optimize robot viewpoints to maximize coverage of the human in cluttered environments. To maintain visibility of the human when using fixed cameras, [4] allow robots to adjust the size or shape of the formation. However, [4] does not consider cluttered environments, and assumes the human follows an ellipsoidal trajectory. A planning approach is presented in [14] that ensures UAVs remain in the same homotopy class as the human, mitigating occlusions. This approach assumes robots are independent during initial trajectory synthesis, which can cause deadlocks in narrow environments. The approach we present in this paper explicitly considers dependencies between robots, reducing the risk of robot collisions while improving tracking performance. In this work, our method explicitly considers inter-robot coordination, reducing the risk of deadlocks in narrow environments and improving overall tracking robustness.

[1] https://github.com/HyPAIR/VisFormationPlanner

III. Problem Statement

In this paper, we consider a formation of N nonholonomic robots equipped with cameras on gimbals that can capture humans moving in a cluttered environment. Our objective is to maximize camera coverage of the human by evenly distributing the robots around them. Moreover, we want to maximize each robot's visibility of the human across the planning horizon. Let $\mathcal{W} \subset SE(3)$ be the $2D$ workspace, and $\mathcal{O} \subset SE(3)$ be the set of static obstacles, which we assume are convex, polyhedral, and known *a priori*. For robot $i \in \{1, ..., N\}$, we define $\mathbf{p}_i(t) = [x_i(t),\, y_i(t)]^T \subset SE(2)$ as the coordinate of the midpoint of the rear axle and $\theta_i(t) \in [-\pi,\, \pi)$ as the robot's orientation in Cartesian space. We assume the motion trajectory of the human is known and estimated using the social force model (SFM) [15]. The human's trajectory is denoted $\mathcal{A}(t) = [x^A(t),\, y^A(t),\, \theta^A(t)]$, $t \in [0,\, t_f]$, where $x^A(t)$ and $y^A(t)$ denote the human's position at time t, $\theta^A(t)$ denotes its orientation, and t_f denotes the time the human's trajectory finishes.. The nonholonomic nature of differential-drive robots makes it challenging to match the human's motion while keeping it within a robot's FOV. Therefore, we attach cameras to a yaw-controllable gimbal for flexible camera movements that always point towards the human. With this, we denote a trajectory for robot i as $\mathcal{T}_i(t) = [\mathbf{x}_i(t),\, \mathbf{u}_i(t)]^T$, $t \in [0,\, t_f]$, where $\mathbf{x}_i(t) = [x_i(t), y_i(t), \theta_i(t), \theta_i^g(t)]^T$ specifies the robot's position, the yaw angle of the robot base, and the yaw angle of the gimbal. The lower and upper bounds on $\mathbf{x}_i(t)$ are denoted $\underline{\mathbf{x}}$ and $\overline{\mathbf{x}}$, respectively. The control input $\mathbf{u}_i(t) = [v_i(t), \omega_i(t), \phi_i(t)]^T$ specifies the robot's linear velocity, its angular velocity, and the angular velocity of the gimbal, where $\underline{\mathbf{u}}$ and $\overline{\mathbf{u}}$ are the lower and upper bounds on $\mathbf{u}_i(t)$, respectively. The kinematic model of robot i with a yaw-controllable gimbal is expressed as:

$$\frac{d}{dt}\begin{bmatrix} x_i(t) \\ y_i(t) \\ \theta_i(t) \\ \theta_i^g(t) \end{bmatrix} = \begin{bmatrix} v_i(t)\cos\theta_i(t) \\ v_i(t)\sin\theta_i(t) \\ \omega_i(t) \\ \phi_i(t) \end{bmatrix}, \quad t \in [0, t_f]. \tag{1}$$

In this paper, we address four challenges for multi-robot tracking and coordination: (1) **Motion Smoothness**, i.e. synthesizing smooth trajectories to prevent abrupt camera movements that degrade observation quality; (2) **Human Visibility**, i.e. minimizing occlusions caused by environmental obstacles to ensure that robots maintain continuous visual contact with the human; (3) **Collision Avoidance**, i.e. preventing collisions among robots, obstacles, and the human to guarantee operational safety; and (4) **Shot Diversity**, i.e. configuring the robots to maximize shot diversity and thus coverage of the human.

To address these challenges, we must compute collision-free trajectories $\mathcal{T}_i$ for each robot that track the moving human from their initial states $\mathbf{x}_i^s$ to their goal states $\mathbf{x}_i^g$ while maximizing shot diversity across the team. We formulate this as a joint optimal control problem (OCP) which optimizes a cost function J_i^c for each robot i:

$$\min_{\mathbf{x}_i,\,\mathbf{u}_i} \sum_{k=1}^{M}\sum_{i=1}^{N}\int_{T_k}^{T_{k+1}} J_i^c(\mathbf{x}_i(t),\,\mathbf{u}_i(t),\,\mathcal{A}(t))\,dt \tag{2a}$$

$$\text{s.t. kinematic constraints (1)}, \tag{2b}$$

$$\underline{\mathbf{x}} \le \mathbf{x}_i(t) \le \overline{\mathbf{x}},\ \underline{\mathbf{u}} \le \mathbf{u}_i(t) \le \overline{\mathbf{u}},\ \forall t \in [T_k, T_{k+1}], \tag{2c}$$

$$\mathbf{x}_i(0) = \mathbf{x}_i^s,\ \mathbf{u}_i(0) = \mathbf{0}, \tag{2d}$$

$$\mathbf{x}_i(t_f) = \mathbf{x}_i^g,\ \mathbf{u}_i(t_f) = \mathbf{0}, \tag{2e}$$

$$\mathcal{G}(\mathbf{x}_i(t),\,\mathbf{u}_i(t),\,T_k) \preceq 0,\ \forall t \in [T_k, T_{k+1}],\ \forall i \in \{1,\dots,N\},\ \forall k \in \{1,\dots,M\}. \tag{2f}$$

In (2), we discretize the human's continuous trajectory into M time points $\mathcal{A}(T_k)\,(k \in \{1,2,...,M\})$. The OCP constraints capture the robots' kinematic constraints (2b), solution feasibility constraints (2c), initial constraints (2d), and terminal constraints (2e). (2f) comprises the robots' safety and visibility constraints, denoted $\mathcal{G}$, which we describe in Sec. V.

