Mixed-Integer Motion Planning for Moving-Target Traveling Salesman Problems with Dynamic Field of View

Journal Title
XX(X):1–18

DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

Runhui Long[1], Lekang Zhou[1], Anoop Bhat[3], Yanzhi Li[1], Bhaskar Vundurthy[3], Howie Choset[3], Zhongqiang Ren[1]

Abstract

This paper considers a Moving-Target Traveling Salesman Problem (MT-TSP), which seeks a 3D path for a drone to capture multiple targets that move along straight lines on the ground, and the drone is equipped with a downward pointing sensor whose field of view (FOV) depends on the altitude of the drone. A target is captured if that target is continuously within the FOV of the drone for a required minimum duration within a required time window. This problem generalizes several existing variants of MT-TSP and is challenging as it not only inherits the difficulty of determining an optimal visiting order of targets as in TSP, but also suffers from time-varying traversal cost between targets due to both the moving targets and the dynamic FOV that depends on the agent's motion. This paper develops exact methods using Mixed-Integer Conic Programs (MICP) to solve the problem to optimality based on both big-M constraints and perspective techniques inspired by the recent work on Graphs of Convex Sets (GCS). We prove the validity of both our big-M based and GCS based programs and their equivalence. Experimental results show that the GCS based MICP outperforms the big-M based MICP in both runtime and optimality gap as the number of targets increases from 5 to 15, especially for instances with more targets, with up to 85.2% of reduction in median runtime, and up to a 100.0% tighter average optimality gap when reaching the runtime limit.

Keywords

Mixed-Integer Programming, Motion Planning, Traveling Salesman Problem

1 Introduction

Given a set of stationary targets and the traversal cost between any pair of these targets, the well-known Traveling Salesman Problem (TSP) seeks a shortest path for an agent from the depot to visit all the targets exactly once and return to the depot. This paper considers a new variant of TSP, which we refer to as Dynamic Range Moving-Target TSP (DR-MT-TSP), where the agent with a maximum speed limit moves in a 3D space (with altitude), equipped with a downward pointing sensor whose detection range, i.e., the field of view (FOV), depends on the altitude of the agent (Fig. 1). In addition, each target moves along a known straight line in the horizontal plane, and must be captured within a certain time window and within a certain altitude window of the agent. A target is regarded as captured if the target is continuously within the FOV of the agent for a required minimum duration. Moving-Target TSP and its variants arise in applications such as defense of dynamic threats like Unmanned Aerial Vehicles (UAV) or hostile rockets (Helvig et al. 2003; Smith 2021; Stieber and Fügenschuh 2022), surveillance of multiple suspicious targets like UAV-based crowd monitoring or aircraft-based maritime patrol (de Moraes and de Freitas 2019; Wang and Wang 2023; Marlow et al. 2007), resupply of moving targets like a supply ship resupplying patrolling boats (Helvig et al. 2003), and planning of industrial robots like a robot arm collecting a number of moving point-objects (Chalasani and Motwani 1999).

DR-MT-TSP generalizes several existing variants of TSP: When the agent altitude is fixed and the FOV is a single point, DR-MT-TSP becomes the existing Moving-Target TSP (MT-TSP) (Stieber and Fügenschuh 2022). When all targets are stationary, and the FOV of the agent are circles with constant radii, one for each target, DR-MT-TSP becomes the existing Close-Enough TSP (CE-TSP) (Di Placido et al. 2023; Carrabs et al. 2017). TSP, MT-TSP and CE-TSP are all NP-hard (Helvig et al. 2003; Dumitrescu and Mitchell 2003) and so is DR-MT-TSP. Intuitively, DR-MT-TSP is computationally difficult as it not only inherits the fundamental challenge of determining an optimal visiting order of targets as in TSP, but also suffers from time-varying traversal cost between targets due to both the moving targets and the dynamic FOV that depends on the agent's motion.

Currently, we are not aware of any research on DR-MT-TSP. This paper, as a first attempt, seeks to develop an exact method to find an optimal solution for DR-MT-TSP. Of close relevance to DR-MT-TSP, MT-TSP has been studied a lot (Helvig et al. 2003; Chalasani and Motwani 1999; Hammar and Nilsson 2002; Hassoun et al. 2020; Philip et al.

[1]Shanghai Jiao Tong University, Shanghai, China

Corresponding author:
Zhongqiang Ren, Global College and Department of Automation, Shanghai Jiao Tong University, No.800 Dong Chuan Road, Minhang District, 200240 Shanghai, China
Email: {zhongqiang.ren}@sjtu.edu.cn

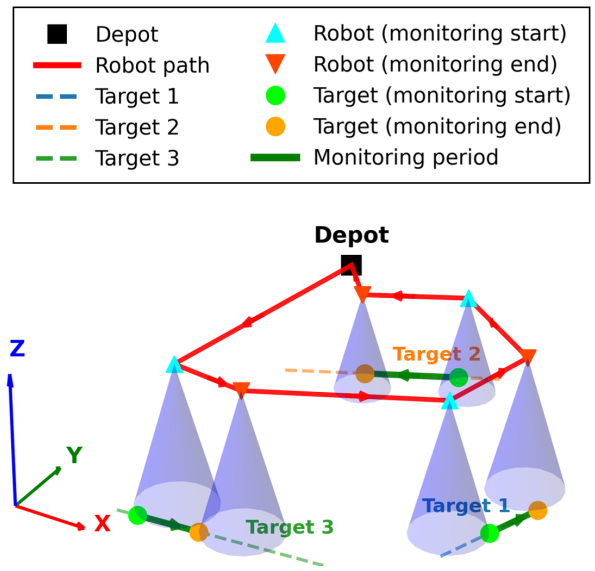

Figure 1. Illustration of DR-MT-TSP where a drone starts from the depot, visits multiple moving targets along known straight lines (in green) on the ground, and returns to the depot. The FOV of the drone depends on the altitude. Each target needs to be monitored for a minimum duration continuously with green and orange dots indicating the start and end of the monitoring of each target. The red line shows an optimal solution path for the drone.

2025a,b; Bhat et al. 2024, 2025a,b; Philip et al. 2024). In MT-TSP, the speed of the targets are generally assumed to be no greater than the agent's maximum speed (Helvig et al. 2003). Currently, the literature presents exact and approximation algorithms for some very restricted cases of the MT-TSP variants where the targets move in the same direction with the same speed (Chalasani and Motwani 1999; Hammar and Nilsson 2002), move along the same line (Helvig et al. 2003; Hassoun et al. 2020), or move along lines through the depot, towards or away from it (Helvig et al. 2003). Several heuristic based approaches have also been introduced in the literature (de Moraes and de Freitas 2019; Wang and Wang 2023; Marlow et al. 2007; Englot et al. 2013; Bourjolly et al. 2006; Choubey 2013; Groba et al. 2015; Jiang et al. 2005; Ucar and Isleyen 2019) that find feasible solutions, but give no information on how far they are from the optimum.

This paper is an extended version of our prior conference paper for MT-TSP (Philip et al. 2024), where we developed an efficient Mixed-Integer Second Order Conic Program (SOCP) leveraging the perspective technique used in the recent work on Graphs of Convex Sets (GCS) (Marcucci et al. 2024), based on the observation that each target's trajectory is a convex set in the space-time, and after combined with the dynamic FOV of the agent, the resulting set of possible positions for the agent to monitor the target is still a convex set. Our GCS based SOCP turns out to converge faster and be able to solve problems with more targets than a former big-M based formulation in the literature (Stieber and Fügenschuh 2022). Different from our prior conference paper (Philip et al. 2024), this paper

- develops both big-M and GCS based formulations for DR-MT-TSP.
- develops both big-M and GCS based formulations for Close-Enough Moving-Target TSP (CE-MT-TSP), which is a special case of DR-MT-TSP with constant FOV when capturing each target.
- conducts a comprehensive evaluation of big-M and GCS based formulations for MT-TSP, CE-MT-TSP and DR-MT-TSP and discuss their pros and cons for different problem variants.
- runs real robot tests for DR-MT-TSP with drones.

Our experimental results show that, the GCS based formulation outperforms the big-M based formulation in both runtime and optimality gap as the number of targets increases, with up to 85.2%, 74.4%, and 83.5% reduction in the median runtime for DR-MT-TSP, CE-MT-TSP, and MT-TSP, respectively. We also show that the GCS based formulation has a tighter optimality gap on average, especially for those hard instances with up to 15 targets where the average optimality gap of GCS based is up to 100.0% smaller than the big-M based formulation for all the three problems.

2 Related Work

2.1 Other Related Variants of TSP

Besides MT-TSP, there are many existing studies on variants of the TSP. Among them, CE-TSP is also closely related to this work, where a node is considered visited once the agent passes through a neighborhood set of that node (Carrabs et al. 2017). Various approaches have been developed to solve CE-TSP. Exact methods include branch-and-bound algorithms that incorporate improved search and pruning strategies (Zhang et al. 2023) and second-order cone programming (Gutow and Choset 2025). Heuristic approaches have also been proposed, such as evolutionary and geometry-based heuristics for constructing high-quality tours (Cariou et al. 2023). Genetic algorithm (GA) has been used to address generalized versions of CE-TSP, where each node is associated with a set of disks of different radii (Di Placido et al. 2023). Another variant of CE-TSP that considers CE-TSP in environments with obstacles has been studied, where improved Mixed Integer Non-Linear Programming (MINLP)-based heuristics are designed to ensure feasibility while navigating around obstacles (Deckerová et al. 2023).

Besides CE-TSP, the time-dependent TSP and the TSP with time windows incorporate time-varying travel costs and temporal feasibility constraints, and their approximability properties have been systematically characterized under a unified definition framework (Saller et al. 2025). The Variable-Speed TSP (VS-TSP) considers curvature-constrained vehicles whose speed can vary along the tour, and trajectory-optimization techniques are used to minimize the total travel time (Kučerová et al. 2021). The TSP-related Coverage Path Planning (CPP) integrates region coverage with tour planning. A grid-based approach and a dynamic programming based approach are introduced to find the optimal solution (Xie et al. 2019), while a Reinforcement Learning (RL) based algorithm is proposed by applying the cellular decomposition methods and recursively solving each

decomposed cell formulated as TSP to generate the coverage path (Kyaw et al. 2020).

MT-TSP variants also arise in patrolling-related problems. To name a few, hybrid GA and simulated annealing heuristic methods were developed to consider dynamically moving targets with distinct threat values, and seeks to minimize the cumulative threat (Wang and Wang 2023). Another example of TSP for patrolling is a traffic patrolling routing problem with drones (TPRP-D) that considers heterogeneous tasks which are points and edges patrolled by a coordinated truck-and-drone team within a road network, and a two-stage heuristic has been proposed to minimize the total completion time (Luo et al. 2019).

2.2 Dynamic Field Of Views

Dynamic sensor FOV was considered in other planning problems. View-planning and active-vision surveys show that sensor pose and FOV choices can be optimized for reconstruction, inspection, and recognition tasks (Zeng et al. 2020). Information-driven planners further embed the sensor's measurement volume into the objective, demonstrating that adjusting the FOV via orientation, zoom, or altitude can improve information gathering (Zhu et al. 2021). Other studies formalize more general sensing geometries, such as rectangular or feature-constrained frusta, enabling their direct use in viewpoint selection (Magaña et al. 2023). In UAV inspection and reconstruction, the benefits of integrating FOV geometry into trajectory design are shown by geometric-primitive-guided and view-aware path planners (Zhou et al. 2023). Finally, zoom-aided coverage and pan-tilt-zoom (PTZ) camera planning illustrate that dynamic FOV control can be scheduled alongside motion to satisfy coverage or quality requirements (Rios et al. 2025).

Besides planning, dynamic FOV has also been studied in several other fields. To name a few, in virtual reality (VR) and head-mounted display (HMD) research, dynamically restricting the user's FOV has been shown to mitigate cybersickness while maintaining presence, motivating software-level FOV modulation techniques (Fernandes and Feiner 2016). Beyond comfort, VR research has also studied how FOV size and the motion of targets influence search performance, particularly for targets that lie outside the user's view (Grinyer and Teather 2022). In surveillance and camera-network literature, PTZ systems treat the FOV as a time-varying system variable, leading to calibration and scheduling algorithms that explicitly model how pan, tilt, and zoom affect the camera frustum and its ground footprint (Song et al. 2021). In visual-inertial odometry and simultaneous localization and mapping (SLAM), wide and variable FOV sensors such as fisheye or omnidirectional cameras require dedicated projection and odometry models to handle changing image overlap and coverage (Xie et al. 2024).

3 Problem Description

Consider a bounded 3D Euclidean workspace $\mathcal{W} = [-L_x/2, L_x/2] \times [-L_y/2, L_y/2] \times [0, L_z] \subseteq \mathbb{R}^3$, where each point has coordinates $(x, y, z) \in \mathbb{R}^3$. Let $V_{tar} = \{1, 2, \cdots, N_{tar}\}$ denote a set of N_{tar} moving targets in the 2D Euclidean plane on the ground with $z = 0$.

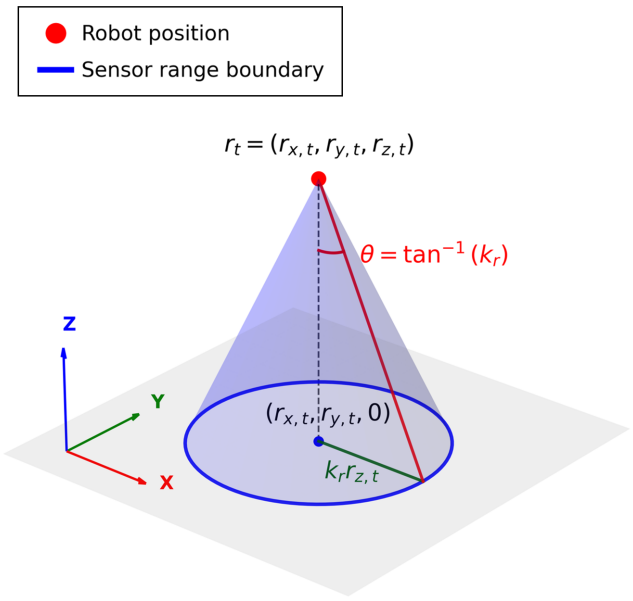

Figure 2. Illustration of the dynamic sensor detection range.

Each target $u \in V_{tar}$ is associated with a time window $[\underline{t}_u, \overline{t}_u]$ within which the robot must catch the target. Each target $u \in V_{tar}$ is also associated with a height window $[\underline{z}_u, \overline{z}_u]$, within which the robot is allowed to catch the target. Each target moves along a straight line with known speed. The coordinates of a target $u \in V_{tar}$ at time points $\underline{t}_u$ are $(\underline{x}_u, \underline{y}_u, 0)$. The velocity of a target $u \in V_{tar}$ is denoted as $(\xi_{x,u}, \xi_{y,u}, 0) \in \mathbb{R}^3$.

