Placement of Unmanned Aerial Vehicles for Directional Coverage in 3D Space

Weijun Wang* † Haipeng Dai* Chao Dong‡ Xiao Cheng* Xiaoyu Wang* Panlong Yang§
Guihai Chen* Wanchun Dou* Xiaoming Fu†

***Abstract*—This paper considers the fundamental problem of Placement of unmanned Aerial vehicles achieviNg 3D Directional coverAge (PANDA), that is, given a set of objects with determined positions and orientations in a 3D space, deploy a fixed number of UAVs by adjusting their positions and orientations such that the overall directional coverage utility for all objects is maximized. First, we establish the 3D directional coverage model for both cameras and objects. Then, we propose a Dominating Coverage Set (DCS) extraction method to reduce the infinite solution space of PANDA to a limited one without performance loss. Finally, we model the reformulated problem as maximizing a monotone submodular function subject to a matroid constraint and present a greedy algorithm with $1-1/e$ approximation ratio to address this problem. We conduct simulations and field experiments to evaluate the proposed algorithm, and the results show that our algorithm outperforms comparison ones by at least 75.4%.**

I. Introduction

Camera sensor network has attracted great attention in recent years as it provides detailed data of environment by retrieving rich information in the form of images and videos [1], [2]. It has found a wide range of applications, such as surveillance, traffic monitoring, and crowd protection, *etc.* For some temporary situations, such as assembly, concerts, matches, and outdoor speeches, establishing stationary camera sensor network in advance may cost too much time and money, and may be inconvenient or even impossible. Fortunately, the development of Unmanned Aerial Vehicle (UAV) technology in the past few years [3]–[6] offers a promising way to address this issue. With the low-cost and agile UAVs, camera sensor network can be deployed dynamically to capture real-time, reliable and high-quality images and videos. For example, *DJI Phantom 4* UAV can fly at $72\,km/h$, rise at $6\,m/s$, swerve at $250\,^\circ/s$, and provide $2K$ real-time images and videos [7].

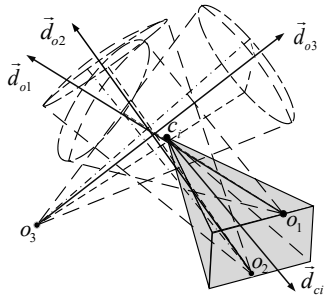


Fig. 1: 3D Directional coverage model

To guarantee high quality of monitoring, UAVs should jointly adjust their coordinates and the orientations of their cameras to cover objects at or near their frontal view [8]. Some sensor placement approaches have been presented in previous works both from the theoretical analysis [9]–[13] and practical experiments [14], [15], but they all focus on omnidirectional sensors and haven't consider the direction of objects. There are some papers that have studied the coverage problem of UAVs [16]–[18], but most of them focus on communication coverage of ground devices and build coverage model as disk which is totally different from our directional coverage. Some sensor placement methods have emerged to study directional sensor or camera sensor coverage problem [19]–[31], but none of them considers that sensors and objects are placed in 3D environment. Although a few camera sensor coverage methods consider 3D environment [32]–[40], almost all of them assume that all objects are still distributed on 2D plane. However, in many scenarios, the objects cannot be served as distributed on 2D plane. For example, in face recognition [41] UAVs won't be more than $15m$ far and $5m$ high from the human faces, which implies the height of humans cannot be ignored, especially they are on different floors. Another example, in crop inspection [42] UAVs need to monitor crop closely to capture the details of them, hence the height of crop also cannot be neglected.

In this paper, we study the problem of Placement of Unmanned Aerial Vehicles achieviNg 3D Directional coverAge (PANDA) [43]. In our considered scenario, some objects are distributed in 3D space with known facing direction, and we have a given number of UAVs to deploy in the free 3D space

This work was supported in part by the National Key R&D Program of China under Grant No. 2018YFB1004704, in part by the National Natural Science Foundation of China under Grant No. 61502229, 61872178, 61832005, 61672276, 61872173, 61472445, 61631020, 61702525, 61702545, 61931011, and 61321491, in part by the Natural Science Foundation of Jiangsu Province under Grant No. BK20181251, in part by the Fundamental Research Funds for the Central Universities under Grant 021014380079, in part by the Natural Science Foundation of Jiangsu Province under Grant No. BK20140076.5, in part by the Nature Science Foundation of Jiangsu for Distinguished Young Scientist under Grant BK20170039. *(Corresponding author: Guihai Chen.)*

W. Wang, H. Dai, X. Cheng, X. Wang, G. Chen and W. Dou are with the State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China.
E-mails:{weijunwang, xiaocheng, xiaoyuwang}@smail.nju.edu.cn, {haipengdai, gchen, wcdou}@nju.edu.cn.

C. Dong is with Nanjing University of Aeronautics and Astronautics, Nanjing, Jiangsu 210007, CHINA. E-mail: dch999@gmail.com

Panlong Yang is with University of Science and Technology of China, Hefei, Anhui 230026, CHINA. E-mail: plyang@ustc.edu.cn

W. Wang and X. Fu is with George-August University Göttingen, Göttingen, 37077, Germany. E-mail: fu@cs.uni-goettingen.de

**As the author changes are not approved by the editor(s) of ToN, there are only 8 authors in the published paper. On *Arxiv*, we list all the authors who contribute to this paper.

TABLE I: Notations

Symbol	Meaning
c_i	UAV i, or its 3D coordinate
o_j	Object j to be monitored, or its 3D coordinate
N	Number of UAVs to be deployed
M	Number of objects to be monitored
$\vec{d}_{ci}$	Orientation of camera of UAV i
γ	Pitching angle of camera
γ_{min}	Minimum pitching angle
γ_{max}	Maximum pitching angle
α	Horizontal offset angles of the FoV around $\vec{d}_{ci}$
β	Vertical offset angles of the FoV around $\vec{d}_{ci}$
$\vec{d}_{oj}$	Orientation of object o_j
θ	Efficient angle around $\vec{d}_{oj}$ for directional coverage
Δ	Farthest sight distance of camera with guaranteed monitoring quality

whose cameras can freely adjust their orientation. The practical 3D directional coverage model for cameras and objects is established as Figure 1. The coverage space of camera is modeled as a straight rectangle pyramid and the efficient coverage space of object is modeled as a spherical base cone, which are both fundamentally different from previous work. Moreover, we define *directional coverage utility* to characterize the effectiveness of directional coverage for all objects. Formally, given a fixed number of UAVs and a set of objects in the space, PANDA problem is to deploy the UAVs in the 3D free space, *i.e.*, to determine their coordinates and orientations (the combinations of which we define as *arrangements*), such that the directional coverage utility for all objects is maximized.

We face two main technical challenges to address PANDA. The first challenge is that the problem is essentially continues and nonlinear. The problem is continues because both coordinates and orientations of UAVs are continuous values. It is also nonlinear because the angular constraints for both camera model and object model, and the constraint of pitching angle of camera in objective function. The second challenge is to develop an approximation algorithm which needs to bound the performance gap to the optimal solution.

To address these two challenges, first, we propose a Dominating Coverage Set extraction method to reduce the continuous search space of arrangements of UAVs to a limited number of arrangements without performance loss. Hence, PANDA problem becomes to select a fixed number of arrangements from a set of candidate ones to maximize the overall object directional coverage utility, which is discrete, and we address the first challenge. Then, we prove that the reformulated problem falls into the scope of the problem of maximizing a monotone submodular function subject to a matroid constraint, which allows a greedy algorithm to achieve a constant approximation ratio . Consequently, we address the second challenge.

We conducted both simulations and field experiments to evaluate our proposed algorithm. The results show that our algorithm outperforms comparison algorithms by at least 75.4%.

II. Related Works

Directional sensor coverage and camera sensor coverage. Directional sensor coverage works can be classified into object coverage and area coverage, whose goals are maximizing the number of covered objects [19], [20] and area coverage ratio [21], [22], respectively. However, most of them do not take the objects' facing direction into account. Some camera sensor coverage works consider objects' facing direction [23]–[31]. Wang *et al.* in [23] proposed a full-view coverage model by introducing objects' facing direction into the coverage model. Then, the full-view coverage model is extended to more scenarios in [1], [24]–[31]. Hu *et al.* in [25] proposed an effective algorithm to solve full-view coverage problem in the mobile heterogeneous camera sensor networks. Some works [24], [26], [27], [29] focus on using minimum number of sensors to achieve barrier coverage, while [27] further considers intruders' faces for most intruders' trajectories crossing the barrier and [29] applies mobile camera sensors. Some works on autonomous cinematography consider the facing direction of multiple objects in one image. Joubert *et al.* in [44] presented a system to figure out the strategy of an UAV for capturing well-composed photos of two objects. Nageli *et al.* in [45] jointly considered monitoring quality and occlusion to optimize UAVs motion plans and associated velocities. But all of them only consider 2D sector model which is also the assumption in most of directional sensor coverage works.

3D camera sensor coverage. There exists a few works focusing on camera sensor coverage in 3D environment [32]–[40]. Ma *et al.* in [32] proposed the first 3D camera coverage model and developed an algorithm for area coverage on 2D plane with the projecting quadrilateral area of 3D camera coverage model. Based on this model, Yang *et al.* in [33] introduced coverage correlation model of neighbor cameras to decrease the number of cameras. Han *et al.* in [34] and Yang *et al.* in [35] took energy and storage of camera sensors into account and proposed high-efficient resource utility coverage algorithm. Si *et al.* in [36] considered the intruders' facing direction and the size of face in barrier coverage. Hosseini *et al.* in [37] addressed the problem of camera selection and configuration problem for object coverage by binary integer programming solution. Li *et al.* in [38] and Peng *et al.* in [39], [40] established a more practical 3D camera coverage model, and studied three area coverage problems based on this model. Specifically, [38], [40] focus on maximize area coverage ratio, while [39] proposed a coverage hole detection and redeployment algorithm. However, most of above works simply use 3D camera coverage model to address those problems whose covered objects or area are only distributed on 2D plane, and thus none of them can solve our problem.

III. Problem Statement

Suppose we have M objects $O = \{o_1, o_2, ..., o_M\}$ to be monitored in a 3D free space, each object o_j has a known orientation, which is denoted by a vector $\vec{d}_{oj}$. We also have N UAVs $C = \{c_1, c_2, ..., c_N\}$ equipped with cameras which can hover anywhere in the 3D free space. Because of hardware limitation, their cameras can only rotate in the vertical plane. However, this limitation has no influence to the coverage orientation, because UAV can hover in the air and rotate itself to face any horizontal orientation. By a little abuse of notation,

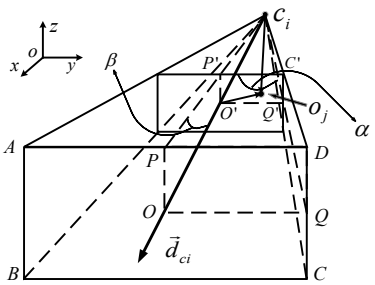


Fig. 2: Camera model

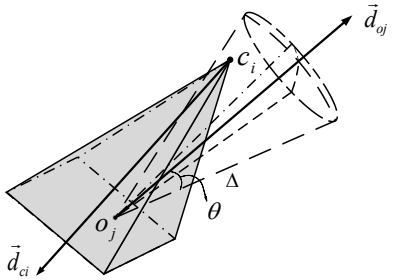


Fig. 3: Directional coverage model

c_i and o_j also denote the coordinate of UAV and object. Table I lists the notations we use in this paper.

A. Camera Model

More complicated than previous sector model of 2D directional sensor, camera model in 3D environment needs to be modeled as a straight rectangular pyramid as shown in Figure 2. Due to cameras only can rotate in vertical plane, edge $\overline{AD}$ and $\overline{BC}$ are always parallel to the ground.

