Enhancing UAV Search Under Occlusion Using Next Best View Planning

Sigrid Helene Strand, Thomas Wiedemann, Bram Burczek, and Dmitriy Shutin, *Senior Member, IEEE*

***Abstract*—Search and rescue missions are often critical following sudden natural disasters or in high-risk environmental situations. The most challenging search and rescue missions involve difficult-to-access terrains, such as dense forests with high occlusion. Deploying uncrewed aerial vehicles for exploration can significantly enhance search effectiveness, facilitate access to challenging environments, and reduce search time. However, in dense forests, the effectiveness of uncrewed aerial vehicles depends on their ability to capture clear views of the ground, necessitating a robust search strategy to optimize camera positioning and perspective. This work presents an optimized planning strategy and an efficient algorithm for the next best view problem in occluded environments. Two novel optimization heuristics, a geometry heuristic, and a visibility heuristic, are proposed to enhance search performance by selecting optimal camera viewpoints. Comparative evaluations in both simulated and real-world settings reveal that the visibility heuristic achieves greater performance, identifying over 90% of hidden objects in simulated forests and offering 10% better detection rates than the geometry heuristic. In addition, real-world experiments demonstrate that the visibility heuristic provides better coverage under the canopy, highlighting its potential for improving search and rescue missions in occluded environments.**

***Index Terms*—Aerial mapping, evolutionary algorithms, next best view, optimal experimental design, search and rescue, uncrewed aerial vehicle.**

I. Introduction

THE equipment and methods currently used in search and rescue (SAR) operations are often inadequate for searches in obstructed or inaccessible terrain. Enhancing strategies for locating missing persons in such environments is therefore a critical objective. Difficult terrain poses significant obstacles to rescue missions, particularly when ground access is limited or impossible. In recent years, uncrewed aerial vehicles (UAVs) have emerged as a valuable asset for SAR, demonstrating remarkable effectiveness across various demanding landscapes and delivering promising results in urgent scenarios. However, certain terrains, such as dense forests, remain problematic for both human teams and UAVs. Forest environments present unique challenges, including reduced visibility for sensors positioned above the canopy. Thermal cameras may struggle with heat reflectance from foliage during daylight hours and laser scanners often fail to differentiate human bodies on the forest floor. Therefore, we propose that strategically angled cameras can still provide useful visual data, improving search efficiency in these complex settings.

In the literature, numerous path planning algorithms have revealed the possibilities of efficient search with UAVs equipped with a camera. Several 3-D exploration issues for aerial robotic platforms have been solved using a technique called next best view (NBV) [1], [2]. The path planning algorithm can be adapted to various applications, such as 3-D exploration of unknown environments and SAR operations with autonomous UAV. While dense forest scenarios exemplify situations where the NBV problem helps overcome severe visibility challenges, the broader question of identifying the most suitable camera perspectives remains open.

Our approach, based on the NBV scheme, attempts to improve SAR by introducing unique heuristics to select the optimal view, which increases the chance of discovering missing persons. Furthermore, the developed heuristics are studied and evaluated according to their performance. In both simulated and real-world investigations, performance is assessed by the outcome of the NBV algorithm's amount of discovered hidden objects while applying the proposed heuristics.

Received 21 March 2025; revised 22 September 2025; accepted 23 November 2025. Date of publication 1 December 2025; date of current version 12 December 2025. This work was supported by the German Aerospace Center (DLR) and the German Research Foundation (DFG) under Grant SH 1743/4-1. *(Corresponding author: Sigrid Helene Strand.)*

Sigrid Helene Strand, Bram Burczek, and Dmitriy Shutin are with the Swarm Exploration Group, Department of Communication Systems, Institute for Communications and Navigation, German Aerospace Center (Deutsches Zentrum für Luft und Raumfahrt, DLR), 82234 Wessling, Germany (e-mail: sigrid.strand@dlr.de; bram.burczek@gmail.com; dmitriy.shutin@dlr.de).

Thomas Wiedemann is with the Swarm Exploration Group, Department of Communication Systems, Institute for Communications and Navigation, German Aerospace Center (Deutsches Zentrum für Luft und Raumfahrt, DLR), 82234 Wessling, Germany, and also with the Perception for Intelligent Systems (PercInS), TUM School of Computation, Information and Technology, Technische Universität München (TUM), 80992 München, Germany (e-mail: thomas.wiedemann@dlr.de).

Digital Object Identifier 10.1109/JSTARS.2025.3638881

II. Related Work

A. Robotic Systems for Search and Rescue

Scientists within the robotics community have been studying robotic data acquisition for various purposes such as environmental monitoring, surveying, and rescue missions [3], [4], [5]. Robotic systems have played a role in conducting rescue efforts as part of SAR missions. In 2001, a single-robot system was first documented as being deployed in a disaster response. Since then, there have been many reported situations of single and multirobotic deployment in response operations [6]. The prerequisites for using robots for this purpose are robustness, speed, and ease of use by human personnel. Meeting this standard of reliability, efficiency, and usability is often challenging,

even with cutting-edge research [7]. uncrewed ground vehicles (UGVs) have been utilized for numerous applications in the rescue response, such as searching for survivors [8], inspecting inside damaged buildings, and other hazardous areas following natural disasters [9]. The main disadvantage is the difficulty of navigation in the complex unstructured environments [7]. Rescue operations for UGVs are limited to locations where the terrain is flat and the capacity to maneuver freely in space is considered limited.

Recent studies show interest in making SAR missions more efficient by using UAVs. They can navigate over terrain and have an overhead perspective, which is especially helpful in many rescue situations, such as dense map constructions for inspection [1] and forest fire monitoring [7], [10]. In addition, UAVs are easy to develop, dynamic, and can be modified for specific tasks in SAR applications [11]. However, sensor payload is often restricted by the size and power of the UAV [7].

There are three major motivations for performing SAR research with UAVs. First, research has shown that the exploration can be improved by making the UAVs autonomous and developing an optimal path for the rescue scenario [12]. Second, by extracting semantic information this solution enhances the SAR team's capacity to quickly locate objects of interest [13]. Finally, establishing a communication framework among SAR mission components can improve operations. In summary, UAVs have proven to be valuable and play an important role in enhancing SAR missions across various applications [11].

UAVs equipped with various sensors have been deployed for a variety of SAR applications. Red green blue (RGB) cameras, thermal cameras, radars and light detection and ranging (LiDAR) are some of the sensors that have been utilized to visualize surroundings or locate missing persons [14]. An RGB camera is often used on a UAV due to its lightweight and ease of usage. The RGB images can then be employed in conventional computer vision applications, such as object recognition, monitoring, and surveillance. One downside of an RGB camera is the limitation of light or occlusion. When there is no visibility, image quality and processing suffer significantly. Furthermore, thermal cameras have demonstrated advantages in the field due to their application for identifying heat sources. Thermal images have the same applications as RGB photos but can capture images in low-light circumstances [14], [15]. However, when the sensor is used in environments with temperatures similar to those of persons, separating the person from the ground is challenging. Radars have proven beneficial in SAR scenarios, such as locating survivors after earthquakes [16]. This sensor operates independently of light and weather conditions, but its resolution is lower compared to that of a LiDAR sensor [17]. LiDAR sensors are frequently utilized to create a 3-D model from the point cloud acquired. The LiDAR sensor offers the advantage of functioning effectively in any lighting conditions while delivering valuable structural insights about the environment. However, it can struggle in adverse weather conditions, such as heavy rain or fog, which scatter the laser beams and reduce accuracy [14].

B. *Next Best View Problem*

The NBV problem has challenged scientists for many years [18], [19]. With solutions that can improve exploration, mapping and SAR tasks, the NBV is highly relevant in the field of robotics. The main challenge of the NBV problem is determining the optimal view position and orientation for a camera or sensor installed on a robot. The optimal technique necessitates the definition of the ideal view; in addition, executing the task and navigating to the waypoints in the most efficient manner [20]. The gain from solving such a problem is to more efficiently create a 3-D object reconstruction [21] and 3-D exploration paths [1]. Previous scientific work has shown various attempts to tackle the problem that was initially addressed in the 1980s by Connolly [22]. Iteratively selecting the optimal viewpoint to maximize the gathered information, ultimately achieves a comprehensive observation. Further, Mavier et al. [23] described an approach for obtaining 3-D data of an unknown scene. The method focuses on occluded regions, which means that only the edges of the occlusions are modeled to determine the next step.

More recent studies have demonstrated the use of robotic platforms for autonomous exploration of unknown environments using the NBV scheme. Scott et al. [24] classified NBV techniques into surface-based and volumetric categories. While these methods show potential for scene exploration, they remain limited in efficiency and are often too slow for real-time applications. They lack robustness in the presence of sensor noise or complex geometries. To address these issues, Scott et al. suggested multistage strategies and the use of heuristic or expert-system designs emulating human operators. These observations indicate the need for NBV methods that are computationally efficient and adaptable to varying scene complexities, which is the focus of this work. Bircher et al. [1] introduced a volumetric receding horizon NBV planner in which a rapidly exploring random tree (RRT) is generated over a voxel-based environment. The best branch is selected based on the volume of unexplored space, and the first edge of this branch from the RRT is executed. The method exhibits short computation times and improved scalability compared to frontier-based planners, enabling online path generation for autonomous aerial robots. However, evaluation was limited to a constrained indoor setting with restricted sensing range, short planning horizons, and moderate motion speeds, which may limit applicability in larger or more complex environments. Moreover, performance in scenarios with long-range sensing or dynamic scenes remains unverified. Krainin et al. [25] proposed a surface-based NBV approach in which a robot grasps an object and actively repositions it in front of a depth camera to construct a complete surface model. Using an information gain metric, the system brings uncertain object regions into view, accounts for manipulator occlusions, and triggers regrasping when necessary. This produces high-quality models for tabletop objects; however, the method relies on external grasping techniques and is tailored to controlled indoor manipulation, limiting applicability to free-space or large-scale exploration. Kriegel et al. [26] combined surface-based and volumetric

strategies to produce high-quality surface models using a next-best-scans (NBS) selection criterion that considers both surface quality and information gain. Their system updates a probabilistic voxel space and a triangle mesh in real time, enabling collision-free scan planning and significantly outperforming human operators in acquisition speed. While effective for static objects in controlled settings, the approach requires object repositioning to capture occluded regions, struggles with complex geometries, and assumes a fixed scanning setup, which limits applicability to mobile or unstructured environments. Informed by these limitations, this work presents a surface-based NBV sampling technique for partially known environments, where unexplored regions are defined by vertices of the surface model rather than volumetric occupancy alone. In addition, a heuristic based on the geometry of camera-view constellations is proposed for efficient viewpoint planning. The methods improve adaptability and planning efficiency, enabling application to SAR missions. While person detection in images is not addressed directly, the proposed approach facilitates rapid identification of optimal viewpoints for taking images, an essential precursor to locating missing persons.