The robot dynamics is first-order integrator in 3D with speed limit $v_{\max}$. Let $r_t = (r_{x,t}, r_{y,t}, r_{z,t})$ denote the position of the robot at time point t. The robot is equipped with a downward facing sensor with a dynamic FOV related to the robot dynamics. At any time t, the FOV is a circle $S(t) \subseteq \mathbb{R}^2$ on the ground centered at $(r_{x,t}, r_{y,t}, 0)$ with radius $k_r r_{z,t}$ where k_r is a known positive real number that characterizes the view angle of the sensor (Fig. 2). Let a non-negative real number $\tau_m \geq 0$ denote the minimum monitoring duration for each target. The robot *catches* a target $u \in V_{tar}$ if there exists a time period no smaller than τ_m such that the target is always within the robot's FOV during that time period.

Let $o = (o_x, o_y, o_z) \in \mathbb{R}^3$ denote the depot from which the robot starts its motion. Let $o' = o$ denote a copy of o with the same coordinate to which the robot must end its motion after catching all moving targets. Let $V = V_{tar} \cup \{o, o'\}$. The time window of o is $\underline{t}_o = \overline{t}_o = 0$ since the robot starts from the depot at time point $t = 0$. The time window of o' is $[0, T]$ with T being the time horizon, so that the robot can return to o' at any time within $[0, T]$. The velocities of o, o' are all zero since they are static points. The height windows of o, o' are both $\underline{z}_o = \overline{z}_o = o_z$ since the robot must start from and return to the exact position of the depot. A feasible tour of the robot starts from o, catches all targets in V_{tar} within their corresponding time windows, and return to o'.

Problem 1. *The goal of the Dynamic Range Moving Target Traveling Salesman Problem (DR-MT-TSP) is to find a feasible tour that minimizes the robot's total travel distance*

among targets, excluding the distance traveled while the robot is monitoring the targets.

Problem 2. *When (i) the robot is limited to a 2D plane wihtout considering its z coordinate (i.e., the depot $o_z = 0$ and the robot $r_{z,t} = 0$ at all time $t \in [0, T]$), and (ii) the sensor FOV is a known and fixed circle with radius $r_c \geq 0$ for each target, DR-MT-TSP becomes the Close-Enough Moving Target Traveling Salesman Problem (CE-MT-TSP).*

Problem 3. *When (i) the robot is limited to a 2D plane wihtout considering its z coordinate, (ii) there is no requirement on monitoring (i.e., the robot catches each target at a specific time instant), and (iii) the sensor range is always a point ($k_r = 0$), DR-MT-TSP becomes the regular Moving Target Traveling Salesman Problem (MT-TSP).*

4 Preliminaries

This section presents the existing mixed-integer conic programming (MICP) formulation for the MT-TSP using big-M constraints (Stieber and Fügenschuh 2022). Let $G = (V, E)$ denote a directed graph that is constructed as follows. The vertex set V is same as the aforementioned $V = V_{tar} \cup \{o, o'\}$ and we therefore use the same notation. For the edges E,

- there is an edge from node o to all nodes in V_{tar},
- there is an edge from each node in V_{tar} to every other node in V_{tar},
- and finally, there is an edge from every node in V_{tar} to node o'.

For any node $u \in V_{tar}$, let E_u^{in} and E_u^{out} denote the sets of all edges entering and exiting u, respectively.

Next, we describe the decision variables in this big-M based MICP formulation. For each node $u \in V$, let t_u, a real number, represent the time point at which the robot visits u. For each edge $e = (u, v) \in E$, let $b_e \in \{0, 1\}$ denote a binary variable that indicates whether or not edge e is chosen in the solution. Let $l(e) \geq 0$ denote the Euclidean distance between node u and node v for each $e = (u, v) \in E$, which will be used to define the second-order cone constraints. Let auxiliary variable $\tilde{l}(e) \geq 0$ for each $e = (u, v) \in E$ represent $b_e l(e)$, i.e. $\tilde{l}(e)$ equals to $l(e)$ if edge e is selected otherwise zero.

Let $R = \sqrt{L_x^2 + L_y^2}$ denote the length of the diagonal of the square area containing the depot and all the target trajectories. The fact that the Euclidean length of any line segment within the square area cannot exceed R will be used in formulating one of the big-M constraints in the formulation. Now, the MICP formulation for the MT-TSP is as follows:

$$\min \sum_{e=(u,v)\in E} \tilde{l}(e) \tag{1}$$

subject to constraints

$$\sum_{e\in E_s^{out}} b_e = 1, \tag{2}$$

$$\sum_{e\in E_{s'}^{in}} b_e = 1, \tag{3}$$

$$\sum_{e\in E_u^{in}} b_e = 1, \ \forall\, u \in V_{tar}, \tag{4}$$

$$\sum_{e\in E_u^{in}} b_e = \sum_{e\in E_u^{out}} b_e, \ \forall\, u \in V_{tar}, \tag{5}$$

$$\underline{t}_u \leq t_u \leq \bar{t}_u, \ \forall u \in V, \tag{6}$$

$$x_{c,u} = \underline{x}_u + \xi_{x,u}(t_u - \underline{t}_u), \ \forall u \in V, \tag{7}$$

$$y_{c,u} = \underline{y}_u + \xi_{y,u}(t_u - \underline{t}_u), \ \forall u \in V, \tag{8}$$

$$(x_{c,v} - x_{c,u})^2 + (y_{c,v} - y_{c,u})^2 \leq l(e)^2, \ \forall e = (u, v) \in E, \tag{9}$$

$$l(e) = \tilde{l}(e) + R(1 - b_e), \ \forall e = (u, v) \in E, \tag{10}$$

$$\tilde{l}(e) \leq v_{\max}(t_v - t_u + T(1 - b_e)), \ \forall e = (u, v) \in E. \tag{11}$$

The objective (1) is to minimize the total tour length of the robot. The condition that the robot departs from the depot once, and arrives at the depot's copy once, is described by (2) and (3) respectively. The constraints (4) ensure that each target is visited exactly once by the robot, and the flow conservation for all the target nodes are ensured by (5). Constraints (2) to (5) ensure a valid path for the robot that starts at the depot, visits each target exactly once, and returns to the depot. Constraints (2) to (5) will re-appear in all formulations in this article. Constraint (6) requires the robot to visit each node within its time-window. The position where each target is caught by the robot is described by (7) and (8). Here, the subscripts c, u in $x_{c,u}, y_{c,u}$ means the x, y coordinates when the robot catches target u. Constraint (9) describes the second-order cone constraints, and the big-M constraints (10) ensure that $\tilde{l}(e)$ is equal to $l(e)$ if $b_e = 1$, and $\tilde{l}(e)$ is free to take any value if $b_e = 0$. The other big-M constraints (11) ensure that the speed limit will only take effect for an edge e if $b_e = 1$. Based on this big-M formulation for MT-TSP, we will then present our formulations for DR-MT-TSP and CE-MT-TSP.

5 Methods: Big-M Based Formulations

5.1 Big-M Based Formulation for DR-MT-TSP

This section presents our big-M formulation for the DR-MT-TSP (Fig. 3).

5.1.1 Decision Variables and Auxiliary Variables Recall that DR-MT-TSP requires that each target $u \in V_{tar}$ is monitored for a minimum duration τ_m. For each $u \in V_{tar}$, let the real variables t_u^s and t_u^f represent the time at which the robot starts to monitor u (i.e., "enters" u) and finishes monitoring u (i.e., "leaves" u), respectively. In addition, let t_o^f denote the time when the robot leaves the depot and starts its path, and let $t_{o'}^s$ denote the time when the robot enters the depot after visiting all targets. Similarly, let $(x_{c,u}^s, y_{c,u}^s, z_{c,u}^s)$ and $(x_{c,u}^f, y_{c,u}^f, z_{c,u}^f)$ denote the positions where the robot enters u at time t_u^s and leaves u at time t_u^f, respectively. Besides, let $(x_{c,o}^f, y_{c,o}^f, z_{c,o}^f)$ and $(x_{c,o'}^s, y_{c,o'}^s, z_{c,o'}^s)$ denote the positions where the robot leaves the depot to starts its path and enters the depot after visiting all the targets, respectively. We also introduce variables δ_u^t, δ_u^x, δ_u^y, and

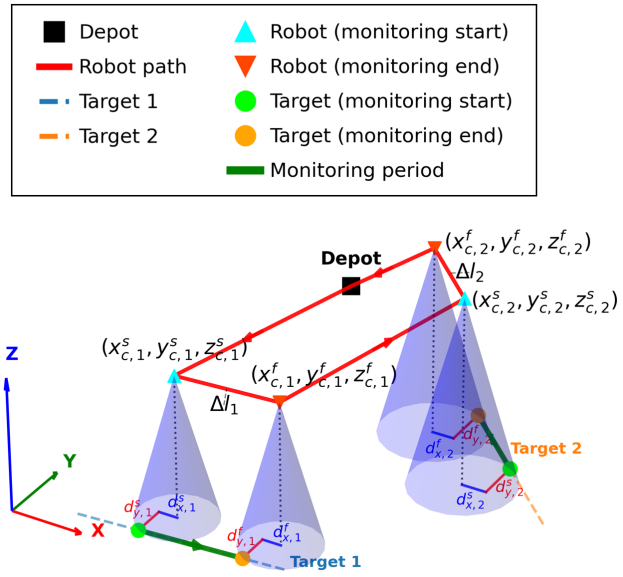

Figure 3. Illustration of the variable definitions of big-M based based formulations for DR-MT-TSP.

δ_u^z to denote the time and position difference between the robot enters and leaves u, which are subject to the following constraints:

$$t_u^f - t_u^s = \delta_u^t, \quad \forall u \in V_{tar}, \tag{12}$$
$$x_{c,u}^f - x_{c,u}^s \leq \delta_u^x, \quad \forall u \in V_{tar}, \tag{13}$$
$$x_{c,u}^f - x_{c,u}^s \geq -\delta_u^x, \quad \forall u \in V_{tar}, \tag{14}$$
$$y_{c,u}^f - y_{c,u}^s \leq \delta_u^y, \quad \forall u \in V_{tar}, \tag{15}$$
$$y_{c,u}^f - y_{c,u}^s \geq -\delta_u^y, \quad \forall u \in V_{tar}, \tag{16}$$
$$z_{c,u}^f - z_{c,u}^s \leq \delta_u^z, \quad \forall u \in V_{tar}, \tag{17}$$
$$z_{c,u}^f - z_{c,u}^s \geq -\delta_u^z, \quad \forall u \in V_{tar}. \tag{18}$$

In addition, let variable $\Delta l_u \geq 0$ represent the Euclidean distance that the robot travels when monitoring the target u, which satisfies:

$$(\delta_u^x)^2 + (\delta_u^y)^2 + (\delta_u^z)^2 \leq \Delta l_u^2, \quad \forall u \in V_{tar}, \tag{19}$$
$$\Delta l_u \leq v_{\max} \delta_u^t, \quad \forall u \in V_{tar}, \tag{20}$$
$$\delta_u^t \geq \tau_m, \quad \forall u \in V_{tar}. \tag{21}$$

Here, the monitoring distance is described by second-order cone constraints (19), and the speed limit is enforced by constraints (20). The minimum monitoring duration requirement for each target is ensured by constraints (21).

Recall that in DR-MT-TSP, the robot has the dynamic sensor FOV. Let $(d_{x,u}^s, d_{y,u}^s)$ and $(d_{x,u}^f, d_{y,u}^f)$ denote the dynamic sensor range in x and y directions when the robot enters and leaves $u \in V_{tar}$, respectively. For consistency, we also define $(d_{x,o}^f, d_{y,o}^f)$ for o and $(d_{x,o'}^s, d_{y,o'}^s)$ for o' which all equal to zero since there is no dynamic FOV at the depot. Finally, let a non-negative integer u_i denote the ordering index when each target $i \in V_{tar}$ is visited, which is used to construct the Miller-Tucker-Zemlin (MTZ) subtour elimination constraints when $\tau_m = 0$. We explain the reason for using these subtour elimination constraints in DR-MT-TSP in the following section. The rest of the variables are the same as MT-TSP.

5.1.2 Big-M Based MICP Formulation Our big-M based MICP formulation for the DR-MT-TSP is as follows:

$$\min \sum_{e=(u,v)\in E} \tilde{l}(e) \tag{22}$$

subject to constraints (2), (3), (4), (5), (12), (13), (14), (15), (16), (17), (18), (19), (20), (21),

$$\underline{t}_u \leq t_u^f \leq \bar{t}_u, \quad \forall u \in V_{tar} \cup \{o\}, \tag{23}$$
$$\underline{t}_u \leq t_u^s \leq \bar{t}_u, \quad \forall u \in V_{tar} \cup \{o'\}, \tag{24}$$
$$x_{c,u}^f = \underline{x}_u + \xi_{x,u}(t_u^f - \underline{t}_u) + d_{x,u}^f, \quad \forall u \in V_{tar} \cup \{o\}, \tag{25}$$
$$x_{c,u}^s = \underline{x}_u + \xi_{x,u}(t_u^s - \underline{t}_u) + d_{x,u}^s, \quad \forall u \in V_{tar} \cup \{o'\}, \tag{26}$$
$$y_{c,u}^f = \underline{y}_u + \xi_{y,u}(t_u^f - \underline{t}_u) + d_{y,u}^f, \quad \forall u \in V_{tar} \cup \{o\}, \tag{27}$$
$$y_{c,u}^s = \underline{y}_u + \xi_{y,u}(t_u^s - \underline{t}_u) + d_{y,u}^s, \quad \forall u \in V_{tar} \cup \{o'\}, \tag{28}$$
$$\underline{z}_u \leq z_{c,u}^f \leq \bar{z}_u, \quad \forall u \in V_{tar} \cup \{o\}, \tag{29}$$
$$\underline{z}_u \leq z_{c,u}^s \leq \bar{z}_u, \quad \forall u \in V_{tar} \cup \{o'\}, \tag{30}$$
$$d_{x,o}^f = d_{y,o}^f = 0, \tag{31}$$
$$d_{x,o'}^s = d_{y,o'}^s = 0, \tag{32}$$
$$(d_{x,u}^f)^2 + (d_{y,u}^f)^2 \leq (k_r z_{c,u}^f)^2, \quad \forall u \in V_{tar}, \tag{33}$$
$$(d_{x,u}^s)^2 + (d_{y,u}^s)^2 \leq (k_r z_{c,u}^s)^2, \quad \forall u \in V_{tar}, \tag{34}$$
$$(x_{c,v}^s - x_{c,u}^f)^2 + (y_{c,v}^s - y_{c,u}^f)^2 + (z_{c,v}^s - z_{c,u}^f)^2 \leq l(e)^2, \quad \forall e = (u,v) \in E, \tag{35}$$
$$l(e) = \tilde{l}(e) + R_c(1 - b_e), \quad \forall e = (u,v) \in E, \tag{36}$$
$$\tilde{l}(e) \leq v_{\max}\big(t_v^s - t_u^f + T(1 - b_e)\big), \quad \forall e = (u,v) \in E, \tag{37}$$
$$u_i - u_j + N_{tar} b_e \leq N_{tar} - 1, \quad \forall e = (i,j) \in E,\ i,j \in V_{tar}, \quad \text{(applied only if } \tau_m = 0\text{)} \tag{38}$$
$$1 \leq u_i \leq N_{tar}, \quad \forall i \in V_{tar}, \quad \text{(applied only if } \tau_m = 0\text{)}. \tag{39}$$

This formulation shares constraints (2) to (5) from the MT-TSP MICP formulation and (12) to (21) mentioned before. The condition requiring the robot to leave and enter each node within its time window is now given by (23) and (24). Constraints (7) and (8) from the MT-TSP big-M based formulation are modified to constraints (25) to (28), where the x, y positions of the robot entering and leaving each node are defined, and the dynamic sensor ranges are added. Constraints (29) and (30) ensure that each target is visited within its height window. Constraints (31) and (32) restrict the dynamic sensor range variables $d_{x,o}^f$, $d_{y,o}^f$, $d_{x,o'}^s$, and $d_{y,o'}^s$ for o and o' to zero. Recall that we define the velocities $\xi_{x,o}$, $\xi_{y,o}$, $\xi_{x,o'}$ and $\xi_{y,o'}$ as zeros, and the height window of o and o' as $\underline{z}_o = \bar{z}_o = \underline{z}'_o = \bar{z}'_o = o_z$. These definitions along with constraints (25) to (32) ensure that $(x_{c,o}^f, y_{c,o}^f, z_{c,o}^f)$ and $(x_{c,o'}^s, y_{c,o'}^s, z_{c,o'}^s)$ are equal to the exact position of the depot. Constraints (33) and (34) describe the second-order cone constraints, ensuring the target is within the sensor detection range of the robot.