We use a 5-tuple (c_i, $\vec{d}_{ci}$, γ, α, β) to denote the camera model. c_i is coordinate (x_0, y_0, z_0) of an UAV in 3D space, $\vec{d}_{ci}$ is the orientation of the camera at the time, γ ($\gamma_{min} \leqslant \gamma \leqslant \gamma_{max}$) is the pitching angle of this orientation, α and β are the camera's horizontal and vertical offset angles of FoV (Field of View) around $\vec{d}_{ci}$. As illustrated in Figure 2, point c_i denotes the coordinate of UAV c_i, its value is (x_0, y_0, z_0). Vector $\vec{d}_{ci}$ denotes the orientation of c_i's camera which is perpendicular to undersurface *ABCD* and its unit vector equals to (x_1, y_1, z_1). Point O is the centre of rectangle *ABCD* and the distance $|\overline{c_iO}| = \Delta$, where Δ is the farthest distance from camera which can guarantee the quality of monitoring of every object on *ABCD*. Thus, the coordinate of point O is $(x_0+\Delta x_1, y_0+\Delta y_1, z_0+\Delta z_1)$. Clearly, we can mathematically express plane $ABCD$ as

$$\vec{d}_{ci} \cdot \begin{pmatrix} x-(x_0+Dx_1) \\ y-(y_0+Dy_1) \\ z-(z_0+Dz_1) \end{pmatrix} = 0. \tag{1}$$

Connecting point c_i to midpoint P of $\overline{AD}$ and Q of $\overline{CD}$ respectively, we can get plane c_iOP and c_iOQ. As cameras only can rotate in vertical field, plane c_iOP is parallel to z-axis. Thus, plane c_iOP can be expressed as

$$-y_1x + x_1y + x_0y_1 - y_0x_1 = 0. \tag{2}$$

As shown in Figure 2, $\overline{OQ} \perp \overline{OP}, \overline{OQ} \perp \overline{Oc_i}$, thus $\overline{OQ} \perp c_iOP$. By Equation (2) and $\angle Oc_iQ$ equals to the horizontal offset angle α, so $|\overline{OQ}| = D \cdot \tan\alpha$. Thus, we can obtain

$$\overrightarrow{OQ} = \Delta \cdot \tan\alpha \cdot (-y_1, x_1, 0). \tag{3}$$

Similar, $\overline{OP} \perp c_iOQ$, $\angle Oc_iP$ equals to the vertical offset angle β, then we have $|\overline{OP}| = \Delta \cdot \tan\beta$. Combine the equation of plane c_iOP, vector $\overrightarrow{OP}$ can be obtained as

$$\overrightarrow{OP} = (x_1z_1, y_1z_1, -y_1^2 - x_1^2) \cdot \Delta \cdot \tan\beta. \tag{4}$$

To a given object o_j, if o_j is covered by c_i, it must be in some rectangle which is parallel to $ABCD$, *i.e.*, rectangle Ω between c_i and $ABCD$ in Figure 2. According to this idea, we can illustrate the camera model as follows. Point O' is the centre of the rectangle and its coordinate is easy to figure out by o_j and normal vector $\vec{d}_{ci}$. Utilize normal vectors $\overrightarrow{OQ}$ in Equation (3) and $\overrightarrow{OP}$ in Equation (4), if o_j satisfies the following constraint, point o_j is covered by camera c_i.

$$F_c(c_i,o_j,\vec{d}_{ci},\vec{d}_{oj}) = \begin{cases} 1, & Prj_{\overrightarrow{OP}}\overrightarrow{O'o_j} \leq |\overline{O'P'}|, \\ & Prj_{\overrightarrow{OQ}}\overrightarrow{O'o_j} \leq |\overline{O'Q'}|, \\ & Prj_{\vec{d}_{ci}}\overrightarrow{c_io_j} \leq \Delta. \\ 0, & otherwise. \end{cases} \tag{5}$$

$$s.t. \quad |\overline{c_iO'}| = Prj_{\vec{d}_{ci}}\overrightarrow{c_io_j}, |\overline{O'P'}| = |\overline{c_iO'}| \cdot \tan\beta, \quad |\overline{O'Q'}| = |\overline{c_iO'}| \cdot \tan\alpha.$$

We use Figure 1 as an example to illustrate our Camera Model. In Figure 1, o_1 and o_2 lie in the straight rectangular pyramid of c_i, thus the coordinates of o_1 and o_2 satisfy the Equation (5) of UAV c_i but o_3 doesn't.

B. 3D Directional Coverage Model

First, we define 3D directional coverage as follows.

Definition 3.1: **(3D directional coverage)** For an given object o_j and its facing direction $\vec{d}(x, y, z)$, there is an UAV c_i with camera orientation $\vec{d}_{ci}$, such that o_j is covered by c_i and $\alpha(\vec{d}, \overrightarrow{o_jc_i}) \leq \theta$ (θ is called the efficient angle), then object o_j is 3D directional covered by c_i.

According to Definition 3.1, object model can be established as a spherical base cone as shown in Figure 3. Let Object o_j be the vertex, rotate a sector of θ central angle and Δ radius around vector $\vec{d}$ for one revolution, then we obtain the object model as follows.

$$\begin{cases} |\overrightarrow{c_io_j}| \leq \Delta, \\ \alpha(\overrightarrow{o_jc_i}, \vec{d}_{oj}) \leq \theta. \end{cases} \tag{6}$$

Based on Equation (6), UAV c_i can efficiently cover object o_j only when it locates in the cone of object o_j where can guarantee $|\overrightarrow{c_io_j}| \leq \Delta$ and $\alpha(\overrightarrow{o_jc_i}, \vec{d}_{oj}) \leq \theta$. Thereby, combine

Equation (5) and (6), we can obtain the *directional coverage function* as

$$F_v(c_i,o_j,\vec{d}_{ci},\vec{d}_{oj}) = \begin{cases} 1, & Prj_{\overrightarrow{OP}}\overrightarrow{O'o_j} \le |\overline{O'P'}|, \\ & Prj_{\overrightarrow{OQ}}\overrightarrow{O'o_j} \le |\overline{O'Q'}|, \\ & Prj_{\vec{d}_{ci}}\overrightarrow{c_io_j} \le |\overrightarrow{c_io_j}| \le \Delta. \\ & \alpha(\overrightarrow{o_jc_i},\vec{d}_{oj}) \le \theta. \\ 0, & otherwise \end{cases}$$
$$s.t. \quad |\overrightarrow{c_iO'}| = Prj_{\vec{d}_{ci}}\overrightarrow{c_io_j}, |\overline{O'P'}| = |\overline{c_iO'}| \times \tan\beta, \quad |\overline{O'Q'}| = |\overline{c_iO'}| \cdot \tan\alpha. \tag{7}$$

Then, the *directional coverage utility* can be defined as

$$\mathcal{U}_v(c_i,\vec{d}_{ci},o_j,\vec{d}_{oj}) = \begin{cases} 1, & \sum_{i=1}^{N} F_v(c_i,o_j,\vec{d}_{ci},\vec{d}_{oj}) \ge 1, \\ 0, & otherwise. \end{cases} \tag{8}$$

Similar to Camera Model, We also use Figure 1 as an example to illustrate our 3D Directional Coverage Model and Directional Coverage Utility. In Figure 1, UAV c_i lies in the spherical base cone of o_1, o_2, and o_3, thus the coordinate of c_i satisfies the Equation (6) of all three objects. Combine the Camera Model, c_i, o_1, and c_i, o_2 both establish a 3D Directional Coverage subject to Equation (7). Consequently, the directional coverage utility of c_i, o_1, $\mathcal{U}_v(c_i,\vec{d}_{ci},o_1,\vec{d}_{o1}) = 1$, and the directional coverage utility of c_i, o_2, $\mathcal{U}_v(c_i,\vec{d}_{ci},o_2,\vec{d}_{o2}) = 1$, but to c_i, o_3, the utility $\mathcal{U}_v(c_i,\vec{d}_{ci},o_3,\vec{d}_{o3}) = 0$.

C. Problem Formulation

In our problem, assume that we have obtained the orientation and coordinates of objects from the location and tracking technology of Internet of Things [46]–[49]. In addition, in many scenarios, such as matches, outdoor concerts, and presidential campaign speeches, people always stand or sit at fixed position and stare at the matches, stage, or speakers, thus, we can obtain their coordinates and orientations beforehand. Formally, we obtain o_j and $\vec{d}_{oj}$ in equations (5) and (7). Note that the parameters of cameras on UAVs are obtained in advance, thus the constants Δ, α, and β are also obtained. Our goal is to determine the coordinates c_i and orientations $\vec{d}_{ci}$ of UAVs.

Let the tuple $\langle c_i,\vec{d}_{ci}\rangle$, called $arrangement$, denotes the coordinate of UAV c_i and orientation of its camera $\vec{d}_{ci}$. Due to the limited resource of UAV, our task is to determine the arrangements for all N UAVs to optimize overall directional coverage utility for all M objects. Formally, the 3D Placement of Unmanned Aerial Vehicle achieviNg Directional coverAge (PANDA) problem is defined as follows.

$$\textbf{(P1)} \quad max \quad \sum_{j=1}^{M}\mathcal{U}_v(\sum_{i=1}^{N}F_v(c_i,o_j,\vec{d}_{ci},\vec{d}_{oj})),$$
$$s.t. \quad \gamma_{min} \le \gamma \le \gamma_{max}.$$

As commonly stated, the goal of PANDA is using N UAVs to directionally cover the maximum number of objects. If an object is covered by an UAV, the utility will increase by one. However, overlapping coverage of the same object won't bring any contribution.

In the following theorem, we prove the PANDA problem is NP-hard.

***Theorem** 3.1:* The PANDA problem is NP-hard.

Proof: To show the difficulty of the PANDA problem, we consider a simple case in which $\alpha = \beta = \theta = \pi$ and $\Delta = 1$, *i.e.*, the camera model is omnidirectional and each object can be covered from any orientation. Namely, as long as an object is located in the coverage of any camera, which is a unit ball, it can be covered by this camera. Our PANDA problem is transformed into using a fixed number of balls with radius of 1 to cover as many as objects in a 3D space. Note that the coverage problem on a 2D plane is a special case of constrained 3D space. This special case is exactly the well-known Unit Disk Coverage problem, which is NP-hard [50].

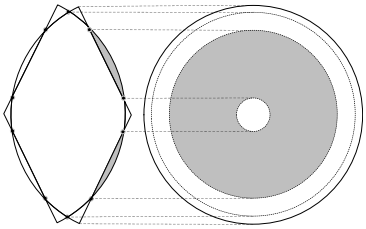
If we can design polynomial algorithm to address the original problem PANDA, obviously, we can address the NP-hard Unit Disk Coverage problem with this same algorithm. However, one NP-hard problem can't be addressed in polynomial time unless $P = NP$. Therefore, the PANDA is NP-hard problem, which doesn't have polynomial algorithm. ■

IV. Solution

In this section, we present an algorithm with approximation ratio $1-1/e$ to address PANDA. Before describing the details of it, we first present the key intuitions and solution overview.

A. Key Intuitions and Solution Overview

1) Key Intuitions: (1) Focus on possible covered sets of objects of UAVs rather than candidate positions and orientations of UAVs. As we can deploy UAVs on any position and set the orientation of their cameras arbitrarily, the number of candidate arrangements of UAVs is infinite, or the solution space of PANDA is infinite. However, many arrangements are essentially equivalent if they cover the same set of objects. Apparently, we only need to consider one representative arrangement among its associated class of all equivalent arrangements, and the number of all such representative arrangements is finite because the number of all possible covered sets of objects is finite. *(2) Focus on those arrangements that cover larger sets of objects.* If a representative arrangement covers the set of objects $\{o_1,o_2,o_3,o_4\}$, undoubtedly you don't need to consider arrangements that cover its subsets, such as $\{o_1,o_2\}$ or $\{o_2,o_3,o_4\}$. Our goal is to find those larger representative arrangements that possibly cover maximal covered sets of objects while avoiding the others. *(3) Find or "create" constraints to help determine representative arrangements.* As the equations listed in seven-objet case in Section IV.C, uniquely determine an arrangement, mathematically we need at least 9 equations. However, even if we know the associated covered set of objects for a class of equivalent arrangements how can we efficiently determine at least one representative arrangement? Our solution is to imagine that given a feasible arrangement, we can adjust its position and orientation such that one or more objects touch some sides of the pyramid of the arrangement while keeping no objects



(a) Side view (b) Top view

Fig. 4: Intersection of two cone model

out of coverage. Obviously, the obtained arrangement after the adjustment is also feasible and can be selected as a representative arrangement. Then, we in turn use the touching conditions as constraints and formulate them as equations to help determine the representative arrangements. Apart from such kind of constraints, we also find an additional constraint regarding positions of representative arrangements for their determination in this paper.

2) Solution Overview: First, any object only can be efficiently covered by an UAV in its efficient coverage space, which is modeled as a spherical base cone, by the 3D Directional Coverage Model. The whole coordinate solution space of UAV is thus the union of all spherical base cones for all objects. We further present a space partition approach to partition the whole coordinate solution space into multiple cells, which can be considered separately. Especially, for each cell, the set of all possibly covered objects by adjusting the orientation of an UAV at any coordinate in the cell is exactly the same.

