Robotics and Computer–Integrated Manufacturing 98 (2026) 103149

Contents lists available at ScienceDirect

Robotics and Computer-Integrated Manufacturing

journal homepage: www.elsevier.com/locate/rcim

A novel coverage path planning method based on shrink-wrapping technique for autonomous inspection of complex structures using unmanned aerial vehicle

Burak Kaleci [a], Gulin Elibol Secil [a], Sezgin Secil [a,*], Zühal Kartal [b], Metin Ozkan [c]

[a] Department of Electrical and Electronics Engineering, Eskisehir Osmangazi University, Eskisehir 26480, Türkiye
[b] Department of Industrial Engineering, Eskisehir Technical University, Eskisehir 26555, Türkiye
[c] Department of Computer Engineering, Eskisehir Osmangazi University, Eskisehir 26480, Türkiye

ARTICLE INFO

Keywords:
Inspection
Coverage path planning
Unmanned aerial vehicle
Viewpoints
Set covering problem
Traveling salesman problem

ABSTRACT

The inspection of large-scale structures can be challenging, time-consuming, costly, and dangerous. Autonomous robotic systems can provide an effective solution for performing such tasks by overcoming the negative aspects. In this paper, we present a novel coverage path planning method for complete sensor scanning of the outer surface of complex structures using an unmanned aerial vehicle (UAV) with a depth camera. The proposed method introduces a new approach by applying the shrink-wrapping technique to construct a 3D triangular mesh representing the structure's surface boundary. Viewpoints are then generated based on this mesh. Additionally, the triangles within the depth camera's field of view for each viewpoint are determined. The set covering problem (SCP) accepts the set of triangles covered by each viewpoint and reduces the number of viewpoints to decrease the flight distance and time. Finally, the coverage route that includes all the selected viewpoints is defined as the solution to the traveling salesman problem (TSP). We conduct extensive experiments to demonstrate the effectiveness of the proposed method across three different large-scale structures. The results show the validity and effectiveness of the proposed method.

1. Introduction

In recent years, there has been an increase in popularity for studies on the inspection of structures such as bridges, steel buildings, and dams in construction sectors; quality control of large-scale vehicles such as ships and aircraft produced in shipyard-type production industries and medium-scale vehicles such as trucks, buses, and wagons produced by industrial production. Fig. 1 shows some samples of these structures and vehicles. For safety, structures and vehicles must be inspected during their construction and also periodically during their use. Checking whether these structures and vehicles are manufactured by their design conformance quality is another important area of study [1].

Human operators usually inspect whether the thousands of parts of various sizes are assembled correctly to construct the structures and vehicles. In this process, human operators visually check all the parts and decide on their suitability. Inspection tasks carried out by human operators are processes that take a long time, are costly, and carry risks to human health. Although there have been significant developments in occupational safety and health in recent years, unfortunately, injuries and deaths occur due to reasons such as human operators having to check hard-to-reach places. In addition, the standardization of quality inspection becomes difficult because operators cannot reach some areas, and the inspections made by eye or physical contact are open to factors such as human errors, and this significantly negatively affects the quality and duration of the inspection [2].

In order to overcome these negative situations caused by human operators in the inspection of large-scale structures and vehicles, studies on the use of robots in inspection tasks have gained importance. In the studies conducted for the inspection of bridges [9], buildings [10], aircraft [11], and ships [12], unmanned aerial vehicles (UAVs) are generally used due to their prominent features such as high mobility and ability to work in hard-to-reach places. In particular, rotary wing UAVs are preferred because they have advantages such as vertical flight capabilities including vertical takeoff and landing, hovering, and point rotation capabilities [13].

The surface inspection of the large-scale structures is implemented

* Corresponding author.
E-mail address: ssecil@ogu.edu.tr (S. Secil).

https://doi.org/10.1016/j.rcim.2025.103149

Available online 18 October 2025

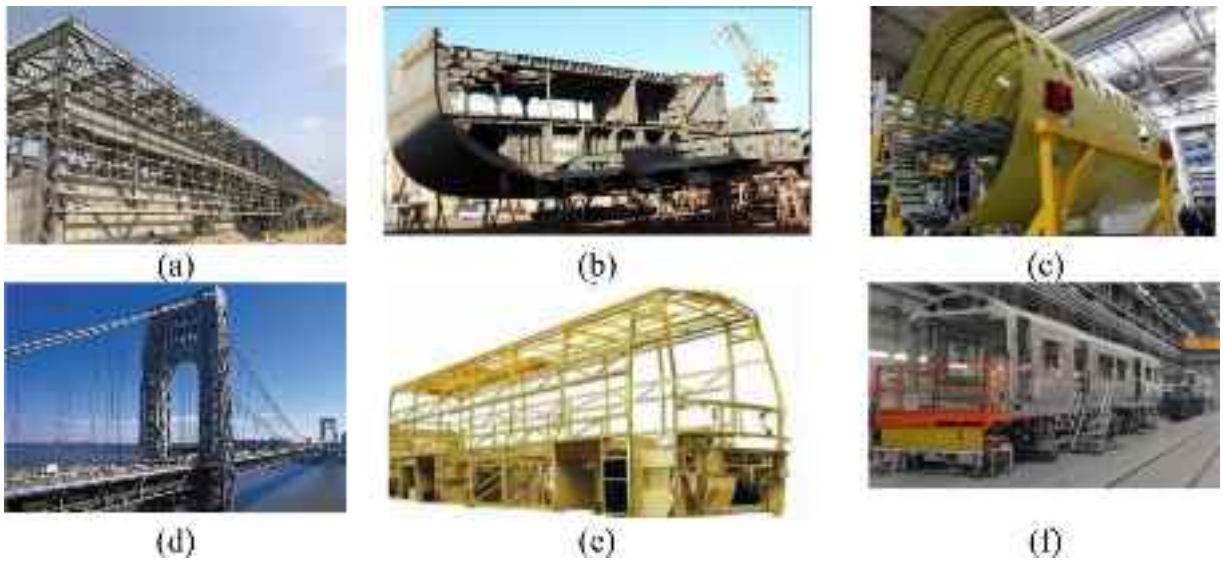

Fig. 1. Various structures and vehicles: (a) Steel building [3], (b) ship [4], (c) plane [5], (d) bridge [6], (e) bus [7], (f) wagon [8].

based on the data collection with the sensors carried by UAVs. The complete coverage of the structure surface by optimizing the required objectives, such as minimum flight time, traveled distance, etc., is referred to as the coverage path planning problem (CPP) [14]. The CPP is a process of determining a suitable path that includes a set of waypoints, some of which can be viewpoints, that the UAV must track to scan the surface of the structure completely. The CPP is divided into two categories in the literature: model-based and sensor-based [15]. Similarly, model-based approaches are generally referred to as offline methods, while sensor-based approaches are referred to as online methods [16,17].