Constraint (9) from the MT-TSP big-M based MICP formulation is modified to (35), describing the distance traveled by the robot between it leaves the last target and enters the next target. The big-M constraint (36) makes $\tilde{l}(e)$ equal to $l(e)$ if $b_e = 1$, and the parameter $R_c = \sqrt{L_x^2 + L_y^2 + L_z^2}$ denotes the length of the diagonal of the cuboid workspace, which ensures that $\tilde{l}(e)$ is free to take any value if $b_e = 0$. The other big-M constraint (37) ensures that the speed limit will only be placed if $b_e = 1$. The MTZ subtour elimination constraints (38) and (39), which are only applied if $\tau_m = 0$, enforce u_j greater than u_i by one if $b_e = 1$, ensuring node v is visited after node u and thereby preventing subtours.

5.1.3 Dynamic FOV Constraints The dynamic FOV constraints (33) and (34) are enforced only at the entering and leaving instants of the monitoring interval, which is sufficient to guarantee that the target remains within the robot's sensing cone during the entire interval $[t_u^s, t_u^f]$. This result follows from the fact that all targets move along straight lines and the convexity of the FOV of the robot.

Lemma 1. *If the dynamic FOV constraints (33) and (34) are satisfied at time points t_u^s, t_u^f respectively, then the target must remains within the robot FOV during the entire time interval $[t_u^s, t_u^f]$.*

Proof. Let $\mathbf{p}_{c,u}^s = (x_{c,u}^s, y_{c,u}^s, z_{c,u}^s)^\top$ and $\mathbf{p}_{c,u}^f = (x_{c,u}^f, y_{c,u}^f, z_{c,u}^f)^\top$ denote the robot positions when the robot starts and ends monitoring target u, respectively. The target position $\mathbf{p}_{t,u}(t)$ moves along a straight line:

$$\mathbf{p}_{t,u}(t) = \underline{\mathbf{p}}_u + \boldsymbol{\xi}_u(t - \underline{t}_u),$$

where $\underline{\mathbf{p}}_u = (\underline{x}_u, \underline{y}_u, 0)^\top$ and $\boldsymbol{\xi}_u = (\xi_{x,u}, \xi_{y,u}, 0)^\top$ are the initial position and velocity of the target u. For any $\theta \in [0,1]$ and the corresponding time $t = t_u^s + \theta(t_u^f - t_u^s)$, the robot and target positions are

$$\begin{aligned}\mathbf{p}_{c,u}(t) &= \mathbf{p}_{c,u}^s + \theta(\mathbf{p}_{c,u}^f - \mathbf{p}_{c,u}^s),\\ \mathbf{p}_{t,u}(t) &= (1-\theta)\,\mathbf{p}_{t,u}(t_u^s) + \theta\,\mathbf{p}_{t,u}(t_u^f).\end{aligned}$$

Correspondingly, the relative vector $\mathbf{d}_u(t) = \mathbf{p}_{t,u}(t) - \mathbf{p}_{c,u}(t)$ and robot height $z_{c,u}(t)$ can be expressed as follows:

$$\begin{aligned}\mathbf{d}_u(t) &= (1-\theta)\mathbf{d}_u^s + \theta\mathbf{d}_u^f,\\ z_{c,u}(t) &= (1-\theta)z_{c,u}^s + \theta z_{c,u}^f,\end{aligned}$$

where $\mathbf{d}_u^s = \mathbf{p}_{t,u}(t_u^s) - \mathbf{p}_{c,u}^s$ and $\mathbf{d}_u^f = \mathbf{p}_{t,u}(t_u^f) - \mathbf{p}_{c,u}^f$.

The sensing cone of the robot is defined by the convex set

$$\mathcal{C} = \{(d_x, d_y, z) \mid d_x^2 + d_y^2 \le (k_r z)^2,\ z \ge 0\}.$$

If both boundary points satisfy $(\mathbf{d}_u^s, z_{c,u}^s) \in \mathcal{C}$ and $(\mathbf{d}_u^f, z_{c,u}^f) \in \mathcal{C}$, then for any $\theta \in [0,1]$, using the triangle inequality and the boundary conditions,

$$\begin{aligned}\|\mathbf{d}_u(t)\|_2 &= \left\|(1-\theta)\mathbf{d}_u^s + \theta\mathbf{d}_u^f\right\|_2\\ &\le (1-\theta)\|\mathbf{d}_u^s\|_2 + \theta\|\mathbf{d}_u^f\|_2\\ &\le (1-\theta)k_r z_{c,u}^s + \theta k_r z_{c,u}^f = k_r z_{c,u}(t). \quad (40)\end{aligned}$$

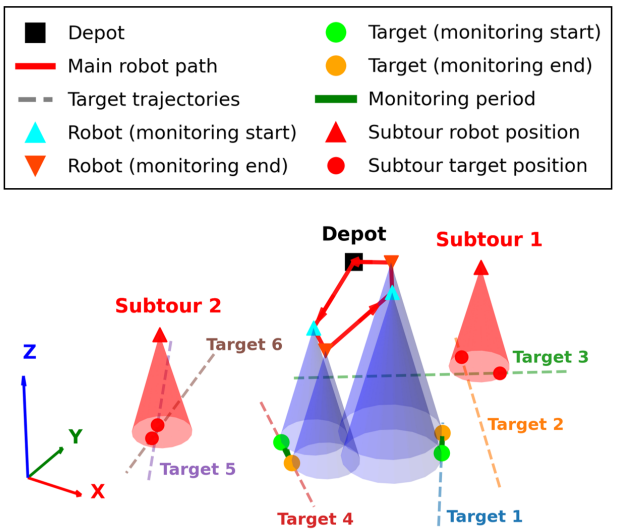

Figure 4. A toy example of DR-MT-TSP that leads to subtour (and hence invalid solution) when there is no MTZ subtour elimination constraints. The robot may catch more than one target at the same time point due to its sensor FOV. E.g., the robot catches targets 5 and 6 in Subtour 2.

Therefore, $(\mathbf{d}_u(t), z_{c,u}(t)) \in \mathcal{C}$ for all $\theta \in [0,1]$, i.e., $(d_x(t))^2 + (d_y(t))^2 \le (k_r z_{c,u}(t))^2$ for all $t \in [t_u^s, t_u^f]$. Consequently, checking the dynamic FOV constraints only at the two boundary instants is sufficient to ensure that the target is within the sensor FOV at any intermediate time point inbetween.

5.1.4 Subtour Elimination Constraints The subtour elimination constraints (38) and (39) is often unnecessary in MT-TSP. In MT-TSP, each edge $e = (u, v)$ connects two locations where the robot catches targets u and v. Only when u, v coincide (i.e., these two locations exactly coincide at the same position at the same time point), then $\tilde{l}(e) = 0$. Otherwise, in general, $\tilde{l}(e) > 0$. Combined with constraint (11), t_v is strictly larger than t_u for any pair of targets. As a result, the time point t_v when the robot catches any target v must be different from t_u for any other target u, which inherently eliminates subtours in the solution.

However, in DR-MT-TSP when $\tau_m = 0$, the subtour elimination constraints are necessary, since the robot has a FOV and it is possible that two targets u, v are caught at the same time point (Fig. 4). For an edge $e = (u, v)$, it is possible that the robot positions $(x_{c,u}^f, y_{c,u}^f, z_{c,u}^f)$ and $(x_{c,v}^s, y_{c,v}^s, z_{c,v}^s)$ simultaneously satisfy the constraints for two distinct targets u and v (Constraints (25)-(30), (33) and (34)), as long as u and v are within the FOV of the robot, which then leads to $\tilde{l}(e) = 0$ and $t_u^f = t_v^s$.

Additionally, when the minimum monitoring duration $\tau_m = 0$, Constraints (12) and (21) only place restrictions on t_u^s and t_u^f as $t_u^f - t_u^s \ge 0$, which allows $t_u^s = t_u^f$. Then, for each edge $e = (u, v)$, the constraints do not forbid the case

$$t_u^s = t_u^f = t_v^s = t_v^f.$$

which leads to $\tilde{l}(u, v) = 0$ and $\tilde{l}(v, u) = 0$, making both directed edges (u, v) and (v, u) feasible at zero cost. As a result, the solution may contain one or more subtours

$\{(u,v),(v,u)\}$, if targets u and v are both within the robot sensor range at some time point.

Fig.4 visualizes such a situation: the red cones represent the robot positions where the robot catches two targets simultaneously, forming subtours which are actually infeasible solutions, since the robot cannot "teleport" across subtours. Therefore, we need to add MTZ subtour elimination constraints (38) and (39) to prevent subtours for DR-MT-TSP when $\tau_m = 0$.

In contrast, when $\tau_m > 0$, constraints (12) and (21) enforce $t_u^f - t_u^s \geq \tau_m$, i.e. $t_u^f > t_u^s$. For an inter-target edge $e = (u,v)$, the non-negativity of $\tilde{l}(e)$ enforces $t_v^s \geq t_u^f$ by constraints (37). Therefore, the time t_v^f when the robot leaves v must satisfy $t_v^f > t_v^s \geq t_u^f > t_u^s$, which implies that for any pair (u,v), the reversed edge (v,u) (which requires $t_u^s \geq t_v^f$) is prevented. In other words, $\tilde{l}(v,u)$ becomes infeasible whenever $\tilde{l}(u,v)$ is valid. The requirement for a positive minimum monitoring duration naturally achieve subtour elimination and thereby the MTZ subtour elimination constraints are unnecessary when $\tau_m > 0$.

We summarize the above analysis with the following theorem.

Lemma 2. *When $\tau_m = 0$, the subtour elimination constaints are necessary for DR-MT-TSP. When $\tau_m > 0$, the subtour elimination constaints are no more needed for DR-MT-TSP.*

5.2 Big-M Based Formulation for CE-MT-TSP

This section presents the big-M based MICP formulation for the CE-MT-TSP. CE-MT-TSP can be seemed as a 2D more restriced version of the DR-MT-TSP, so most of the constraints in MICP formulation for DR-MT-TSP can be reused and some of the constraints need to be slightly modified. The formulation is presented below:

$$\min \sum_{e=(u,v)\in E} \tilde{l}(e) \tag{41}$$

subject to constraints (2), (3), (4), (5), (12), (13), (14), (15), (16), (20), (21), (23), (24), (25), (26), (27), (28), (31), (32), (36), (37), (38), (39),

$$(\delta_u^x)^2 + (\delta_u^y)^2 \leq \Delta l_u^2, \quad \forall u \in V, \tag{42}$$
$$(d_{x,u}^f)^2 + (d_{y,u}^f)^2 \leq (r_c)^2, \quad \forall u \in V_{tar}, \tag{43}$$
$$(d_{x,u}^s)^2 + (d_{y,u}^s)^2 \leq (r_c)^2, \quad \forall u \in V_{tar}, \tag{44}$$
$$(x_{c,v}^s - x_{c,u}^f)^2 + (y_{c,v}^s - y_{c,u}^f)^2 \leq l(e)^2, \quad \forall e = (u,v) \in E. \tag{45}$$

Compared with the big-M based MICP formulation for DR-MT-TSP, the major differences are the modifications from constraints (33) and (34) to (43) and (44), where the target is restricted within the fixed circle with radius r_c instead of the sensor FOV. Besides, constraints (19) and (35) are modified to (42) and (45), respectively, where the terms of z direction are deleted. In addition, constraints (17), (18), (29), and (30) related to z direction and height windows are removed. The rest of the formulation shares the same constraints as those in the DR-MT-TSP big-M based MICP formulation.

6 Methods: GCS Based Formulations

This section first restates the big-M based MICP for the DR-MT-TSP as a biconvex binary program, and then presents the GCS based formulation. This biconvex binary program is just an "intermediate helper formulation" that helps explain the GCS based formulation.

6.1 Biconvex Binary Program for DR-MT-TSP

For simplicity, we refer to this program as the biconvex formulation. For each $e = (u,v) \in E$, we reuse the variables t_u^f, t_v^s, $x_{c,u}^f$, $x_{c,v}^s$, $y_{c,u}^f$, $y_{c,v}^s$, $z_{c,u}^f$, $z_{c,v}^s$, $d_{x,u}^f$, $d_{x,v}^s$, $d_{y,u}^f$, $d_{y,v}^s$, and $\tilde{l}(e)$, as well as the binary variable b_e, as defined in the big-M based MICP formulation. The auxiliary variables δ_u^t, δ_u^x, δ_u^y, δ_u^z in the big-M based MICP formulation are also reused to denote the time and position difference between the robot enters and leaves each $u \in V_{tar}$. The biconvex program formulation for the DR-MT-TSP is as follows:

$$\min \sum_{e=(u,v)\in E} \tilde{l}(e) \tag{46}$$

subject to constraints (2), (3), (4), (5), (12), (13), (14), (15), (16), (17), (18), (19), (20), (21), (23), (24), (25), (26), (27), (28), (29), (30), (31), (32), (33), (34), (38), (39),

$$(b_e x_{c,v}^s - b_e x_{c,u}^f)^2 + (b_e y_{c,v}^s - b_e y_{c,u}^f)^2 + (b_e z_{c,v}^s - b_e z_{c,u}^f)^2 \leq \tilde{l}(e)^2, \quad \forall e = (u,v) \in E, \tag{47}$$
$$\tilde{l}(e) \leq v_{\max}(b_e t_v^s - b_e t_u^f), \quad \forall e = (u,v) \in E. \tag{48}$$

This formulation shares flow constraints (2) to (5), constraints (12) to (21), constraints (23) to (34) from the big-M based formulations. The key modification of the biconvex formulation is the modification of the second-order cone constraints (35) in the MICP formulation for inter-target traveling distance of each edge to (47), by multiplying b_e with $x_{c,u}^f$ $x_{c,v}^s$, $y_{c,u}^f$, $y_{c,v}^s$, $z_{c,u}^f$, and $z_{c,v}^s$. Besides, constraints (48) are modified from (37) by multiplying b_e with t_u^f and t_v^s. The purpose for doing so is to remove the big-M constraints (36) and (37). Specifically, if $b_e = 1$, the second-order cone constraint (47) becomes effective and constraint (48) will place the speed limit on $\tilde{l}(e)$. If $b_e = 0$, constraint (48) sets $\tilde{l}(e)$ to zero and the second-order cone constraint (47) also holds for both sides equal to zero. The subtour elimination constraints are the same to the constraints (38) and (39) in the big-M based formulation.