To solve (2), we propose Alg. 1, which we describe throughout the remainder of this paper. In summary, we begin by generating a sequence of polygons named view regions for each robot (line 1). View regions are constructed along the human's path and keep the human in view while maximizing shot diversity. We then construct collision-free paths for each robot that connect their view regions while remaining in the same homotopy class as the human (line 2-4). These paths are then used as an initial guess for the joint OCP in (2) (lines 6 and 7). If we fail to solve (2) or sharp turns are detected, we iteratively relax constraints over robot viewpoints and re-solve (2) until a valid solution is found (lines 8-10). We proceed by describing how to synthesize view regions and the paths between them in Sec. IV, and then discuss the OCP objectives and constraints in Sec. V.

Algorithm 1: The Proposed Formation Planner.

1 $\mathcal{V} \leftarrow$ **IdentifyVisRegions**$(map,\,\mathcal{A},\,\{\mathbf{x}_i^s\}_{i=1}^N)$;
2 $\{\mathbf{x}_i^g\}_{i=1}^N \leftarrow$ **GenerateSubGoals**$(\mathcal{V})$;
3 $\mathcal{SC}^H \leftarrow$ **GenerateSafeCorridors**$(map,\,\{\mathbf{x}_i^g\}_{i=1}^N,\,\mathcal{A})$;
4 $\mathbf{path} \leftarrow$ **PlanFormationPath**$(map,\,\{\mathbf{x}_i^g\}_{i=1}^N,\,\mathcal{SC}^H)$;
5 $[\mathbf{InitTraj},\,\mathcal{SC}^S] \leftarrow$ **IntialGuess**$(\mathbf{path},\,\mathcal{A})$;
6 **GenerateOCP**$(\mathbf{InitTraj},\,\mathcal{SC}^S)$;
7 $[\mathcal{T},\,is\_failed] \leftarrow Solve$ (2);
8 **while** is_failed *or* ***SharpTurnDetected***$(\mathcal{T})$ **do**
9 Relax a View Region Constraint in (2);
10 $[\mathcal{T},\,is\_failed] \leftarrow Solve$ (2);
11 **return** $\mathcal{T}$;

IV. Initial Trajectory Synthesis

In this section, we describe how to synthesize an initial joint trajectory which forms the initial guess for the OCP in (2). This requires generating a set of viewpoints for each robot along the human's trajectory, and then planning collision-free paths between them.

A. Generating Visible Regions

Recall that we discretize the human's trajectory into M predicted positions. To synthesize the robot viewpoints, we first define a two-dimensional *annular region* around each human position for each robot, similar to [7]. Annular regions are ring-shaped areas centered on the human with an inner radius of d_{min} and outer radius of d_{max}. Robots within an annular region remain within a fixed range of the human, as shown in pink in Fig. 2(a). For each annular region and robot, we then identify visible regions where the robot has a clear line of sight on the human. We compute these visible regions using Alg. 2.

Alg. 2 computes the visible regions for a single robot at the next timestep given its previous angle relative to the human ψ^{prev} (see Fig. 2(a)). We begin by defining a range of angles around ψ^{prev} which are computed using a fixed neighborhood angle ψ_{nb} (line 3). We then sweep through these angles in fixed increments to identify the visible regions (line 4-11). At each angle, we check for obstacles along the line between the human's position $\mathcal{A}(T_k)$ and the edge of the annular region at that angle (line 6). If an obstacle is detected, we create a region from the start of the current sweep to the last collision-free angle (line 9). If this region can fit the robot's footprint, it is added to the set of visible regions (lines 8 and 12). We then continue the sweep from the latest angle to find more visible regions. After the sweeping process is complete, we test the cumulative angle range covered by the visible regions against a threshold ψ_s (line 15). We do this to build a sufficient solution space for the optimization in Sec. IV-B. If the total angle range exceeds ψ_s, we return the visible regions. Otherwise, we decrease a scaling factor $\mathcal{S}$ and re-run the sweeping process (line 18). Scaling factor S decreases the diameter of the annular region, bringing the robot closer to the human. This helps mitigate occlusions in cluttered environments, improving the size or number of visible regions. Fig. 2(b) shows the visible regions for three robots, where robot 3 has two visible regions due to the presence of obstacles.