Sensor-based approaches are commonly employed to explore unknown environments or to scan objects by implementing next-best view algorithms. While these methods yield high-quality and detailed reconstructions, they are typically better suited for indoor settings and smaller objects. Their application to large-scale outdoor environments is often constrained by significant computational complexity. Furthermore, these approaches may fail to ensure complete coverage due to the local optimum issue [18]. Conversely, model-based approaches rely on prior information, such as computer-aided design (CAD) models of structures and parts or building information modeling (BIM) for structures like bridges and buildings, to plan the coverage path. These approaches, in turn, offer the advantage of guaranteeing full coverage and deriving an optimal path based on predefined criteria. In this study, we have chosen to adopt model-based approaches to achieve our objective of inspecting large-scale structures and vehicles while ensuring optimal path planning.

Studies on model-based coverage path planning generally consist of two steps [19–21]. These steps include viewpoint generation and path planning. The process of viewpoint generation focuses on determining the appropriate sensor locations at which a UAV can collect data to achieve coverage of the entire structure or specific areas of interest. Following the generation of viewpoints, the subsequent and critical step is path planning. The primary objective of this step is to design a continuous path that traverses all predetermined viewpoints while minimizing associated costs, such as travel distance, travel time, or energy consumption.

In this paper, we propose a model-based coverage path planning method for complete sensor scanning of the outer surface of complex structures using a UAV. The proposed method leverages the 3D mesh model of the structure and a depth camera mounted on a UAV. It introduces a novel approach by applying the shrink-wrapping technique to construct a 3D triangular mesh representing the structure's surface boundary. The viewpoints are generated by applying a series of operations on the 3D triangular mesh of surface boundary. Initial viewpoints are determined using the center and normal of each triangle in the triangular mesh model. However, as the depth sensor on the UAV is constrained to maintain a focal line parallel to the horizontal X-Y plane, the initial viewpoints may not align with the sensor's orientation. To address this, the initial viewpoints are transformed into feasible viewpoints that are compatible with the sensor constraints. Additionally, triangles within the depth camera's field of view (FOV) are identified for each viewpoint. Due to the sensor's FOV, more than one triangle can be covered within the FOV from a single viewpoint, or identical triangles can be encompassed from different viewpoints. Therefore, the redundant viewpoints can be eliminated. The problem of selecting the minimal set of viewpoints required to cover all triangles is formulated as a set covering problem (SCP), aiming to reduce flight distance and time. Subsequently, the optimal coverage path that traverses all remaining viewpoints is computed by solving the traveling salesman problem (TSP), with the cost function based on Euclidean distance. To ensure collision-free navigation, potential collisions between viewpoints are detected using the 3D triangular mesh, and a high penalty cost is assigned to such connections to prevent their inclusion in the path.

Extensive experiments were conducted on three different structures to demonstrate the effectiveness of the proposed method. First, hyperparameter optimization for the shrink-wrapping technique was performed across these structures. For each parameter combination, SCP and TSP solutions were computed, and the shortest paths were identified. For the shortest paths, the 3D triangular mesh of each structure was scanned and evaluated based on the number of points and the Uniformity Index (UI). Then, we scanned the model and generated the distance map for each structure. The experimental results demonstrated the method's effectiveness in achieving efficient and uniform coverage of the 3D models.

The main contributions of this paper include:

- In order to determine viewpoints for complex structures, we propose a novel approach called surface boundary cover obtained by applying the shrink-wrapping technique.
- In complex structures where internal and external scanning is required but it is difficult to distinguish between internal and external, the bounding surface approach is proposed. Thus, different coverage path planning methods are applicable for internal and external scanning.

- A new viewpoint generation method is proposed for the sensor coverage of the structures with highly challenging surfaces from the outside. The method considers the constraints on the FOV of the sensor and the motion of the UAV.
- During the viewpoint generation process, we identify a set of triangles for each viewpoint by determining the surfaces of triangles visible within the given viewpoint. These sets of triangles are then utilized as inputs for the SCP. The solution of this problem ensures that the selected viewpoints provide complete coverage of all visible triangles.
- The standard TSP is adapted to account for potential collisions between viewpoints.
- A detailed analysis is provided to reveal the effects of some manually assigned parameters in the method on the coverage performance through travel distance, number of viewpoints, and coverage rate. The effectiveness and applicability of the proposed method are demonstrated with 3D simulations based on a real sensor, a UAV, and different structure models.

The paper is organized as follows: Section 2 reviews related works on model-based coverage path planning. The proposed method and the viewpoint generation are presented in Section 3. The coverage path planning is given in Section 4. The simulated experiments and results to verify the proposed method are presented in Section 5. Finally, our conclusions and proposed future work are presented in Section 6.

2. Related works

Studies on model-based coverage path planning generally consist of two steps: viewpoint generation and path planning. Therefore, we organized this section under two topics: (1) viewpoint generation in model-based approaches and (2) coverage path planning.