Apart from the binary variables b_e, the non-convexity of this program comes from the bilinear terms, i.e. $b_e t_u^f$, $b_e t_v^s$, $b_e x_{c,u}^f$, $b_e x_{c,v}^s$, $b_e y_{c,u}^f$, $b_e y_{c,v}^s$, $b_e z_{c,u}^f$, and $b_e z_{c,v}^s$. In the next section, we introduce our GCS based formulation for the DR-MT-TSP, which bypasses these biconvex terms by using perspective and leads to a new MICP formulation.

6.2 GCS based Formulation for DR-MT-TSP

This section first introduces the notion of perspective and then presents the entire formulation.

6.2.1 Perspective Sets Let $A_u, u \in V$ denote the line segment (which vanishes to a point for o and o') between the two end points $(\underline{t}_u, \underline{x}_u, \underline{y}_u), (\bar{t}_u, \bar{x}_u, \bar{y}_u)$ of any $u \in V$,

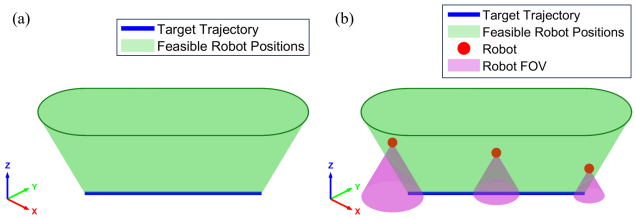

Figure 5. (a) Illustration of $\overline{A}_u$, i.e. the augmented set of A_u; (b) Illustration of robot positions in $\overline{A}_u$. As long as the robot is within $\overline{A}_u$, the target is within its FOV.

where

$$\overline{x}_u = \underline{x}_u + \xi_{x,u}(\overline{t}_u - \underline{t}_u), \quad \overline{y}_u = \underline{y}_u + \xi_{y,u}(\overline{t}_u - \underline{t}_u).$$

Let $\overline{A}_u = \{(t,x,y,z) \mid \exists (d_x,d_y)$ s.t. $(t, x-d_x, y-d_y) \in A_u,\ z \in [\underline{z}_u, \overline{z}_u],\ d_x^2 + d_y^2 \le (k_r z)^2\}$ denote the augmented set of A_u, which adds the z-dimension to A_u while z is restricted to be within the height window of u, and x and y are allowed in a feasible range that satisfies the dynamic sensor range constraints. Fig. 5 (a) illustrates such an $\overline{A}_u$ for a target u moving along a straight line. Fig. 5 (b) shows that as long as the robot is in $\overline{A}_u$, its FOV will cover at least a portion of the line segment representing the target trajectory, i.e., the target is in its FOV at a certain time instant. This augmented set provides an alternative way to represent the constraints related to the dynamic FOV of the robot. In other words, constraints (23) to (34) are equivalent to the following set of constraints:

$$(t_u^f, x_{c,u}^f, y_{c,u}^f, z_{c,u}^f) \in \overline{A}_u, \quad \forall e = (u,v) \in E, \tag{49}$$
$$(t_v^s, x_{c,v}^s, y_{c,v}^s, z_{c,v}^s) \in \overline{A}_v, \quad \forall e = (u,v) \in E. \tag{50}$$

Note that this augmented set is convex (Boyd and Vandenberghe 2004) and to make the article self-contained, we briefly show the proof here.

Lemma 3. *The augmented set* $\overline{A}_u = \{(t,x,y,z) \mid \exists (d_x,d_y)$ *s.t.* $(t, x-d_x, y-d_y) \in A_u,\ z \in [\underline{z}_u, \overline{z}_u],\ d_x^2 + d_y^2 \le (k_r z)^2\}$ *is convex.*

Proof. Let two arbitrary points

$$p^1 = (t^1, x^1, y^1, z^1), \quad p^2 = (t^2, x^2, y^2, z^2)$$

belong to $\overline{A}_u$. By definition, there exist vectors $d^1 = (d_x^1, d_y^1)$, $d^2 = (d_x^2, d_y^2)$ and corresponding points $a^1 = (t^1, x_a^1, y_a^1)$, $a^2 = (t^2, x_a^2, y_a^2) \in A_u$ such that

$$x^1 - d_x^1 = x_a^1, \quad y^1 - d_y^1 = y_a^1, \quad z^1 \in [\underline{z}_u, \overline{z}_u],$$
$$x^2 - d_x^2 = x_a^2, \quad y^2 - d_y^2 = y_a^2, \quad z^2 \in [\underline{z}_u, \overline{z}_u],$$
$$\|d^1\| \le k_r z^1, \quad \|d^2\| \le k_r z^2.$$

For any $\theta \in [0,1]$, define convex combinations

$$p^\theta = \theta p^1 + (1-\theta)p^2, \quad a^\theta = \theta a^1 + (1-\theta)a^2,$$
$$z^\theta = \theta z^1 + (1-\theta)z^2, \quad d^\theta = \theta d^1 + (1-\theta)d^2.$$

Recall that A_u represents a line segment corresponding to the position of target u in its feasible time window, so A_u is convex and $a^\theta \in A_u$. So we have:

$$\begin{aligned}(t^\theta, x^\theta - d_x^\theta, y^\theta - d_y^\theta) &= \theta(t^1, x_a^1, y_a^1) + (1-\theta)(t^2, x_a^2, y_a^2) \\ &= a^\theta \in A_u.\end{aligned}$$

Next, the interval $[\underline{z}_u, \overline{z}_u]$ is also convex, so $z^\theta \in [\underline{z}_u, \overline{z}_u]$. Finally, we verify the convexity of the displacement variable:

$$\begin{aligned}\|d^\theta\| = \|\theta d^1 + (1-\theta)d^2\| &\le \theta\|d^1\| + (1-\theta)\|d^2\| \\ &\le \theta k_r z^1 + (1-\theta)k_r z^2 = k_r z^\theta,\end{aligned}$$

where the first inequality follows from the triangle inequality and the last equality from linearity of the convex combination.

Therefore, $p^\theta = (t^\theta, x^\theta, y^\theta, z^\theta)$ satisfies all defining conditions of $\overline{A}_u$, i.e. $p^\theta \in \overline{A}_u$ for any $\theta \in [0,1]$, and the set $\overline{A}_u$ is convex.

6.2.2 Auxiliary Variables Now we can define some new auxiliary variables for the bilinear terms as follows:

$$\begin{aligned}\alpha_e &= (\alpha_{t,e}, \alpha_{x,e}, \alpha_{y,e}, \alpha_{z,e}) \\ &= (b_e t_u^f, b_e x_{c,u}^f, b_e y_{c,u}^f, b_e z_{c,u}^f), \quad \forall e = (u,v) \in E,\end{aligned} \tag{51}$$

$$\begin{aligned}\alpha'_e &= (\alpha'_{t,e}, \alpha'_{x,e}, \alpha'_{y,e}, \alpha'_{z,e}) \\ &= (b_e t_v^s, b_e x_{c,v}^s, b_e y_{c,v}^s, b_e z_{c,v}^s), \quad \forall e = (u,v) \in E,\end{aligned} \tag{52}$$

$$\begin{aligned}\beta_e &= (\beta_{x,e}, \beta_{y,e}) \\ &= (b_e d_{x,u}^f, b_e d_{y,u}^f), \quad \forall e = (u,v) \in E,\end{aligned} \tag{53}$$

$$\begin{aligned}\beta'_e &= (\beta'_{x,e}, \beta'_{y,e}) \\ &= (b_e d_{x,v}^s, b_e d_{y,v}^s), \quad \forall e = (u,v) \in E.\end{aligned} \tag{54}$$

Although we can substitute all the biconvex terms in the biconvex formulation by these variables, the non-convexity still exists when we define these variables by non-convex constraints (51) to (54). In the GCS based formulation, we will introduce perspective, which bypasses these non-convexity.

Let $\tilde{A}_u$ denote the perspective of $\overline{A}_u$, which is $\tilde{A}_u = \{(\lambda, t, x, y, z) \mid \lambda \ge 0, (t,x,y,z) \in \lambda \overline{A}_u\}$ (Fig. 6). The non-convex constraints (51) to (54) can be replaced by the

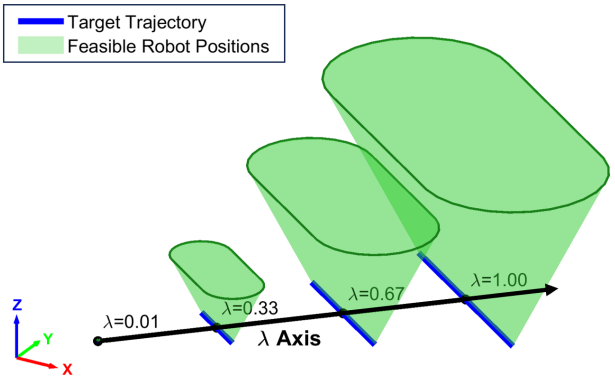

Figure 6. Illustration of $\tilde{A}_u$, i.e., the perspective of $\overline{A}_u$. When $\lambda = 0$, the set $\tilde{A}_u$ contains only a single zero point.

following perspective set constraints:

$$(b_e, \alpha_{t,e}, \alpha_{x,e}, \alpha_{y,e}, \alpha_{z,e}) \in \tilde{A}_u, \quad \forall e = (u,v) \in E, \quad (55)$$
$$(b_e, \alpha'_{t,e}, \alpha'_{x,e}, \alpha'_{y,e}, \alpha'_{z,e}) \in \tilde{A}_v, \quad \forall e = (u,v) \in E. \quad (56)$$

Lemma 4. *$\tilde{A}_u$ is a convex set.*

Proof. Take any two points $(\lambda_1, w_1), (\lambda_2, w_2) \in \tilde{A}_u$, where $\lambda_1, \lambda_2 \geq 0$ and $w_1, w_2 \in \mathbb{R}^4$. By definition of the perspective set, there exist $q_1, q_2 \in \overline{A}_u$ such that

$$w_1 = \lambda_1 q_1, \qquad w_2 = \lambda_2 q_2.$$

Let $\gamma \in [0,1]$ and consider the convex combination

$$\begin{aligned}(\lambda, w) &= \gamma(\lambda_1, w_1) + (1-\gamma)(\lambda_2, w_2) \\ &= \big(\gamma\lambda_1 + (1-\gamma)\lambda_2,\ \gamma\lambda_1 q_1 + (1-\gamma)\lambda_2 q_2\big).\end{aligned}$$

Then

$$\begin{aligned}\lambda &= \gamma\lambda_1 + (1-\gamma)\lambda_2 \geq 0, \\ w &= \gamma\lambda_1 q_1 + (1-\gamma)\lambda_2 q_2.\end{aligned}$$

Define η as

$$\lambda\eta = \gamma\lambda_1,$$

so that $\eta \in [0,1]$, and

$$\begin{aligned}\lambda(1-\eta) &= \lambda - \lambda\eta = (\gamma\lambda_1 + (1-\gamma)\lambda_2) - \gamma\lambda_1 \\ &= (1-\gamma)\lambda_2.\end{aligned}$$

So we have

$$\begin{aligned}w &= \lambda\eta q_1 + \lambda(1-\eta) q_2 \\ &= \lambda(\eta q_1 + (1-\eta) q_2).\end{aligned}$$

Since $\overline{A}_u$ is convex (Lemma 3), $\eta\, q_1 + (1-\eta)\, q_2 \in \overline{A}_u$, hence $w \in \lambda\overline{A}_u$ and thus $(\lambda, w) \in \tilde{A}_u$.

Therefore every convex combination of two points in $\tilde{A}_u$ remains in $\tilde{A}_u$, proving that $\tilde{A}_u$ is convex.

Lemma 5. *$\tilde{A}_u$ is a convex cone.*

Proof. By Lemma 3 we have $\overline{A}_u$ is convex. In Lemma 4 we already established that $\tilde{A}_u$ is convex; here we strengthen that result by showing $\tilde{A}_u$ is a cone (and hence a convex cone).

Recall the perspective set definition

$$\tilde{A}_u = \{(\lambda, w) \in \mathbb{R}_{\geq 0} \times \mathbb{R}^4 \mid w \in \lambda\overline{A}_u\}.$$

Take any $(\lambda, w) \in \tilde{A}_u$ and any scalar $\zeta \geq 0$. By definition there exists $q \in \overline{A}_u$ such that $w = \lambda q$. Then

$$\zeta w = \zeta(\lambda q) = (\zeta\lambda) q.$$

Since $\zeta\lambda \geq 0$ and $q \in \overline{A}_u$, it follows from the definition of $\tilde{A}_u$ that $(\zeta\lambda, \zeta w) \in \tilde{A}_u$. Thus $\tilde{A}_u$ is closed under multiplication by nonnegative scalars, i.e. it is a cone.

Combining this cone property with the convexity already shown in Lemma 4, we conclude that $\tilde{A}_u$ is a convex cone.