III. Problem Description

For the scope of this article, let us assume an outdoor environment cluttered with obstacles, such as trees or bushes, that obstruct the view. We assume that the scenery has been roughly mapped and reconstructed using photogrammetry methods from $\mathcal{C}_k = \{\boldsymbol{o}_k, \theta_k, \gamma_k\}$, $k = 1, \ldots, K$, initial camera views. Here, each kth camera $\mathcal{C}_k$ is parameterized with a camera position $\boldsymbol{o}_k \in \mathbb{R}^3$ in 3-D space, as well as camera orientation defined in terms of pitch θ_k and yaw γ_k angles.[1] The collection of the views is used to construct an initial, or *a priori*, 3-D model of the environment. This initial model we represent by its surface as a triangular mesh $\mathcal{M} = \{\mathcal{V}, \mathcal{F}\}$, where $\mathcal{V} = \{\boldsymbol{n}_1, \ldots, \boldsymbol{n}_N\}$ is a set of vertices with coordinates $\boldsymbol{n}_i \in \mathbb{R}^3$, $i = 1, \ldots, N$, and $\mathcal{F}$ is a set of corresponding triangular faces.

Within the scenery, an unknown number of objects of interest, e.g., missing people or damaged structures, are located at unknown positions. Due to the obstacles and obstructed view, these objects might not be fully visible in the initial top-down views $\mathcal{C}_k$, $k = 1, \ldots, K$; instead, they can be obstructed by foliage or other obstacles within the camera's field of view. Our goal is to iteratively calculate an NBV configuration $\mathcal{C}_{K+1}$ that includes new positions $\boldsymbol{o}_{K+1}$ and orientations $(\theta_{K+1}, \gamma_{K+1})$ of a camera carried by a UAV, which would increases the visibility of hidden objects in one of the camera's views.

Finding the NBV can be formulated as an optimization problem. However, since the location of the hidden objects is generally unknown, these cannot be included directly in the optimization objective. Instead, we propose and investigate optimization heuristics that can be used to score different views in the optimization routine. Our goal is thus to place cameras optimally so that they can better "illuminate" hidden objects in the scene. We propose two heuristics that, when optimized, indirectly improve the view of the scene and thus increase the chance of finding hidden objects. This procedure is detailed in the following section.

[1]Note that the roll angle of the camera is assumed to be fixed since it does not change the visibility of the scene substantially. Thus, only two angles are used to specify the orientation.

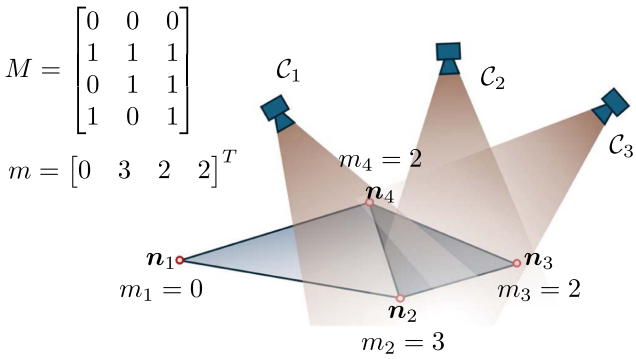

Fig. 1. Example Visibility Fitness calculation for $K = 3$ cameras observing a mesh with $N = 4$ vertices.

IV. NBV Optimization Approach

When formulated as an optimization problem, finding the NBV requires an appropriate definition of a fitness (objective) function. In this work, we introduce two different heuristics to serve as fitness functions for the NBV optimization problem. Our goal is to develop an effective yet straightforward heuristic that indirectly enhances object visibility.

A. Visibility Fitness

The motivation for the first proposed heuristic is to maximize the number of vertices in $\mathcal{V}$ that are seen with a new camera view $\mathcal{C}_{K+1}$ by changing its position and orientation. Moreover, the heuristic should reward a new camera view, if a vertex is not readily seen by the other, already placed cameras $\mathcal{C}_k$, $k = 1, \ldots, K$. At the same time, we penalize the objective if too many cameras observe the same vertex, thus not delivering any additional information.

Let us calculate the visibility of all vertices in $\mathcal{V}$ by a new view $\mathcal{C}_{K+1}$. We represent the visibility as an N-dimensional binary vector $\boldsymbol{w} \in \{0, 1\}^{|\mathcal{V}|}$. Each element $w_i \triangleq [\boldsymbol{w}]_i$, $i = 1, \ldots, N$, encodes if a vertex i is visible by the camera $\mathcal{C}_{K+1}$. In particular

$$w_i = \begin{cases} 1; \text{ if } \boldsymbol{n}_i \text{ is visible by } \mathcal{C}_{K+1} \\ 0; \text{ else} \end{cases} \quad \forall \boldsymbol{n_i} \in \mathcal{V}. \tag{1}$$

To check for visibility we use a simple ray tracer. Namely, if the ray emitted from the camera position $\boldsymbol{o}_{K+1}$ to the vertex $\boldsymbol{n}_i \in \mathcal{V}$ intersects with any face in $\mathcal{F}$, we consider the vertex as not visible. The same goes for the case where the vertex $\boldsymbol{n}_i \in \mathcal{V}$ is outside the field of view of a new camera $\mathcal{C}_{K+1}$.

In the same way, we define the visibility matrix for already available camera views $\boldsymbol{M} \in \{0, 1\}^{|\mathcal{V}| \times K}$, with $i - k$th element $[\mathbf{M}]_{i,k}$ given as (see also Fig. 1)

$$[\mathbf{M}]_{i,k} = \begin{cases} 1; \text{ if } \boldsymbol{n}_i \text{ is visible by } \mathcal{C}_k \\ 0; \text{ else} \end{cases}$$
$$\forall \boldsymbol{n}_i \in \mathcal{V}, \quad k = 1, \ldots, K. \tag{2}$$

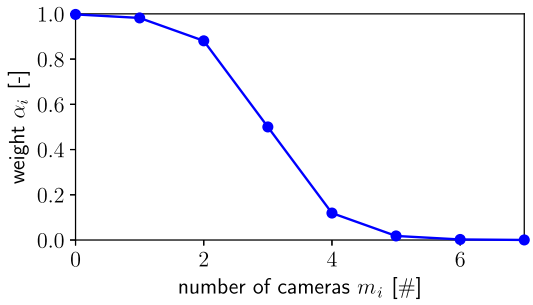

Fig. 2. Scaling function for the weights α_i in (5) versus vertex visibility.

For each vertex, we can calculate the number of cameras that can see the respective vertex as

$$m_i = \sum_{k=1}^{K} [\mathbf{M}]_{i,k} \tag{3}$$

which is an ith row-sum of the matrix $\boldsymbol{M}$. In Fig. 1 we illustrate this computation for a simplified case. As can be seen, the variable m_i just counts how many views of a vertex have been collected so far. Intuitively, we want to design $\mathcal{C}_{K+1}$ so as to focus on vertices that were observed insufficiently often, or were not observed at all (such as, e.g., vertex $\boldsymbol{n}_1$ in Fig. 1). To this end, we define a Visibility Fitness function as follows:

$$J_v(\mathcal{C}_{K+1}) = \sum_{i=1}^{N} \alpha_i w_i \tag{4}$$

where the weights α_i, $i = 1, \ldots, N$, are now specifically designed to give preference to vertices with low values of m_i

$$\alpha_i = \frac{1 - \tanh(m_i - 3)}{2}. \tag{5}$$

The choice of the weighting function is empiric, yet it serves its purpose: the weights α_i, $i = 1, \ldots, N$, are selected to reward vertices with low visibility m_i, while at the same time scaling down vertices that are already visible from multiple cameras. The scaling function (5) is visualized in Fig. 2 to illustrate this behavior. As one can see, we have chosen the function (5) and its parameter, so that the reward drops after a vertex has been seen by at least two cameras and reaches nearly zero for more than four cameras. These values are motivated by photogrammetric applications, where two cameras are sufficient to determine a 3-D position of a vertex.

Finally, placing a new camera can be posed as a problem of maximizing the Visibility Fitness $J_v(\mathcal{C}_{K+1})$

$$\max_{\boldsymbol{o}_{K+1}, \theta_{K+1}, \gamma_{K+1}} J_v(\mathcal{C}_{K+1}) \tag{6}$$

where the optimization is performed with respect to the parameters $\boldsymbol{o}_{K+1}$, θ_{K+1}, and γ_{K+1} of the next $(K+1)$th camera.