2.1. Viewpoint generation

Previous studies that addressed model-based coverage path planning employed different methods, such as sampling-based, voxel-based, and triangular mesh-based, to determine the viewpoints. Englot and Hover [22] performed a sampling through a polygonal mesh model of a ship's hull. In this way, a set of fixed views that provide full coverage on the hull was obtained. Then, a road map containing viewpoints was created by solving the set cover problem. Dornhege et al. [23] used the OctoMap map in a sampling-based manner to obtain candidate viewpoints and determine candidate points with the help of the ray tracing function. Then, the solution to the set coverage problem was used to create the set of viewpoints. In the manufacturing domain, Li et al. [24] presented a sampling based discretization of CAD surfaces to generate inspection poses. An oriented bounding box was constructed and a line triangle intersection procedure was applied to obtain a representative surface point set, from which feasible scanner poses were computed from surface normals subject to kinematic constraints, and candidate viewpoints were subsequently obtained. In a robotic inspection study, Glorieux et al. [25] replaced random sampling with a targeted viewpoint-sampling strategy that maximizes coverage of yet-unseen primitives while penalizing travel time; the final set was selected via set covering. In the study conducted by Jung et al. [19], a volumetric map was used to determine viewpoints. According to the method, a set of normal vectors was created using the centers of the voxels in the volumetric map. Then, the vectors perpendicular to these normal vectors were determined as viewpoints. Due to the formation of too many viewpoints and overlapping surfaces, the obtained viewpoints were then subjected to subsampling with a voxel grid filter. In the recent study conducted by Yan et al. [18], a different type of approach was followed to determine viewpoints. In this study, to reduce computational complexity, the gaze sampling area was divided into voxels of equal size. Then, the centers of these voxels were considered candidate gaze points. These candidate gaze points were also grouped as local gaze points that rescan missing and low-quality regions and global gaze points that serve for overall coverage. Similarly, in the study conducted by Almadhoun et al. [26], a grid resolution adjusted according to the relevant structure model was used and the study area where the structure is located was discretized. Then, clusters were created in these discrete areas where sample road points and corresponding sensor gaze points were sampled. Bircher et al. [27] used the triangular mesh model of the structure and a viewpoint was sampled for each triangle in the network structure. Then, the best configurations among these viewpoints were determined by solving an Art Gallery Problem (AGP). In another study, Bircher et al. [28] sampled a viewpoint for each triangular network. However, in this study, instead of finding a minimal set that provides full coverage, the aim was to find examples that include shorter paths. Therefore, an iterative resampling scheme was used instead of solving an AGP. In mobile visual inspection with a mobile manipulator, Kong et al. [29] uniformly discretized the surface into triangular meshes and took triangle centers as samples; candidate viewpoints were evaluated with a visibility and image-quality metric, and an alternating evolution genetic algorithm (GA) selected a compact set of high-quality viewpoints. Xi et al. [30] generated candidate measurement poses on polyhedral surfaces by flattening the triangular surface mesh to a 2D parameter domain, distributing curvature-oriented sampling points, mapping them back onto the 3D surface to define measurement points, and computing feasible inspection poses via a one-step inverse approach subject to kinematic constraints. Claro et al. [31] generated viewpoints based on the characteristics of the sensors used for inspection and the geometry of the structure. They defined a projected area representing the region where the sensor collects data. The model's surface was then discretized using these projected areas. Candidate viewpoints were determined based on the sensor points corresponding to these projections. Additionally, the maximum navigation speed of the UAV served as a final constraint for the selection of viewpoints. Huang et al. [32] converted the BIM representation of the structure into point cloud data and proposed a method to extract the overall building geometry as a surface model. For each point in the point cloud, a viewpoint was calculated based on the point's normal vector, ensuring orientation towards the building's exterior. They then implemented a sequence of operations to eliminate occluded viewpoints and identify collision-free viewpoints. Finally, a rule-based approach was employed to determine the final set of viewpoints.

2.2. Path planning

The main goal in path planning is generally expressed as creating a continuous path passing through all viewpoints with the lowest cost such as minimum travelled distance, minimum travelled time, or minimum energy. Although previous studies that addressed model-based coverage path planning employed different methods to determine the viewpoints, they have mainly considered the TSP for path planning. These studies, however, incorporated diverse cost functions and approaches to address the TSP. For instance, Englot and Hover [22] applied the Euclidean distance and the chained Lin-Kernighan heuristic (LKH) [33]. They adjusted the costs for certain viewpoints due to their use of back-and-forth sweep paths, where some viewpoints were already sequenced. Xi et al. [30] solved a TSP in 3D with a genetic algorithm to minimize total travel distance when sequencing the sampled measurement points along the inspection path. Glorieux et al. [25] performed sequencing with an iterative TSP, where the edge cost between any two viewpoints was computed on demand by a collision free path planner; the objective was to minimize cycle time (travel plus measurement), yielding notable reductions on industrial parts. In a visual inspection setting with multiple mobile-base configurations, Kong et al. [29] formulated a clustered TSP and solved it with a GA to minimize overall path length by combining inter-station and inter-viewpoint costs. Dornhege et al. [23] utilized an elevation map to differentiate critical structural features like stairs and ramps, as their robot operates in 3D

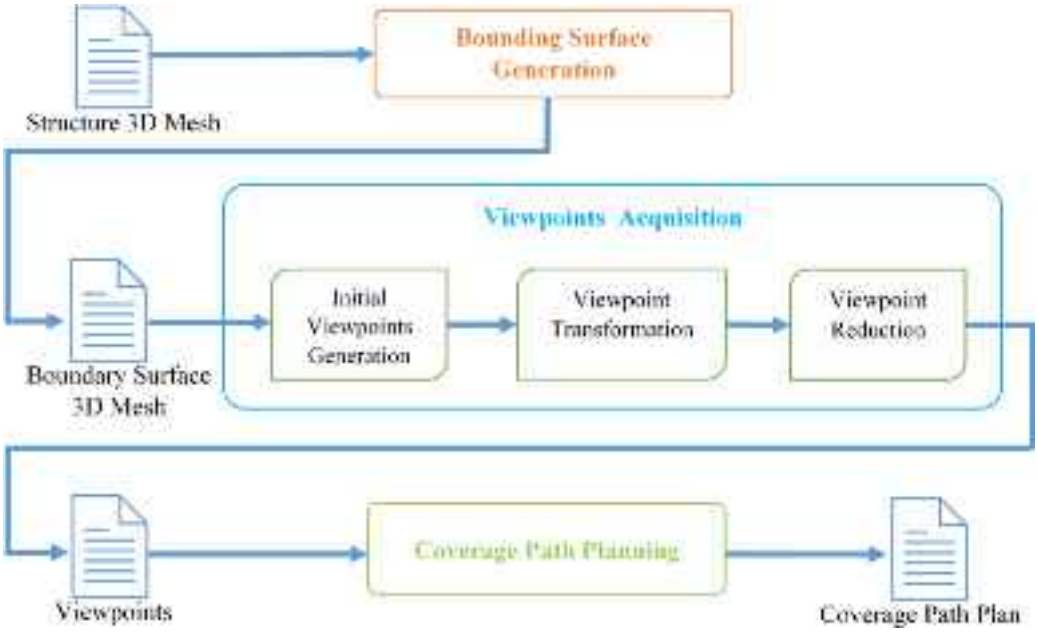

Fig. 2. Framework for the proposed method.