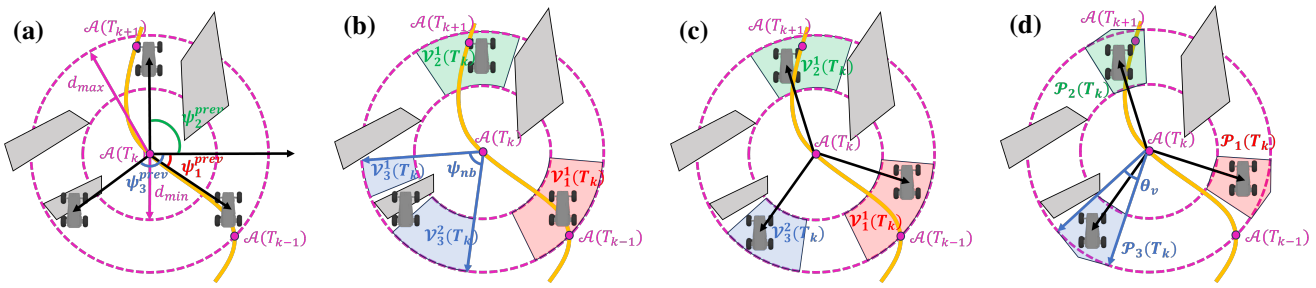

Fig. 2. Generating the optimal formation configuration. The orange line shows the trajectory of the human. (a) The previous formation configuration projected onto the obstacle landscape at the current timestep. (b) The visible regions generated in Sec. IV-A. (c) The optimal formation configuration obtained in Sec. IV-B. (d) The generated convex pentagons approximated by [16] based on the optimal formation configuration.

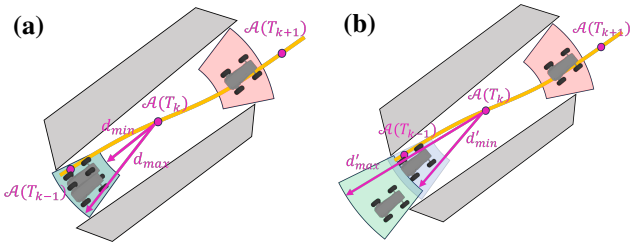

Fig. 3. Resolving infeasible formation configurations. (a) An infeasible formation configuration caused by a narrow environment. (b) The formation configuration after adjusting the scaling factor.

B. Identifying the Optimal Sequence of Visible Regions

Given a set of visible regions for each robot at each timestep, we now synthesize a sequence of configurations for the entire formation that maximizes the distribution of the robots around the human. For this, we employ a sequential strategy that synthesizes the next formation configuration using only the previous configuration. We formalize the problem of synthesizing the next formation configuration as a mixed-integer nonlinear programming (MINLP) problem:

$$\min_{\psi_i,\, z_{i,j}} \quad \sum_{i=1}^{N} \left[(\Delta_i - \Delta^*)^2 + \lambda_1(\psi_i - \psi_i^{\text{prev}})^2\right] \tag{3a}$$

$$\text{s.t.} \quad \sum_{j=1}^{K_i} z_{i,j} = 1, \quad \forall i, \tag{3b}$$

$$\psi_i \in [\psi_L^{i,j}, \psi_R^{i,j}], \quad \text{if } z_{i,j} = 1, \tag{3c}$$

$$\Delta_i = ((\psi_{N_i} - \psi_i + \pi) \bmod 2\pi) - \pi, \forall i. \tag{3d}$$

Here, Δ_i is the angular difference between robot i and its neighboring robot $N_i = (i+1) \mod N$ (see (3d)). The optimal separation angle is $\Delta^* = \frac{2\pi}{N}$ for $N > 2$ and $\Delta^* = \frac{\pi}{2}$ for $N = 2$, i.e. where the robots are equally spread around the human. The objective function (3a) trades between optimizing the separation angle and how much each robot adjusts its angle, improving motion smoothness. Constraint (3c) forces each robot to select a single visible region from Alg. 2. For example, in Fig. 2(c), robot 3 selects the lower visible region in Fig. 2(b) as it maximizes the angular distribution of the formation while avoiding collisions and occlusions. We solve the MINLP in (3) sequentially for each discrete timestep/human position.

Algorithm 2: Identifying a Single Robot's Visible Regions for the Next Timestep.

Input: $\mathcal{A}(T_k)$, ψ^{prev}, ψ_{nb}, $d_{\max}$, ψ_s, α
Output: $\mathcal{V}$

Initialize: $\mathcal{V} \leftarrow \emptyset$, $\mathcal{S} \leftarrow 1$, found $\leftarrow$ false
while *not found* **do**
  $\psi_L \leftarrow \psi^{prev} - \psi_{nb}/2$, $\psi_R \leftarrow \psi^{prev} + \psi_{nb}/2$, $\psi_{\text{total}} \leftarrow 0$;
  for $\psi \in [\psi_L, \psi_R]$ *with step* d_ψ **do**
    $\mathbf{p}_{\max} \leftarrow \mathcal{A}(T_k) + \mathcal{S} \cdot d_{\max}[\cos\psi, \sin\psi]$;
    if *isObstacleOnLine*$(\overline{\mathcal{A}(T_k)\mathbf{p}_{\max}})$ **then**
      if $\psi_L < \psi$ **then**
        if *robotCanFit*$(\psi_L, \psi - d_\psi, \mathcal{S})$ **then**
          $\mathcal{V} \leftarrow \mathcal{V} \cup (\psi_L, \psi - d_\psi, \mathcal{S})$;
          $\psi_{\text{total}} \leftarrow \psi_{\text{total}} + (\psi - d_\psi - \psi_L)$;
        $\psi_L \leftarrow \psi + d_\psi$;
  if *robotCanFit*$(\psi_L, \psi_R, \mathcal{S})$ **then**
    $\mathcal{V} \leftarrow \mathcal{V} \cup (\psi_L, \psi_R, \mathcal{S})$;
    $\psi_{\text{total}} \leftarrow \psi_{\text{total}} + (\psi_R - \psi_L)$;
  if $\psi_{total} \geq \psi_s$ **then**
    found $\leftarrow$ true, **return** $\mathcal{V}$;
  else
    $\mathcal{S} \leftarrow \alpha \cdot \mathcal{S}$; // Reduce distance
    if $\mathcal{S}$ *is too small* **then**
      $\mathcal{V} \leftarrow \emptyset$, **return** $\emptyset$;
    $\mathcal{V} \leftarrow \emptyset$;