With the perspective set constraints that preserve the convexity, we can now replace all the bilinear terms in the biconvex formulation by $(\alpha_{t,e}, \alpha_{x,e}, \alpha_{y,e}, \alpha_{z,e})$, $(\alpha'_{t,e}, \alpha'_{x,e}, \alpha'_{y,e}, \alpha'_{z,e})$, $(\beta_{x,e}, \beta_{y,e})$, and $(\beta'_{x,e}, \beta'_{y,e})$, without adding the non-convex constraints (51) to (54) for each edge $e = (u,v) \in E$. Recall that (49) and (50) are a compact way of writing constraints (23) to (34). Similarly, the perspective set constraints (55) and (56) are also a compact way of writing a set of constraints, including the bilinear variable definition constraints (51) to (54) and the constraints shown below:

$$\underline{t}_u b_e \leq \alpha_{t,e} \leq \overline{t}_u b_e, \quad \forall e = (u,v) \in E, \quad (57)$$
$$\underline{t}_v b_e \leq \alpha'_{t,e} \leq \overline{t}_v b_e, \quad \forall e = (u,v) \in E, \quad (58)$$
$$\alpha_{x,e} = \xi_{x,u}\alpha_{t,e} + b_e(\underline{x}_u - \xi_{x,u}\underline{t}_u) + \beta_{x,e}, \quad \forall e = (u,v) \in E, \quad (59)$$
$$\alpha'_{x,e} = \xi_{x,v}\alpha'_{t,e} + b_e(\underline{x}_v - \xi_{x,v}\underline{t}_v) + \beta'_{x,e}, \quad \forall e = (u,v) \in E, \quad (60)$$
$$\alpha_{y,e} = \xi_{y,u}\alpha_{t,e} + b_e(\underline{y}_u - \xi_{y,u}\underline{t}_u) + \beta_{y,e}, \quad \forall e = (u,v) \in E, \quad (61)$$
$$\alpha'_{y,e} = \xi_{y,v}\alpha'_{t,e} + b_e(\underline{y}_v - \xi_{y,v}\underline{t}_v) + \beta'_{y,e}, \quad \forall e = (u,v) \in E, \quad (62)$$
$$\underline{z}_u b_e \leq \alpha_{z,e} \leq \overline{z}_u b_e, \quad \forall e = (u,v) \in E, \quad (63)$$
$$\underline{z}_v b_e \leq \alpha'_{z,e} \leq \overline{z}_v b_e, \quad \forall e = (u,v) \in E, \quad (64)$$
$$\beta_{x,e} = \beta_{y,e} = 0, \quad \forall e = (u,v) \in E,\ u = o, \quad (65)$$
$$\beta'_{x,e} = \beta'_{y,e} = 0, \quad \forall e = (u,v) \in E,\ v = o', \quad (66)$$
$$\beta^2_{x,e} + \beta^2_{y,e} \leq (k_r\alpha_{z,e})^2, \quad \forall e = (u,v) \in E, \quad (67)$$
$$\beta'^2_{x,e} + \beta'^2_{y,e} \leq (k_r\alpha'_{z,e})^2, \quad \forall e = (u,v) \in E. \quad (68)$$

If $b_e = 1$, (57) to (68) are equivalent to (23) to (34) in the big-M based formulation, i.e. $(\alpha_{t,e}, \alpha_{x,e}, \alpha_{y,e}, \alpha_{z,e})$, $(\alpha'_{t,e}, \alpha'_{x,e}, \alpha'_{y,e}, \alpha'_{z,e})$, $(\beta_{x,e}, \beta_{y,e})$, and $(\beta'_{x,e}, \beta'_{y,e})$ are equivalent to $(t^f_u, x^f_{c,u}, y^f_{c,u}, z^f_{c,u})$, $(t^s_v, x^s_{c,v}, y^s_{c,v}, z^s_{c,v})$, $(d^f_{x,u}, d^f_{y,u})$, and $(d^s_{x,v}, d^s_{y,v})$. If $b_e = 0$, $\alpha_{t,e}$, $\alpha'_{t,e}$, $\alpha_{z,e}$, and $\alpha'_{z,e}$ become zero due to constraints (57), (58), (63), and (64). Constraints (67) and (68), combined with $\alpha_{z,e} = 0$ and $\alpha'_{z,e} = 0$, enforce $\beta_{x,e}$, $\beta'_{x,e}$, $\beta_{y,e}$, and $\beta'_{y,e}$ to be zero. All these zeros combined with constraints (59) to (62) make $\alpha_{x,e}$, $\alpha'_{x,e}$, $\alpha_{y,e}$, and $\alpha'_{y,e}$ equal to zero. These show that constraints (57) to (68) not only contain constraints (23) to (34) in the big-M based formulation, but also the bilinear variable definition constraints (51) to (54). Based on these, and together with the $\tilde{l}(e)$ and binary variable b_e that we have introduced in the previous formulation, we can replace the bilinear terms in the biconvex formulation.

6.2.3 GCS-Based MICP Formulation Based on perspective, our GCS based formulation is as follows:

$$\min \sum_{e=(u,v)\in E} \tilde{l}_\alpha(e) \quad (69)$$

subject to constraints (2), (3), (4), (5), (19), (20), (21), (38), (39), (57), (58), (59), (60), (61), (62), (63), (64), (65), (66), (67), (68),

$$\sum_{e\in E_u^{out}} \alpha_{t,e} - \sum_{e\in E_u^{in}} \alpha'_{t,e} = \delta^t_u, \quad \forall u \in V_{tar}, \quad (70)$$
$$\sum_{e\in E_u^{out}} \alpha_{x,e} - \sum_{e\in E_u^{in}} \alpha'_{x,e} \leq \delta^x_u, \quad \forall u \in V_{tar}, \quad (71)$$

$$\sum_{e\in E_u^{out}} \alpha_{x,e} - \sum_{e\in E_u^{in}} \alpha'_{x,e} \geq -\delta_u^x, \quad \forall u \in V_{tar}, \tag{72}$$

$$\sum_{e\in E_u^{out}} \alpha_{y,e} - \sum_{e\in E_u^{in}} \alpha'_{y,e} \leq \delta_u^y, \quad \forall u \in V_{tar}, \tag{73}$$

$$\sum_{e\in E_u^{out}} \alpha_{y,e} - \sum_{e\in E_u^{in}} \alpha'_{y,e} \geq -\delta_u^y, \quad \forall u \in V_{tar}, \tag{74}$$

$$\sum_{e\in E_u^{out}} \alpha_{z,e} - \sum_{e\in E_u^{in}} \alpha'_{z,e} \leq \delta_u^z, \quad \forall u \in V_{tar}, \tag{75}$$

$$\sum_{e\in E_u^{out}} \alpha_{z,e} - \sum_{e\in E_u^{in}} \alpha'_{z,e} \geq -\delta_u^z, \quad \forall u \in V_{tar}, \tag{76}$$

$$(\alpha'_{x,e} - \alpha_{x,e})^2 + (\alpha'_{y,e} - \alpha_{y,e})^2 + (\alpha'_{z,e} - \alpha_{z,e})^2 \leq \tilde{l}_\alpha(e)^2, \quad \forall e = (u,v) \in E, \tag{77}$$

$$\tilde{l}_\alpha(e) \leq v_{\max}(\alpha'_{t,e} - \alpha_{t,e}), \quad \forall e = (u,v) \in E, \tag{78}$$

This formulation shares constraints (2) to (5), (19) to (21), (38), and (39) from the previous formulations, as well as the perspective set constraints constraints (57) to (68) mentioned before. Constraints (70) to (76) are modified from constraints (12) to (18) in the big-M based formulation. Take constraint (70) as an example, which contains two steps of modifications from constraint (12) in the big-M based formulation. Constraint (12) is first modified by taking the sum of the multiplication of t_u^f and t_u^s with the b_e corresponding to all incoming edges and outgoing edges of $u \in V_{tar}$, respectively:

$$\sum_{e\in E_u^{out}} b_e t_u^f - \sum_{e\in E_u^{in}} b_e t_u^s = \delta_u^t, \quad \forall u \in V_{tar}. \tag{79}$$

The flow constraints (4) and (5) that only allow one selected incoming edge and one selected outgoing edge for each $u \in V_{tar}$, ensure that $\sum_{e\in E_u^{out}} b_e t_u^f$ and $\sum_{e\in E_u^{in}} b_e t_u^s$ in constraint (79) equal to t_u^f and t_u^s in constraint (12), respectively. Then we substitute the bilinear terms $b_e t_u^f$ and $b_e t_u^s$ by $\alpha_{t,e}$ and $\alpha'_{t,e}$, respectively, which forms constraint (70). In the same way, constraints (13) to (18) in the MICP formulation can be modified to constraints (71) to (76), by substituting $(x_{c,u}^f, y_{c,u}^f, z_{c,u}^f)$ and $(x_{c,v}^s, y_{c,v}^s, z_{c,v}^s)$ by the sum of $(\alpha_{x,e}, \alpha_{y,e}, \alpha_{z,e})$ and $(\alpha'_{x,e}, \alpha'_{y,e}, \alpha'_{z,e})$, respectively. Constraints (77) and (78) are modified from (47) and (48) in the biconvex formulation, by replacing the bilinear terms $(b_e t_u^f, b_e x_{c,u}^f, b_e y_{c,u}^f, b_e z_{c,u}^f)$ and $(b_e t_v^s, b_e x_{c,v}^s, b_e y_{c,v}^s, b_e z_{c,v}^s)$ with $(\alpha_{t,e}, \alpha_{x,e}, \alpha_{y,e}, \alpha_{z,e})$ and $(\alpha'_{t,e}, \alpha'_{x,e}, \alpha'_{y,e}, \alpha'_{z,e})$, respectively.

6.3 GCS Based Formulation for CE-MT-TSP

This section presents the GCS based formulation for the CE-MT-TSP. Similar as what we have done in big-M based formulation, most of the constraints in DR-MT-TSP GCS based formulation can be reused and some of the constraints need to be slightly modified. The formulation is presented below:

$$\min \sum_{e=(u,v)\in E} \tilde{l}(e) \tag{80}$$

subject to constraints (2), (3), (4), (5), (20), (21), (38), (39), (42), (57), (58), (59), (60), (61), (62), (65), (66), (70), (71), (72), (73), (74), (78),

$$\beta_{x,e}^2 + \beta_{y,e}^2 \leq (b_e r_c)^2, \quad \forall e = (u,v) \in E, \tag{81}$$

$$\beta_{x,e}'^2 + \beta_{y,e}'^2 \leq (b_e r_c)^2, \quad \forall e = (u,v) \in E, \tag{82}$$

$$(\alpha'_{x,e} - \alpha_{x,e})^2 + (\alpha'_{y,e} - \alpha_{y,e})^2 \leq \tilde{l}_\alpha(e)^2, \quad \forall e = (u,v) \in E. \tag{83}$$

Compared with the DR-MT-TSP GCS based formulation, the major modifications are the changes of constrains (67) and (68) to (81) and (82), where the target is restricted within the fixed circle with radius r_c instead of the sensor FOV, and $b_e r_c$ ensures the perspective set property of $(\beta_{x,e}, \beta_{y,e})$, and $(\beta'_{x,e}, \beta'_{y,e})$. Besides, constraints (19) and (77) are modified to (42) and (83), respectively, where the terms of z direction are deleted. In addition, constraints (63), (64), (75), and (76), related to height windows and z direction are removed. The rest of the formulation shares the same constraints as those in the DR-MT-TSP GCS based formulation.

6.4 GCS Based Formulation for MT-TSP

As presented in our prior work (Philip et al. 2024), this section presents the GCS based formulation for the MT-TSP. Recall that there is no monitoring requirement and the circular close-enough range for MT-TSP, the formulation can be modified from the GCS based formulation for the CE-MT-TSP, by removing the variables and constraints related to monitoring and the close-enough range. The formulation is presented below:

$$\min \sum_{e=(u,v)\in E} \tilde{l}(e) \tag{84}$$

subject to constraints (2), (3), (4), (5), (57), (58), (78), (83),

$$\sum_{e\in E_u^{in}} \alpha'_{t,e} = \sum_{e\in E_u^{out}} \alpha_{t,e}, \quad \forall u \in V_{tar}, \tag{85}$$

$$\alpha_{x,e} = \xi_{x,u}\alpha_{t,e} + b_e(\underline{x}_u - \xi_{x,u}\underline{t}_u), \quad \forall e = (u,v) \in E, \tag{86}$$

$$\alpha'_{x,e} = \xi_{x,v}\alpha'_{t,e} + b_e(\underline{x}_v - \xi_{x,v}\underline{t}_v), \quad \forall e = (u,v) \in E, \tag{87}$$

$$\alpha_{y,e} = \xi_{y,u}\alpha_{t,e} + b_e(\underline{y}_u - \xi_{y,u}\underline{t}_u), \quad \forall e = (u,v) \in E, \tag{88}$$

$$\alpha'_{y,e} = \xi_{y,v}\alpha'_{t,e} + b_e(\underline{y}_v - \xi_{y,v}\underline{t}_v), \quad \forall e = (u,v) \in E. \tag{89}$$

The flow conservation constraints (2), (3), (4), and (5), share the same constraints as the previous formulations. Since there is no monitoring period, constraint (85) is added to ensure time continuity, i.e., the robot enters and leaves a target at the same time instance. Constraints (57) and (58) share the same constraints as the previous GCS based formulations to ensure the robot visit each target within its corresponding time window as well as the perspective set property of the time variables $\alpha_{t,e}$ and $\alpha'_{t,e}$. Constraints (86) to (89) are modified from constraints (59) to (62) in the previous GCS based formulations, by removing the

$(\beta_{x,e}, \beta_{y,e})$ and $(\beta'_{x,e}, \beta'_{y,e})$ to ensure the robot visits the exact position of each target. Constraints (78) and (83) share the same constraints in the CE-MT-TSP GCS based formulation, representing the inter-target traveling distance of the robot and the speed restriction.

7 Analysis

We first show that the big-M based formulation is valid for the problem statement. We then show that the GCS based formulation is equivalent to the big-M formulation and is thus also valid. Finally, we show that the GCS based formulation is a mixed-integer conic program.

7.1 Validity of the Big-M Based Formulation

Theorem 1. *The big-M based formulation for the DR-MT-TSP is valid.*

Proof. To show validity, we prove that any feasible solution returned by the big-M based formulation indeed corresponds to a feasible solution of the DR-MT-TSP described in the problem statement. To verify this, we identify the correspondence between each component of DR-MT-TSP and the constraints in the big-M based formulation.

First, the robot leaves the depot once, and enters the depot's copy once after finishing its tour, which are described by constraints (2) and (3), respectively. Recall that the definition of the time window of o and o' as $(\underline{t}_o, \overline{t}_o) = (0, 0)$ and $(\underline{t}_{o'}, \overline{t}_{o'}) = (0, T)$, respectively. Combined with constraints (23) and (24), we have $t^f_o = 0$ and $t^s_{o'} \in [0, T]$, i.e., the robot starts its tour at $t = 0$ and ends the tour within the time horizon. The definition of the velocities of o and o' as zeros and the height windows of o and o' as $\underline{z}_o = \overline{z}_o = \underline{z}'_o = \overline{z}'_o = o_z$, along with constraints (25) to (32), ensure that $(x^f_{c,o}, y^f_{c,o}, z^f_{c,o})$ and $(x^s_{c,o'}, y^s_{c,o'}, z^s_{c,o'})$ are equal to the exact position (o_x, o_y, o_z) of the depot, i.e., the robot starts and ends the tour exactly at the depot.

Constraint (4) ensures that the robot enters each target exactly once, along with the flow conservation constraint (5) for all the target nodes, ensuring that the robot leaves each target exactly once. Constraints (2) to (5) ensure a valid path for the robot that starts at the depot, visits each target exactly once, and returns to the depot. For every target $u \in V_{tar}$, constraint (4) requires the robot visit every target exactly once, and constraint (5) ensures that the robot leaves each visited target exactly once.

Next, for each target $u \in V_{tar}$, the robots entering and leaving u within the time window $[\underline{t}_u, \overline{t}_u]$ are ensured by constraints (23) and (24). The x and y positions of the robot entering and leaving each target, i.e., $(x^s_{c,u}, y^s_{c,u}, z^s_{c,u})$ and $(x^f_{c,u}, y^f_{c,u}, z^f_{c,u})$, are depicted by constraints (25) to (28), while its z positions within the height window $[\underline{z}_u, \overline{z}_u]$ of the target are ensured by constraints (29) and (30). Constraints (33) and (34) ensure u is within the target's FOV at the time instance when the robot leaves and enters u, i.e., at t^s_u, and t^f_u, respectively. According to Lemma 1, this ensures that the target is monitored by the robot (within the robot FOV) during the entire time period between t^s_u and t^f_u.

During the monitoring period of each target u, constraints (13) to (18) define the displacement $(\delta^x_u, \delta^y_u, \delta^z_u)$ of the robot in x, y, and z direction. Second-order cone constraint (19) depicts the displacement during the monitoring period, and constraint (20) ensures the velocity of the robot is within its speed limit. Constraints (12) and (21) define the monitoring time δ^t_u and ensure it is larger than τ_m, which accords to the minimum monitoring time requirement of DR-MT-TSP.

So far, the robot will start from the depot, enter, monitor, and leave each target, and return to the depot. Except for the condition $\tau_m = 0$, subtours are inherently prevented according to Lemma 2. Otherwise, constraints (38) and (39) will be added to eliminate subtours and ensure all the targets are visited.