B. Geometry Fitness

Another criterion that we propose is based on geometrical properties. Specifically, we consider the information of the geometric constellation between the camera views to find the next best suitable camera $\mathcal{C}_{K+1}$, while accounting for the geometry of already placed cameras $\mathcal{C}_k$, $k = 1, \ldots, K$. This geometry plays a role in the photogrammetric reconstruction of an individual vertex $\boldsymbol{n}_i \in \mathcal{V}$, $i = 1, \ldots, N$ (see [27]). Indeed, the computation of the vertex $\boldsymbol{n}_i$ using cameras $\mathcal{C}_k$, $k = 1, \ldots, K$, can be interpreted as (in general) a *nonlinear* transformation between multiple 2-D image spaces into a global 3-D space. The curvature of this transformation in the vicinity of vertex coordinates in the object space can be interpreted as a precision of the transformation; the inverse curvature can then be interpreted as a (local) uncertainty of the transformation between image space and the global coordinates of the vertex. The measure of the uncertainty, as described by, e.g., the determinant of the resulting inverse curvature matrix, can be naturally used as a criterion for camera placement: new camera position $\mathcal{C}_{K+1}$ is selected to reduce the resulting uncertainty. Such approaches are formally known as *optimal experiment design* (see, e.g., [28], [29]). We will develop an intuition for this perspective on the transformation later, as details of the transformation are stated more formally, which we now discuss in the following.

1) Relation Between Image and Object Space: To understand the relation between the coordinates in the camera's image space and the coordinates in global space, we need to formally define a coordinate transformation. As a first step, we define the transformation of the *global* 3-D space into a *local* camera 3-D frame. The local camera frame is defined with respect to camera orientation, such that its origin is located in the camera's focal point and the positive direction of the z-axis is alight with the direction of view. The corresponding transformation specifically for the kth camera can be expressed as

$$\boldsymbol{n} = \boldsymbol{o}_k + \boldsymbol{R}_k(\theta_k, \gamma_k)\boldsymbol{n}' \tag{7}$$

where $\boldsymbol{n}$ and $\boldsymbol{n}'$ are vertices in the global and local coordinate frames, respectively, and $\boldsymbol{R}_k(\theta_k, \gamma_k)$ is (orthogonal) rotation matrix defined by the orientation angles θ_k and γ_k of the corresponding camera $\mathcal{C}_k$. Note that we implicitly assume that the local coordinate system is a function of a particular camera; as such, we generally have K different local coordinate frames, and a single global one.

From (7), it becomes obvious that

$$\boldsymbol{n}' = \boldsymbol{R}_k(\theta_k, \gamma_k)^T (\boldsymbol{n} - \boldsymbol{o}_k) \tag{8}$$

where we used the fact that for orthogonal matrices $\boldsymbol{R}_k^{-1} = \boldsymbol{R}_k^T$. The expression (8) provides a mapping from the global (or object) coordinates $\boldsymbol{n}$ into the local camera frame $\boldsymbol{n}'$. Now, assuming that we have a nondistorted image with vertex coordinates $\boldsymbol{n}' = [x', y', z']^T$ and a camera modeled as a pin-hole camera with a focal length f, we can express 2-D pixel coordinates $\boldsymbol{p}(\boldsymbol{n}) = [u(\boldsymbol{n}), v(\boldsymbol{n})]^T$ of a vertex $\boldsymbol{n}$ as

$$u(\boldsymbol{n}) = f\frac{x'}{z'}, \quad v(\boldsymbol{n}) = f\frac{y'}{z'}. \tag{9}$$

The latter transformation expressions are also known as co-linearity equations [30].

2) Linearization of the Co-Linearity Equation: Inspecting (9) reveals that the transformation between an arbitrary point $\boldsymbol{n}$ in a global coordinate system into pixel coordinates (u, v) is nonlinear. Let us now consider a linearization of this transformation. For this purpose, we consider the kth camera $\mathcal{C}_k$ and compute a first-order Taylor expansion of this mapping around

a mesh vertex $\boldsymbol{n}_i \in \mathcal{V}$ for the kth camera

$$\boldsymbol{p}_k(\boldsymbol{n}) \approx \boldsymbol{A}_{k,i}\boldsymbol{n} + \boldsymbol{c}_{k,i} \tag{10}$$

where $\boldsymbol{c}_{k,i}$ is some constant term, and $\boldsymbol{A}_{k,i}$ is a Jacobian of the co-linearity equations defined as

$$\boldsymbol{A}_{k,i} = \left[\begin{matrix}\nabla u(\boldsymbol{n})^T \\ \nabla v(\boldsymbol{n})^T\end{matrix}\right]\Bigg|_{\boldsymbol{n}=\boldsymbol{n}_i}. \tag{11}$$

The matrix $\boldsymbol{A}_{k,i}$ thus represents a projection of a single vertex to the corresponding image coordinates. If the vertex is not visible, all elements in $\boldsymbol{A}_{k,i} = 0$. Visibility is determined by projecting the vertex into the image plane of $\mathcal{C}_k$ and checking whether it lies within the camera frustum and is not occluded by other surfaces. This is performed as described in (2). For all available cameras $k = 1, \ldots, K$, we can equivalently write

$$\underbrace{\left[\begin{matrix}\boldsymbol{p}_1(\boldsymbol{n}) \\ \vdots \\ \boldsymbol{p}_K(\boldsymbol{n})\end{matrix}\right]}_{\triangleq \widetilde{\boldsymbol{p}}(\boldsymbol{n})} \approx \underbrace{\left[\begin{matrix}\boldsymbol{A}_{1,i} \\ \vdots \\ \boldsymbol{A}_{K,i}\end{matrix}\right]}_{\triangleq \widetilde{\boldsymbol{A}}_i} \boldsymbol{n} + \widetilde{\boldsymbol{c}}_i \tag{12}$$

which summarizes coordinates of the *same* vertex $\boldsymbol{n}$ as seen by different cameras in the corresponding K camera images.

For all available vertices $\boldsymbol{n}_i \in \mathcal{V}$ we can generalize (12) as follows:

$$\left[\begin{matrix}\widetilde{\boldsymbol{p}}(\boldsymbol{n}_1) \\ \widetilde{\boldsymbol{p}}(\boldsymbol{n}_2) \\ \vdots \\ \widetilde{\boldsymbol{p}}(\boldsymbol{n}_N)\end{matrix}\right] \approx \underbrace{\left[\begin{matrix}\widetilde{\boldsymbol{A}}_1 & & & \\ & \widetilde{\boldsymbol{A}}_2 & & \\ & & \ddots & \\ & & & \widetilde{\boldsymbol{A}}_N\end{matrix}\right]}_{\overline{\boldsymbol{A}}} \left[\begin{matrix}\boldsymbol{n}_1 \\ \boldsymbol{n}_2 \\ \vdots \\ \boldsymbol{n}_N\end{matrix}\right] + \text{const} \tag{13}$$

which is essentially an approximate affine mapping of the vertices $\mathcal{V}$ onto the pixel coordinates of the images for all K cameras.

In practice, a classical photogrammetry application would target estimation of vertex coordinates $\boldsymbol{n}_i$, $i = 1, \ldots, N$, from collected images (or rather corresponding pixels) by "inverting" (13), which typically amounts to determining a least squares solution $\widehat{\boldsymbol{n}}_i$, $i = 1, \ldots, N$. It is well-known [31] that the associated estimation error is zero-mean, with covariance (or rather dispersion) matrix given by $\boldsymbol{Q} = (\overline{\boldsymbol{A}}^T\overline{\boldsymbol{A}})^{-1}$, with $\overline{\boldsymbol{A}}^T\overline{\boldsymbol{A}}$ being an approximation of a curvature of the underlying nonlinear transformation between global coordinates and image pixels. As such, we can measure the accuracy of this inverse mapping based on some measure of the "size" of $\boldsymbol{Q}$ or, more specifically, the change of its "size" due to a new camera $\mathcal{C}_{K+1}$.

It it easy to see that $\boldsymbol{Q}$ can be computed as

$$\boldsymbol{Q} = \left(\overline{\boldsymbol{A}}^T\overline{\boldsymbol{A}}\right)^{-1} = \left[\begin{matrix}(\widetilde{\boldsymbol{A}}_1^T\widetilde{\boldsymbol{A}}_1)^{-1} & & \\ & \ddots & \\ & & (\widetilde{\boldsymbol{A}}_N^T\widetilde{\boldsymbol{A}}_N)^{-1}\end{matrix}\right]. \tag{14}$$

Let $\boldsymbol{o}_{K+1}$ and $\theta_{K+1}, \gamma_{K+1}$ denote some arbitrary position and orientation of yet-to-be-placed camera $\mathcal{C}_{K+1}$. From (11), we can compute the corresponding Jacobian $\boldsymbol{A}_{K+1,i}$ as if we have a camera placed at $\boldsymbol{o}_{K+1}$ and oriented it according to angles $(\theta_{K+1}, \gamma_{K+1})$. With the new camera, the matrices $\widetilde{\boldsymbol{A}}_i$ in (12) and the corresponding products $\widetilde{\boldsymbol{A}}_i^T\widetilde{\boldsymbol{A}}_i$, $i = 1, \ldots, N$, in (14) are extended as follows:

$$\widetilde{\boldsymbol{A}}_i(\mathcal{C}_{K+1}) \triangleq \left[\begin{matrix}\widetilde{\boldsymbol{A}}_i \\ \boldsymbol{A}_{K+1,i}\end{matrix}\right]$$

$$\widetilde{\boldsymbol{A}}_i(\mathcal{C}_{K+1})^T\widetilde{\boldsymbol{A}}_i(\mathcal{C}_{K+1}) \triangleq \sum_{k=1}^{K+1}\left(\boldsymbol{A}_{k,i}^T\boldsymbol{A}_{k,i}\right). \tag{15}$$

Note that we now indicate an explicit dependency of $\widetilde{\boldsymbol{A}}_i(\mathcal{C}_{K+1})$ on the location and orientation of the new camera. This allows us to compute the corresponding would-be dispersion matrix $\boldsymbol{Q}(\mathcal{C}_{K+1})$ simply by including an additional contribution from the new camera.