and they employed OctoMap to identify viewpoints. Then, they calculated the cost between pairs of viewpoints using an A* planner on the elevation map. To solve the TSP, they applied the LKH [34]. Jung et al. [19] implemented a multi-layer approach for path planning, utilizing two distinct cost functions based on Euclidean distance. In the first layer, they solved the TSP using the LKH solver and updated the viewpoints in the second layer by examining overlapping areas. This approach effectively minimizes redundant voxels and results in a more efficient tour path. In the context of 5-axis on-machine inspection, Li et al. [35] proposed an orientation-point (O-P) relation schema and two complementary algorithms, namely the minimum-orientation algorithm (MOA) and the shortest-path algorithm (SPA), which jointly optimize the number of probe orientations and the inspection tour length using a TSP solver based on LKH while enforcing collision free motion with a safe box model. In another study, Yan et al. [18] formulated a cost function that accounts for both the short distance between viewpoints and minimal camera orientation changes switching between viewpoints. They employed a standard TSP framework and utilized the Ant Colony Optimization (ACO) algorithm [36] to solve it. Bircher et al. [27,28] used the LKH TSP solver to determine the optimal tour for viewpoint sampling, incorporating factors such as the heading, height, and distance between adjacent viewpoints. Claro et al. [31] presented a novel formulation of the TSP incorporating a UAV energy model. This formulation, termed Asymmetric Traveling Salesman Problem with Precedence Loss (ATSP-PL), accounted for movement costs influenced by the UAV's preceding position. The energy model established a relationship between each UAV movement and its energy consumption, while the path planning algorithm was designed to minimize the UAV's overall energy loss. Huang et al. [32] utilized an alternative cost function to account for potential collisions with obstacles. To compute the distance between pairs of viewpoints, they first assessed whether the line of sight between them was obstructed by buildings or barriers. If unobstructed, the Euclidean distance was used; otherwise, the shortest path was determined using a 3D A* algorithm [37]. This algorithm operated on a 3D map constructed from cubic nodes that represented the buildings and their surrounding environments.

3. Proposed method

This section describes our model-based coverage path planning method designed for comprehensive sensor scanning of the outer surfaces of complex structures using a UAV. The primary objective is to devise the shortest path that ensures complete coverage of the structure's exterior while accounting for the sensor's FOV constraints and the UAV's motion limitations. The proposed coverage path planning method comprises three steps: 1) Bounding surface generation, 2) viewpoint acquisition, and 3) coverage path planning. Each step is described in detail in the following subsections. The framework for the proposed method is given in Fig. 2.

The proposed method is carried out in three stages as seen in Fig. 2. In the first stage, the 3D complete model of the structure represented by a triangle mesh is used as input. The structure boundary cover is obtained in the form of a 3D triangle mesh, which constitutes the basis for generating the viewpoints for sensor scanning. At the same time, the boundary cover provides the opportunity to distinguish between the interior and exterior of the structure and to use different strategies for sensor scanning inside and outside.

In the second stage, the viewpoints are obtained by applying a series of operations. First, the initial viewpoints are obtained by using the center and normal of each triangle in the triangular mesh model of the boundary cover. Then, each viewpoint is transformed by considering the orientation of the initial viewpoints and the orientational constraint of the sensor. Thus, the viewpoints are replaced by new transformed viewpoints. These viewpoints generated based on the triangles of the boundary cover may be redundant. For each viewpoint, more than one triangle may be in the FOV of the sensor. Therefore, the triangles seen from each viewpoint are listed for the related viewpoint. The viewpoints and related lists including the visible triangles are modelled as a set coverage problem. The solution of the set coverage problem reduces the number of viewpoints.

Finally, path planning for the viewpoints is formulated as the TSP. Solving the TSP, the coverage path plan is obtained.

3.1. Bounding surface generation

3D alpha wrapping is among the most effective techniques for generating a triangle mesh that accurately represents the structure's boundary cover. The 3D alpha wrapping algorithm is proposed by Portaneri et al. [38] to generate valid wrapped surface meshes from 3D point clouds or meshes. The algorithm takes an input 3D geometry such as point sets or a triangle mesh, which can have defects like gaps, missing data, self-interaction, degeneration, and non-manifold features,