Recall that the goal of DR-MT-TSP is to find a feasible tour that minimizes the robot's total inter-agent travel distance, where the distance traveled during the monitoring periods are excluded. For each edge $e = (u, v) \in E$, the Euclidean inter-target travel distance is depicted by second-order cone constraint (35). By big-M constraints (36) and (37), we have $\tilde{l}(e)$ equal to $l(e)$ and robot speed within the speed limit when $b_e = 1$, and these constraints are relaxed when $b_e = 0$. So the objective $\sum_{e=(u,v)\in E} \tilde{l}(e)$ equals the total inter-agent travel distance, hence the program minimizes exactly the objective stated in the problem description of DR-MT-TSP.

Conclusion. To conclude, every feasible solution of the big-M based formulation is a feasible solution of the DR-MT-TSP, and its objective matches the goal of DR-MT-TSP. Therefore, the big-M based formulation is valid for DR-MT-TSP.

7.2 Equivalency between the Big-M Based and GCS Based Formulations

Theorem 2. *The GCS based formulation is equivalent to the big-M based formulation of the DR-MT-TSP. Consequently, since the big-M based formulation is valid, the GCS based formulation is also valid.*

Proof. Let $S = b_e \cup (t^f_u, x^f_{c,u}, y^f_{c,u}, z^f_{c,u}) \cup (t^s_u, x^s_{c,u}, y^s_{c,u}, z^s_{c,u})$ denote a feasible solution of the big-M based formulation. Let $S' = b_e \cup \alpha_e \cup \alpha'_e$ denote a feasible solution of the GCS based formulation, where $\alpha_e = (\alpha_{t,e}, \alpha_{x,e}, \alpha_{y,e}, \alpha_{z,e})$, and $\alpha'_e = (\alpha'_{t,e}, \alpha'_{x,e}, \alpha'_{y,e}, \alpha'_{z,e})$. To show the equivalency, we prove that every S can be mapped to an S' with the same objective value, and every S' can recover to an S.

Construct an S' from S:

To construct an S' from S, for any S, we first define for every edge $e = (u, v) \in E$:

$$(\alpha_{t,e}, \alpha_{x,e}, \alpha_{y,e}, \alpha_{z,e}) = (b_e t^f_u, b_e x^f_{c,u}, b_e y^f_{c,u}, b_e z^f_{c,u}),$$
$$(\alpha'_{t,e}, \alpha'_{x,e}, \alpha'_{y,e}, \alpha'_{z,e}) = (b_e t^s_v, b_e x^s_{c,v}, b_e y^s_{c,v}, b_e z^s_{c,v}).$$

Using the values of $(d^f_{x,u}, d^f_{y,u})$ and $(d^s_{x,v}, d^s_{y,v})$ that correspond to solution S, we also define for every edge $e = (u, v) \in E$:

$$(\beta_{x,e}, \beta_{y,e}) = (b_e d^f_{x,u}, b_e d^f_{y,u}),$$
$$(\beta'_{x,e}, \beta'_{y,e}) = (b_e d^s_{x,v}, b_e d^s_{y,v}).$$

We now show that each constraint in the GCS based formulation is satisfied for $(\alpha_{t,e}, \alpha_{x,e}, \alpha_{y,e}, \alpha_{z,e})$ and $(\alpha'_{t,e}, \alpha'_{x,e}, \alpha'_{y,e}, \alpha'_{z,e})$.

Since b_e is unchanged, the shared flow constraints (2) to (5) hold immediately.

As for subtour elimination, since b_e and u_i are unchanged, (38) and (39) remain satisfied.

As for the time windows requirement, since $\alpha_{t,e} = b_e t_u^f$ and $\alpha'_{t,e} = b_e t_v^s$, the constraints in the big-M based formulation

$$t_u^f \in [\underline{t}_u, \bar{t}_u], \qquad t_v^s \in [\underline{t}_v, \bar{t}_v]$$

can imply

$$\underline{t}_u b_e \le \alpha_{t,e} \le \bar{t}_u b_e, \qquad \underline{t}_v b_e \le \alpha'_{t,e} \le \bar{t}_v b_e,$$

Thus constraints (57) and (58) follow.

As for the position constraints, big-M based formulation imposes, e.g.,

$$x_{c,u}^f = \underline{x}_u + \xi_{x,u}(t_u^f - \underline{t}_u) + d_{x,u}^f.$$

Multiplying by b_e and substituting the definitions of $\alpha_{t,e}$, $\alpha_{x,e}$, $\beta_{x,e}$ yields constraint (59), i.e.,

$$\alpha_{x,e} = \xi_{x,u}\alpha_{t,e} + b_e(\underline{x}_u - \xi_{x,u}\underline{t}_u) + \beta_{x,e}.$$

In the same way, constraints (60) to (62) hold. As for the height windows requirement, since $\alpha_{z,e} = b_e z_{c,u}^f$ and $\alpha'_{z,e} = b_e z_{c,v}^s$, the constraints in the big-M based formulation

$$z_{c,u}^f \in [\underline{z}_u, \bar{z}_u], \qquad z_{c,v}^s \in [\underline{z}_v, \bar{z}_v]$$

can imply

$$\underline{z}_u b_e \le \alpha_{z,e} \le \bar{z}_u b_e, \qquad \underline{z}_v b_e \le \alpha'_{z,e} \le \bar{z}_v b_e,$$

Thus, constraints (63) and (64) follow.

As for the dynamic FOV constraints, big-M based formulation enforces

$$(d_{x,u}^f)^2 + (d_{y,u}^f)^2 \le (k_r z_{c,u}^f)^2,$$

$$(d_{x,v}^s)^2 + (d_{y,v}^s)^2 \le (k_r z_{c,v}^s)^2.$$

Multiplying $d_{x,u}^f$, $d_{y,u}^f$, $z_{c,u}^f$, $d_{x,v}^s$, $d_{y,v}^s$, and $z_{c,v}^s$ by b_e gives

$$\beta_{x,e}^2 + \beta_{y,e}^2 \le (k_r \alpha_{z,e})^2,$$

$$\beta_{x,e}'^2 + \beta_{y,e}'^2 \le (k_r \alpha'_{z,e})^2,$$

which matches constraints (67) and (68). Similarly, constraints (65) and (66) hold. Thus, the dynamic FOV constraints in GCS based formulation hold.

As for the monitoring period of each target $u \in V_{tar}$, take the definition for monitoring time δ_u^t as an example. As mentioned before, the constraint in big-M based formulation

$$t_u^f - t_u^s = \delta_u^t$$

first implies the bilinear constraint

$$\sum_{e \in E_u^{out}} b_e t_u^f - \sum_{e \in E_u^{in}} b_e t_u^s = \delta_u^t,$$

where $\sum_{e \in E_u^{out}} b_e t_u^f$ and $\sum_{e \in E_u^{in}} b_e t_u^s$ are equal to t_u^f and t_u^s, respectively. Substitute the bilinear terms $b_e t_u^f$ and $b_e t_u^s$ by $\alpha_{t,e}$ and $\alpha'_{t,e}$, respectively, which forms constraint (70) in the GCS based formulation. In the same way, constraints (71) to (76) hold, implied by constraints (13) to (18) in the big-M based formulation. Since the definitions for monitoring time and displacement variables δ_u^t, δ_u^x, δ_u^y, and δ_u^z are equivalent between both formulations, the shared constraints (19) to (21) for minimum monitoring time requirement and speed limit hold as well.

Regarding the inter-target traveling distance and travel speed limit constraints for each edge $e = (u, v) \in E$, as mentioned before, constraints (35), (36), and (37) in the big-M based formulation, imply biconvex constraints (47) and (48), by multiplying b_e with $x_{c,u}^f$ $x_{c,v}^s$, $y_{c,u}^f$, $y_{c,v}^s$, $z_{c,u}^f$, and $z_{c,v}^s$. Substituting the bilinear terms $b_e x_{c,u}^f$ $b_e x_{c,v}^s$, $b_e y_{c,u}^f$, $b_e y_{c,v}^s$, $b_e z_{c,u}^f$, and $b_e z_{c,v}^s$ by $\alpha_{x,e}$, $\alpha'_{x,e}$, $\alpha_{y,e}$, $\alpha'_{y,e}$, $\alpha_{z,e}$, and $\alpha'_{z,e}$ yields constraints (77) and (78) in the GCS based formulation. Thus, the objectives for both formulations which minimize the sum of the inter-target traveling distance of the selected edges as $\sum_e \tilde{l}(e)$ are equivalent, so that both solutions have identical objective values.

Therefore, any feasible solution S of the big-M based formulation induces a feasible solution S' of the GCS based formulation with the same objective values.

Recover an S from S':

Given a feasible solution S' from GCS based formulation, we can recover a corresponding feasible solution S for the big-M based formulation as follows:

$$(t_u^f, x_{c,u}^f, y_{c,u}^f, z_{c,u}^f) = \sum_{e \in E_u^{out}} \alpha_e, \quad \forall u \in V_{tar} \cup \{o\},$$

$$(t_u^s, x_{c,u}^s, y_{c,u}^s, z_{c,u}^s) = \sum_{e \in E_u^{in}} \alpha'_e, \quad \forall u \in V_{tar} \cup \{o'\}.$$

Flow constraints (2) to (5) imply exactly one outgoing selected edge for each $u \in V_{tar} \cup \{o\}$ and one incoming selected edge for each $u \in V_{tar} \cup \{o'\}$, so the above summations exactly reconstruct a solution S of the big-M based formulation.

Conclusion. Every feasible solution S of the big-M based formulation corresponds to a feasible solution S' of the GCS based formulation with identical objective values, and vice versa. Therefore, we conclude that: The GCS based formulation is equivalent to the big-M based formulation, and is hence valid.

7.3 Validation of MICP for the GCS Based Formulation

Theorem 3. *The GCS based formulation is an MICP.*

Proof. To prove that the GCS based formulation is an MICP, we show that every constraint in the GCS based formulation belongs to one of the following classes:

- linear equality or inequality constraints,
- second-order cone constraints,
- perspective set constraints whose feasible sets are convex cones.

Since the formulation also contains binary variables b_e, we can prove that it is an MICP.

The GCS based formulation contains the following linear constraints:

- flow constraints (2) to (5),
- MTZ subtour elimination constraints (38) and (39),
- linear monitoring variables constraints (70) to (76),
- linear monitoring speed limit and time requirement constraints (20) and (21),
- linear travel speed limit constraint (78).

The GCS based formulation contains the following second-order cone constraints:

- monitoring travel distance constraint (19),
- inter-target travel distance constraint (77).

Recall that constraints (57) to (68) in the GCS based formulation can be written in a compact way as the perspective set constraints (55) and (56), i.e.,

$$(b_e, \alpha_{t,e}, \alpha_{x,e}, \alpha_{y,e}, \alpha_{z,e}) \in \tilde{A}_u, \quad \forall e = (u, v) \in E,$$
$$(b_e, \alpha'_{t,e}, \alpha'_{x,e}, \alpha'_{y,e}, \alpha'_{z,e}) \in \tilde{A}_v, \quad \forall e = (u, v) \in E.$$

By Lemma 5 that $\tilde{A}_u$ is a convex cone, constraints (55) and (56) are conic constraints (Boyd and Vandenberghe 2004). Therefore, constraints (57) to (68) are conic constraints.

Conclusion. The GCS based formulation consists of linear constraints, second-order cone constraints, perspective set constraints whose feasible sets are convex cones, and binary decision variables b_e. Therefore, we conclude that: The GCS based formulation is an MICP.

8 Experimental Results

We first describe the test settings, instance generation and evaluation metrics that are shared by all experiments and then discuss the results for DR-MT-TSP, CE-MT-TSP and MT-TSP.

8.1 Common Test Setup

8.1.1 Hardware and Test Settings All tests were run on a laptop with an Intel(R) Core(TM) Ultra 7 155H CPU with 16 cores and 22 threads, with a maximum turbo frequency of 4.80 GHz, and 32 GB RAM. The implementation was in Python 3.12.6, and all formulations were solved using Gurobi 12.0.3 optimizer. All the Gurobi parameters were set to their default values, except for TimeLimit, which was set to 300 seconds.

8.1.2 Instance Generation The instances used for the tests were defined by the number of targets N_{tar}, a cuboid workspace of fixed size $L_x = L_y = 100$ units and $L_z = 40$ units, a fixed time horizon $T = 150$ seconds, the depot location fixed at the bottom center (0, 0, 0) of the workspace, and a set of N_{tar} randomly generated linear trajectories corresponding to the time window $[\underline{t}_u, \bar{t}_u]$ of the targets such that each target had a constant speed within $[0.5, 1] unit/sec$ and was confined within the square area of the workspace with $z = 0$ during $[\underline{t}_u, \bar{t}_u]$.

The test instances were generated as follows. N_{tar} varied among 5, 10, and 15. Under each N_{tar}, we varied other parameters, such as the time window duration and the maximum speed v_{max}. The time window duration were varied to be 25, 50, and 75 secs, and v_{max} was varied to be 4, 6, and 8 unit/sec. For each parameter combination, 10 instances were generated and tested. Feasibility was ensured in all generated instances in a sense that a feasible solution could be found within the time limit for both big-M based and GCS based formulations, otherwise the generated instance would be discarded and regenerated. In addition, the best objective value of the feasible solution should be larger than zero, otherwise the instance was considered trivial and was also abandoned and regenerated.

8.1.3 Evaluation Metrics To evaluate the big-M based and GCS based formulations, we consider the Runtime and Average Gap (%): Given each parameter combination of N_{tar}, time-window duration, v_{max}, and τ_m, the 10 instances were run by the solver using the two formulations respectively. The optimality gap value from the solver for an instance is defined as $\frac{|z_P - z_D|}{|z_P|} \times 100\%$, where z_P is the primal (feasible) objective, and z_D is the dual (lower-bound) objective. Let Average Gap (%) denote the average of gap values output by the solver for all the 10 instances.

8.2 DR-MT-TSP

Recall that in DR-MT-TSP, the robot must visit each target within its corresponding height window $[\underline{z}_u, \bar{z}_u]$, so we set the upper bound of each target's height window as a random integer within $[10, 35]$, and the lower bound of the height window of all the targets as zero. We set the view angle of the sensor $k_r = 0.5$. For each N_{tar}, besides varying the time window duration and v_{max}, we varied the minimum monitoring duration τ_m to be 2, 5, and 8 secs. A total of 270 instances were generated for DR-MT-TSP.

8.2.1 Varying the Time Window Duration In this section, we consider the experiments where the time window durations are varied. We do this by fixing v_{max} at 8 unit/s and τ_m at 5 secs, and solving all the instances for the time window durations (25, 50, and 75 secs). The results for this experiment are illustrated in Fig. 7 (a), (b), and (c), which corresponds to $N_{tar} = 5$, 10, and 15, respectively. We observe that the problem becomes more challenging to solve for both formulations as the time window duration increases. This difficultly becomes more prominent as N_{tar} increases.