Second, we propose the notion of Dominating Coverage Set (DCS), which covers the maximal set of objects and has no proper superset of covered objects by other arrangements. Then, our goal turns to find all candidate DCSs and their associated representative arrangements. Specifically, we first prove that the coordinates of associated arrangements of DCSs must lie on the boundaries of cells, which serves as a constraint to help determine the representative arrangements. Next, we imagine that we adjust the coordinates and orientations of arrangements to create touching conditions and thus new constraints. More importantly, in Section IV.C we intuitively give the formulation in seven-object case, as well as prove the Theorem 4.2 that it is sufficient to enumerate the seven different cases, where 1 to 7 objects touch on the sides of pyramid, to extract all possible DCSs. We also analyze the enumeration process must be done in polynomial time. Consequently, our problem is transformed into choosing N arrangements among the obtained candidate DCS arrangements to maximize the number of covered objects.

Finally, we prove that the transformed problem falls into the realm of maximizing a monotone submodular optimization problem subject to a uniform matroid, and propose a greedy algorithm to solve it with performance guarantee.

B. 3D Space Partition

As mentioned in 3D Directional Coverage Model, efficient coverage space of each object is modeled as a spherical base cone. These spherical base cones intersect among each other and form many 3D partitions called *cells*.

Due to geometric symmetry, only the UAVs locating in cells have chance to cover objects, and their potentially covered objects vary from one cell to another. For example, in Figure 1, UAV c_i locates in the common cell of o_1 and o_2 and it can cover o_1 and o_2 simultaneously. Then we focus on the upper bound of the number of cells.

***Theorem** 4.1:* The number of partitioned cells is subject to $Z = O(M^2)$.

Proof: We first decrease the dimensions to 2D plane and analyze the upper bound of the number of partitioned cells on 2D plane by M uniform sectors intersecting with each other. Then, we prove that this upper bound is also the upper bound of the original 3D scenario by reduction.

***Claim** 4.1:* The number of partitioned cells on 2D plane by n uniform sectors intersecting with each other is at most $5n^2 - 5n + 2$.

Proof: First, we analyze the relationship between the number of cells and that of intersection points. Obviously, if there are three or more edges or arcs intersecting at same point, the number of cells must not be maximized. Thus, consider the condition that there are only two edges or arcs intersecting at one point, then one intersection point divides each edge into two parts, *i.e.*, the total number of added edges is 2 times that of intersection point. Let e denote the initial total number of edges, v denote the initial total number of vertices, f denote the initial total number of cells, (*i.e.*, faces in Graph Theory), and x denote the added intersection point. Due to the Euler characteristic [51], we have $f = e - v + 2 = (e + 2x) - (v + x) + 2 = x + 2$.

Furthermore, we observe that when the radian of sector is in $(\pi/2, \pi)$, there are the most intersection points for two sectors intersects with each other, *i.e.*, 10 intersection points. Thus, any pair among n sectors intersect at 10 different points, and there are at most $10 \cdot \binom{n}{2} = 5n^2 - 5n$ intersection points. By $f = x + 2$, the total number of cell is at most $5n^2 - 5n + 2$. ■

In original 3D scenario, the side view and the top view of two cones intersecting with each other are depicted in Figure 4. As shown in Figure 4(a), from the side view, two cones intersect with each other by 10 intersection points. However, from the top view as shown in Figure 4(b), the cells with grey color are also connected with each other in another dimension. This connection condition also happen in other symmetric cells. Therefore, the number of cells is less than the number of cells $5n^2 - 5n + 2$, *i.e.*, the number of cells $Z = O(M^2)$ ■

C. Dominating Coverage Set (DCS) Extraction

After the space partition, we only need to consider the relationship between objects and UAVs in each cell, which depends on the coordinates and orientations of UAVs. In this subsection, we show that instead of enumerating all possible covered sets of objects, we only need to consider a limited number of representative covered sets of objects, which are

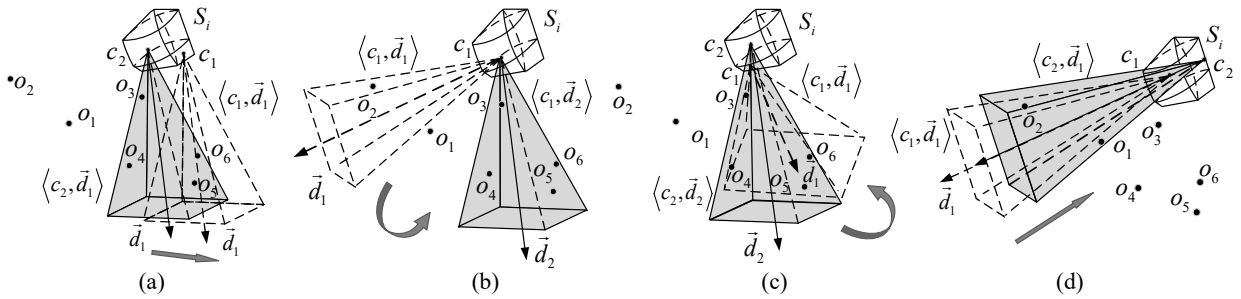


Fig. 5: Four kinds of transformations: (a) Translation, (b) Rotation Around Camera, (c) Rotation Around object(s), (d) Projection

defined as Dominating Coverage Sets (DCSs), and figure out their corresponding arrangements. Our ultimate goal is to reduce the problem to a combinatorial optimization problem which is selecting N arrangements from a limited number of arrangements obtained by DCS extraction.

1) Preliminaries: To begin with, we give the following definitions to assist analysis.

***Definition** 4.1:* **(Dominance)** Given two arrangements $\langle c_1, \vec{d}_{c1}\rangle$, $\langle c_2, \vec{d}_{c2}\rangle$ and their covered sets of objects O_1 and O_2. If $O_1 = O_2$, $\langle c_1, \vec{d}_{c1}\rangle$ is equivalent to $\langle c_2, \vec{d}_{c2}\rangle$, or $\langle c_1, \vec{d}_{c1}\rangle \equiv \langle c_2, \vec{d}_{c2}\rangle$; If $O_1 \supset O_2$, $\langle c_1, \vec{d}_{c1}\rangle$ dominates $\langle c_2, \vec{d}_{c2}\rangle$, or $\langle c_1, \vec{d}_{c1}\rangle \succ \langle c_2, \vec{d}_{c2}\rangle$; And if $O_1 \supseteq O_2$, $\langle c_1, \vec{d}_{c1}\rangle \succeq \langle c_2, \vec{d}_{c2}\rangle$.

***Definition** 4.2:* **(Dominating Coverage Set)** Given a set of objects O_i covered by an arrangement $\langle c_i, \vec{d}_{ci}\rangle$, if there does not exist an arrangement $\langle c_j, \vec{d}_{cj}\rangle$ such that $\langle c_j, \vec{d}_{cj}\rangle \succ \langle c_i, \vec{d}_{ci}\rangle$, then O_i is a Dominating Coverage Set (DCS).

For a given cell, it is possible only a few objects in the ground set of objects can be covered by an UAV locating in this cell. We formally give the following definition.

***Definition** 4.3:* **(Candidate Covered Set of Objects)** The candidate covered set of objects $\hat{O}_i$ for cell S_k are those objects possible to be covered by UAV c_i with some orientation $\vec{d_{ci}}$ in S_k.

Obviously, any DCS of a cell is a subset of its candidate covered set of objects $\hat{O}_i$.

As selecting DCSs is always better than selecting its subsets, we focus on figuring out all DCSs as well as their arrangements. In what follows, we first study two special cases where a coverage cell is reduced to a vertice (vertice case) and a line (line case) to pave the way for analyzing the general case.

2) DCS Extraction for the Vertice Case: First, we define four kinds of transformations as follows.

***Definition** 4.4:* **(Translation)** Given an arrangement $\langle c_1, \vec{d}_{c1}\rangle$, keep the orientation unchanged and move the UAV from coordinate c_1 to coordinate c_2.

***Definition** 4.5:* **(Rotation Around Camera (RAC))** Given an arrangement $\langle c_1, \vec{d}_{c1}\rangle$, keep the coordinate unchanged and rotate the orientation from $\vec{d}_{c1}$ to $\vec{d}_{c2}$.

***Definition** 4.6:* **(Rotation Around Objects (RAO))** Given an arrangement $\langle c_1, \vec{d}_{c1}\rangle$, keep the object(s) on the touching side of pyramid, *i.e.*, left side in Figure 5(c), and move the UAV from $\langle c_1, \vec{d}_{c1}\rangle$ to $\langle c_2, \vec{d}_{c2}\rangle$.

***Definition** 4.7:* **(Projection)** Given an arrangement $\langle c_1, \vec{d}_{c1}\rangle$, keep the orientation unchanged and move the UAV along the reverse direction of orientation $\vec{d}_{c1}$ until reaching some point c_2 on the boundary of cell, *i.e.*, $\langle c_2, \vec{d}_{c1}\rangle = f_{\perp}(\langle c_1, \vec{d}_{c1}\rangle)$.

Figure 5 depicts four instances of these four transformations. Obviously, *Projection* is a special case of *Translation*. Figure 5(c) illustrates the *RAO* subject to objects o_3 and o_4.

Then, we present the DCS extraction algorithm for point case as shown in Algorithm 1. Basically, the algorithm is a greedy algorithm which lets the UAV locate at the vertice and rotate around $o_j \in \hat{O}_i$ for a circle. Object o_j will slide on left, up, right, down sides orderly as illustrated in Figure 7. During this process, Algorithm 1 tracks the current set of covered objects, and records all DCSs. The input of DCS extraction for point case is the vertice S_i and its candidate covered set of objects $\hat{O}_i$. The output is the set of all DCSs.

We use a toy example to illustrate the key idea of the DCS extraction for the vertice case. Figure 6 illustrates the loop for object o_3, and the initial arrangement is $\langle S_i, \vec{d}_3^{\,left}\rangle$ as shown in Figure 6(a). First, keep o_3 sliding on the left side of the straight rectangle pyramid and execute RAC transformation until o_3 touching up side, whose orientation is $\vec{d}_3^{\,up}$, as illustrated in Figure 6(a). Second, keep o_3 sliding on the up side and execute RAC as shown in Figure 6(b). During this sliding process, o_5 will get out of the pyramid. When o_5 touching the right side, add current DCS $\{o_3, o_4, o_5\}$ to the set of DCSs, then continue to slide until o_3 touching right side, whose orientation is $\vec{d}_3^{\,right}$. Third, keep o_3 sliding on the right side and execute RAC as shown in Figure 6(c). Then, add DCSs $\{o_1, o_4, o_5\}$, $\{o_1, o_2, o_5\}$ to the set of DCSs orderly when o_4 and o_1 touch the down side in order. Forth, keep o_3 sliding on the down side and execute RAC as shown in Figure 6(d). When o_2 touches the left side, add DCS $\{o_2, o_3\}$. At last, slide o_3 such that it assumes its initial arrangement as shown in Figure 6(e), and add $\{o_3\}$. Note that for better readability, we just present the final situation of sliding on each side in Figure 6.

3) DCS Extraction for Line Case: Line case is a specific instance of cell case. All the approaches to extract DCSs for any subcase are the same as following cell case. Thus, we omit this part in this paper to save space.

4) DCS Extraction for the Cell Case: Now, we consider extracting DCSs for cell case. According to the definition of *projection* , we have the following lemma.

***Lemma** 4.1:* If $\langle c_2, \vec{d}_{c1}\rangle = f_{\perp}(\langle c_1, \vec{d}_{c1}\rangle)$, then $\langle c_2, \vec{d}_{c1}\rangle \succeq \langle c_1, \vec{d}_{c1}\rangle$.