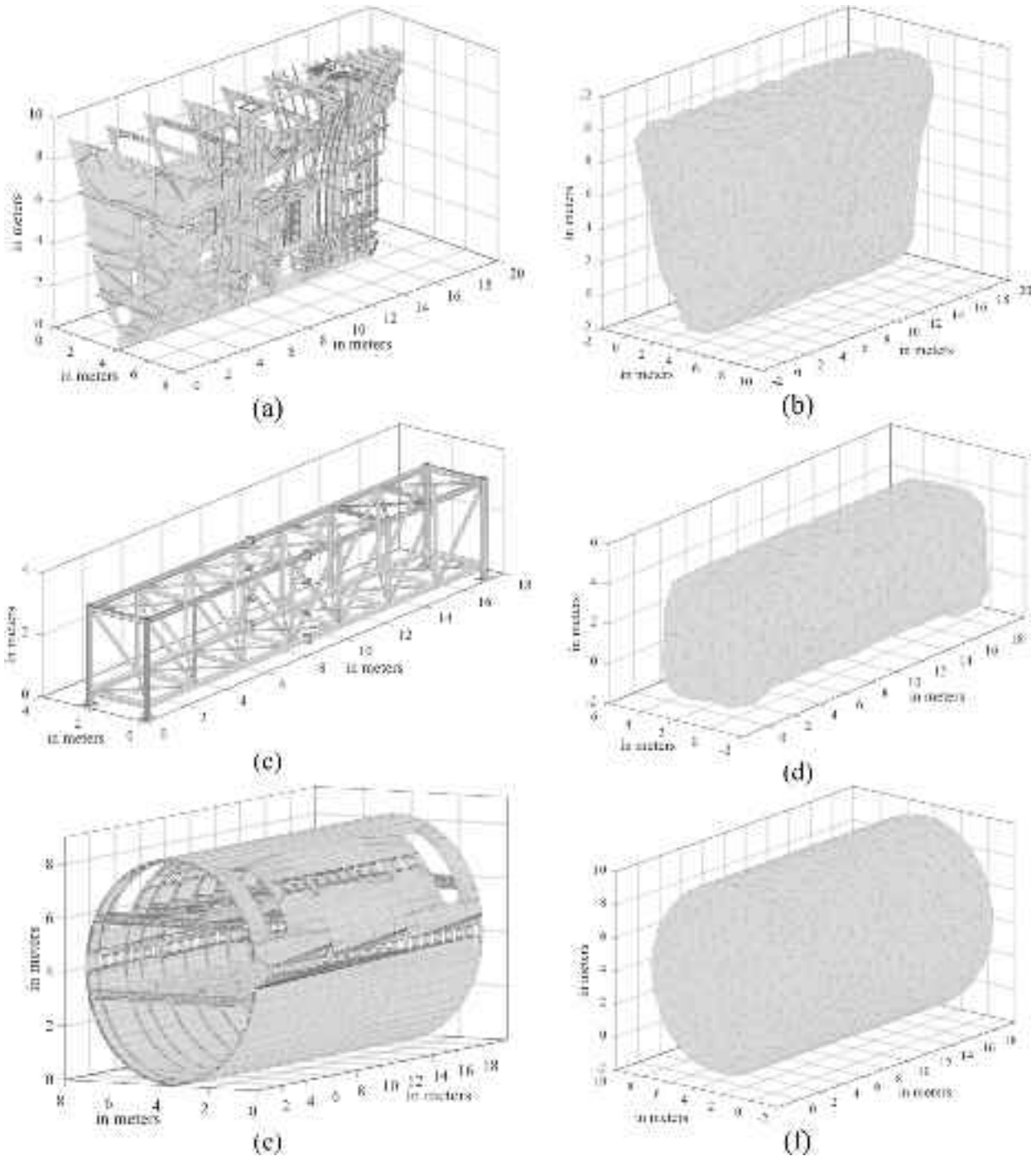

Fig. 3. Illustration of the outputs of the 3D alpha wrapping algorithm for some structures taken from GrabCAD: (a)-(b) Original input model and output model of a destroyer bow (H: 9.79867 m, W: 17.94920 m, D: 7.96925 m), (c)-(d) Original input model and output model of a pedestrian bridge (H:3.31500 m, W: 17.50020 m, D: 2.56000 m), (e)-(f) Original input model and output model of a fuselage (H: 8.39360 m, W: 18.19000 m, D: 7.15000 m).

and generates a watertight and orientable surface triangle mesh that strictly encloses the input. The idea behind the algorithm bridges the gap between a number of existing approaches by combining alpha shapes, mesh carving, and Delaunay-based meshing with the purpose of creating an enclosing mesh with an adjustable margin.

The algorithm produces the output triangular mesh through a two-step process. First, it computes the Delaunay triangulation on an offset surface derived from the input. Then, this triangulation is greedily refined and carved using empty balls with a radius defined by the parameter alpha. Thus, the level of detail and complexity of the resulting triangular mesh can be controlled via two user-defined parameters: alpha and offset. The parameter alpha determines the maximum size of cavities or holes that can be traversed during the carving process, while offset specifies the distance between the vertices of the output mesh and the original input.

Fig. 3 illustrates three different structural mesh models alongside their corresponding 3D triangular meshes obtained utilizing the 3D wrapping algorithm. The first column depicts the original mesh models of the structures, which, as shown, consist of hundreds of components and exhibit intricate surface geometries. These structures lack regular geometric features, making it challenging to propose a formalized approach for sensor scanning using a UAV. Nevertheless, the 3D

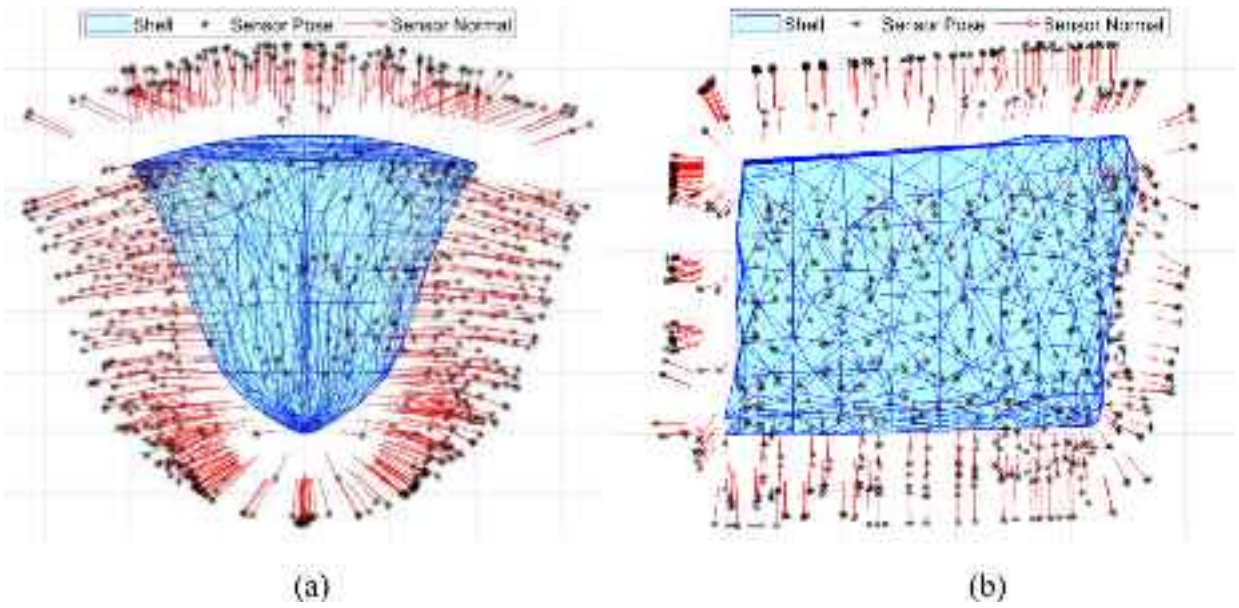

Fig. 4. Position (with black colored star) and orientation (with red colored arrows) of the initial viewpoints for destroyer bow whose boundary surface mesh is presented in blue colored: (a) Front view, (b) Side view.

triangular meshes, as shown in the second column, effectively simplify these complex structures into representations suitable for UAV coverage path planning, enabling comprehensive scanning of their external surfaces.

3.2. Viewpoints acquisition

Based on the boundary surface cover, the viewpoints are generated by subsequent operations. As a starting point, initial viewpoints are obtained. Then, a series of operations are performed to refine and improve the viewpoints to achieve the minimum number of viewpoints for the best coverage.

3.2.1. Initial viewpoints

The set of triangles of the boundary surface mesh is used to calculate the initial viewpoints. The position of a viewpoint for a triangle is determined as the point shifted by the distance d_s from the centroid of the triangle throughout the surface normal vector of the same triangle. The orientation of the viewpoint is determined as the inverse of the surface normal vector. d_s is determined by considering the range of the depth camera sensor and calculated by

$$d_s = \frac{d_{max} - d_{min}}{2} + d_{min} \tag{1}$$

where d_{max} and d_{min} are the maximum and minimum measurable (or the range for the best accuracy) depth distance of the sensor.

Fig. 4 shows the position and orientation of the initial viewpoints for the destroyer bow seen in Fig. 3(a)-(b). The triangles of the boundary surface mesh are shown in blue.