In narrow passages, we may observe robots that intersect and collide, as shown in Fig. 3(a). Recall that a scaling factor $\mathcal{S}$ is used in Alg. 2 to adjust a robot's distance to the human. To fix intersections in narrow passages, we increase $\mathcal{S}$ by a factor β for one of the intersecting robots. If the two robots use a different value of $\mathcal{S}$, we select the robot with the larger $\mathcal{S}$ to reduce the chance of occlusions. This is because Alg. 2 has already had to bring the robot with the smaller $\mathcal{S}$ further forward to avoid occlusions. If the robots have the same scaling factor, we select the robot with the smaller visible region to improve overall visibility. If both robots have equal scaling factors and visible regions, we select a robot at random. After adjusting the scaling factors, we re-run Alg. 2 and re-solve (3) to synthesize a collision-free formation configuration, as shown in Fig. 3(b).

C. Synthesizing Paths to Connect Formation Configurations

Given the sequence of formation configurations in Sec. IV-B, our final step in synthesizing the initial trajectory is to plan collision-free paths between each formation configuration. This produces a number of multi-robot path planning problems where the goal location in one problem is the initial location of the next. For each path planning problem, we discretize the robot's environment into a 2D occupancy map. We then use enhanced conflict-based search (ECBS) [8] for collision-free path planning. ECBS is a centralized, complete, and suboptimal algorithm that uses a high-level search tree to resolve potential robot conflicts. For each planning problem we optimize the *makespan*, i.e. the time for the last robot to reach its goal. Each robot must finish its current path before moving onto the next. By optimizing the makespan robot paths will be more balanced, reducing long robot waiting times at intermediate goal locations.

To maximize visibility of the human, we want to remove the chance of occlusions that block a robot's FOV. For this, we modify ECBS to ensure that each robot's path remains in the same homotopy class as the human at all times. Consider the planning problem at timestep T_k. Recall that $\mathbf{p}_i(T_k)$ and $\mathcal{A}(T_k)$ are the positions of robot i and the human at time T_k, respectively. We begin by constructing a reference trajectory ref_i^k for each robot i, where $ref_i^k = \{\mathbf{p}_i(T_k), \mathcal{A}(T_k), ..., \mathcal{A}(T_{k+1})\, \mathbf{p}_i(T_{k+1})\}$. Trajectory ref_i^k follows the shortest path from $\mathbf{p}_i(T_k)$ to $\mathcal{A}(T_k)$, follows the human's trajectory to $\mathcal{A}(T_{k+1})$, and then follows the shortest path to $\mathbf{p}_i(T_{k+1})$, as shown in Fig. 4. The reference trajectories are collision-free, as the human's trajectory is collision-free, and the routes from $\mathbf{p}_i(T_k)$ and $\mathbf{p}_i(T_{k+1})$ to the human are occlusion-free as described in Sec. IV-A and Sec. IV-B.

Given a reference trajectory ref_i^k, we construct a safe corridor along ref_i^k using the method in [17] (line 3 in Alg. 1), which expands obstacle-free line segments into a series of connected polygons. Each polygon is represented using the $\mathcal{H}$-representation [18], i.e. the j-th polygon is given by $\mathcal{SC}_j^H = \{q \in \mathbb{R}^2 : \mathbf{A}_j q \leq \mathbf{b}_j\}$, where $\mathbf{A}_j \in \mathbb{R}^{n_j \times 2}$ and $\mathbf{b}_j \in \mathbb{R}^{n_j}$ describe the hyperplane of a convex polygon, and n_j is the number of hyperplanes. The set of polygons that form the safe corridor is given by $\mathcal{SC}^H$. During ECBS occupancy map construction, any cell outside of $\mathcal{SC}^H$ is treated as an obstacle, forcing the robot into the same homotopy class as the human. We relax this constraint if a feasible solution cannot be found, e.g. if the environment is too narrow to admit all robots within their safe corridors. This allows us to find solutions even if total visibility is not possible.

We run our modified ECBS for each pair of formation configurations and record the paths in **path** (line 4 in Alg. 1). From this, a set of initial trajectories and safe corridors are generated using the time profile of the human (line 5).

V. Trajectory Optimization

In this section, we use the initial trajectories in Sec. IV as an initial guess for trajectory optimization, which is formulated in the OCP in (2). The trajectories we synthesize are smooth and kinematically feasible while satisfying the visibility requirements of the human. We proceed by describing the cost functions and constraints in (2) in detail.

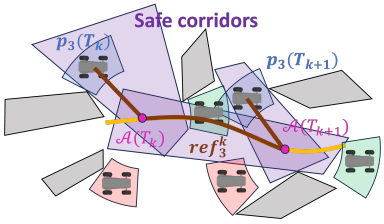

Fig. 4. An example reference trajectory and corresponding safe corridors for maintaining topological equivalence with the human.

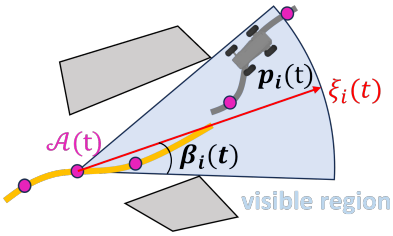

Fig. 5. The visible region for occlusion avoidance.