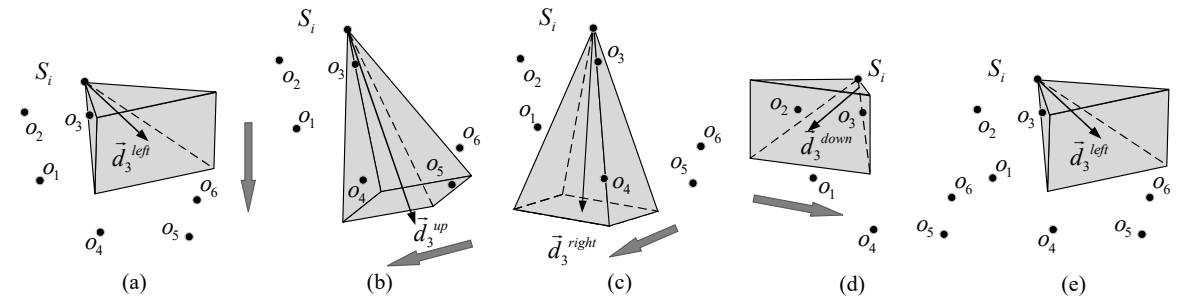


Fig. 6: DCSs extraction for vertice case (object o_3 loop) (a) Rotate pyramid down as well as keep o_3 sliding on the left side of pyramid. (b) Rotate pyramid left as well as keep o_3 sliding on the up side of pyramid. (c) Rotate pyramid up as well as keep o_3 sliding on the right side of pyramid. (d) Rotate pyramid right as well as keep o_3 sliding on the down side of pyramid. (e) Rotate pyramid down as well as keep o_3 sliding on the left side of pyramid until back to the original place.

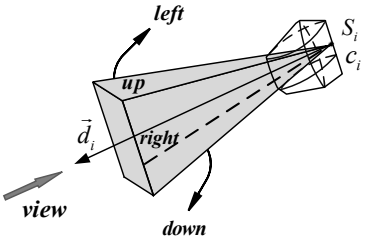
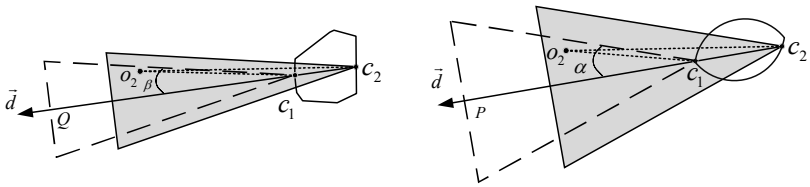


Fig. 7: View orientation



(a) Right view figure

(b) Up view figure

Fig. 8: Illustration for Lemma 4.1

Algorithm 1: DCSs Extraction for the Vertive Case

Input: The vertice S_i, the candidate covered set of objects $\hat{O}_i$

Output: All DCSs

1 Computer pitching angle of each object with $\vec{d}_{S_i o_j}$.

2 **for** *every* o_j *in* $\hat{O}_i$ **do**

3 Initialize the rotated horizonal angle as the horizonal component of $\vec{d}_{S_i o_j} - \alpha$ and the vertical angle as γ_j.

4 Keep object o_j sliding on the sides of the straight rectangle pyramid orderly, execute RAC transformation, where pitching angle changes between $\gamma_j + \beta$ and $\gamma_j - \beta$ and horizonal angle changes between $\vec{d}_{S_i o_j} - \alpha$ and $\vec{d}_{S_i o_j} + \alpha$, until there is at least one object will be uncovered. During the sliding process, if o_j returns to the initialization value, add the current covered set of objects to the collection of DCSs and break.

5 Add the current covered set of objects to the collection of DCSs.

6 Keep o_j continue sliding on the four sides of the straight rectangle pyramid orderly until a new object is covered. During the sliding process, if o_j returns to the initialization value, break. If not, goto step 4.

Proof: First, we prove that no object will fall out of the straight rectangle pyramid through undersurface by the distance from objects to UAV. As mentioned in Section IV-C, every cell is formed by several spherical base cones with height of Δ. Thus, any point in given cell won't be farther than Δ from any object in $\hat{O}_i$ of this cell. As a result, no object will fall out of the pyramid through its undersurface during the process of projection. Second, we prove that no object will fall out of the pyramid through its four sides by the angle between objects and UAV. Figure 8(a) illustrates the view from right of Figure 5(d). In Figure 8(b), it is obvious that $\angle o_2 c_2 Q < \angle o_2 c_1 Q$ since $\angle o_2 c_2 Q = \angle o_2 c_1 Q - \angle c_1 o_2 c_2$. So, the condition $\angle o_2 c_1 Q < \beta$ ensures $\angle o_2 c_2 Q < \beta$, *i.e.*, *projection* won't make objects fall out of pyramid through up or down side. Figure 8(b) illustrates the view from up of Figure 5(d). Similarly, as shown in Figure 8(b), no objects will fall out of the pyramid through right or left side because $\angle o_2 c_2 P < \angle o_2 c_1 P < \alpha$. In summary, *projection* won't drop any initially covered object out of coverage, but, in contrast, it leads new objects to be covered. ■

As Figure 5(d) shows, after *projection* transformation, arrangement $\langle c_2, \vec{d}_{c1} \rangle$ covers o_1 and o_2 which are not covered by $\langle c_1, \vec{d}_{c1} \rangle$ before.

By Lemma 4.1 we can get the following corollary.

Corollary 4.1: Considering the case wherein UAVs lying on the boundaries of a cell is equivalent to considering the whole cell in terms of DCS extraction.

By Corollary 4.1, we only need to consider the arrangements wherein cameras lying on the boundaries of cell. We can perform the following transformation that begins with an arbitrary arrangement $\langle c, \vec{d}_c \rangle$ where c lies on the boundary. First, we execute *RAC* until there is at least one object touches some side of the straight rectangle pyramid (note that an object will never fall out of the pyramid through its undersurface as we discussed before). Next, keeping c lying on the boundary and former touched objects lying on their former touching sides, execute *RAO* and *translation* such that there is at least another object touches some side of pyramid. Execute above transformation of *RAO* and *translation* under given constraints repeatedly, such that as many as possible objects touch sides of pyramid until there is no objects will touch any side, we call it final condition. Finally, the position and orientation of straight

rectangle pyramid, namely arrangement, can be either uniquely determined or not. For the former case, we can directly extract DCS of the unique arrangement. For the latter, we can select an arbitrary arrangement of final condition and extract DCS.

Because that during above transformation there is no object falling out of the pyramid, the set of covered objects of final condition dominates all sets under conditions of the process of transformation. Thus we only need to analyze the final condition which generate representative arrangement. In particular, we can enumerate all possible cases of final conditions for which there are 1 to 7 objects touching sides of the pyramid. Besides, one may be concerned about the possible performance loss as we select an arbitrary arrangement if a unique arrangement cannot be uniquely determined. We argue that there is NO performance loss and will prove it in Theorem 4.3.

In the following analysis, we use **(a, b, c)** to denote the case of final condition where a sides have three objects touching each of them, b sides have two objects touching each of them, and c sides have one object touching each of them. We only analyze typical cases and omit the discussion of similar cases in this paper to save space. For example, in three-object cases, the solution of three coplanar objects lies on any one side of four sides are the same, so we just analyze they lying on the up side as show in Figure 9 **(1, 0, 0)**. Figure 9 to 13 depict the typical cases of final conditions, whose view is from the inverse direction of $\vec{d}_{ci}$ as shown in Figure 7. The crossing dotted lines denote four edges of straight rectangle pyramid and their intersection point denotes the vertex of it.

To one-object and two-object case, we only need to choose one point c_i on the boundary of cell S_i arbitrarily and execute the algorithm for point case.

In three-object case, there exists three typical subcases.

(1) **(1, 0, 0)**. As **(1, 0, 0)** in Figure 9, o_1, o_2, and o_3 lie on the up side. Clearly, with the coordinates of three objects and expression of camera model, we have

$$\begin{cases} \vec{n}_{up} \cdot \overrightarrow{o_1o_2} = 0, \vec{n}_{up} \cdot \overrightarrow{o_2o_3} = 0, \\ \vec{d}_{ci} \cdot \vec{n}_{up} = \sin\beta, \vec{d}_{ci} \cdot \vec{n}_l = 0, \\ |\vec{n}_{up}| = 1, |\vec{d}_{ci}| = 1, |\vec{n}_l| = 1, \vec{n}_l // xOy. \end{cases} \tag{9}$$

where $\vec{n}_{up}$ is the normal vector of up side, $\vec{n}_l$ is the direction vector of the intersecting line of up side and the horizonal plane, and $\vec{d}_{ci} \cdot \vec{n}_l = 0$ describes camera can only rotate in the vertical plane we have discuss in Section III . Hence, we can obtain the orientation $\vec{d}_{ci}$ with Equation (9) and the candidate coordinates c_i can be expressed as follows:

$$\begin{cases} |\overrightarrow{c_io_1}| \le \Delta, |\overrightarrow{c_io_2}| \le \Delta, |\overrightarrow{c_io_3}| \le \Delta, \\ \alpha(\overrightarrow{o_1c_i}, \vec{d}_{oj}) \le \theta, \alpha(\overrightarrow{o_2c_i}, \vec{d}_{oj}) \le \theta, \alpha(\overrightarrow{o_3c_i}, \vec{d}_{oj}) \le \theta. \end{cases} \tag{10}$$

Then we only need to pick an arbitrary critical value of c_i that satisfies Inequality (10) to determine the arrangement.

(2) **(0, 1, 1)**. First, as **(0, 1, 1)** shown in Figure 9, we can give the following equation:

$$\begin{cases} \vec{n}_{lf} \cdot \overrightarrow{o_1o_2} = 0, \vec{n}_{lf} \cdot \vec{n}_{up} = \frac{\tan\alpha \tan\beta}{\sqrt{\sec^2\alpha}\sqrt{\sec^2\beta}}, \\ \vec{n}_{lf} \cdot \vec{n}_l = \cos\alpha, \vec{n}_{up} \cdot \vec{n}_l = 0, \\ |\vec{n}_{up}| = 1, |\vec{n}_{lf}| = 1, |\vec{n}_l| = 1, \vec{n}_l // xOy. \end{cases} \tag{11}$$

where $\vec{n}_{lf}$ is the normal vector of left side. With Equation (11), we can obtain an single-variable expression of the intersection line of left and up side. Then, combining Inequality (10) and the constraint of γ, we can determine the range of this parameter. Finally, selecting a legal parameter to determine intersection line, we can determine c_i easily with Inequality (10). Therefore, the arrangement $\langle c_i, \vec{d}_{ci} \rangle$ can be determined. Subcases **(0, 1, 2)** and **(0, 1, 3)** can be solved by the same way. Here, we omit their analysis to save space.

(3) **(0, 0, 3)**. As **(0, 0, 3)** in Figure 9, select a point c_i on the boundary of cell arbitrarily and connect c_io_1, c_io_2, and c_io_3, respectively. Then, this subcase is transformed into **(0, 3, 0)** with two objects on each side. We can obtain the equation

$$\begin{cases} \vec{n}_{up} \cdot \overrightarrow{c_io_1} = 0, \vec{n}_{rg} \cdot \overrightarrow{c_io_2} = 0, \vec{n}_{bt} \cdot \overrightarrow{c_io_3} = 0, \\ \vec{n}_{up} \cdot \vec{n}_{rg} = \frac{\tan\alpha \tan\beta}{\sqrt{\sec^2\alpha}\sqrt{\sec^2\beta}}, \vec{n}_{up} \cdot \vec{n}_{bt} = \cos 2\beta, \\ |\vec{n}_{up}| = 1, |\vec{n}_{rg}| = 1, |\vec{n}_{bt}| = 1. \end{cases} \tag{12}$$

where $\vec{n}_{rg}$ is the normal vector of right side. Thus, selecting a feasible solution arbitrarily, we can get an arrangement. Finally, execute *projection* until c_i reaching on the boundary of cell to determine the final $\langle c_i, \vec{d}_{ci} \rangle$. Moreover, subcase **(0, 0, 4)** can be solved by the same way.

In four-object case, we have two typical subcases.

(1) **(1, 0, 1)**. Similar to **(1, 0, 0)**, $\vec{d}_{ci}$ can be obtained, then normal vectors of four sides are easily to get. As **(1, 0, 1)** in Figure 10, with the normal vector of down side and o_4, we can obtain the intersection line expression of up side and down side. Then, selecting a point on this intersection line and execute *projection*, we can determine the arrangement. Subcases **(1, 1, 0)** in five-object and **(2, 0, 0)** in six-object cases can be solved by the same way.

(2) **(0, 2, 0)**. As **(0, 2, 0)** in Figure 10, combining $\vec{n}_{up} \cdot \overrightarrow{o_1o_2} = 0$ and Equation (11), we can obtain the intersection line of up side and left side. Then, selecting c_i and $\vec{d}_{ci}$ by the same way as **(1, 0, 1)**, the arrangement can be determined.

In five-object case, we have two typical cases.