3) Optimality Criterion: Now we can measure how the total dispersion changes due to placing a new camera $\mathcal{C}_{K+1}$. This can be achieved using classical criteria from optimal experiment design theory [28], [29]. In particular, we will use a so-called D-optimality criterion to measure the size of the dispersion matrix. The D-optimality criterion measures the volume of the resulting uncertainty ellipsoid created by the eigenvalues λ_i of the dispersion matrix $\boldsymbol{Q}$ (see also Fig. 3). The camera position $\mathcal{C}_{K+1}$ is then selected to minimize the resulting volume of the new dispersion matrix $\boldsymbol{Q}(\mathcal{C}_{K+1})$ as given by its determinant. Naturally, a smaller volume of the uncertainty ellipsoid obtained by adding a new camera $\mathcal{C}_{K+1}$ implies a more precise reconstruction of the 3-D points in the global coordinates from individual camera images. Let us now define the corresponding NBV cost function with respect to a new camera $\mathcal{C}_{K+1}$ as follows:

$$\begin{aligned} J_d(\mathcal{C}_{K+1}) &= \det\left(\boldsymbol{Q}(\mathcal{C}_{K+1})^{-1}\right) \\ &= \prod_{i=1}^{N}\det\left(\widetilde{\boldsymbol{A}}_i(\mathcal{C}_{K+1})^T\,\widetilde{\boldsymbol{A}}_i(\mathcal{C}_{K+1})\right) \\ &= \prod_{i=1}^{N}\det\left(\sum_{k}^{K+1}\boldsymbol{A}_{k,i}^T\boldsymbol{A}_{k,i}\right) \end{aligned} \tag{16}$$

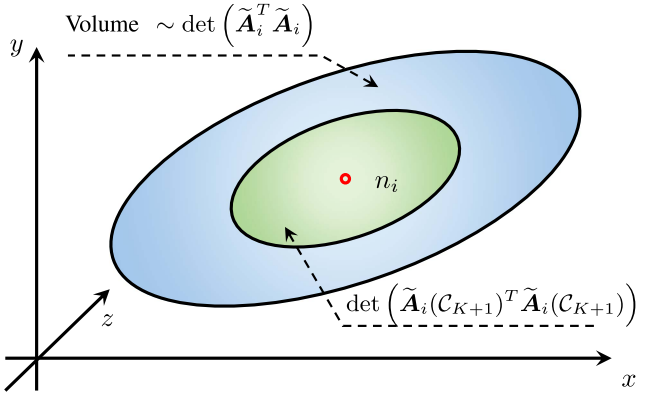

Fig. 3. Illustration of D-optimality criterion for a single vertex n_i to position the next $\mathcal{C}_{K+1}$ camera. The determinant can be interpreted as the volume of the uncertainty ellipsoid.

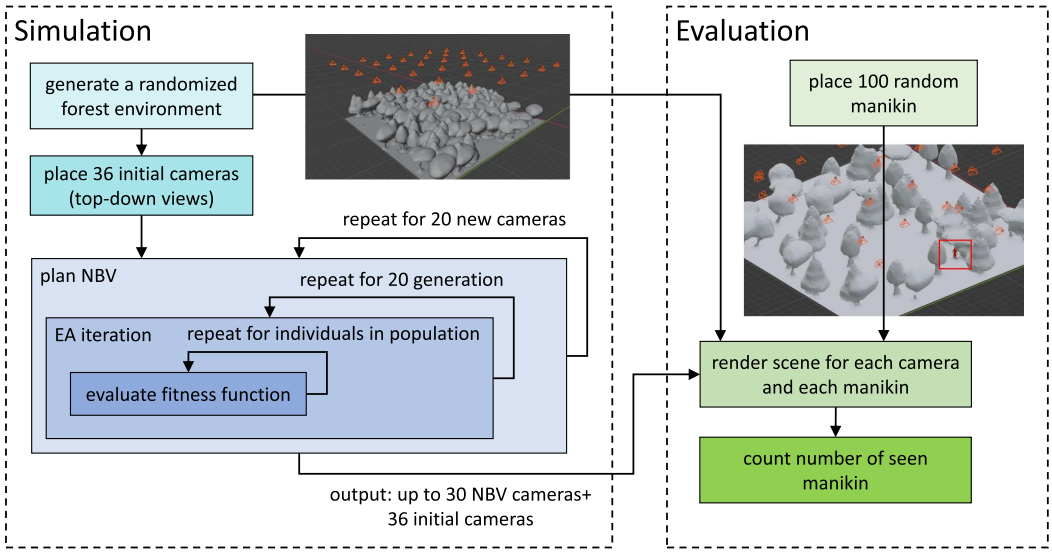

Fig. 4. Workflow of simulation experiments to evaluate the proposed NBV planning.

where we exploit the fact that $\boldsymbol{Q}(\mathcal{C}_{K+1})$ is a diagonal block matrix. Note that we use an inverse of $\boldsymbol{Q}(\mathcal{C}_{K+1})$, which is equivalent to a precision (or information) matrix. As a result, maximizing $J_d(\mathcal{C}_{K+1})$ with respect to the parameter $\boldsymbol{o}_{K+1}$ and $\theta_{K+1}, \gamma_{K+1}$ of the $(K+1)$th camera is equivalent to increasing the information about reconstructed 3-D points in global coordinate system. It is also equivalent to minimizing the D-optimality criterion, defined with respect to $\boldsymbol{Q}(\mathcal{C}_{K+1})$ rather than with respect to its inverse. Now, the placement of the new camera can be posed as the following optimization problem:

$$\max_{\boldsymbol{o}_{K+1}, \theta_{K+1}, \gamma_{K+1}} J_d(\mathcal{C}_{K+1}) \tag{17}$$

where the optimization is performed over the position $\boldsymbol{o}_{K+1}$ and orientation $\theta_{K+1}, \gamma_{K+1}$ of the new $K+1$ camera.

Let us stress that although (17) does require computing the determinant for each potential camera position, individual terms $\boldsymbol{A}_{k,i}^T \boldsymbol{A}_{k,i}$ as defined in (11) are only 3×3 matrices. The corresponding determinant can be easily computed in closed form and is thus easy to evaluate for each vertex $i = 1, \ldots, N$, in the mesh $\mathcal{M}$.

C. Optimization Algorithm

In the search of the NBV our goal is to maximize the fitness functions (4) or (17) by placing new cameras $\mathcal{C}_{K+1}$ in an optimal way. While the computation of the fitness function for each individual camera position is moderate, the search through the parameter space $\{\boldsymbol{o}_{K+1}, \theta_{K+1}, \gamma_{K+1}\}$ to maximize the corresponding fitness functions can be quite prohibitive. The brute-force optimization of the constructed objective functions is impractical, as well as, gradient-based optimization algorithms. Both (6) and (17) are nonconvex, having multiple local optima, which renders the use of gradient-based approaches inappropriate. Thus, we propose to utilize an evolutionary algorithm (EA) to address the considered optimization problems [32], [33]. In this way, we can cope with the high number of local optima in our optimization objective. Note that while EA is a stochastic optimization approach, in general, it does not guarantee reaching a globally optimal solution in a finite time. However, it might nonetheless be "sufficiently" close to the optimum, ensuring a good *practical* solution rather than the best possible one.

In our case for the EA algorithm we define an "individual" as a particular combination of parameters $\boldsymbol{o}_{K+1}, \theta_{K+1}$, and γ_{K+1} of a potential next camera view $\mathcal{C}_{K+1}$. We will restrict the parameter space of possible camera positions to certain values. This is done not only to reduce computational complexity but also to account for the physical constraints of a 3-D scenery. First, we bound the z-component of $\boldsymbol{o}_{K+1}$—the camera height—to a certain range above ground to avoid collision with the scene. Namely, we restrict the height to be between 10 and 16 m. Also, X- and Y-component of $\boldsymbol{o}_{K+1}$ are restricted to a range close to the area of interest since camera locations that are far away from the scene could be excluded immediately as suboptimal. Therefore, we limit the X- and Y-coordinates to be within ± 20 m with respect to the center of the area of interest.

A population of a single generation in our EA optimization includes I such individuals. Each individual in the population is initialized with random values for the first generation, yet adhering to the constraints of the search space, as defined previously. We then use either the Visibility Fitness J_v or Geometry Fitness, J_d, to determine the performance of each individual in the population, i.e., how well the camera is placed according to the proposed heuristic. Depending on the fitness, the best individuals from the population are selected for the reproduction of the next generation. We randomly sample three individuals from the previous population and select the one with the highest fitness. This process is repeated until the new generation is fully populated. Selected individuals are then consecutively paired and recombined to produce new "offspring" individuals, a process known as crossover [32]. Crossover allows the algorithm to explore new areas of the search space by combining different characteristics (genes) of the parent solutions. In our case, we make use of a two-point crossover. The process involves selecting two points (or positions) in the parent genetic representation (here camera

parameters) and swapping the genetic material between these points to generate a new individual. In addition, a small portion of the offspring is randomly altered, which in an EA algorithm is known as mutation. We use Gaussian mutation, where a gene, i.e., camera parameter, in an individual's genome, is altered by a small randomly chosen value from a Gaussian distribution with zero mean and variance one. While mutation enables exploration of new regions in the search space through random changes, its primary role is to fine-tune promising solutions by introducing small adjustments. The overall cycle is then repeated for multiple generations.