A. Trajectory Cost Function

The cost function J_i^c in (2a) is a weighted sum of terms which capture smoothness, the angular separation of the robots, occlusion avoidance, and the deviation of the human within the robots' FOVs.

Smoothness. To synthesize smooth trajectories and minimize the impact of camera jitter on filming, we minimize each robot's control effort:

$$J_i^{ce} = \mathbf{u}_i(t)^T \mathbf{u}_i(t). \tag{4}$$

Angular Separation. Recall from Sec. IV-B that we want to maximize the angular separation between robots to maximize shot diversity. For this, we adopt the angular distance cost function proposed in [19]:

$$J_i^{sep} = \left(\frac{(\mathbf{p}_i(t) - \mathcal{A}(t)) \cdot (\mathbf{p}_{N_i}(t) - \mathcal{A}(t))}{\|\mathbf{p}_i(t) - \mathcal{A}(t)\| \|\mathbf{p}_{N_i}(t) - \mathcal{A}(t)\|} - \cos \Delta^* \right)^2. \tag{5}$$

The left hand term is the cosine of the angular distance between robot i and its neighboring robot $N_i = (i+1) \mod N$, and Δ^* is the optimal angular separation.

Occlusion Avoidance. To avoid occlusions at time t, we construct the vector between the human's position $\mathcal{A}(t)$ and robot i's position on its initial trajectory. We then follow a sweeping motion as in Alg. 2 to generate a single visible region from the human's position. To minimise the risk of occlusions, we encourage the robot to align with the bisector of the visible region $\xi_i(t)$, which lies at angle $\beta_i(t)$. This is visualised in Fig. 5, and the corresponding cost term is given by:

$$J_i^{occ} = \left(\cos \beta_i(t) - \frac{(\mathbf{p}_i(t) - \mathcal{A}(t)) \cdot \xi_i(t)}{\|\mathbf{p}_i(t) - \mathcal{A}(t)\| \|\xi_i(t)\|} \right)^2. \tag{6}$$

Deviation in the FOV. We lose visibility of the human if it leaves a robot's FOV. Therefore, we add a cost term which

penalizes a robot if their gimbal angle $\theta_i^g(t)$ deviates away from the human. This encourages robots to keep the human in the center of their FOV:

$$J_i^{fov} = (\theta_i^g(t) - atan2(\mathbf{p}_i(t) - \mathcal{A}(t)))^2. \quad (7)$$

Given these four terms, the cost function J_i^c is written as follows, where the λ terms are positive weights:

$$J_i^c = \lambda_1 J_i^{ce} + \lambda_2 J_i^{sep} + \lambda_3 J_i^{occ} + \lambda_4 J_i^{fov}. \quad (8)$$

B. Trajectory Optimization Constraints

View Region Constraints. To maximize shot diversity, we constrain the robots to their visible regions. After synthesizing the optimal formation configurations in Sec. IV-B, we obtain precise angles for each robot in addition to their annular region. From these angles, we perform a sweeping motion similar to Alg. 2 to generate a single refined visible region for each robot with a maximum angle of θ_v, which is user-defined. To use this in the OCP in (2), we approximate these visible regions as convex pentagons using the method in [16], and constrain the robot within them. We demonstrate this process in Fig. 2(d).

Environmental Obstacle Avoidance Constraints. View regions constrain the robot positions, but do not guarantee obstacle avoidance. For this, we constrain robots to the safe corridors constructed from the initial trajectories in Sec. IV-C ($\mathcal{SC}^S$ in lines 5 and 6 of Alg. 1). In particular, we ensure that the complete footprint of each robot lies within their safe corridor at all times, similar to [20].

Human Distance Constraints. During execution, each robot must maintain an appropriate distance from the human:

$$\mathcal{G}_i^{dis} = \{d_{min}^a < \|\mathbf{p}_i(t) - \mathcal{A}(t)\| < d_{max}^a\}, \quad (9)$$

where d_{min}^a and d_{max}^a define the minimum and maximum distances from the human similar to annular regions.

Inter-Robot Collision Avoidance Constraints. To avoid collisions, robot footprints should never intersect. Therefore, we apply the collision avoidance constraints in [21], which check that each vertex of a robot's footprint lies outside the footprints of others.

C. Setting the Decision Variables per Trajectory Segment

Recall that in the OCP in (2) we split the human's trajectory into M segments, where segment k covers the trajectory from time T_k to T_{k+1}. In practice, we solve (2) as a nonlinear program (NLP). As a result, the integral in (2a) is solved over a set of discrete t_k which dictate the number of decision variables. For segment k we use $\frac{T_{k+1}-T_k}{\max_i |path_i^k|}$ discrete timesteps, where $\max_i |path_i^k|$ is the longest path length for the segment as computed by ECBS in Sec. IV-C. This allows longer paths to have more control inputs, which retains the precision of decision-making used in ECBS.

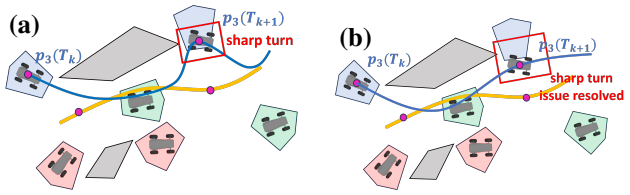

Fig. 6. Resolving sharp turns or kinematic infeasibility through constraint relaxation. (a) A trajectory with a sharp turn that is difficult to track. (b) The revised trajectory sacrifices visibility to remove the sharp turn.