(1) **(1, 0, 2)**. Similar to **(1, 0, 1)**, we can obtain the orientation $\vec{d}_{ci}$ and every normal vector of four sides. Thus, with the coordinates of o_4, o_5 and normal vectors of left and down sides as **(1, 0, 2)** in Figure 11, we can determine c_i as well as an arrangement. Then, execute *projection* until c_i reaching on the boundary of cell to determine the final $\langle c_i, \vec{d}_{ci} \rangle$.

(2) **(0, 2, 1)**. Similar to **(0, 2, 0)**, we can obtain $\vec{d}_{ci}$ and normal vectors of four sides. With coordinates of o_1, o_3, o_5 and normal vectors of left, down, and right sides, we can obtain the intersection point of these three sides, saying c_i. Then, execute *projection* until c_i reaching on the boundary of cell, we can obtain the final arrangement.

Six-object case can be classified into two kinds of subcases. The first is **(2, 0, 0)**, it can be solved by the same way as **(1, 0, 1)**. The second kind includes **(1, 1, 1)**, **(0, 3, 0)**, and **(0,**

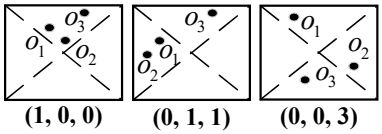


Fig. 9: Typical three-object cases

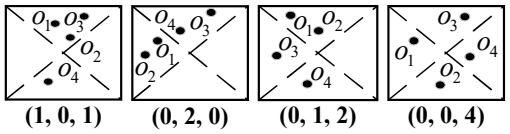


Fig. 10: Typical four-object cases

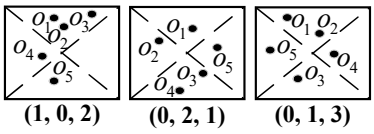


Fig. 11: Typical five-object cases

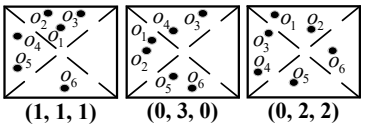


Fig. 12: Typical six-object cases

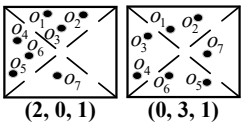


Fig. 13: Typical seven-object cases

2, 2), which has the only arrangement. Thus, we only need to solve their equations and execute *projection* to obtain the final arrangements.

In seven-object case, every subcase has the only arrangement, no matter how distributed on four sides. Thus, we only need to solve their equations and execute *projection* to obtain the final arrangements. There are two typical subcases in seven-object case. To **(2, 0, 1)**, we can obtain the following equations:

$$\begin{cases} \vec{n}_{up} \cdot \overrightarrow{o_1o_2} = 0, \vec{n}_{up} \cdot \overrightarrow{o_1o_3} = 0, |\vec{n}_{up}| = 1, \\ \vec{n}_{lf} \cdot \overrightarrow{o_4o_5} = 0, \vec{n}_{lf} \cdot \overrightarrow{o_4o_6} = 0, \vec{n}_{lf}| = 1, \\ \vec{d}_{ci} \cdot \vec{n}_{up} = \sin\beta, \vec{d}_{ci} \cdot \vec{n}_{lf} = \sin\alpha, |\vec{d}_{ci}| = 1. \end{cases} \tag{13}$$

With Equation (13), we can obtain $\vec{d}_{ci}$. Due to o_7 lying on the bottom side, utilizing the positional relation of $\overrightarrow{c_io_7}$ and $\vec{n}_{up}$, $\overrightarrow{c_io_7}$ and $\vec{n}_{lf}$, and c_i must lie on the crossing line of up and left side, we can derive c_i. Then, combine $\vec{d}_{ci}$, the unique arrangement is obtained, *i.e.*, $\langle c_i, \vec{d}_{ci}\rangle$. Similarly, to **(0, 3, 1)**, we can obtain the following equations:

$$\begin{cases} \vec{n}_{up} \cdot \overrightarrow{o_1o_2} = 0, \vec{n}_{lf} \cdot \overrightarrow{o_3o_4} = 0, \vec{n}_{bt} \cdot \overrightarrow{o_5o_6} = 0, \\ \vec{n}_{up} \cdot \vec{n}_{rg} = \frac{\tan\alpha\tan\beta}{\sqrt{\sec^2\alpha}\sqrt{\sec^2\beta}}, \vec{n}_{up} \cdot \vec{n}_{bt} = \cos 2\beta, \\ \vec{d}_{ci} \cdot \vec{n}_{up} = \sin\beta, |\vec{n}_{up}| = 1, |\vec{n}_{lf}| = 1, |\vec{d}_{ci}| = 1. \end{cases} \tag{14}$$

With Equation (14), we can obtain $\vec{d}_{ci}$. Similar to **(2, 0, 1)** case, we also can derive c_i then $\langle c_i, \vec{d}_{ci}\rangle$ as well. Due to space limit, we omit the repeated part.

Based on the above analysis for all cases, we present Algorithm 2. Let Γ be the output set of DCSs in Algorithm 2, then we have the following theorem.

***Theorem** 4.2:* Given any arrangement $\langle c_i, \vec{d}_{ci}\rangle$, there exists $\langle c_k, \vec{d}_{ck}\rangle \in \Gamma$ such that $\langle c_k, \vec{d}_{ck}\rangle \succeq \langle c_i, \vec{d}_{ci}\rangle$.

Proof: Without loss of generality, we start from searching arrangements for three-object cases. Assuming objects o_1, o_2 and o_3 touching one side of straight rectangle pyramid, we select an arbitrary feasible arrangement $\langle c_1, \vec{d}_{c1}\rangle$. Then, keeping the three objects on this side and execute transformations, there exists numerous conditions which can be classified into three classes.

Class 1. *There is the only one arrangement for objects o_1, o_2 and o_3.* Obviously, the selected feasible arrangement $\langle c_1, \vec{d}_{c1}\rangle$ is unique, and it must generate the only DCS such that $\langle c_1, \vec{d}_{c1}\rangle \in \Gamma$.

Class 2. *There won't be any new object touch any side of pyramid.* This condition implies $\langle c_1, \vec{d}_{c1}\rangle = \langle c_k, \vec{d}_k\rangle (k \neq 1, k \in U)$, where U is the universe of all arrangements. Thus arrangement $\langle c_1, \vec{d}_{c1}\rangle$ generates DCS such that $\langle c_1, \vec{d}_{c1}\rangle \in \Gamma$.

Class 3. *Some new object(s) touch some side(s) of pyramid.* Assuming object o_4 touches one side and we arbitrarily select an arrangement $\langle c_2, \vec{d}_{c2}\rangle$ covering these four objects. Then, there exists two subclasses.

Subclass 3.1. $\langle c_2, \vec{d}_{c2}\rangle = \langle c_1, \vec{d}_{c1}\rangle$. This indicates that object o_4 has been covered by $\langle c_1, \vec{d}_{c1}\rangle$, then $\langle c_1, \vec{d}_{c1}\rangle$ generates the DCS such that $\langle c_1, \vec{d}_{c1}\rangle \in \Gamma$.

Subclass 3.2. $\langle c_2, \vec{d}_{c2}\rangle \succeq \langle c_1, \vec{d}_{c1}\rangle$. This indicates that object o_4 isn't covered by $\langle c_1, \vec{d}_{c1}\rangle$, thus $\langle c_1, \vec{d}_{c1}\rangle$ is not a DCS. However, as Algorithm 2, when searching arrangement $\langle c_2, \vec{d}_{c2}\rangle$ for objects o_1, o_2, o_3 and o_4 in the next round of all combination of four objects on the sides, arrangement $\langle c_1, \vec{d}_{c1}\rangle$ for objects o_1, o_2 and o_3 will be replaced by arrangement $\langle c_2, \vec{d}_{c2}\rangle$. If no object will touch any side of pyramid during continuous transformation, arrangement $\langle c_2, \vec{d}_{c2}\rangle$ generates the DCS such that $\langle c_1, \vec{d}_{c1}\rangle \preceq \langle c_2, \vec{d}_{c2}\rangle \in \Gamma$. Otherwise, similar to the above, arrangement $\langle c_2, \vec{d}_{c2}\rangle$ will be replaced by $\langle c_3, \vec{d}_{c3}\rangle, \ldots, \langle c_k, \vec{d}_{ck}\rangle$ iteratively until no object will touch any side or there is the only determined arrangement. arrangement $\langle c_k, \vec{d}_{ck}\rangle$ generates the DCS for $o_1, \ldots, o_k$, as well as, for o_1, o_2 and o_3. Consequently, $\langle c_1, \vec{d}_{c1}\rangle \preceq \langle c_k, \vec{d}_{ck}\rangle \in \Gamma$. ■

D. Problem Reformulation and Solution

In this subsection, we discuss about how to select a given number of arrangements from the obtained ones to maximize the number of coverage objects. We first reformulate the problem, then prove its submodularity, and finally present an effective algorithm to address this problem.

Let x_i be a binary indicator denoting whether the i_{th} arrangement in the arrangement set of DCSs Γ is select or not. For all DCSs from all cells in Γ, we can compute the coverage function with each object. The problem **P1** can be reformulated as

$$\begin{aligned} \textbf{(P2)} \quad max \quad & \sum_{j=1}^{M} \mathcal{U}_v\Big(\sum_{\langle c_i, \vec{d}_{ci}\rangle \in \Gamma} x_i F_v(c_i, o_j, \vec{d}_{ci}, \vec{d}_{oj})\Big), \\ s.t. \quad & \sum_{i=1}^{|\Gamma|} x_i = N(x_i \in \{0,1\}). \end{aligned} \tag{15}$$

The problem is then transformed to a combinatorial optimization problem. Now, we give the following definitions to assist further analysis before addressing **P2**.

Algorithm 2: DCS Extraction for the Area Case

Input: The cell S_i, the candidate covered set of objects $\hat{O}_i$
Output: All DCSs

for $i \leq 7$ *(number of objects ≤ the maximum of minimum number of objects on 4 sides to determine the only one arrangement)* **do**
  for *every combination* $o_{k_1}, ..., o_{k_i}$ *of all objects in* $\hat{O}_i$ **do**
    if $i \geq 3$ **then**
      for *every subcases* $a+b+c$ *subject to* $3a+2b+c=i$ *and* $a+b+c \leq 4$ **do**
        for *every possible arrangement of 4 sides taken* $a+b+c$ *to arrange* $3,2,1$ *objects respectively* **do**
          Execute the process following the corresponding subcase $a+b+c$.
          if *exists corresponding arrangement* $\langle c_i, \vec{d}_{ci} \rangle$ **then**
            Add the results to the candidate DCS set. ***break***
    Select one point p on the boundary of S_i arbitrarily.
    if i = *1* **then**
      Build arrangement $\langle p, \vec{d}_p \rangle$ with object o_k on the surface of straight rectangle pyramid.
    if i = *2* **then**
      Decide if these objects can be in one camera coverage. If it is, build straight rectangle pyramid with line $\overline{po}_{k_1}$ and $\overline{po}_{k_2}$ on the surface.

Algorithm 3: Arrangements Selection

Input: The number of UAVs N, DCSs set Γ, objective function $f(X)$
Output: arrangement set X

$X = \emptyset$.
while $|X| \leq N$ **do**
  $e^* = \arg\max_{e \in \Gamma \setminus X} f(X \cup \{e\}) - f(X)$.
  $X = X \cup \{e^*\}$.

***Definition** 4.8:* [52] Let S be a finite ground set. A real-valued set function $f: 2^S \to \mathbb{R}$ is normalized, monotonic, and submodular if and only if it satisfies the following conditions, respectively: (1) $f(\emptyset) = 0$; (2) $f(A \cup \{e\}) - f(A) \geq 0$ for any $A \subseteq S$ and $e \in S \backslash A$; (3) $f(A \cup \{e\}) - f(A) \geq f(B \cup \{e\}) - f(B)$ for any $A \subseteq B \subseteq S$ and $e \in S \backslash B$.

***Definition** 4.9:* [52] A Matroid $\mathcal{M}$ is an arrangement $\mathcal{M} = (S, L)$ where S is a finite ground set, $L \subseteq 2^S$ is a collection of independent sets, such that (1) $\emptyset \in L$; (2) if $X \subseteq Y \in L$, then $X \in L$; (3) if $X, Y \in L$, and $|X| < |Y|$, then $\exists y \in Y \backslash X$, $X \cup \{y\} \in L$.

***Definition** 4.10:* [52] Given a finite set S and an integer k. A uniform matroid $\mathcal{M} = (S, L)$ is a matroid where $L = \{X \subseteq S : |X| \leq k\}$.