The number of initial viewpoints are equal to the total triangles of the mesh. Let m_t be a total number of triangles in the boundary surface mesh. Then, the number of initial viewpoints m_v will be equal to m_t. The vertices of the triangles can be represented by $P_v^i = \{v_1^i, v_2^i, v_3^i\},\ v_j^i \in \mathbb{R}^3$ and $j \in [1,3]$ where P_v^i is the set of vertices of i^{th} triangle and v_j^i is the 3D position of vertices of i^{th} triangle.

i^{th} triangle of the boundary surface mesh has a normal vector n^i and a centroid c^i, which are calculated by

$$n^i = \frac{(v_2^i - v_1^i) \times (v_3^i - v_1^i)}{\| (v_2^i - v_1^i) \times (v_3^i - v_1^i) \|} \tag{2}$$

$$c^i = \frac{1}{3}\sum_{j=1}^{3} v_j^i \tag{3}$$

The initial viewpoints $V_I^i = \{{}^I p_v^i, {}^I o_v^i\}$, $i \in [1, m_v]$ can be calculated by

$${}^I p_v^i = c^i + d_s \cdot n^i \tag{4}$$

$${}^I o_v^i = -n^i \tag{5}$$

where ${}^I p_v^i$ and ${}^I o_v^i$ are the position and orientation of i^{th} initial viewpoint, respectively.

3.2.2. Transformation of viewpoints

An analysis of the orientations of the viewpoints reveals that certain viewpoints are inaccessible due to the constraints of the UAV and the mounted depth camera. Specifically, some viewpoints are unreachable by the UAV as they are positioned beneath the structure. Furthermore, the depth camera, which is mounted with its viewing direction aligned parallel to the horizontal X-Y plane, is unable to access viewpoints oriented at certain angles relative to the horizontal plane. Consequently, the corresponding triangle cannot be scanned. To address this limitation, the viewpoints are systematically evaluated based on their orientations, and alternative viewpoints are computed to replace those that are incompatible with the constraints of the depth camera.

Fig. 5 illustrates the calculation technique of the transformed viewpoints from the initial viewpoints. $\mathscr{V}^i = \{p_v^i, o_v^i\}$, $i \in [1, m_v]$ is the transformed viewpoint as a substitute for $\mathscr{V}_I^i$. θ_z^i is the angle from the initial viewpoint to the vertical z-axis. θ_z^i is also the slope between the face of the triangle and the x-y plane. If $0 \le \theta_z^i < \pi/2$, then the face of the triangle is facing up in $+z$ direction. If $\pi/2 < \theta_z^i \le \pi$, then the face of the triangle is facing down in -z direction. When $\theta_z^i = \pi/2$, the face of the triangle orients parallel to the x-y plane. The sensor orientation should always be parallel to the horizontal x-y plane such that the triangle face is covered by the sensor FOV. The virtual line $\hat{z}$ is parallel to the axis z, which passes through the centroid of the triangle c^i. The perpendicular distance between the lines $\hat{z}$ and z is d_s. $\hat{c}^i$ is the point at which the line $\hat{z}$ intersects the x-y plane. The projection of $\mathscr{V}_I^i$ onto the x-y plane overlaps the line $\overline{\hat{c}^i c^i}$.

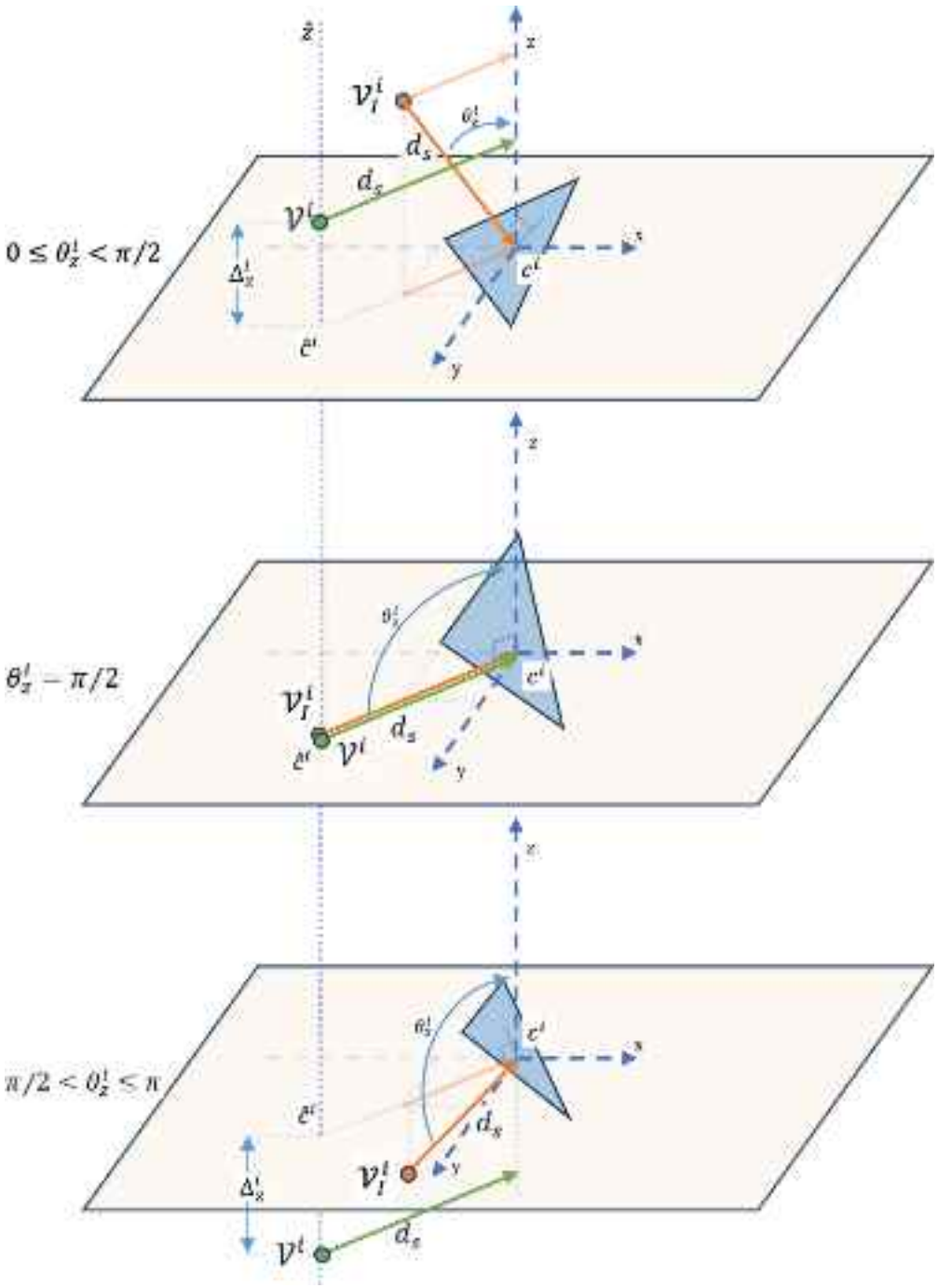

Fig. 5. Illustration for viewpoint transformation.