In our implementation, we make use of the DEAP Python library to implement the algorithm [32]. The performance of the EA algorithm is affected by several parameters, mainly the population size I, number of generations N_G before termination, as well as crossover ρ_c and mutation ρ_m rates. The crossover ρ_c rate defines the probability of a crossover operation to generate a new offspring, while the mutation ρ_m rate defines the probability of a random mutation of an individual in the population. Both ρ_c and ρ_m determine the balance between exploration (discovering new areas of the search space), and exploitation (fine-tuning existing solutions). In our actual practical use, moderate values were selected; namely, for the crossover rate, we used $\rho_c = 0.8$, and for the mutation rate $\rho_m = 0.2$. The population size I and number of generations N_G will be discussed in greater detail in the evaluation section.

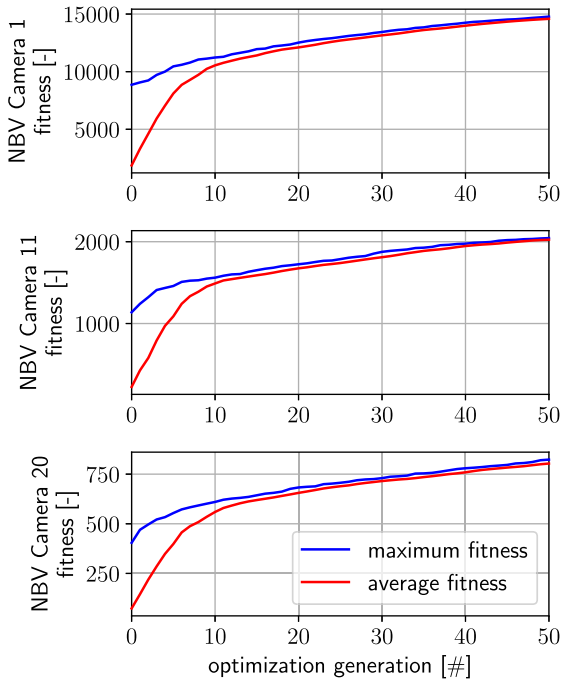

Fig. 5. Convergence of EA for the Visibility Fitness, J_v, plotted over the number of generations. The blue curve represents the maximum achieved fitness in the population for a particular generation, while the red curve shows the average in the population.

V. Experiments and Evaluation

To evaluate our approach, we carried out experiments in simulations as well as in a real-world scenario. The simulations allow us to get a deeper insight into the performance of our NBV planning and to conduct a high number of experiments for different constellations of the environment. On the other hand, our real-world experiments are less comprehensive but act as a proof of concept. They show that our approach also works under real-world conditions.

A. Evaluation in Simulation

The purpose of the evaluation in simulation is twofold: First, we want to verify that our approach can place cameras in a way so that the chosen heuristics are maximized and also converge to an optimum. Second, we want to validate that by maximizing the heuristic, we also increase our chance of finding hidden objects in the camera views. In addition, we also want to compare the two heuristics and determine which one performs better. Let us start by explaining the simulation setup.

1) Simulation Setup: The overall workflow of a single simulation run is depicted in Fig. 4. As mentioned in the Section III, we assume that we start with a rough 3-D model of the environment obtained from first surveillance from a top-down view of pictures taken by a UAV. In the simulation, we just generate a randomization forest scenario that spans 30 m $\times$ 30 m where trees are placed at random positions with random shapes and different densities. Further, we place 36 initial cameras on a regular 6 $\times$ 6 grid above the scenario, which emulates the viewpoints from the first UAV flight. We are supposing that the 3-D model has been created from these 36 camera views. Based on this initial setup, we plan iteratively new camera views given the already placed cameras. As a result, we obtain camera position $\boldsymbol{o}_{K+1}$, and orientation $(\theta_{K+1}, \gamma_{K+1})$. For the implementation, we make use of the Python library DEAP [33]. In total, we plan 20 NBV. To evaluate them, we generated 100 independent scenes, each containing a single virtual red manikin placed at a random position below the trees in the forest environment. For each camera, i.e., 36 initial cameras and 20 planned cameras, we render the scene once for each hidden manikin. If the sum of red pixels in the rendered image is larger than 0, we consider the manikin visible by the camera view. Based on that, we determine how many of the 100 manikins are partially visible to the cameras. Where higher values mean better performance of the NBV planning. With this data, we can quantify the enhancement of visibility offered by the NBV configuration when compared to the initial top-down perspectives.

2) Simulation Results: Let us first have a look at the performance of the EA optimization. In Fig. 5, the fitness value is plotted for finding the first, the 11th, and the 20th NBV after the initial 36 camera views. The figure depicts the average and maximum fitness within one generation of the EA optimization. For these simulation runs, the population size was chosen to be 50 and we made use of the Visibility Fitness, J_v. The curves are averaged over 20 simulation runs with different randomized environments.

While the performance continuously increases, the plots indicate that the rate of improvement decreases after approximately 20 generations. For the Geometry Fitness, J_d, we observed a similar trend, with a steeper improvement in the early

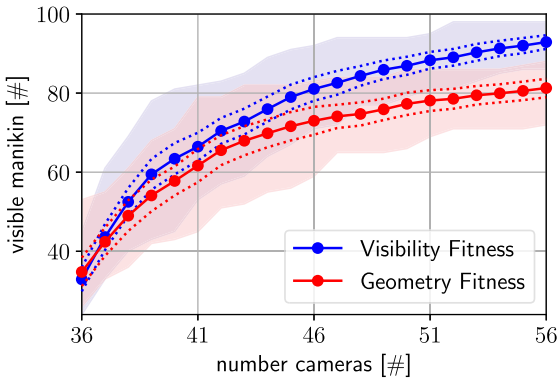

Fig. 6. Curves show how many of the 100 hidden manikins are visible with respect to the placed camera views. For the blue curve, the Visibility Fitness was used during the view optimization, and for the red curve the Geometry Fitness, respectively. The curves are averaged over 18 simulation runs with randomized environments. The transparent area marks the minimum and maximum values, and dotted lines the 95% confidence interval for the 18 runs.

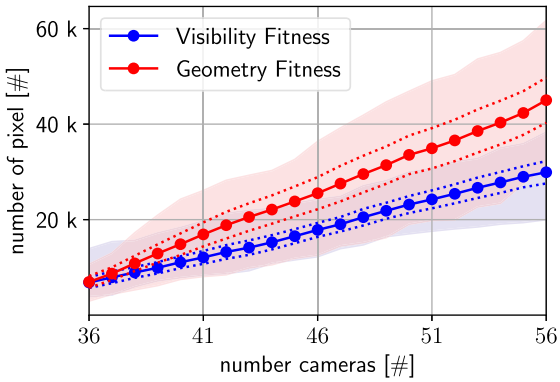

Fig. 7. Curves show how many red pixels (associated with the manikin) are seen when the scene is rendered for the camera views. The number of pixels is summed over all cameras and summed over all 100 hidden manikins. The curves are then averaged over 18 simulation runs with randomized environments. The transparent area marks the minimum and maximum values, and dotted lines the 95% confidence interval for the 18 runs.

generations followed by slower gains thereafter. Beyond generation 20 up to 50, the heuristic value improves by roughly 25%, which, although not negligible, must be weighed against the additional computational cost. For our application, increasing the optimization time over a factor of two was deemed not justifiable for the observed level of improvement, and we therefore terminated further simulation experiments after 20 generations.

The dominant computational cost of the proposed NBV heuristics stems from the ray tracing module, which is required to evaluate the visibility of each candidate solution. For every individual in the population, one ray is cast per vertex n_i of the environment model. The total number of ray tracing operations scales as $O(\mathrm{N_{pop}} \times \mathrm{N_{gen}} \times \mathrm{n_i})$, where N_{pop} is the population size and N_{gen} the number of generations. Increasing the population or number of generations thus linearly increases the computational burden, while more complex environments with a larger number of vertices proportionally raise the number of ray casts. Since ray tracing dominates runtime, other steps such as the selection between the Visibility Fitness and Geometry Fitness have negligible impact on the overall cost. Moreover, since multiple NBVs are computed sequentially, the overall computational cost becomes $O(\mathrm{N_{pop}} \times \mathrm{N_{gen}} \times \mathrm{n_i} \times (\mathrm{N_{NBV}} + \mathrm{N_{init}}))$, where N_{NBV} denotes the number of selected next best views and N_{init} the number of initial camera views. Several NBV approaches in the literature, such as the voxel-based receding-horizon planner of Bircher et al. [1], operate on volumetric occupancy grids whose computational cost scales as $O(N_v^3)$ with the linear voxel grid dimension N_v, since all voxels must be evaluated when computing information gain. In contrast, the proposed surface-based method scales linearly as $O(n_i)$ with the number of mesh vertices, typically satisfying $n_i \ll N_v^3$. This yields a substantially lower complexity than voxel-based strategies for comparable scenes, thus trading accuracy of the representation for computational efficiency.

The outcome of the NBV planning is depicted in Fig. 8, as an example, with a hidden virtual manikin in red. The scene shows the initial cameras with a top-down view in blue and the 20 planned camera views in green. The manikin is not visible from the top-down views, since it is occluded by the trees. However, one of the NBV has a perfect view of the manikin.

Fig. 6 shows how many of the 100 hidden manikins are visible in the placed camera views. The curves start with the initial 36 top-down cameras and show how the iteratively placed NBV increase the chance of seeing manikins in the camera pictures. The two curves utilize the two different heuristics in the EA optimization of the views. The plots are generated based on 18 simulation runs for each heuristic. The initial views capture no more than a maximum of 55 and an average of 35 out of the 100 placed manikins. This indicates that approximately half of the manikins remain hidden from the initial views. By adding 20 cameras with our proposed NBV approach based on the Visibility Fitness, we can see more than 90% of the manikins on average in the camera views. For the Geometry Fitness, we increase the chance to 80% on average.