The main advantages of the GCS based formulation here are that its Average Gap (%) is better than the big-M based formulation in nearly all the tests and its Runtime is better than the big-M based formulation against a larger N_{tar} and bigger time window durations. We see this especially in the case of 15 targets where its median Runtime and Average Gap (%) are 65.8% and 81.8% smaller than the big-M based formulation when time window duration is 50 secs, respectively, and when time window duration is 75 secs, although both formulations are unable to get the optimal solution within the time limit, the Average Gap (%) of the GCS based formulation is 46.7% smaller.

8.2.2 Varying the Robot Maximum Speed In this section, we consider the second set of experiments where the v_{max}

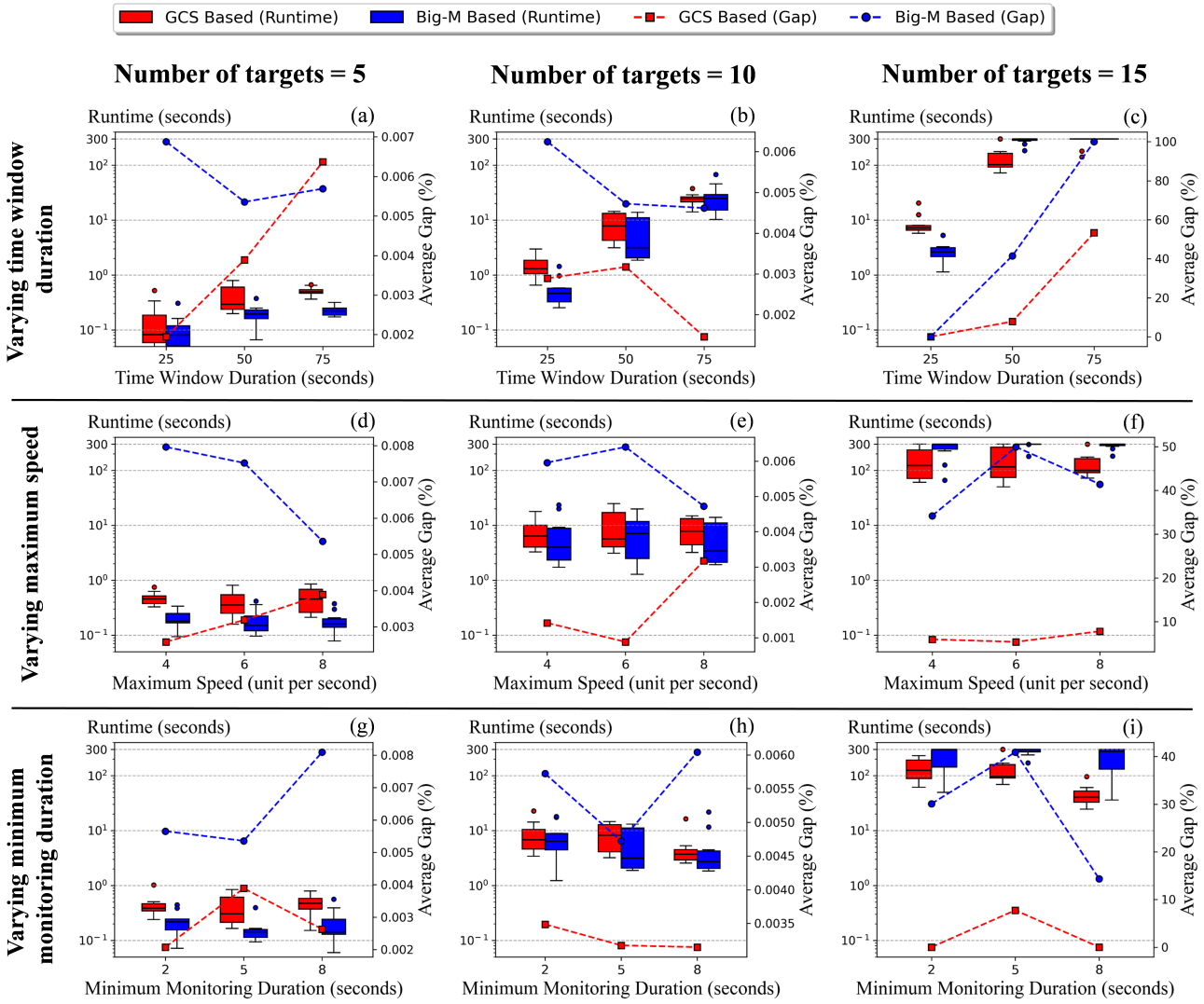

Figure 7. Results of DR-MT-TSP comparing Runtime and Average Gap (%) of the big-M based and GCS based formulation. The Runtime increases with N_{tar} and the time window duration, while it is only weakly affected by variations in v_{max} and τ_m. The GCS based formulation outperforms the big-M based formulation as N_{tar} increases, especially for N_{tar} = 15 exemplified in (i), where its median Runtime and Average Gap (%) are up to 85.2% and 100.0% smaller than the big-M based formulation, respectively.

is varied. We do this by fixing the time window duration at 50 secs and τ_m at 5 secs, and varying v_{max} to 4, 6, and 8 unit/sec. The results for this experiment are illustrated in Fig. 7 (d), (e), and (f), which corresponds to $N_{tar} = 5$, 10, and 15, respectively. Similar as the results in the previous subsection, the Runtime increases as N_{tar} increases.

However, we observe that as for each N_{tar}, the Runtime changes only slightly as v_{max} increases for both formulations. When N_{tar} is 15, the instances become more challenging for the big-M based formulation, but the median Runtime of the GCS based formulation is 59.4%, 61.3%, and 66.8% smaller than the big-M based formulation for $v_{max} =$ 4, 6, and 8 unit/sec. In all these plots, we again observe that the GCS based formulation vastly outperforms the big-M based formulation as for Average Gap (%), especially for $N_{tar} = 15$ where its Average Gap (%) is 82.6%, 89.3%, and 81.2% smaller than the big-M based formulation.

8.2.3 Varying the Minimum Monitoring Duration In this section, we consider the third set of experiments where τ_m is varied. We do this by fixing the time window duration at 50 secs and v_{max} at 8 unit/sec, and varying τ_m to 2, 5, and 8 secs. The results for this experiment are illustrated in Fig. 7 (g), (h), and (i), which corresponds to $N_{tar=}$ 5, 10, and 15, respectively.

The Runtime increases more obviously for the big-M based formulation than the GCS based formulation as N_{tar} increases, and as τ_m increases, the Runtime slightly decreases for both formulations when N_{tar} is 10 and 15. When $N_{tar} = 15$, the median Runtime of the GCS based outperforms the big-M based by 58.5%, 67.7%, and 85.2% for $\tau_m = 2$, 5, and 8 secs, respectively. In addition, the Average Gap (%) of the GCS based formulation is smaller than the big-M based formulation under all the nine configurations, especially for $N_{tar} = 15$ where its Average Gap (%) is 100.0%, 81.1%, and 100.0% smaller.

8.3 CE-MT-TSP

Recall that CE-MT-TSP is a 2D problem, so the workspace became a square of fixed size $L_x = L_y = 100$ units and the depot location was fixed at the center (0, 0) of the workspace.

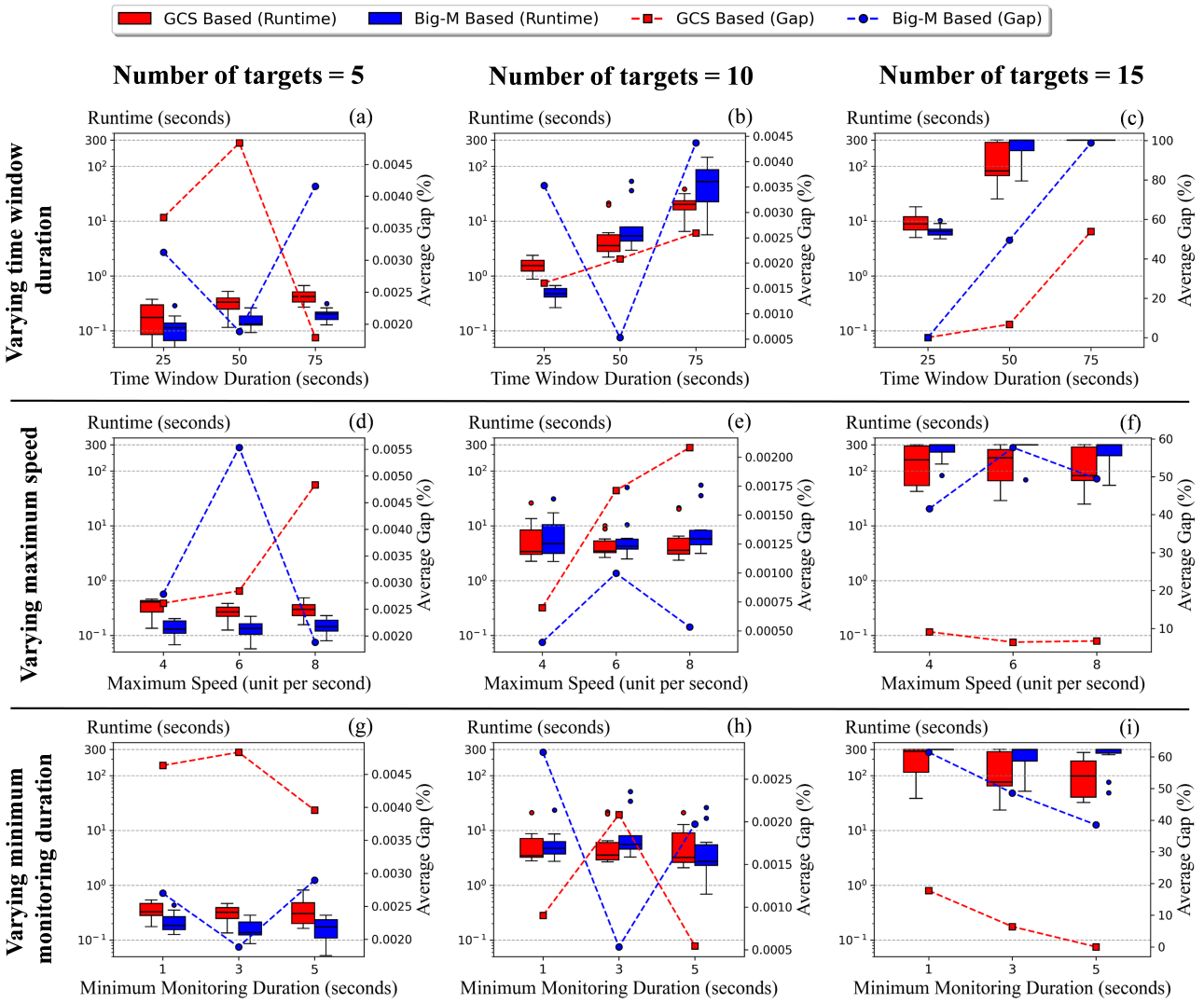

Figure 8. Results of CE-MT-TSP comparing Runtime and Average Gap (%) of the big-M based and GCS based formulation. The Runtime increases with N_{tar} and the time window duration, while it is only weakly affected by variations in v_{max} and τ_m. The GCS based formulation outperforms the big-M based formulation as N_{tar} increases, especially for N_{tar} = 15 exemplified in (i), where its median Runtime and Average Gap (%) are up to 74.4% and 100.0% smaller than the big-M based formulation, respectively.

We set the radius of the circle as $r_c = 5$ within which the robot is regarded as "close-enough" and catches a target. For each N_{tar}, besides varying the time window duration and v_{max}, we varied the minimum monitoring duration τ_m to be 1, 3, and 5 secs. A total of 270 instances were generated for CE-MT-TSP.

8.3.1 Varying the Time Window Duration In this section, we consider the experiments where the time window durations are varied. We do this by fixing v_{max} at 8 unit/sec and τ_m at 3 secs, and solving all the instances for the time window durations (25, 50, and 75 secs). The results for this experiment are illustrated in Fig. 8 (a), (b), and (c), which corresponds to $N_{tar} = 5$, 10, and 15 respectively.

We observe that the Runtime of both formulations increases as the time window duration increases, and this increase becomes more prominent as N_{tar} increases. When N_{tar} is 10, the median Runtime of the GCS based formulation outperforms the big-M based formulation by 33.7% and 61.7% when the time window duration is 50 and 75. When N_{tar} is 15 and the time window duration is 50, the median Runtime and the Average Gap (%) of the GCS based formulation outperform the big-M based formulation by 72.3% and 86.4%, respectively. When $N_{tar} = 15$ and the time window duration increases to 75, although both formulations cannot get the optimal solutions within the time limit, the Average Gap (%) of the GCS based formulation is 45.5% smaller than the big-M based formulation.

8.3.2 Varying the Robot Maximum Speed In this section, we consider the second set of experiments where the v_{max} is varied. We do this by fixing the time window duration at 50 secs and τ_m at 3 secs, and varying v_{max} to 4, 6, and 8 unit/sec. The results for this experiment are illustrated in Fig. 8 (d), (e), and (f), which corresponds to $N_{tar} = 5$, 10, and 15, respectively.

Similar to the results of varying the time window duration, the Runtime increases as N_{tar} increases. However, as for each N_{tar}, the Runtime changes very slightly as v_{max} increases for both formulations. The GCS based formulation outperforms the big-M based formulation regarding median Runtime under all the v_{max} settings when N_{tar} is 10 and

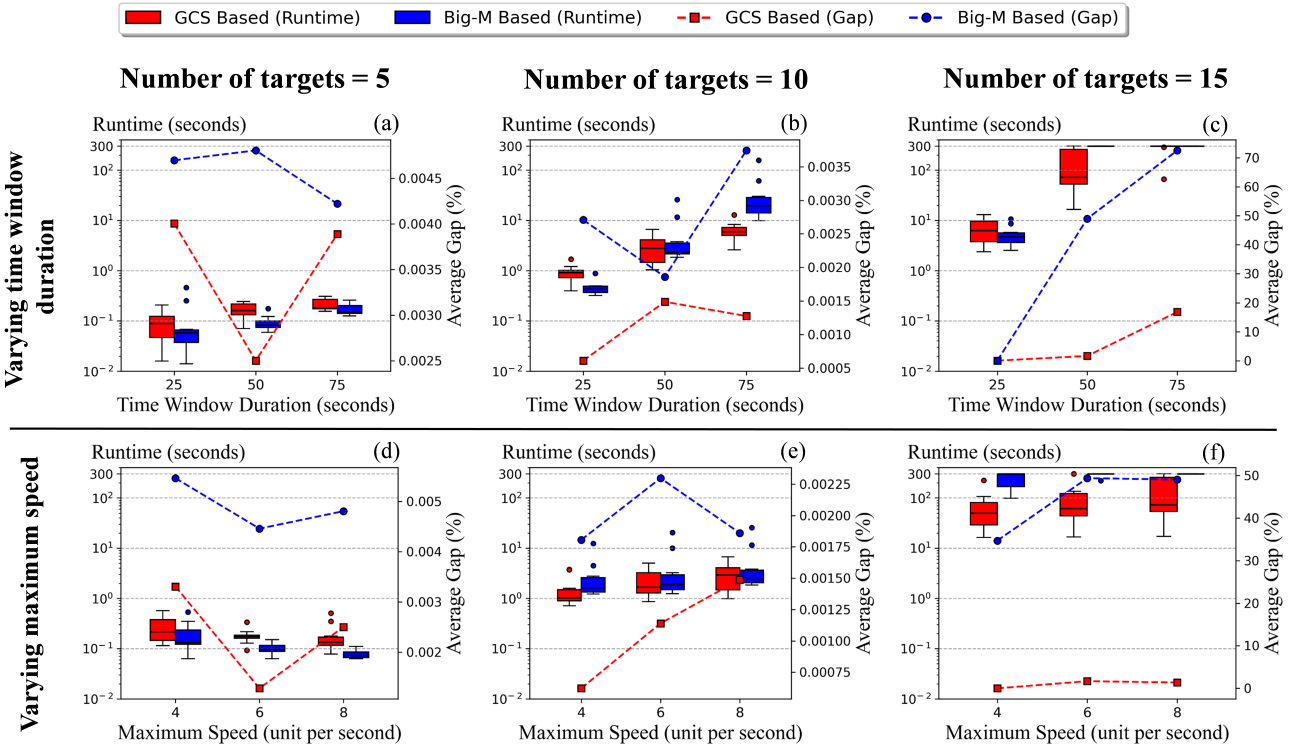

Figure 9. Results of MT-TSP comparing Runtime and Average Gap (%) of the big-M based and GCS based formulation. The Runtime increases with N_{tar} and the time window duration, while it is only weakly affected by variations in v_{max}. The GCS based formulation outperforms the big-M based formulation as N_{tar} increases, especially for N_{tar} = 15 exemplified in (f), where its median Runtime and Average Gap (%) are up to 83.5% and 100.0% smaller than the big-M based formulation, respectively.