Assume that all of the points shown in Fig. 5 are defined with respect to a common coordinate frame. The direction of each coordinate axis (x, y, and z) coincides with that of the corresponding axis of the common coordinate frame. The angle θ_z^i can be calculated as the angle between two vectors. One of the vector is directed from the centroid of the triangle to the initial viewpoints, which is $-o_v^i$. The other vector has the direction to $+z$. Thus, the angle θ_z^i becomes

$$\theta_z^i = \cos^{-1}\left(\frac{-o_v^i \cdot [\,0 \quad 0 \quad 1\,]^T}{|o_v^i|}\right) \tag{6}$$

The projection of ${}^I p_v^i$ onto the x-y plane can be calculated by

$${}^I\widehat{p}_v^i = {}^I p_v^i - \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \cdot d_s \cdot \cos\theta_z^i \tag{7}$$

Thus, by substituting (7), the orientation vector of the viewpoint $\mathscr{V}^i$ becomes

$$o_v^i = \frac{c_i - {}^I\widehat{p}_v^i}{\| \, c_i - {}^I\widehat{p}_v^i \, \|} \tag{8}$$

By substituting (8), the projection of p_v^i, which is the position of the viewpoint $\mathscr{V}^i$, becomes

$$\widehat{c}^i = c_i - o_v^i \cdot d_s \tag{9}$$

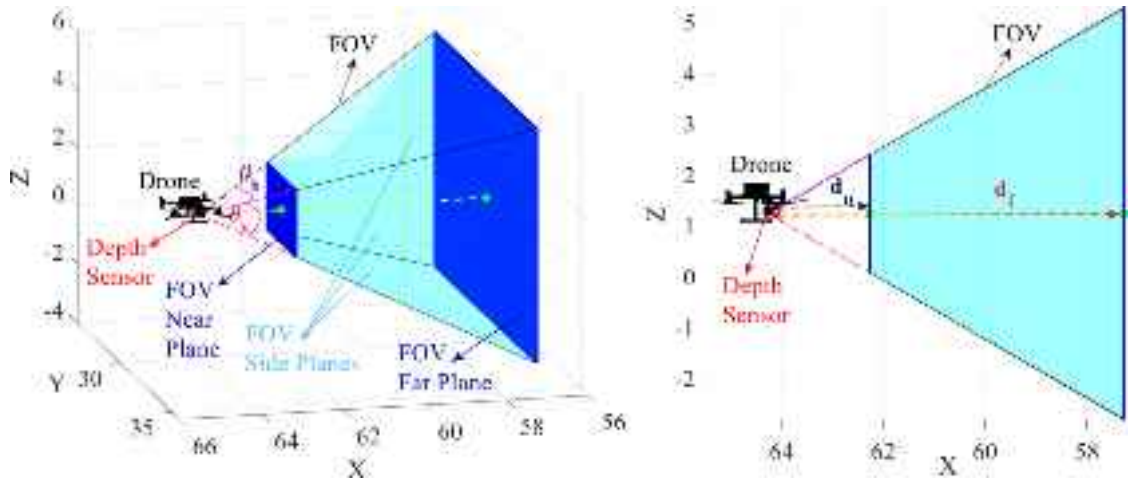

Fig. 6. Illustration of the FOV Model of a depth camera.

The position of the viewpoint $\mathscr{V}^i$ becomes

$$p_v^i = \hat{c}^i + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \cdot \Delta_z^i \tag{10}$$

where Δ_z^i is a scalar that takes value according to θ_z^i. Let θ_v be the vertical angle of the depth camera. Then, it is calculated by

$$\Delta_z^i = d_s.\tan\left(\left\{\frac{\pi - 2\theta_z^i}{\pi}\right\}\frac{\theta_v}{\alpha}\right) \tag{11}$$

where $\alpha \geq 2$ is any scalar. For $\alpha = 2$, the centroid of the triangle c_i may lie at the lower or upper limits of the FOV depending on the valuation of Δ_z^i.

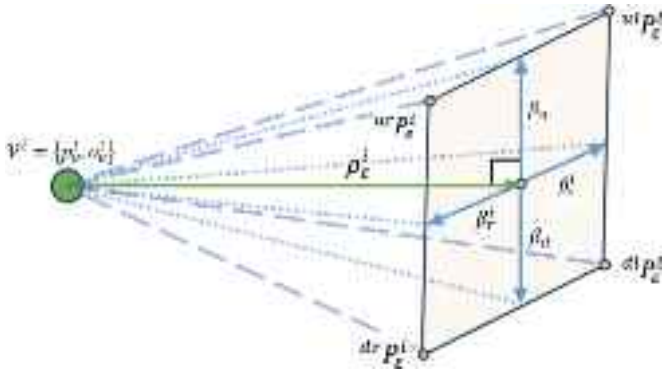

Fig. 7. The parameters of the FOV Model.

3.3. Determination of triangles covered by viewpoints based on the FOV

The transformed viewpoints, $\mathscr{V}^i = \{p_v^i, o_v^i\}, i \in [1, m_v]$ are calculated considering the depth camera's constraints. Each viewpoint is designed to encompass at least the triangle associated with it. However, due to the FOV of the depth camera, adjacent triangles may also fall within the coverage of the camera when positioned at a given viewpoint. As a result, numerous viewpoints may be redundant, and these can be eliminated to optimize scanning efficiency by reducing the required time and traversal distance. In this subsection, the FOV of the depth camera is modeled first. Then, the determination of triangles covered by the FOV on viewpoints is explained.

3.3.1. Depth camera FOV model

The FOV model of the depth camera is calculated using some sensor parameters. θ_v and θ_h are the vertical and horizontal FOV of the depth camera, respectively. The depth cameras generally have the minimum and maximum measurable depth distances, which can be called the near and far plane. Thus, d_n and d_f are the perpendicular distances of the near and far plane to the depth camera frame.

As seen in Fig. 6, the FOV of the depth camera is in the form of a trapezoidal prism. The locations of the corner points of the prism can be calculated relative to the depth camera at the viewpoints. It is assumed that the z-axis is vertically upward, and the orientations of the viewpoints are always parallel to the horizontal $x - y$ plane.