Then, our problem can be reformulated as

$$\begin{aligned} \textbf{(P3)} \quad & max \ f(X) = \sum_{j=1}^{M} \mathcal{U}_v\Big(\sum_{\langle c_i, \vec{d}_{ci} \rangle \in X} F_v(c_i, o_j, \vec{d}_{ci}, \vec{d}_{oj}) \Big), \\ & s.t. \quad X \in L, \\ & \quad L = \{X \subseteq \Gamma : |X| \leq \mathcal{M}\}. \end{aligned} \tag{16}$$

***Lemma** 4.2:* The objective function $f(X)$ in **P3** is a monotone submodular function, whose constraint is a uniform matroid.

Proof: According to Definition 4.8, we need to verify the three listed requirements of $f(X)$ in order to prove that it is monotone submodular.

First, when the number of deployed UAVs is 0, obviously $\mathcal{U}_v(\cdot) = 0$, thus we have $f(\emptyset) = 0$.

Second, let A be a set of arrangements in Γ, $e \in \Gamma \setminus A$ and $\varphi(X, i) = \mathcal{U}_v(\sum_{\langle c_i, \vec{d}_{ci} \rangle \in X} F_v(c_i, o_j, \vec{d}_{ci}, \vec{d}_{oj}))$. We first observe that the directional coverage utility $\mathcal{U}(\cdot)$ is non-decreasing. Moreover, it is clear that $\sum_{\langle c_i, \vec{d}_{ci} \rangle \in A \cup \{e\}} F_v(c_i, o_j, \vec{d}_{ci}, \vec{d}_{oj}) \geq \sum_{\langle c_i, \vec{d}_{ci} \rangle \in A} F_v(c_i, o_j, \vec{d}_{ci}, \vec{d}_{oj})$. Then, we have $\varphi(A \cup \{e\}, i) - \varphi(A, i) \geq 0$. Therefore,

$$f(A \cup \{e\}) - f(A) = \sum_{j=1}^{M} (\varphi(A \cup \{e\}, i) - \varphi(A, i)) \geq 0.$$

Third, let A and B be two sets where $A \subseteq B \subseteq \Gamma$ and element $e \in \Gamma \setminus B$. Since $\mathcal{U}_v(\cdot)$ is a binary function, we can analyze using exhaustive approach. If $\mathcal{U}_v(A, \cdot) = 0$ and $\mathcal{U}_v(A \cup \{e\}, \cdot) = 1$, then there must exist $\mathcal{U}_v(B \cup \{e\}, \cdot) = 1$. Moreover, regardless of the value of $\mathcal{U}_v(B, \cdot)$, we always have $(\mathcal{U}_v(A \cup \{e\}, \cdot) - \mathcal{U}_v(A, \cdot)) - (\mathcal{U}_v(B \cup \{e\}, \cdot) - \mathcal{U}_v(B, \cdot)) \geq 0$. For other cases, *i.e.*, $\mathcal{U}_v(A, \cdot) = \mathcal{U}_v(A \cup \{e\}, \cdot)$, there must exist $(\mathcal{U}_v(A \cup \{e\}, \cdot) - \mathcal{U}_v(A, \cdot)) = (\mathcal{U}_v(B \cup \{e\}, \cdot) - \mathcal{U}_v(B, \cdot))$. Thus, $(\varphi(A \cup \{e\}, i) - \varphi(A, i)) - (\varphi(B \cup \{e\}, i) - \varphi(B, i)) \geq 0$. Therefore,

$$\begin{aligned} & (f(A \cup \{e\}) - f(A)) - (f(B \cup \{e\}) - f(B)) \\ =& \sum_{j=1}^{M} [(\varphi(A \cup \{e\}, i) \\ & - \varphi(A, i)) - (\varphi(B \cup \{e\}, i) - \varphi(B, i))] \\ \geq& 0. \end{aligned}$$

To sum up, $f(X)$ is a monotone submodular function. ■

Therefore, the reformulated problem falls into the scope of maximizing a monotone submodular function subject to matroid constraints, and we can use a greedy algorithm to achieve a good approximation [52]. The pseudo code of this arrangement selecting algorithm is shown in Algorithm 3. In every round, Algorithm 3 greedily adds an arrangement e^* to X to maximize the increment of function $f(X)$. We omit the proof to save space.

E. Theoretical Analysis

***Theorem** 4.3:* Algorithm PANDA achieves an approximation ratio of $1-1/e$, and its time complexity is $O(NM^9)$.

Proof: First, we bound the approximation ratio of PANDA algorithm. Denote the overall directional coverage utility for all N UAVs under optimal solution to problem **P1** and the reformulated problem **P2** as $\mathbf{OPT}_{p2}$ and $\mathbf{OPT}_{p1}$, respectively. According to Corollary 4.1 and Theorem 4.2, Algorithm 2 extracts all DCSs without loss. Thus, $\mathbf{OPT}_{p1} = \mathbf{OPT}_{p2}$. Denote the overall directional coverage utility by Algorithm 3 to the problem **P2** (or **P3**) as **SOL**. According to the fact that a greedy algorithm of maximizing a monotone submodular function subject to a uniform matroid achieves $1-1/e$ approximation ratio [52], thus the approximation ratio of PANDA is $1-1/e$, formally,

$$\frac{SOL}{OPT_{p1}} = \frac{SOL}{OPT_{p2}} = 1 - \frac{1}{e}, \tag{17}$$

where e is the Napier's constant.

Next, we analyze the time complexity of PANDA. First, PANDA computes the total number of cells intersected by the cone of each object in 3D space, whose complexity is $O(M^2)$ according to Theorem 4.1. Second, in each cell, Algorithm 2 will extract DCSs for each subcase. The total number of case is $\sum_{i=1}^{7}\binom{M}{i} = O(M^7)$ and each case has $O(1)$ subcases which generate the corresponding arrangements. Thus, the total time complexity of Algorithm 2 is $O(M^7)$. Last, Algorithm 3 will perform N times loop and in each time it will select the best one from the current remaining arrangements. Thereby, the time complexity of PANDA algorithm is $O(NM^9)$. ■

V. Discussion

A. Deploying Minimum UAVs to Achieve a Required Coverage Utility

Consider a problem which is slightly different from PANDA, that is, placing minimum UAV to achieve a required coverage utility. The solution is almost the same as PANDA, except Algorithm 3. Instead, we greedily select arrangements for UAV one by one until the required coverage utility is achieved, then output selected arrangements. According to the classical results in [53], the adapted algorithm based on Algorithm 3 achieves $\frac{1}{\ln n}$ approximation ratio, where n is the number of candidate arrangements. With similar analysis, we can prove that the overall solution for this variant problem can also achieve $\frac{1}{\ln n}$ approximation ratio.

B. Heterogeneous UAVs

In this section, we consider the case where UAVs are heterogeneous. In particular, heterogeneous camera models are consisted of different parameters, *i.e.*, α, β, and Δ. Formally, the PANDA problem of heterogeneous camera version is, given a set of UAVs containing Φ types of cameras, the ϕ-th ($1 \le \phi \le \Phi$) type of camera model with parameter α^ϕ, β^ϕ, and Δ^ϕ, and the number of UAVs of this type is N_ϕ. The objective function is the same as the one of PANDA, which is placing these heterogeneous UAVs in 3D space to maximize the overall directional coverage utility for all M objects.

Algorithm 4: Arrangements Selection for Heterogeneous UAVs

Input: The number of UAVs N_ϕ for the ϕ-th type, DCSs set Γ_ϕ, objective function $f(X)$
Output: arrangement set $X_\phi (1 \le \phi \le \Phi)$

```
X_φ = ∅(1 ≤ φ ≤ Φ).
for all φ ∈ [Φ] do
    while |X_φ| ≤ Nφ do
        X = ⋃_{φ=1}^{Φ} X_φ.
        e* = arg max_{e∈Γ_φ \ X_φ} f(X ∪ {e}) − f(X).
        X_φ = X_φ ∪ {e*}.
```

For this heterogeneous UAVs version problem, we can adapt the same solution of PANDA to address it. First, for each type of camera, we divide the whole 3D space into multiple cells with each Δ^ϕ and the homogenous efficient angle θ. Then, we extract DCSs of each type of camera model, *i.e.*, α^ϕ and β^ϕ, and obtain the set of DCSs, *i.e.*, Γ_ϕ, of each type of camera. Finally, we obtain Φ different sets of DCSs because of the heterogeneous parameters of camera models.

To PANDA problem, we reformulate it into a problem of maximizing a monotone submodular function subject to a uniform matroid constraint. To this heterogeneous UAVs version problem, we can reformulate it into a maximizing monotone submodular function subject to a different constraint. First we give the following definition.

***Definition** 5.1:* [52] Given $\mathcal{S} = \bigcup_{i=1}^{k} \mathcal{S}'_i$ is the disjoint union of k sets, $l_1, l_2, \cdots, l_k$ are positive integers, a Partition Matroid $\mathcal{M} = (\mathcal{S}, \mathcal{I})$ is a matroid where $\mathcal{I} = \{X \subset \mathcal{S} : |X \cap \mathcal{S}'_i| \le l_i \textit{ for } i \in [k]\}$.

We first define a universal set Γ which is the disjoint union of obtained Φ DCS sets, *i.e.*, $\Gamma = \bigcup_{\phi=1}^{\Phi} \Gamma_\phi$. Then, by Definition 5.1, the partition matroid can be established as $\mathcal{N} = (\Gamma, \mathcal{I})$, where $\mathcal{I} = \{X \subset \Gamma : |X \cap \Gamma_\phi| \le N_\phi \textit{ for } i \in [\Phi]\}$. Then, using the similar proof of Lemma 4.2, we can derive the following lemma.

***Lemma** 5.1:* The objective function of placing heterogeneous UAVs is a monotone submodular function, whose constraint is a partition matroid.

By Lemma 5.1, we can design a greedy algorithm to maximize the monotone submodular function with performance bound [52]. Algorithm 4 shows our greedy algorithm. Essentially, Algorithm 4 traverses all types of UAVs and greedily selects the best arrangement with maximum directional coverage utility for each type . By similar analysis to Theorem 4.3 and the classical theoretical results in Chapter 2 in [52], we can obtain the following theorem.

***Theorem** 5.1:* The heterogeneous UAVs version PANDA algorithm achieves an approximation ratio of $1/2$, and its time complexity is $O(tM^9)$ where $t = \max\{N, \Phi\}$.

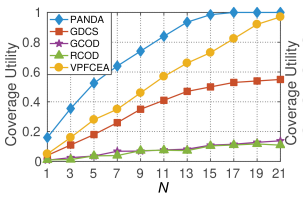


Fig. 14: N vs. utility

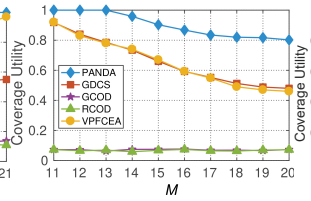


Fig. 15: M vs. utility

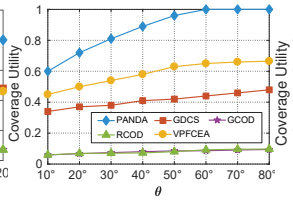


Fig. 16: θ vs. utility

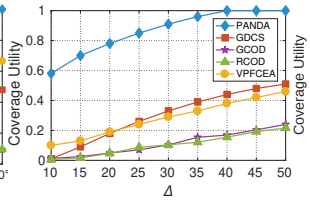


Fig. 17: Δ vs. utility

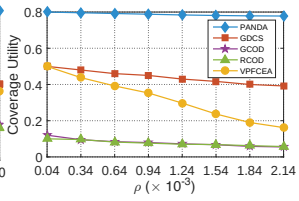


Fig. 18: ρ vs. utility

C. Heterogeneous Objects

In this section, we consider the case where objects are heterogeneous. This problem is much more easier than heterogeneous UAVs version PANDA, because the only heterogeneous parameter of objects is θ. The other parameter Δ in objects model is determined by the parameter of UAVs.

To address this heterogeneous objects version PANDA problem, we can also use the same method of PANDA. The only difference is the space discretization step. Formally, given a set of objects containing Ψ types, the ψ-th ($1 \leq \psi \leq \Psi$) type of object model with parameter θ^{ψ} and Δ, and the number of objects of this type is M_{ψ}. Then, we divide the whole 3D space with all objects but corresponding heterogeneous θ not the homogeneous one in PANDA. The following step is the same as the method of PANDA. Thus, the time complexity and the approximation ratio are both the same as PANDA, which is $O(NM^9)$ and $1 - 1/e$.