D. Relaxing the View Region Constraints

In some instances, we may fail to find a solution for the OCP in (2). In dense, cluttered environments, robots may also have to execute very sharp turns to satisfy the view region constraints (see Fig. 6(a)). These turns may be difficult to track during execution. To address this, we iteratively relax the view region constraints in Alg. 1 (lines 8-10). The OCP in (2) has a view region constraint for each robot i and viewpoint k. We relax these constraints by reformulating them into the cost function using a small positive weight, where we penalize robot i if it deviates from its optimal configuration $\mathbf{p}_i^{opt}(T_k)$ for viewpoint k:

$$J_i^{soft,k} = \|\mathbf{p}_i(T_k) - \mathbf{p}_i^{opt}(T_k)\|. \quad (10)$$

Sharp turns occur if a robot's turning radius falls below a predefined threshold. Upon observing a sharp turn, we relax the constraint for the view region nearest that turn in time. Upon failing to solve (2) we relax the most violating view region constraint. This can be computed by evaluating the constraints along the infeasible trajectory. By relaxing these constraints into the cost function, we slightly sacrifice shot diversity to synthesize trajectories that can be easily tracked and are kinematically feasible. Fig. 6(b) illustrates the result of view region constraint relaxation.

VI. Experiments

In this section, we demonstrate the efficacy of our approach in simulation. All simulations are run on Ubuntu 20.04 with an Intel Core i9 processor @ 3.2GHz and 16GB of RAM. All software is implemented in C++ using the robot operating system (ROS) [22]. To solve the OCP in (2), we use the explicit Runge-Kutta method [23] to construct the NLP, CppAD [24] for automatic differentiation, and the primal-dual interior-point solver IPOPT [25] to solve the NLP. We use Gurobi [26] to solve the MINLP in (3). For our experiments, we simulate differential wheeled robots with a yaw-controllable gimbal in Gazebo. Experimental runs can be viewed in the supplementary video[2], and parameter settings are specified in the source code.

We compare our method against three representative state-of-the-art optimization-based methods [9], [13], [14]. With this, we aim to demonstrate improved tracking success and efficiency under our method. [9] tracks a single human using a leader-follower formation but ignores occlusions during planning. [13] mitigates occlusions by adjusting the

[2] https://youtu.be/JThzKUf0beA

TABLE I

COMPARISON AGAINST STATE-OF-THE-ART METHODS. N IS THE NUMBER OF ROBOTS, AND OBS IS THE NUMBER OF OBSTACLES.

N	Obs	30				60				90				120				150			
	Method	Ours	[9]	[13]	[14]	Ours	[9]	[13]	[14]	Ours	[9]	[13]	[14]	Ours	[9]	[13]	[14]	Ours	[9]	[13]	[14]
3	SR (%)	**100**	**100**	87	**100**	**100**	**100**	44	97	**100**	**100**	17	85	**100**	**100**	0	71	**100**	**100**	0	63
	VR (%)	**99.54**	96.61	98.09	99.14	98.32	89.16	95.67	**98.77**	98.17	85.17	91.43	**98.24**	97.26	81.35	–	**97.43**	96.89	77.40	–	**97.09**
	TL (m)	85.22	**83.94**	89.30	86.65	87.72	**86.13**	93.17	88.28	94.10	**90.47**	99.19	94.26	94.26	**91.25**	99.41	95.30	100.89	**96.67**	106.44	102.71
4	SR (%)	**100**	**100**	74	**100**	**100**	**100**	23	88	**100**	**100**	8	81	**100**	**100**	0	74	**100**	**100**	0	56
	VR (%)	**99.26**	94.50	98.05	98.80	**98.13**	88.76	96.66	98.01	97.48	83.56	95.51	**97.63**	96.18	81.16	–	**96.78**	95.76	75.34	–	**96.54**
	TL (m)	86.97	**85.04**	91.80	87.15	91.40	**90.49**	96.689	91.36	95.95	**93.79**	102.25	96.16	100.48	**98.05**	107.40	101.41	104.84	**103.14**	112.62	105.29

formation orientation in the initial trajectory but does not ensure topological equivalence with the human. [14] treats topological equivalence as a hard constraint during initial trajectory synthesis but does not consider inter-robot coordination. Each of these approaches are designed for aerial vehicles, and so we adapt their kinematic models and motion constraints to support ground robots. For fairness, we extend these methods to use gimbal-mounted cameras on the robots and incorporate the FOV constraint (7) into their respective cost functions.

For our experiments, we consider $50m \times 50m$ environments with 30, 60, 90, 120, and 150 randomly generated cuboid obstacles respectively, where obstacles are $1m \times 1m \times 2m$. We also consider formations of 3 and 4 robots, where each robot's footprint is a $1m \times 0.8m$ rectangle. For each number of robots and obstacles, we randomly generate 100 problem instances which are consistent between methods. Each problem instance considers a different human trajectory. For each problem, we allow each method 60 seconds to find a solution before recording a failure. For each method, we record the success rate (SR), i.e. the percentage of problems where a solution was found; the visibility ratio (VR), i.e. the proportion of time where the human is visible, averaged over all robots and problems; and the average trajectory length (TL). We present our results in Table I.