So, from a comparison of the two heuristics, we can conclude that the Geometry Fitness heuristic does not enhance the ability to detect hidden individuals as much as the Visibility Fitness heuristic. On average, the Visibility Fitness reveals 10% more individuals than the Geometry Fitness. However, it is noteworthy that the NBVs calculated by the Geometry Fitness expose significantly more red pixels of hidden manikins when rendered, as can be seen from Fig. 7. This suggests that while the Visibility Fitness is more effective in detecting at least some part of the manikins, the Geometry Fitness may offer a better view of the manikins at the price of missing some.

B. *Evaluation in Real World Experiment*

The real-world evaluation was conducted in a forest area characterized by dense bushes, small trees, and irregular terrain, bordered by open space and a gravel road. The 3-D reconstruction of the area can be seen in Fig. 9. This environment was deliberately selected to introduce substantial real-world challenges, including strong occlusion from vegetation, heterogeneous backgrounds, and constrained sensor viewpoints. These factors are difficult to fully replicate in

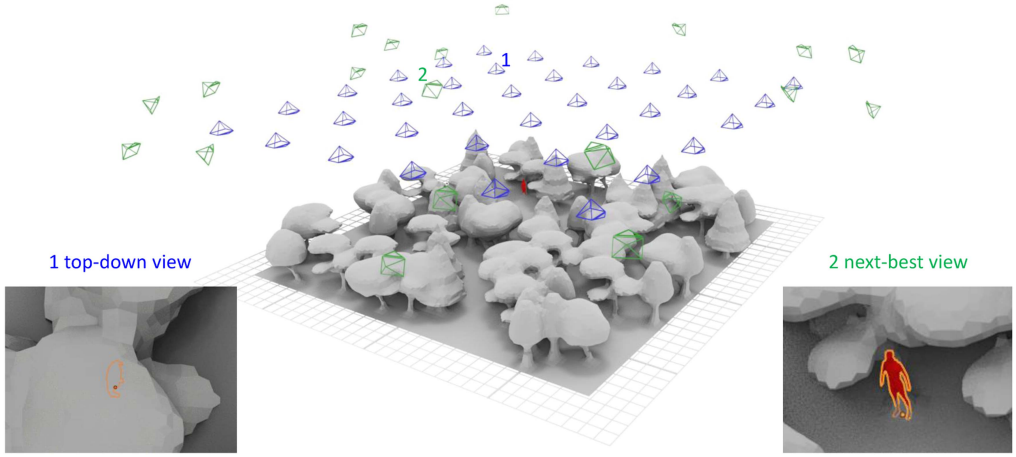

Fig. 8. Example of the outcome of a simulation run. The picture shows the randomized forest scene with one hidden manikin marked in red. The 36 blue cameras represent the initial views, while the 20 green cameras represent planned NBV according to Visibility Fitness. On the left, a section of the top-down camera illustrates how the manikin is obstructed by the trees. In contrast, on the right, one of the planned camera views shows a clear visibility of the manikin.

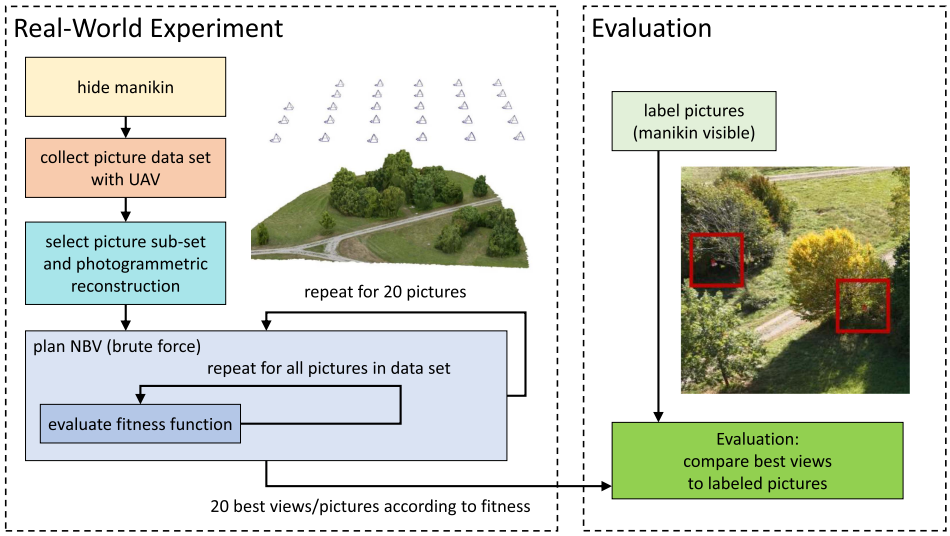

Fig. 9. Workflow of real-world experiments to evaluate the proposed NBV planning.

simulation and degrade the effectiveness of purely top-down search strategies.

Seven tripod-mounted markers wearing high-visibility vests were deployed to emulate the manikins used in the simulation study. The tripods were placed at varying distances and partially concealed within bushes or behind tree trunks, with positions chosen so that no targets were visible from any top-down viewpoint. This placement ensured that successful detection required nontrivial lateral or oblique viewpoints, replicating the partial observability conditions encountered in real SAR missions.

A DJI Mavic 3 Pro UAV was used to acquire a dense dataset of 1032 geotagged and orientation-tagged images from a variety of positions and with varying yaw and pitch angles. Each image was manually annotated to indicate whether a target was visible. From this dataset, a subset of 31 nadir-oriented images positioned on a regular grid were selected to serve as an *initial view set* for photogrammetric 3-D reconstruction of the scene (see Fig. 9).

In contrast to the simulations, we do not run the EA algorithm to find the NBV. Instead, we follow a brute-force approach and evaluate the fitness functions, Visibility Fitness J_v and Geometry Fitness J_d, for all pictures in the dataset.

Using the dataset with prerecorded images is advantageous compared to running the EA online and repeating the full experiment for every parameter change. The data only needs to be collected once, and the performance of different parameters and fitness functions can then be evaluated in postprocessing.

For each NBV iteration, the image with the highest fitness score was selected. In total, 20 NBVs were identified per trial, motivated by battery constraints to ensure completion within the UAV's endurance limit. Capturing each image required approximately one minute, resulting in a total flight time of $\sim$ 20 min, which lies well within the operational range of common UAVs, such as the DJI Mavic 3 Pro.

The coverage results are visualized in Fig. 10, where dark blue indicates regions seen by few or no cameras, and red areas

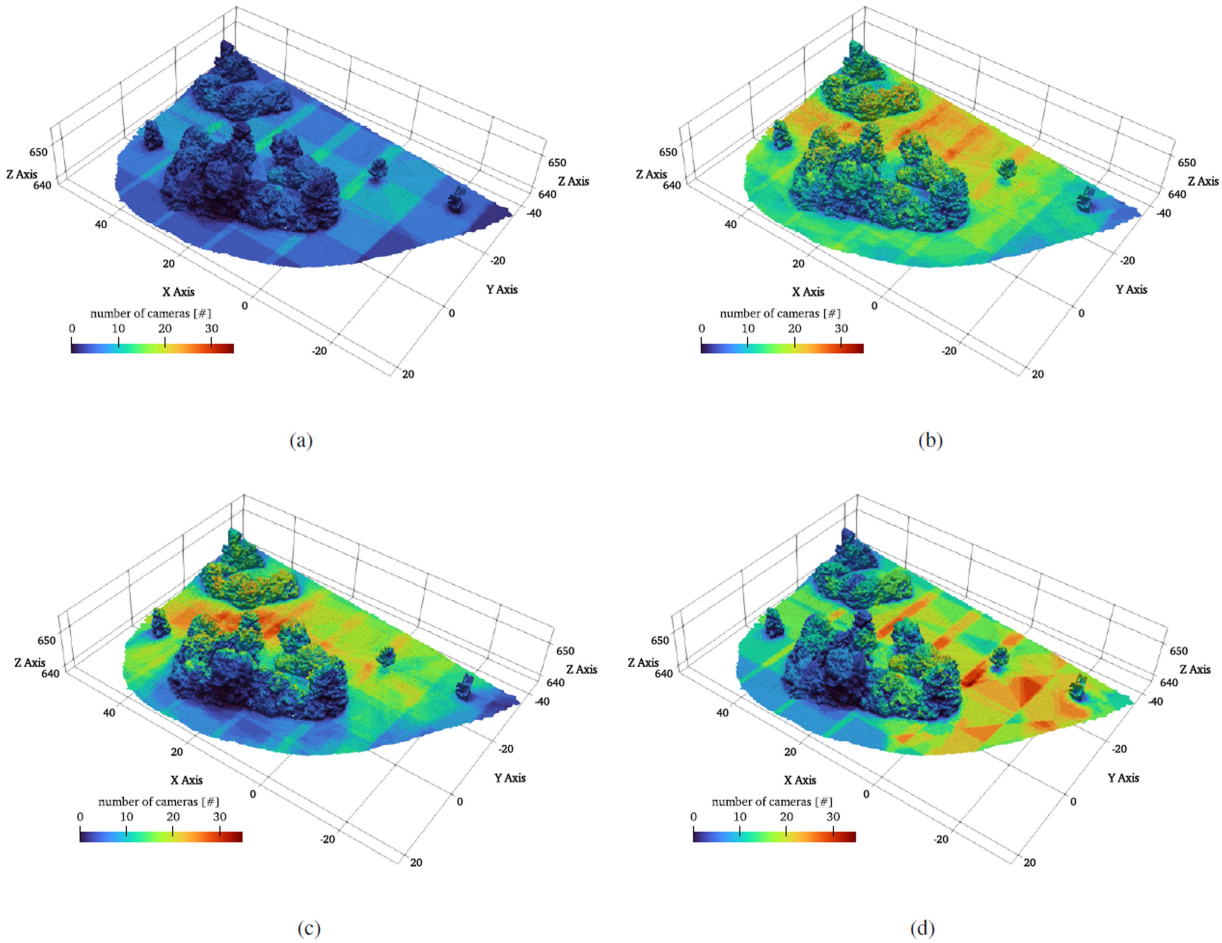

Fig. 10. Change in mesh vertex coverage after adding NBV to the initial camera configuration. (blue: low coverage, red: high coverage). (a) shows the coverage for the initial camera configuration; (b) shows the coverage for 20 NBVs planned based on the Visibility Fitness and (c) for NBV planned based on the Geometry Fitness and (d) for planned with the Agisoft Metashape software.