D. Non-ignorable Size of Objects

In this section, we consider the case where the size of objects can't be ignored. To this case, we can also use our method addressed PANDA problem but need some preprocessing. This preprocessing includes two steps. First, we discretizes the surface of each objects into λ_j subareas. Second, to each subarea, we randomly select a point on it and compute the norm vector at this point of this subarea. After this preprocessing, we obtain $\sum_{i=1}^{M} \lambda_j$ subareas. As long as the subarea is sufficient small, then we can treat them as point as how we do in our paper. Therefore, we can use the method of PANDA to address the following issues.

Now, we briefly analyze our method to address Non-ignorable Size of Objects version PANDA problem. The denser discretization we cut the surface of one object, the more exact results we can obtain but also the more computation cost we should pay. Fortunately, we can bound both the time complexity and the approximation ratio with the parameter λ_j and Theorem 4.3. Due to the limit space, we omit the detailed analysis.

In Non-ignorable Size of Objects version PANDA, an important issue need to be considered that is the obstruction of one object by another. There are some papers considering this problem and getting some results both in theory and practice. [54] is a representative paper study the obstruction in covering problem of UAV.

E. Experiment in Actual Scenarios

In this section, we consider the case where the 3D directional coverage in actual scenarios. In actual scenarios, there will be many problems occurred, such as obstacles, electronic interference, and sky above the sidewalk, etc. However, PANDA is a basic problem and our algorithm of PANDA can also address these issues occurred in actual scenarios. In briefly, these actual problems introduce some constraints to PANDA that the obstacles, the regions of electronic interference, and the sky above the sidewalks can't place UAVs or obstruct the field of view. Mathematically, they decrease the solution space of PANDA, but union with these constraints the algorithm of PANDA can also work because we just need to search the remaining solution space.

VI. Simulation Results

A. Evaluation Setup

In our simulation, objects are uniformly distributed in a $100\,m \times 100\,m \times 50\,m$ cuboid space. If no otherwise stated, we set $\alpha = \pi/3$, $\beta = \pi/12$, $\Delta = 25\,m$, $N = 10$, $\gamma_{min} = \pi/6$, $\gamma_{max} = \pi/3$, $\theta = \pi/6$, and $M = 20$, respectively. Note that both of the orientations of cameras and objects are considered with respect to the North. The orientations of objects are randomly selected from $[0, 2\pi]$ in horizonal plane and $[0°, 90°]$ in vertical plane. Each data point in evaluation figures is computed by averaging the results of 200 random topologies. As there are no existing approaches for PANDA, we compare four algorithms including three presented algorithms and an existing algorithm VPFCEA proposed by [38]. VPFCEA algorithm solves the optimal coverage problem on 2D plane. Randomized Coordinate with Orientation Discretization (RCOD) randomly generates coordinates of UAVs, and randomly selects orientation of UAVs from $\{0, \alpha, ..., k\alpha, ..., 2\pi\}$ in horizonal plane and $\{\gamma_{min}, \gamma_{min}+\beta, ..., \gamma_{min}+\lfloor(\gamma_{max}-\gamma_{min})/\beta\rfloor\beta, ..., \gamma_{max}\}$ in vertical plane. Grid Coordinate with Orientation Discretization (GCOD) improves RCOD by placing the UAVs at grid points. Grid Coordinate with Dominating Coverage Set (GDCS) further improves GCOD. It utilize DCS extraction algorithm for point case to generate candidate orientations and greedily selects the orientation with best coverage utility. z

B. Performance Comparison

1) Impact of Number of UAVs N: Our simulation results show that on average, PANDA outperforms RCOD, GCOD, GDCS, and VPFCEA by 12.35 *times,* 10.27 *times,* 3.51 *times, and* 87.56% *respectively, in terms of* N. Figure 14 shows that the coverage utility for all algorithms increase monotonically with N. In particular, the coverage utility of PANDA first fast increases and approaches 1 when $N = 15$, and then becomes stable. GDCS increases relatively linearly because it can only choose among given grid coordinates for placing

UAVs. VPFCEA performs better than GDCS because it can place UAVS at any location, but much worse than PANDA because it only considers the object located on 2D plane. In contrast, the coverage utility of RCOD and GCOD always remain low, because their candidate coordinates of UAVs are limited and orientations are predetermined or randomly generated.

2) Impact of Number of Objects M: Our simulation results show that on average, PANDA outperforms RCOD, GCOD, GDCS, and VPFCEA by 12.37 *times,* 11.74 *times,* 41.5%*, and* 42.3%*, respectively, in terms of* M. From Figure 15, the coverage utility decreases monotonically with increasing M. PANDA first performs well for no more than 13 objects but then decreases when M is larger than 13. The decreasing rate tends to be gentle and around 0.8. In contrast, GDCS and VPFCEA invariably degrades while RCOD and GCOD always keep low performance.

3) Impact of Efficient Angle θ: Our simulation results show that on average, PANDA outperforms RCOD, GCOD, GDCS, and VPFCEA by 10.01 *times,* 9.94 *times,* 110.36%*, and* 86.36%*, respectively, in terms of* θ. As shown in Figure 16, the coverage utility of four algorithms increases monotonically with θ. The coverage utility of PANDA first increases at a fast speed and approaches 1 when θ increases from 10° to 60°, and then keeps stable. However, the other four comparison algorithms increase slowly.

4) Impact of Farthest Sight Distance Δ: Our simulation results show that on average, PANDA outperforms RCOD, GCOD, GDCS, and VPFCEA by 11.20 *times,* 13.53 *times,* 84.35%*, and* 82.35%*, respectively, in terms of* Δ. Figure 17 shows that the coverage utility of PANDA invariably increases with Δ until it approaches 1, while that of RCOD, GCOD, GDCS, and VPFCEA increase to about 0.2, 0.2, 0.55, and 0.55 respectively, and then keeps relatively stable.

5) Impact of Farthest Sight Distance ρ: Our simulation results show that on average, PANDA outperforms RCOD, GCOD, GDCS, and VPFCEA by 6.18 *times,* 6.64 *times,* 81.35%*, and* 97.65%*, respectively, in terms of ρ (= number of objects ÷ volume of whole space).* Figure 18 shows that the coverage utility of PANDA fluctuates slightly when ρ grows, but it is almost always near 0.8. The coverage utility of GDCS also fluctuates slightly, because GDCS uses our DCS Extraction algorithm at each grid. VPFCEA decreases a lot, because it doesn't consider the objects distributed in 3D space. When ρ increases, it covers objects much more difficultly. RCOD and GCOD always maintain bad coverage utility, because along with rho increasing they can cover objects more easier but the number of objects also increase. As coverage utility is the covered number of objects divides the sum number of objects, randomized placement like RCOD and GCOD can't get high coverage utility.

6) Impact of Horizonal Angle α and Vertical Angle β: Here we study the impact of α and β on coverage utility. Suppose the horizonal offset angle α and vertical offset angle β vary from 10° to 80°, respectively. Figure 19 depicts the results and each point on the surface denote an average value of 100 experiment results. We observe that the coverage utility increases monotonically while either α or β increases. Indeed, with a larger α or β, the 3D coverage space gets larger and more potential objects can be covered. In this subsection, we evaluate the performance of the proposed algorithm in Section V-B under the condition of heterogeneous UAVs.

C. Performance Evaluation for Heterogeneous UAVs

In this simulation, there are three types of UAVs with different coverage angles, they are $\alpha = \pi/3$, $\beta = \pi/12$; $\alpha = \pi/4$, $\beta = \pi/10$; and $\alpha = \pi/6$, $\beta = \pi/8$, and the number of each type of UAVs is 4. The other simulation settings are the same as that in Section VI-A. Due to that VPFCEA algorithm doesn't consider the heterogeneity of UAVs, we just compare PANDA with RCOD, GCOD, and GDCS. To make ROCD, GCOD, and GDCS suitable for heterogeneous UAVs, we first execute them for all three types of UAVs, and then greedily choose the combination of arrangements with best coverage utility.

1) Impact of Farthest Sight Distance Δ: Our simulation results show that on average, PANDA outperforms RCOD, GCOD, and GDCS by 21.72 *times,* 13.36 *times, and* 80.75%*, respectively, in terms of* Δ. Figure 20 shows that the coverage utility increases with increasing Δ. The coverage utility of PANDA increases invariably until it approaches 1 when $\Delta = 43$. The coverage utility of RCOD, GCOD, and GDCS also increase, but their initial coverage utilities are very low because the locations their arrangements are limited.

2) Impact of Number of UAVs N: Our simulation results show that on average, PANDA outperforms RCOD, GCOD, and GDCS by 11.97 *times,* 10.41 *times, and* 3.47 *times, respectively, in terms of* N. In this simulation, three types of UAVs are added orderly one by one until 7 UAVs of each type are selected. As shown in Figure 21, the coverage utility for all algorithms increase monotonically with N. The coverage utility of PANDA increases dramatically until approaching one, then it becomes stable. The other three compared algorithms perform relatively worse because they only place UAVs at grid coordinates, which make them suboptimal.

3) Impact of Number of Objects M: Our simulation results show that on average, PANDA outperforms RCOD, GCOD, and GDCS by 12.33 *times,* 11.69 *times, and* 43%*, respectively, in terms of* M. As illustrated in Figure 22, the coverage utility monotonically decreases with M. The coverage utility of PANDA first performs well when the number of objects is no more than 13. In contrast, GDCS invariably degrades while RCOD and GCOD always keep low performance.

VII. Field Experiments

As shown in Figure 23, our testbed consists of 7 DJI Phantom 4 advanced UAVs and 15 randomly distributed face figures as objects, and our experimental site is the playground of our school including its stands, whose size is $110\,m \times 80\,m$. Specifically, we set $\alpha = 35^\circ$, $\beta = 20^\circ$, $\Delta = 10\,m$, $\gamma_{min} =$

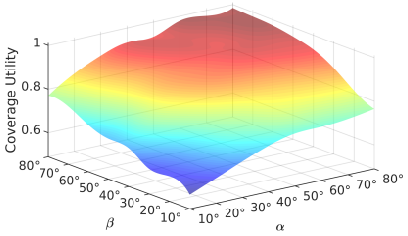


Fig. 19: α and β vs. utility

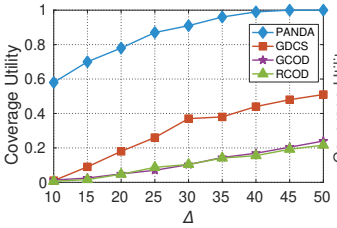


Fig. 20: Δ vs. utility

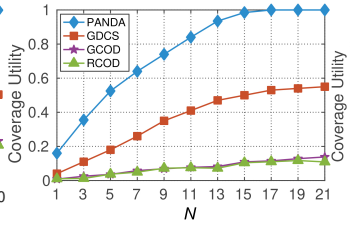


Fig. 21: N vs. utility

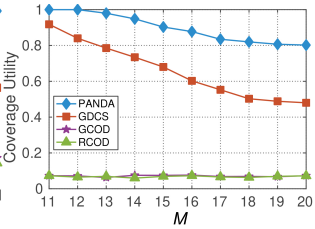


Fig. 22: M vs. utility

(a) UAV

(b) Object

(c) Experiment site

Fig. 23: Testbed

TABLE II: Coordinate and orientation of objects

Object	Coordinate	Orientation	Object	Coordinate	Orientation
o_1	(19.4,0.7.9,5.8)	($7\pi/4$,$\pi/2$)	o_9	(83.0,2.7.9,5.0)	(0,$\pi/2$)
o_2	(21.0,5.9,3.3)	($4\pi/5$,$2\pi/6$)	o_{10}	(84.3,3.2,4.6)	($\pi/4$,$\pi/3$)
o_3	(2.1,0.9,5.8)	(0,$\pi/6$)	o_{11}	(81.5,19.3,1.7)	($3\pi/4$,$\pi/2$)
o_4	(9.6,1.2,5.4)	($3\pi/4$,0)	o_{12}	(16.2,66.9,1.7)	($\pi/2$,0)
o_5	(11.4,3.9,4.2)	($3\pi/4$,0)	o_{13}	(9.94,53.4,0.5)	($\pi/2$,$\pi/2$)
o_6	(18.7,2.9,4.6)	($\pi/4$,$\pi/6$)	o_{14}	(83.2,28.9,0.5)	(π,$\pi/3$)
o_7	(3.0,6.1,3.3)	($3\pi/2$,$\pi/2$)	o_{15}	(36.6,63.8,0.5)	($3\pi/4$,0)
o_8	(84.4,3.0.9,4.6)	(π,$\pi/6$)			