The probability of occlusion increases with the number of obstacles, reducing visibility for all four methods. Recall that [13] mitigates occlusions by rotating the formation. This is effective in sparse environments, but may cause failure as the environment becomes more cluttered. This is demonstrated in the 120 and 150 obstacle environments, where the success rate reaches zero. Further, these rotation maneuvers increase the trajectory length; [13] consistently synthesizes the longest trajectories. In comparison, [9] tracks the human without considering occlusions during planning. This often produces the shortest trajectories, but also results in the lowest visibility ratio out of the approaches that find a solution. Our method and [14] mitigate occlusions by maintaining topological equivalence with the human. This increases the trajectory length, but significantly improves the visibility ratio. [14] often produces slightly higher visibility rates than our approach. This is because we relax the topological equivalence constraint during ECBS if a solution cannot be found, and relax view region constraints if a solution cannot be found or a sharp turn is detected. However, because of this and the lack of robot coordination during initial trajectory synthesis in [14], our approach has a significantly higher success rate in many environments.

In Fig. 7 we show an example joint trajectory for each of the four experimental methods. Both our approach and [14] maintain topological equivalence with the human, avoiding occlusions and maximizing visibility. However, this occasionally requires robots to deviate around obstacles, slightly decreasing trajectory smoothness. [13] produces the least smooth trajectories due to frequent rotations for occlusion avoidance. Conversely, [9] produces the smoothest trajectories, though this is because of the lack of occlusion avoidance during planning. Both [9] and [13] suffer from occlusions, breaking visibility with the human (see the red lines in Fig. 7(a) and 7(b)). Each of the four methods aim to maintain the human in the center of the camera's FOV. However, [9] and [13] do not consider topological equivalence with the human during obstacle avoidance. This can cause abrupt changes in motion which the camera gimbal cannot promptly adapt to, resulting in the human moving away from the FOV center or even leaving the FOV entirely. Since our approach and [14] explicitly consider topological equivalence with the human, it is often straightforward to keep the camera pointed at the human. This is demonstrated by the lack of invisible points in Fig. 7(c) and 7(d). In addition, the view region constraints in our method position the human centrally within the FOV, improving the stability of footage captured by the camera.

VII. CONCLUSION

In this paper, we consider using multiple ground robots for autonomous tracking in cluttered environments. We present a novel planning framework for differential-drive robots equipped with movable cameras. Our framework begins by planning a sequence of viewpoints for each robot using annular regions. We then plan collision-free paths between each viewpoint which maintain topological equivalence with the human. These paths then form an initial guess for joint trajectory optimization, which synthesizes stable trajectories under the motion constraints of the robots and gimbals. Our framework maximizes shot diversity while maintaining safety and visibility. Though we have focused on cinematography domains, our approach can be applied to any target tracking problem. Our method assumes the environment and human trajectory are known, and relies on a centralized planner, which limits its applicability in dynamic, unknown, or large environments. Therefore, in future work, we will extend our method to *a priori* unknown environments and human trajectories, and explore decentralized planning methods to improve scalability. We will also deploy our approach on real robotic platforms.

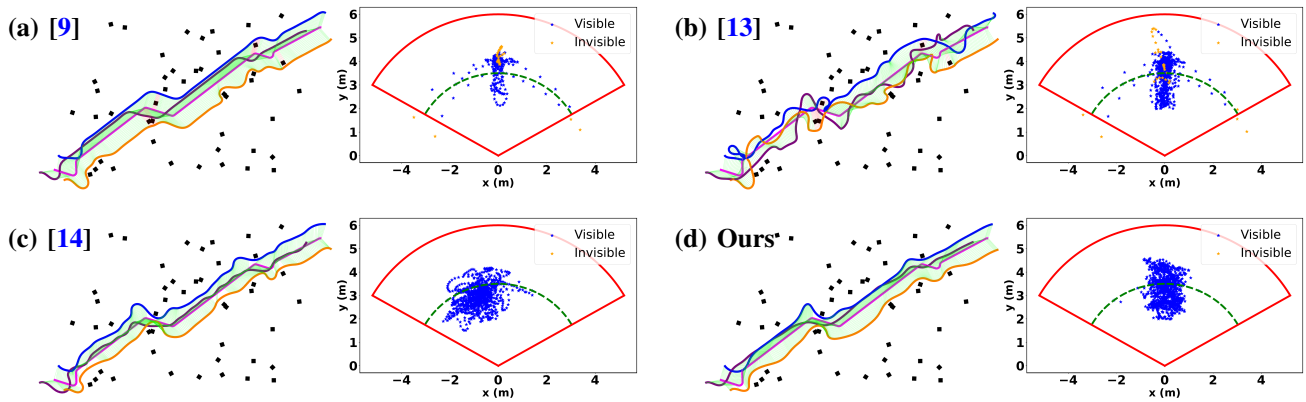

Fig. 7. Example trajectories synthesized by the experimental methods in an environment with 60 cuboid obstacles. The purple, orange, and blue lines show each robot's trajectory, and the pink line shows the human's trajectory. The green lines show unobstructed lines of sight, and the red lines show occluded lines of sight due to obstacles. To the right of each joint trajectory we show the spatial distribution of the human during execution relative to the robots' FOVs. The red sector shows the camera's FOV, and the green dashed arc shows the desired distance. Blue markers denote positions where the human was visible, and orange markers show instances where the human was invisible due to occlusions or being outside the FOV.

REFERENCES

[1] R. Tallamraju, E. Price, R. Ludwig, K. Karlapalem, H. H. Bülthoff, M. J. Black, and A. Ahmad, "Active perception based formation control for multiple aerial vehicles," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 4491–4498, 2019.

[2] A. Alcántara, J. Capitán, R. Cunha, and A. Ollero, "Optimal trajectory planning for cinematography with multiple unmanned aerial vehicles," *Robotics and Autonomous Systems*, vol. 140, p. 103778, 2021.