indicate high visibility from many viewpoints. Compared to the initial view set, both Visibility Fitness J_v and Geometry Fitness J_d improve scene coverage, but with distinct spatial effects. As shown in Fig. 10(b), the Visibility Fitness J_v selection of images produced more uniform coverage improvements, particularly beneath tree canopies and in heavily occluded regions. In contrast, Fig. 10(c) illustrates that the Geometry Fitness J_d, achieved higher peak coverage in open areas, but reduced coverage under vegetation. Notably, the NBV selection process successfully captured five of the seven targets that were occluded from top-down imagery, confirming the method's robustness under severe partial observability. Overall, these results indicate that the Visibility Fitness J_v offers more consistent coverage gain in complex cluttered environments, whereas the Geometry Fitness J_d yields localized improvements in less obstructed regions.

For reference, we also compare against a camera configuration generated by a commercial photogrammetry software package, in addition to our Geometry and Visibility Fitness heuristics. The software used is Agisoft Metashape[2] with the tool *Mission Plan*.

The results in Table I underline the limitations of the initial configuration of 31 cameras, where most vertices are seen by only one or two views and no vertex is observed by more than five. In contrast, adding 20 NBVs markedly improves redundancy, with all three extended configurations yielding a median of seven cameras per vertex and more than 70% of vertices visible from at least five viewpoints. Among these, the Visibility Fitness heuristic achieves the most consistent performance, with nearly complete three-camera coverage (96%) and over 80% of vertices covered by five or more cameras. The Metashape Software-based and Geometry Fitness approaches also provide strong improvements, though with slightly lower redundancy. These findings suggest that while any of the strategies substantially strengthen mesh visibility compared to the baseline, the Visibility Fitness heuristic offers the most balanced tradeoff between widespread coverage and robust redundancy, which is essential for achieving views below canopy or other obstacles.

TABLE I
VERTEX VISIBILITY METRICS ACROSS DIFFERENT CAMERA STRATEGIES

Model	Avg	Median	≥3 (%)	≥5 (%)	≥10 (%)
Initial	1.65	2	6.84	0.00	0.00
Agisoft Metashape	6.63	7	89.02	70.85	20.01
Geometry Fitness	6.72	7	89.04	71.60	21.69
Visibility Fitness	6.96	7	96.04	80.39	20.91

Values are the average/median number of cameras per vertex and the percentage of vertices seen by at least 3, 5, or 10 cameras.

[2][Online]. Available: https://www.agisoft.com/

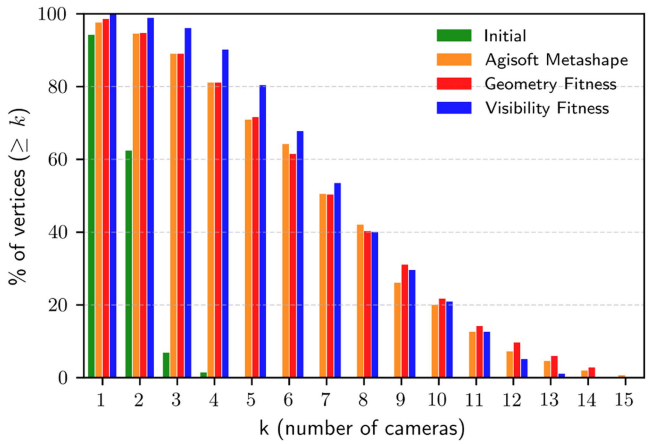

Fig. 11. Cumulative coverage of vertices: for each k, the curves show the percentage of vertices visible from at least k cameras.

The cumulative coverage curves in Fig. 11 provide a more detailed view behind the summary statistics of Table I. Each curve shows, for a given k, the percentage of mesh vertices that are visible from at least k different cameras. This metric captures not just whether a vertex is visible, but how many independent viewpoints, which is particularly relevant when available objects of interest are partially occluded. The initial 31-camera configuration shows a rapid dropoff, with almost no vertices seen by more than three cameras, highlighting its weakness in resolving hidden regions. By contrast, all extended 51-camera strategies significantly increase redundancy, ensuring that most vertices are observed from five or more viewpoints. Among these, the Visibility Fitness heuristic provides the most consistent improvement in the critical range from $k = 3$ to $k = 8$, where its curve dominates over the Agisoft Metashape- and Geometry-based strategies. In practical terms, this indicates a higher likelihood of detecting occluded or hidden persons, since vertices corresponding to them are less likely to disappear from view if multiple independent observations exist.

VI. Conclusion

In this work, we focused on the NBV problem for selecting strategic viewing positions for UAVs, increasing the likelihood of locating missing persons in SAR operations, where visibility is obstructed. We developed and tested two novel heuristics for optimizing cameras' positions and orientations in both real-world and simulated applications. We demonstrated that these heuristics enhance UAV-assisted searches and can be seamlessly integrated into SAR planning. The Visibility Fitness heuristic, J_v, has shown greater scene coverage, identifying over 90% of the hidden manikins on average in simulations, compared to 80% detected by the Geometry Fitness function, J_d. Also in real-world experiments, J_v achieved better area coverage beneath the canopy. In addition, our analysis of cumulative coverage curves has shown that heuristic-based NBV planning not only increases the number of visible targets but also improves the redundancy of viewpoints per target. This redundancy is critical for detecting partially occluded individuals, as it ensures that a person hidden from one camera is still likely to be detected from another. Among the tested strategies, the Visibility Fitness heuristic provided the strongest overall performance.

Despite these advancements, several methodological challenges remain. The computational effort of NBV calculations increases with the number of camera views, which can limit real-time applicability on resource-constrained platforms. While the real-world experiments in this work relied on offline evaluation to enable systematic parameter tuning, future work will focus on integrating the EA-based NBV optimizer directly onboard the UAV to enable real-time, adaptive viewpoint selection during flight in practical deployments. In addition, the current approach assumes static battery constraints and does not dynamically adapt to in-flight conditions. Future work can focus on improving the efficiency of NBV optimization and incorporating battery-aware route planning.

Beyond SAR, the proposed method has potential applications in environmental monitoring of dense vegetation, precision agriculture, and infrastructure inspection under partial occlusions. The approach can also be considered an efficient way to collect image data for emerging AI-driven object detection. Further, real-time adaptive planning frameworks could enhance UAV autonomy and operational effectiveness in challenging environments.

References

[1] A. Bircher, M. Kamel, K. Alexis, H. Oleynikova, and R. Y. Siegwart, "Receding horizon "next-best-view" planner for 3D exploration," in *Proc. IEEE Int. Conf. Robotics. Autom.*, 2016, pp. 1462–1468, doi: 10.1109/ICRA.2016.7487281.

[2] N. Mandischer, M. Gürtler, S. Döbler, M. Hüsing, and B. Corves, "Finding moving operators in firefighting operations based on multi-goal next-best-view exploration," in *Proc. IEEE Int. Symp. Safety, Secur., Rescue Robot.*, 2022, pp. 47–52, doi: 10.1109/SSRR56537.2022.10018673.

[3] M. D. Dunbabin and L. Marques, "Robots for environmental monitoring: Significant advancements and applications," *IEEE Robot. Automat. Mag.*, vol. 19, pp. 24–39, Mar. 2012, doi: 10.1109/MRA.2011.2181683.

[4] M. A. Hsieh et al., "Adaptive teams of autonomous aerial and ground robots for situational awareness," *J. Field Robot.*, vol. 24, no. 11–12, pp. 991–1014, 2007. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.20222

[5] J. A. Sánchez-Rojas, J. A. Arias-Aguilar, H. Takemura, and A. E. Petrilli-Barceló, "Staircase detection, characterization and approach pipeline for search and rescue robots," *Appl. Sci.*, vol. 11, no. 22, 2021, Art. no. 10736, doi: 10.3390/app112210736.

[6] D. S. Drew, "Multi-agent systems for search and rescue applications," *Curr. Robot. Rep.*, vol. 2, no. 2, pp. 189–200, 2021, doi: 10.1007/s43154-021-00048-3.

[7] J. Delmerico et al., "The current state and future outlook of rescue robotics," *J. Field Robot.*, vol. 36, no. 7, pp. 1171–1191, 2019. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.21887

[8] R. Murphy, "Trial by fire [rescue robots]," *IEEE Robot. Automat. Mag.*, vol. 11, no. 3, pp. 50–61, Sep. 2004, doi: 10.1109/MRA.2004.1337826.

[9] R. R. Murphy, *Disaster Robotics*. Cambridge, MA, USA: MIT Press, Feb. 2014, pp. 111–116, doi: 10.7551/mitpress/9407.001.0001.