15. Especially when N_{tar} is 15, the instances become much more challenging for the big-M based formulation. Besides smaller median Runtime, the Average Gap (%) of the GCS based formulation is vastly smaller than the big-M based formulation by 78.1%, 88.9%, and 86.4% when v_{max} is 4, 6, and 8 unit/sec, respectively.

8.3.3 Varying the Minimum Monitoring Duration In this section, we consider the third set of experiments where τ_m is varied. We do this by fixing the time window duration at 50 secs and v_{max} at 8 unit/sec, and varying τ_m to 1, 3, and 5 secs. The results for this experiment are illustrated in Fig. 8 (g), (h), and (i), which corresponds to $N_{tar} = 5$, 10, and 15, respectively.

The Runtime increases more obviously for the big-M based formulation than the GCS based formulation as N_{tar} increases. When N_{tar} is 10, the median Runtime of the GCS based formulation outperforms the big-M based formulation by 27.1% and 36.1% when $\tau_m = 1$ and 3 secs, respectively. When N_{tar} is 15, the median Runtime of the GCS based formulation outperforms the big-M based formulation under all the three τ_m settings, specifically by 7.6%, 74.4%, and 67.1% for $\tau_m = 1$, 3, and 5 secs, respectively. Besides, the Average Gap (%) of the GCS based formulation obviously outperforms the big-M based formulation when N_{tar} is 15, by 71.1%, 86.9%, and 100.0% for $\tau_m = 1$, 3, and 5 secs, respectively.

8.4 MT-TSP

Recall that MT-TSP is a 2D problem, so we set the workspace as a square of fixed size $L_x = L_y = 100$ units. The depot location was fixed at the center (0, 0) of the workspace. For each N_{tar}, we varied the time window duration and v_{max}. Therefore, a total of 180 instances were generated for MT-TSP.

8.4.1 Varying the Time Window Duration In this section, we consider the experiments where the time window durations are varied. We do this by fixing v_{max} at 8 unit/sec, and solving all the instances for the time window durations (25, 50, and 75 secs). The results for this experiment are illustrated in Fig. 9 (a), (b), and (c), which corresponds to $N_{tar} = 5$, 10, and 15, respectively.

We observe that the Runtime of both formulations increases as the time window duration increases, and this increase becomes more prominent as N_{tar} increases. When N_{tar} is 10 and the time window duration is 75 secs, the median Runtime of the GCS based formulation outperforms the big-M based formulation by 69.4%. When N_{tar} is 15, the median Runtime of the GCS based formulation outperforms the big-M based formulation by 75.8% when the time window duration is 50 secs. As for the Average Gap (%), the GCS based formulation outperforms the big-M based formulation under all the nine configurations, especially for the case of 15 targets where its Average Gap (%) is significantly smaller by 96.7% and 76.8% when the time window duration is 50 and 75, respectively, as compared to the big-M based formulation whose Average Gap (%) increases much more dramatically as the time window duration increases.

8.4.2 Varying the Robot Maximum Speed In this section, we consider the second set of experiments where v_{max} is

varied. We do this by fixing the time window duration at 50 secs, and varying v_{max} to 4, 6, and 8 unit/sec. The results for this experiment are illustrated in Fig. 9 (d), (e), and (f), which corresponds to $N_{tar} = 5$, 10, and 15, respectively.

Similar to the results of varying the time window duration, the Runtime increases as N_{tar} increases. However, as for each N_{tar}, the Runtime changes slightly as v_{max} increases for both formulations. When $N_{tar} = 10$, the median Runtime of the GCS based formulation outperforms the big-M based formulation by 37.0% and 11.3% when v_{max} is 4 and 6 unit/sec, respectively. When $N_{tar} = 15$, the instances become extremely challenging for the big-M based formulation to solve within the time limit, but the GCS based formulation is able to get the optimal solutions within the time limit for most of the instances, achieving 83.5%, 79.5%, and 75.5% smaller median Runtime and 100.0%, 96.6%, and 97.2% smaller Average Gap (%) than the big-M based formulation when $v_{max} = 4$, 6, and 8 unit/sec, respectively. In addition, it is worth noting that the Average Gap (%) of the GCS based formulation is smaller than the big-M based formulation under all the nine configurations, which is similar to the results of varying the time window duration.

9 Conclusion and Future Work

This paper introduced an exact method based on Mixed Integer Conic Program (MICP) for the Dynamic Range Moving-Target Traveling Salesman Problem (DR-MT-TSP), which optimizes the path of a drone equipped with a downward pointing dynamic field of view (FOV) sensor to capture and monitor multiple moving targets. We developed two formulations, a big-M based MICP and a Graph of Convex Sets (GCS) based MICP, and proved their validity and equivalence. Experimental results demonstrate that the GCS based formulation significantly outperforms the big-M based formulation in both runtime and optimality gap as the number of targets increases.

The proposed formulations effectively handle the challenges of dynamic FOV and the monitoring requirements. For future work, we plan to extend the GCS based MICP to multi-agent scenarios, where multiple drones cooperate to monitor moving targets. Additionally, we aim to incorporate more complex target motion patterns, such as piecewise-linear or stochastic trajectories, and integrate collision avoidance and communication constraints to enhance the practicality of the proposed program in real-world deployments.

Acknowledgements

The main ideas in this paper originated during Dr. Ren and Allen's work at CMU and TAMU respectively. This material is partially based on the work supported by the National Science Foundation (NSF) under Grant No. 2120219 and 2120529. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the author(s) and do not necessarily reflect views of the NSF. To conduct this work, Dr. Ren's group at SJTU was also partially supported by the Natural Science Foundation of China under Grant 62403313.

References

Bhat A, Gutow G, Vundurthy B, Ren Z, Rathinam S and Choset H (2024) A complete algorithm for a moving target traveling salesman problem with obstacles. In: *Algorithmic Foundations of Robotics XVI*. Cham: Springer International Publishing.

Bhat A, Gutow G, Vundurthy B, Ren Z, Rathinam S and Choset H (2025a) A complete and bounded-suboptimal algorithm for a moving target traveling salesman problem with obstacles in 3d*. In: *2025 IEEE International Conference on Robotics and Automation (ICRA)*. pp. 6132–6138. DOI:10.1109/ICRA55743.2025.11128828.

Bhat A, Gutow G, Vundurthy B, Ren Z, Rathinam S and Choset H (2025b) Parallel, asymptotically optimal algorithms for moving target traveling salesman problems. *arXiv preprint arXiv:2509.08743* .

Bourjolly JM, Gurtuna O and Lyngvi A (2006) On-orbit servicing: a time-dependent, moving-target traveling salesman problem. *International Transactions in Operational Research* 13(5): 461–481.

Boyd SP and Vandenberghe L (2004) *Convex Optimization*. The Edinburgh Building, Cambridge, CB2 8RU, UK: Cambridge University Press. ISBN 978-0-521-83378-3. Seventh printing with corrections 2009.

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

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

Chalasani P and Motwani R (1999) Approximating capacitated routing and delivery problems. *SIAM Journal on Computing* 28(6): 2133–2149.

Choubey NS (2013) Moving target travelling salesman problem using genetic algorithm. *International Journal of Computer Applications* 70(2).

de Moraes RS and de Freitas EP (2019) Experimental analysis of heuristic solutions for the moving target traveling salesman problem applied to a moving targets monitoring system. *Expert Systems with Applications* 136: 392–409.

Deckerová J, Kučerová K and Faigl J (2023) On improvement heuristic to solutions of the close enough traveling salesman problem in environments with obstacles. In: *2023 European Conference on Mobile Robots (ECMR)*. pp. 1–6.

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

Dumitrescu A and Mitchell JS (2003) Approximation algorithms for tsp with neighborhoods in the plane. *Journal of Algorithms* 48(1): 135–159. Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms.

Englot B, Sahai T and Cohen I (2013) Efficient tracking and pursuit of moving targets by heuristic solution of the traveling salesman problem. In: *52nd IEEE Conference on Decision and Control*. pp. 3433–3438.

Fernandes AS and Feiner SK (2016) Combating vr sickness through subtle dynamic field-of-view modification. In: *2016 IEEE Symposium on 3D User Interfaces (3DUI)*. pp. 201–210.

Grinyer K and Teather RJ (2022) Effects of field of view on dynamic out-of-view target search in virtual reality. In: *2022 IEEE Conference on Virtual Reality and 3D User Interfaces (VR)*. pp. 139–148.

Groba C, Sartal A and Vázquez XH (2015) Solving the dynamic traveling salesman problem using a genetic algorithm with trajectory prediction: An application to fish aggregating devices. *Computers Operations Research* 56: 22–32.

Gutow G and Choset H (2025) Efficient second-order cone programming for the close enough traveling salesman problem. In: *2025 IEEE International Conference on Robotics and Automation (ICRA)*. pp. 12979–12985. DOI:10.1109/ICRA55743.2025.11127623.

Hammar M and Nilsson BJ (2002) Approximation results for kinetic variants of TSP. *Discrete & Computational Geometry* 27(4): 635–651.

Hassoun M, Shoval S, Simchon E and Yedidsion L (2020) The single line moving target traveling salesman problem with release times. *Annals of Operations Research* 289(2): 449–458.

Helvig CS, Robins G and Zelikovsky A (2003) The moving-target traveling salesman problem. *Journal of Algorithms* 49(1): 153–174.

Jiang Q, Sarker R and Abbass H (2005) Tracking moving targets and the non-stationary traveling salesman problem. *Complexity* 11.

Kučerová K, Váňa P and Faigl J (2021) Variable-speed traveling salesman problem for vehicles with curvature constrained trajectories. In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. pp. 4714–4719. DOI: 10.1109/IROS51168.2021.9636762.

Kyaw PT, Paing A, Thu TT, Mohan RE, Vu Le A and Veerajagadheswar P (2020) Coverage path planning for decomposition reconfigurable grid-maps using deep reinforcement learning based travelling salesman problem. *IEEE Access* 8: 225945–225956.

Luo H, Zhang P, Wang J, Wang G and Meng F (2019) Traffic patrolling routing problem with drones in an urban road system. *Sensors* 19(23).

Magaña A, Vlaeyen M, Haitjema H, Bauer P, Schmucker B and Reinhart G (2023) Viewpoint planning for range sensors using feature cluster constrained spaces for robot vision systems. *Sensors* 23(18).

Marcucci T, Umenberger J, Parrilo P and Tedrake R (2024) Shortest paths in graphs of convex sets. *SIAM Journal on Optimization* 34(1): 507–532.

Marlow D, Kilby P and Mercer G (2007) The travelling salesman problem in maritime surveillance - techniques, algorithms and analysis. *MODSIM07 - Land, Water and Environmental Management: Integrated Systems for Sustainability, Proceedings* .

Philip AG, Ren Z, Rathinam S and Choset H (2024) A mixed-integer conic program for the moving-target traveling salesman problem based on a graph of convex sets. In: *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 8847–8853.

Philip AG, Ren Z, Rathinam S and Choset H (2025a) C*: A new bounding approach for the moving-target traveling salesman problem. *IEEE Transactions on Robotics* 41: 4663–4678. DOI: 10.1109/TRO.2025.3588754.

Philip AG, Ren Z, Rathinam S and Choset H (2025b) A mixed-integer conic program for the multi-agent moving-target traveling salesman problem. In: *2025 IEEE 21st International Conference on Automation Science and Engineering (CASE)*. pp. 492–498. DOI:10.1109/CASE58245.2025.11163754.

Rios NC, Mondal S and Tsourdos A (2025) A simulation framework for zoom-aided coverage path planning with uav-mounted ptz cameras. *Sensors* 25(17).

Saller S, Koehler J and Karrenbauer A (2025) A survey on approximability of traveling salesman problems using the TSP-T3CO definition scheme. *Annals of Operations Research* 351(3): 2129–2190.

Smith CD (2021) *Assessment of genetic algorithm based assignment strategies for unmanned systems using the multiple traveling salesman problem with moving targets*. University of Missouri-Kansas City.

Song J, Song H and Wang S (2021) Ptz camera calibration based on improved dlt transformation model and vanishing point constraints. *Optik* 225: 165875.

Stieber A and Fügenschuh A (2022) Dealing with time in the multiple traveling salespersons problem with moving targets. *Central European Journal of Operations Research* 30(3): 991–1017.

Ucar U and Isleyen S (2019) A meta-heuristic solution approach for the destruction of moving targets through air operations. *The International Journal of Industrial Engineering: Theory, Applications and Practice* 26: 986–1004.

Wang Y and Wang N (2023) Moving-target travelling salesman problem for a helicopter patrolling suspicious boats in antipiracy escort operations. *Expert Systems with Applications* 213: 118986.

Xie J, Carrillo LRG and Jin L (2019) An integrated traveling salesman and coverage path planning problem for unmanned aircraft systems. *IEEE Control Systems Letters* 3(1): 67–72.

Xie W, Chu G, Qian Q, Yu Y, Zhai S, Chen D, Wang N, Bao H and Zhangv G (2024) Omnidirectional dense slam for back-to-back fisheye cameras. In: *2024 IEEE International Conference on Robotics and Automation (ICRA)*. pp. 1653–1660.

Zeng R, Wen Y, Zhao W and Liu YJ (2020) View planning in robot active vision: A survey of systems, algorithms, and applications. *Computational Visual Media* 6(3): 225–245.

Zhang W, Sauppe JJ and Jacobson SH (2023) Results for the close-enough traveling salesman problem with a branch-and-bound algorithm. *Computational Optimization and Applications* 85(2): 369–407.

Zhou H, Ji Z, You X, Liu Y, Chen L, Zhao K, Lin S and Huang X (2023) Geometric primitive-guided uav path planning for high-quality image-based reconstruction. *Remote Sensing* 15(10).

Zhu H, Chung JJ, Lawrance NR, Siegwart R and Alonso-Mora J (2021) Online informative path planning for active information gathering of a 3d surface. In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. pp. 1488–1494.