[3] A. Bucker, R. Bonatti, and S. Scherer, "Do you see what i see? coordinating multiple aerial cameras for robot cinematography," in *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pp. 7972–7979, IEEE, 2021.

[4] G. López-Nicolás, M. Aranda, and Y. Mezouar, "Adaptive multirobot formation planning to enclose and track a target with motion and visibility constraints," *IEEE Transactions on Robotics*, vol. 36, no. 1, pp. 142–156, 2019.

[5] J. Gemerek, B. Fu, Y. Chen, Z. Liu, M. Zheng, D. van Wijk, and S. Ferrari, "Directional sensor planning for occlusion avoidance," *IEEE Transactions on Robotics*, vol. 38, no. 6, pp. 3713–3733, 2022.

[6] K. Suresh, A. Rauniyar, M. Corah, and S. Scherer, "Greedy perspectives: Multi-drone view planning for collaborative perception in cluttered environments," in *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, pp. 10990–10997, IEEE, 2024.

[7] J. Ji, N. Pan, C. Xu, and F. Gao, "Elastic tracker: A spatio-temporal trajectory planner for flexible aerial tracking," in *Proceedings of the International Conference on Robotics and Automation (ICRA)*, pp. 47–53, IEEE, 2022.

[8] M. Barer, G. Sharon, R. Stern, and A. Felner, "Suboptimal variants of the conflict-based search algorithm for the multi-agent pathfinding problem," in *Proceedings of the International Symposium on Combinatorial Search*, vol. 5, pp. 19–27, 2014.

[9] X. Zhou, X. Wen, Z. Wang, Y. Gao, H. Li, Q. Wang, T. Yang, H. Lu, Y. Cao, C. Xu, *et al.*, "Swarm of micro flying robots in the wild," *Science Robotics*, vol. 7, no. 66, p. eabm5954, 2022.

[10] X. Xu, G. Shi, P. Tokekar, and Y. Diaz-Mercado, "Interactive multi-robot aerial cinematography through hemispherical manifold coverage," in *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, pp. 11528–11534, IEEE, 2022.

[11] T. Nägeli, L. Meier, A. Domahidi, J. Alonso-Mora, and O. Hilliges, "Real-time planning for automated multi-view drone cinematography," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 4, pp. 1–10, 2017.

[12] R. Tallamraju, N. Saini, E. Bonetto, M. Pabst, Y. T. Liu, M. J. Black, and A. Ahmad, "Aircaprl: Autonomous aerial human motion capture using deep reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6678–6685, 2020.

[13] C. Ho, A. Jong, H. Freeman, R. Rao, R. Bonatti, and S. Scherer, "3d human reconstruction in the wild with collaborative aerial cameras," in *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, pp. 5263–5269, IEEE, 2021.

[14] H. Wang, X. Zhang, Y. Liu, G. Sun, X. Zhang, and Y. Zhuang, "Pcdct: Perception-complementarity-driven collaborative trajectory generation for vision-based aerial tracking," *IEEE Transactions on Automation Science and Engineering*, 2025.

[15] M. Moussaïd, N. Perozo, S. Garnier, D. Helbing, and G. Theraulaz, "The walking behaviour of pedestrian social groups and its impact on crowd dynamics," *PloS one*, vol. 5, no. 4, p. e10047, 2010.

[16] Z. Zhang, Y. Zhong, J. Guo, Q. Wang, C. Xu, and F. Gao, "Auto filmer: Autonomous aerial videography under human interaction," *IEEE Robotics and Automation Letters*, vol. 8, no. 2, pp. 784–791, 2022.

[17] S. Liu, M. Watterson, K. Mohta, K. Sun, S. Bhattacharya, C. J. Taylor, and V. Kumar, "Planning dynamically feasible trajectories for quadrotors using safe flight corridors in 3-d complex environments," *IEEE Robotics and Automation Letters*, vol. 2, no. 3, pp. 1688–1695, 2017.

[18] C. D. Toth, J. O'Rourke, and J. E. Goodman, *Handbook of discrete and computational geometry*. CRC press, 2017.

[19] L. Yin, F. Zhu, Y. Ren, F. Kong, and F. Zhang, "Decentralized swarm trajectory generation for lidar-based aerial tracking in cluttered environments," in *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)*, pp. 9285–9292, IEEE, 2023.

[20] W. Zhang, C. Street, and M. Mansouri, "Multi-nonholonomic robot object transportation with obstacle crossing using a deformable sheet," in *2025 IEEE International Conference on Robotics and Automation*, IEEE, 2025.

[21] W. Zhang, C. Street, and M. Mansouri, "A decoupled solution to heterogeneous multi-formation planning and coordination for object transportation," *Robotics and Autonomous Systems*, vol. 180, p. 104773, 2024.

[22] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, A. Y. Ng, *et al.*, "Ros: An open-source robot operating system," in *Proceedings of the ICRA Workshop on Open Source Software*, vol. 3, p. 5, 2009.

[23] P. J. van der Houwen and B. P. Sommeijer, "Explicit runge–kutta (–nyström) methods with reduced phase errors for computing oscillating solutions," *SIAM Journal on Numerical Analysis*, vol. 24, no. 3, pp. 595–617, 1987.

[24] B. M. Bell, "Cppad: A package for c++ algorithmic differentiation," *Computational Infrastructure for Operations Research*, vol. 57, no. 10, p. 3, 2012.

[25] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, pp. 25–57, 2006.

[26] Gurobi Optimization, LLC, "Gurobi Optimizer Reference Manual," 2024.