[10] C. Yuan, Z. Liu, and Y. Zhang, "Uav-based forest fire detection and tracking using image processing techniques," in *Proc. Int Conf. Unmanned Aircr. Syst.*, 2015, pp. 639–643, doi: 10.1109/ICUAS.2015.7152345.

[11] S. H. Alsamhi et al., "UAV computing-assisted search and rescue mission framework for disaster and harsh environment mitigation," *Drones*, vol. 6, no. 7, 2022, Art. no. 154, doi: 10.3390/drones6070154.

[12] J. Wu et al., "An adaptive conversion speed Q-learning algorithm for search and rescue UAV path planning in unknown environments," *IEEE Trans. Veh. Technol.*, vol. 72, no. 12, pp. 15391–15404, Dec. 2023, doi: 10.1109/TVT.2023.3297837.
[13] R. D. G. de Aquino, J. B. Barreto, A. D. N. Rodrigues, F. A. N. Verri, and C. H. Q. Forster, "CNN classification subject to practical aspects of search and rescue UAV missions," in *Proc. IEEE Latin Amer. Conf. Comput. Intell.*, 2023, pp. 1–6, doi: 10.1109/LA-CCI58595.2023.10409392.
[14] A. N. Wilson, A. Kumar, A. Jha, and L. R. Cenkeramaddi, "Embedded sensors, communication technologies, computing platforms and machine learning for UAVs: A review," *IEEE Sensors J.*, vol. 22, no. 3, pp. 1807–1826, Feb. 2022, doi: 10.1109/JSEN.2021.3139124.
[15] R. J. Amala Arokia Nathan and O. Bimber, "Synthetic aperture anomaly imaging for through-foliage target detection," *Remote Sens.*, vol. 15, no. 18, 2023, Art. no. 4369, doi: 10.3390/rs15184369.
[16] A. A. Pramudita et al., "Radar system for detecting respiration vital sign of live victim behind the wall," *IEEE Sensors J.*, vol. 22, no. 15, pp. 14670–14685, Aug. 2022, doi: 10.1109/JSEN.2022.3188165.
[17] A. Moreira, P. Prats-Iraola, M. Younis, G. Krieger, I. Hajnsek, and K. P. Papathanassiou, "A tutorial on synthetic aperture radar," *IEEE Geosci. Remote Sens. Mag.*, vol. 1, no. 1, pp. 6–43, Mar. 2013, doi: 10.1109/MGRS.2013.2248301.
[18] L. M. Wong, C. Dumont, and M. A. Abidi, "Next best view system in a 3 D object modeling task," in *Proc. 1999 IEEE Int. Symp. Comput. Intell. Robot. Autom.*, 1999, pp. 306–311, doi: 10.1109/CIRA.1999.810066.
[19] V. J. Lumelsky, "Effect of robot kinematics on motion planning in unknown environment," in *Proc. 1985 24th IEEE Conf. Decis. Control*, 1985, pp. 338–343.
[20] M. Lauri, J. Pajarinen, J. Peters, and S. Frintrop, "Multi-sensor next-best-view planning as matroid-constrained submodular maximization," *IEEE Robot. Automat. Lett.*, vol. 5, no. 4, pp. 5323–5330, Oct. 2020.
[21] D. Yan, J. Liu, F. Quan, H. Chen, and M.-Y. Fu, "Active implicit object reconstruction using uncertainty-guided next-best-view optimization," *IEEE Robot. Automat. Lett.*, vol. 8, no. 10, pp. 6395–6402, Oct. 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID
[22] C. I. Connolly, "The determination of next best views," in *Proc. IEEE Int. Conf. Robot. Autom.*, 1985, pp. 432–435, vol. 2, doi: 10.1109/ROBOT.1985.1087372.
[23] J. Maver and R. Bajcsy, "Occlusions as a guide for planning the next view," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 15, no. 5, pp. 417–433, May 1993, doi: 10.1109/34.211463.
[24] W. R. Scott, G. Roth, and J.-F. Rivest, "View planning for automated three-dimensional object reconstruction and inspection," *ACM Comput. Surv.*, vol. 35, no. 1, pp. 64–96, 2003, doi: 10.1145/641865.641868.
[25] M. Krainin, B. Curless, and D. Fox, "Autonomous generation of complete 3 D object models using next best view manipulation planning," in *Proc. 2011 IEEE Int. Conf. Robot. Autom.*, 2011, pp. 5031–5037, doi: 10.1109/ICRA.2011.5980429.
[26] S. Kriegel, C. Rink, T. Bodenmüller, and M. Suppa, "Efficient next-best-scan planning for autonomous 3 D surface reconstruction of unknown objects," *J. Real-Time Image Process.*, vol. 10, no. 4, pp. 611–631, 2015, doi: 10.1007/s11554-013-0386-6.
[27] M. Deseilligny, L. Luca, and F. Remondino, "Automated image-based procedures for accurate artifacts 3 D modeling and orthoimage," *Geoinformatics FCE CTU*, vol. 6, pp. 291–299, 2011.
[28] V. V. Fedorov, W. J. Studden, and E. M. Klimko, *Theory of Optimal Experiments*. New York, NY, USA: Academic Press, 1972, p. 57. [Online]. Available: http://lib.ugent.be/catalog/rug01:000474727
[29] V. Fedorov, "Optimal experimental design," *WIREs Comput. Statist.*, vol. 2, no. 5, pp. 581–589, 2010. [Online]. Available: https://wires.onlinelibrary.wiley.com/doi/abs/10.1002/wics.100
[30] T. Luhmann, S. Robson, S. Kyle, and I. Harley, *Close Range Photogrammetry: Principles, Techniques and Applications*. Dunbeath, U.K.: Whittles Publishing, 2011.
[31] S. M. Kay, *Fundamentals of Statistical Signal Processing: Estimation Theory*. Hoboken, NJ, USA: Prentice Hall, 1993, vol. 1.
[32] F.-M. De Rainville, F.-A. Fortin, M.-A. Gardner, M. Parizeau, and C. Gagné, "Deap: A python framework for evolutionary algorithms," in *Proc. 14th Annu. Conf. Companion. Genet. Evol. Comput.*, New York, NY, USA: Assoc. Comput. Machinery, 2012, pp. 85–92, doi: 10.1145/2330784.2330799.
[33] F.-A. Fortin, F.-M. De Rainville, M.-A. Gardner, M. Parizeau, and C. Gagné, "DEAP: Evolutionary algorithms made easy," *J. Mach. Learn. Res.*, vol. 13, pp. 2171–2175, 2012.

Sigrid Helene Strand received the M.Sc. degree in engineering and ICT, with a specialization in geomatics, LiDAR, and remote sensing, from the Norwegian University of Science and Technology (NTNU), Trondheim, Norway, in 2023. She is currently working toward the Doctoral degree in remote sensing with the Swarm Exploration Group, Institute of Communications and Navigation, German Aerospace Center (DLR), Wessling, Germany.

From 2019 to 2021 and from 2022 to 2023, she was a part-time Recruiter and Teaching Assistant with NTNU. In 2020, she did a software development internship with Signicat AS, Trondheim, Noway, in 2021, she was a Research Assistant with Norwegian Mapping Authority (Kartverket), Kristiansand, Norway, and in 2022, she did a geomatics internship with Geodata AS, Oslo, Norway. From 2021 to 2023, she was also a Research Assistant with Norkart AS, Oslo, Norway, under the KartAI project, where she worked on photogrammetric modelling and airborne imagery and LiDAR data processing. .

Thomas Wiedemann received the bachelor's and master's degrees in mechatronics from the Faculty of Mechanical Engineering, Technical University of Munich, Munich, Germany, in 2012 and 2014, respectively, and the Ph.D. degree in computer science from the Center for Applied Autonomous Sensor Systems, Örebro University, Örebro, Sweden, in 2021.

Since 2014, he has been a Scientist with Swarm Exploration Group, Institute of Communications and Navigation, German Aerospace Center (DLR), Wessling, Germany. Since 2023, he has been affiliated as a Postdoctoral Researcher with the Chair "Perception for Intelligent Systems" with the Technical University of Munich. His research interests include multi-robot systems, robotic exploration, and gas source localization.

Bram Burczek was born in Solingen, Germany, on 9 August 2002. He received the B.Sc. degree in computer science from Baden-Wuerttemberg Cooperative State University (DHBW) Mannheim, Mannheim, Germany, in 2023.

As part of his dual study program, he completed his bachelor thesis from Institute of Communications and Navigation, German Aerospace Center (DLR), Wessling, Germany. He is currently a Software Developer in the public sector.

Dmitriy Shutin (Senior Member, IEEE) received the master's degree in computer science from Dniepropetrovsk State University, Dnipro, Ukraine, in 2000, and the Ph.D. degree in electrical engineering from the Graz University of Technology, Graz, Austria, in 2006.

From 2001 to 2006 and 2006 to 2009, he was a Teaching Assistant and an Assistant Professor, respectively, with the Signal Processing and Speech Communication Laboratory, Graz University of Technology. From 2009 to 2011, he was a Research Associate with the Department of Electrical Engineering, Princeton University, Princeton, NJ, USA. Since 2011, he has been with the Institute of Communications and Navigation, German Aerospace Center (DLR), Wessling, Germany, where he is currently a Leader of the Swarm Exploration Group. His research interests include machine learning for signal processing, Bayesian inference, and statistical signal processing over networks.

Dr. Shutin was a recipient of the Best Student Paper Award at the 2005 IEEE International Conference on Information, Communications and Signal Processing (ICICS). In 2009, he was a recipient of the Erwin Schroedinger Research Fellowship. From 2012 to 2014, he was a selected Advisor of German air navigation service provider, within the Navigational System Panel of ICAO. He is a Steering Committee Member of IEEE Autonomous Systems Initiative (ASI).