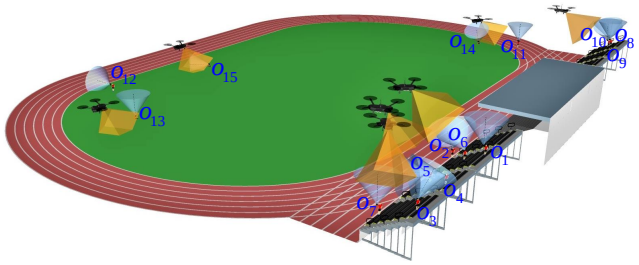


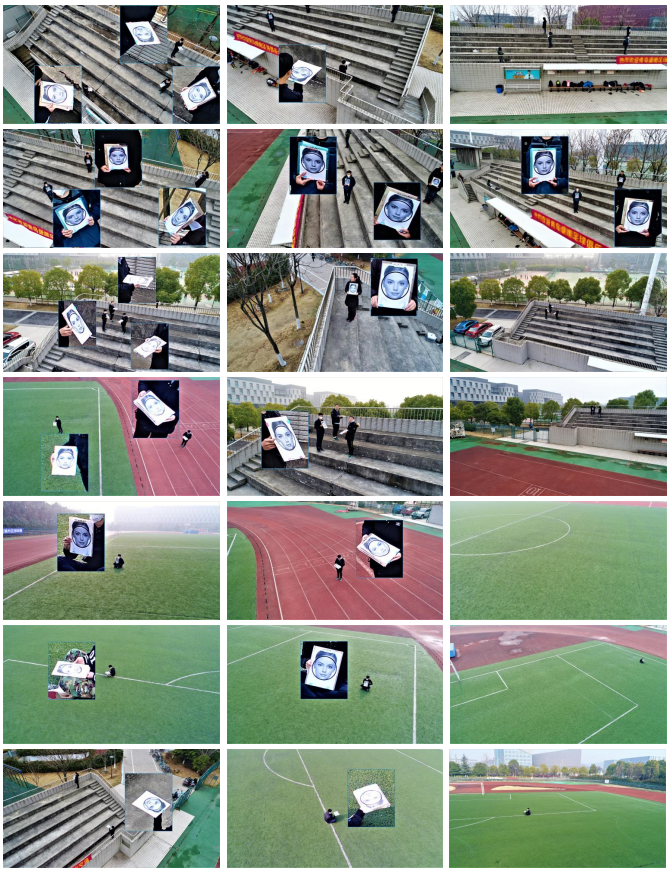
Fig. 24: Objects distribution and UAV placement by PANDA

$10°$, $\gamma_{max} = 70°$, and $\theta = \pi/6$ based on real hardware parameters. The orientations of objects (θ, φ) are randomly generated where $\theta \in \{k\pi/4, k \in \{1,2,3,4,5,6,7,8\}\}$ is the angle between orientation and xOz and $\varphi \in \{k\pi/6, k \in \{0,1,2,3\}\}$ is the angle between orientation and xOy. Table II lists the obtained coordinates and orientations of all objects. Moreover, we draw a circle around the face on each face figure as shown in Figure 23(b) to help demonstrate the coverage result by observing the circle's distortion degree.

We respectively execute PANDA, GDCS, and GCOD offline and obtain their corresponding strategies. Figure 24 illustrates the placement results for PANDA. Note that the spherical base cones of objects are depicted in grey while the straight rectangle pyramids of UAVs are in yellow. Figure 25 shows the 7 pictures took by 7 UAVs for each of the three algorithms PANDA, GDCS, and GCOD. The rectangular enlarged views in each figure demonstrate the details of successfully efficient covered objects for the corresponding UAV. From Figure 25, we can see that PANDA covers the most objects among all the three algorithms. The coverage utilities of PANDA, GDCS, and GCOD are 0.93, 0.53, and 0.20, respectively, which means PANDA outperforms GDCS and GCOD by 75.4% and 3.65 times.

VIII. Conclusion

In this paper, we solve the problem of 3D placement of UAVs to achieve directional coverage. The key novelty of this paper is on proposing the first algorithm for UAV placement with optimized directional coverage utility in 3D environment. The key contribution of this paper is building the practical 3D directional coverage model, developing an approximation algorithm, and conducting simulation and field experiments for evaluation. The key technical depth of this paper is in reducing the infinite solution space of this optimization problem to a limited one by utilizing the techniques of space partition and Dominating Coverage Set extraction, and modeling the reformulated problem as maximizing a monotone submodular function subject to a matroid constraint. Our evaluation results show that our algorithm outperforms the other comparison algorithms by at least 75.4%.



(a) PANDA (b) GDCS (c) GCOD

Fig. 25: Experimental results of different algorithms

REFERENCES

[1] S. He *et al.*, "Full-view area coverage in camera sensor networks: Dimension reduction and near-optimal solutions," *IEEE TVT*, 2016.

[2] X. Gao *et al.*, "Optimization of full-view barrier coverage with rotatable camera sensors," in *IEEE ICDCS*, 2017.

[3] S. Hayat *et al.*, "Survey on unmanned aerial vehicle networks for civil applications: A communications viewpoint," *IEEE Communications Surveys & Tutorials*, 2016.

[4] L. Gupta *et al.*, "Survey of important issues in UAV communication networks," *IEEE Communications Surveys & Tutorials*, 2016.

[5] Y. Zeng *et al.*, "Wireless communications with unmanned aerial vehicles opportunities and challenges.pdf," *IEEE Communication Magzine*, 2016.

[6] H. Menouar *et al.*, "UAV-enabled intelligent transportation systems for the smart city: Applications and challenges," *IEEE Communications Magazine*, 2017.

[7] "https://www.dji.com/phantom-4-adv/info."

[8] V. Blanz *et al.*, "Face recognition based on frontal views generated from non-frontal images," in *IEEE CVPR*, 2005.

[9] H. Dorit *et al.*, "Approximation Schemes for Covering and Packing Problems in Image Processing and VLSI," *Journal of ACM*, 1985.

[10] T. Shi *et al.*, "The Energy-Data Dual Coverage in Battery-free Sensor Networks," in *IEEE ICDCS*, 2019.

[11] J. Gao *et al.*, "Composite event coverage in wireless sensor networks with heterogeneous sensors," in *IEEE INFOCOM*, 2015.

[12] A. Efrat *et al.*, "Improved Approximation Algorithms for Relay Placement," *ACM Transactions on Algorithms*, 2015.

[13] K. Alexander *et al.*, "Deterministic Boundary Recognition and Topology Extraction for Large Sensor Networks," in *ACM SODA*, 2006.

[14] K. Andreas *et al.*, "Near-Optimal Sensor Placements in Gaussian Processes: Theory, Efficient Algorithms and Empirical Studies," *Journal of Machine Learning*, 2008.

[15] G. Carlos *et al.*, "Near-optimal Sensor Placements in Gaussian Processes," in *ACM ICML*, 2005.

[16] M. Mozaffari *et al.*, "Efficient deployment of multiple unmanned aerial vehicles for optimal wireless coverage," *IEEE Communication Letters*, 2016.

[17] C. Liu *et al.*, "Energy-efficient uav control for effective and fair communication coverage: A deep reinforcement learning approach," *IEEE JSAC*, 2018.

[18] J. Cui *et al.*, "Power-efficient deployment of a uav for emergency indoor wireless coverage," *IEEE Access*, 2018.

[19] Y. C. Wang *et al.*, "Using Rotatable and Directional (R & D) Sensors to Achieve Temporal Coverage of Objects and Its Surveillance Application," *IEEE TMC*, 2012.

[20] G. Fusco *et al.*, "Selection and orientation of directional sensors for coverage maximization," in *IEEE SECON*, 2009.

[21] Z. Wang and pthers, "Achieving k-barrier coverage in hybrid directional sensor networks," *IEEE TMC*, 2014.

[22] T. Y. Lin *et al.*, "Enhanced deployment algorithms for heterogeneous directional mobile sensors in a bounded monitoring area," *IEEE TMC*, 2017.

[23] Y. Wang *et al.*, "On full-view coverage in camera sensor networks," in *IEEE INFOCOM*, 2011.

[24] ——, "Barrier coverage in camera sensor networks," in *ACM MobiHoc*, 2011.

[25] Y. Hu *et al.*, "Critical sensing range for mobile heterogeneous camera sensor networks," in *IEEE INFOCOM*, 2014.

[26] H. Ma *et al.*, "Minimum camera barrier coverage in wireless camera sensor networks," in *IEEE INFOCOM*, 2012.

[27] Z. Yu *et al.*, "Local face-view barrier coverage in camera sensor networks," in *IEEE INFOCOM*, 2015.

[28] Q. Zhang *et al.*, "Toward optimal orientation scheduling for full-view coverage in camera sensor networks," in *IEEE GLOBECOM*, 2016.

[29] X. Liu *et al.*, "Achieving full-view barrier coverage with mobile camera sensors," in *IEEE NaNA*, 2016.

[30] Y. Wu *et al.*, "Achieving full view coverage with randomly-deployed heterogeneous camera sensors," in *IEEE ICDCS*, 2012.

[31] R. Yang *et al.*, "Distributed algorithm for full-view barrier coverage with rotatable camera sensors," in *IEEE GLOBECOM*, 2015.

[32] H. Ma *et al.*, "A coverage-enhancing method for 3d directional sensor networks," in *IEEE INFOCOM*, 2009.

[33] X. Yang *et al.*, "3D visual correlation model for wireless visual sensor networks," in *IEEE ICIS*, 2017.

[34] C. Han *et al.*, "An Energy Efficiency Node Scheduling Model for Spatial-Temporal Coverage Optimization in 3D Directional Sensor Networks," *IEEE Access*, 2016.

[35] X. Yang *et al.*, "3-D Application-Oriented Visual Correlation Model in Wireless Multimedia Sensor Networks," *IEEE Sensors Journal*, 2017.

[36] P. Si *et al.*, "Barrier coverage for 3d camera sensor networks," *Sensors*, 2017.

[37] M. Hosseini *et al.*, "Sensor selection and configuration in visual sensor networks," in *IST*, 2012.

[38] L. Yupeng *et al.*, "A virtual potential field based coverage-enhancing algorithm for 3d directional sensor networks," in *ISSDM*, 2012.

[39] J. Peng *et al.*, "A coverage detection and re-deployment algorithm in 3d directional sensor networks," in *CCDC*, 2015.

[40] ——, "A coverage-enhance scheduling algorithm for 3d directional sensor networks," in *CCDC*, 2013.

[41] H.-J. Hsu *et al.*, "Face recognition on drones: Issues and limitations," in *ACM DroNet*, 2015.

[42] "https://enterprise.dji.com."

[43] W. Wang *et al.*, "PANDA: Placement of unmanned aerial vehicles achieving 3d directional coverage," *IEEE INFOCOM*, 2019.

[44] N. Joubert *et al.*, "Towards a Drone Cinematographer: Guiding Quadrotor Cameras using Visual Composition Principles," *CoRR*, 2016.

[45] T. Nägeli *et al.*, "Real-time Planning for Automated Multi-view Drone Cinematography," *ACM Transactions on Graphics*, 2017.

[46] L. Catarinucci *et al.*, "An IoT-Aware Architecture for Smart Healthcare Systems," *IEEE Internet of Things Journal*, 2015.

[47] T. He *et al.*, "Range-free Localization Schemes for Large Scale Sensor Networks," in *ACM MOBICOM*. ACM, 2003.

[48] K. Akkaya *et al.*, "IoT-based occupancy monitoring techniques for energy-efficient smart buildings," in *IEEE WCNC)*, 2015.

[49] Z. Chen *et al.*, "A Localization Method for the Internet of Things," *Springer The Journal of Supercomputing*, 2013.

[50] R. G. Michael *et al.*, "Computers and intractability: a guide to the theory of np-completeness," *WH Free. Co., San Fr*, 1979.

[51] J. D.S, "Euler's gem: The polyhedron formula and the birth of topology," *Elsevier*, vol. 6, 2008.

[52] S. Fujishige, *Submodular functions and optimization*, 2005.

[53] V. Chvatal, "A greedy heuristic for the set-covering problem," *Mathematics of operations research*, 1979.

[54] A. Saeed *et al.*, "Argus: realistic target coverage by drones," in *IEEE IPSN*, 2017.