Assume that the subscript symbol $\varepsilon \in \{n, f\}$ represents n and f for the near and far planes of the FOV model. In order to explain the calculation of the corner points, Fig. 7 shows the representation of parameters for both the near and far planes relative to the viewpoints. The vectors shown in Fig. 7 are calculated as

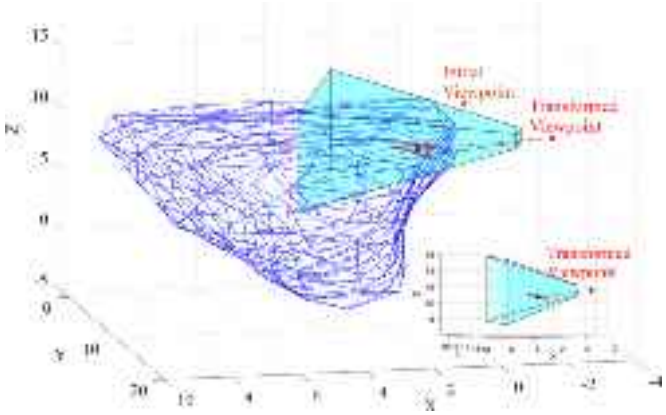

Fig. 8. Illustration of the FOV model of the depth camera.

$$\rho_\varepsilon^i = d_\varepsilon.o_v^i \tag{12}$$

$$\beta_u = -\beta_d = d_\varepsilon \tan\frac{\theta_v}{2}\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{13}$$

$$\beta_r^i = -\beta_l^i = \rho_\varepsilon^i \times \beta_u \tag{14}$$

$$^{xy}p_\varepsilon^i = p_v^i + \rho_\varepsilon^i + \beta_x + \beta_y^i,\ x \in \{u, d\} \text{ and } y \in \{l, r\} \tag{15}$$

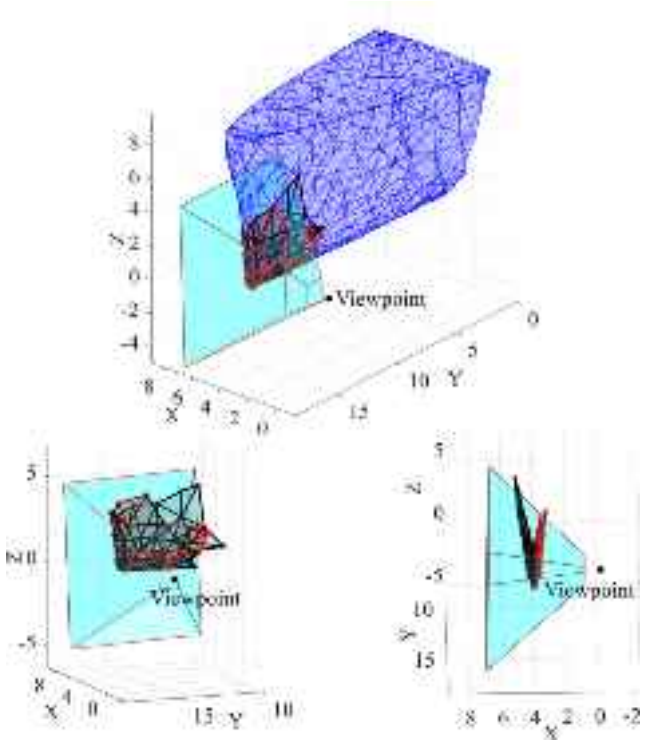

Fig. 9. Illustration of visible triangles that are covered by the FOV of the sensor for an example case: (red) visible triangles, (black) invisible triangles.

where the letters in the scripts represent ***u***pward, ***d***ownward, ***l***eft, and ***r***ight from the perspective of the viewpoints. Thus, ${}^{x}P_{\varepsilon}^{i}$ represents the corner point of the plane ε located at the corner x, which is calculated for the i[th] viewpoint. It is defined as the common coordinate frame of the viewpoint.

An example of the FOV model of the depth camera is presented in Fig. 8, utilizing the triangular mesh of the destroyer bow shown in Fig. 3 (b). In Fig. 8, the triangle corresponding to the viewpoint is highlighted in red, with its center indicated by a black point. The FOV model of the depth camera is represented as a trapezoidal prism. As seen from the figure, the near and far planes of the FOV model are oriented perpendicular to the X-Y plane, consistent with the configuration of the depth sensor mounted on the UAV.

3.3.2. Determination of triangles covered by the FOV on viewpoints

The triangles in the coverage area of i^{th} viewpoint are determined using the FOV model of the depth camera. First, a convex hull is created by using the eight corner points ${}^{x}P_{\varepsilon}^{i}$ for all $x \in \{ur, ul, dr, dl\}$ and $\varepsilon \in \{n, f\}$. Then, for i^{th} viewpoint, it is checked whether any of the vertices of the triangles lie within the convex hull. This approach ensures that all triangles located partially or entirely within the convex hull are accounted for. The indexes of the triangles that satisfy this condition are then recorded in a list associated with the i^{th} viewpoint. After performing the procedure for all the viewpoints, the corresponding triangles covered by each viewpoint will be identified.

Some triangles within the FOV may be obscured by others and, therefore not visible from the viewpoint, such as those located on the opposite side of the structure relative to the viewpoint. To exclude such occluded triangles within the FOV, a method based on the visibility graph algorithm [39] is employed. This approach receives the viewpoint and the set of triangles within the FOV. The procedure begins by selecting a triangle within the FOV of i^{th} viewpoint. Then, three straight lines are drawn connecting each vertex of the triangle to the viewpoint. If none of these lines intersect other triangles within the FOV, the triangle is considered visible and added to the list of visible triangles for i^{th} viewpoint. Additionally, it is verified that all vertices of the triangle lie within the convex hull to ensure that the triangle is fully contained within it. Fig. 9 shows the visible and invisible triangles for a sample case. The visible triangles are shown in red while the invisible ones are in black.

3.4. Set covering problem

Identifying triangles covered by viewpoints reveals that several triangles are visible from a single viewpoint. Many of these viewpoints capture data from the same area of the structure's surface. As a result, selecting only the essential viewpoints will reduce both the flight time of the UAV and the volume of data to be processed. In the proposed method, we prefer to handle the problem as a Set Covering Problem (SCP) where we determine the minimum number of viewpoints needed to cover all visible triangles. The objective of the SCP is to find the minimum number of subsets from a given collection of sets that cover all elements in a universal set. In other words, it aims to ensure that every element in the universal set is included in at least one selected subset while minimizing the number of subsets used.

In this study, the mathematical model of the SCP was employed to select the minimum number of viewpoints needed to cover all visible triangles. The inputs to the SCP consist of the set of visible triangles that need to be covered (U) and the collection of subsets (S), where each subset represents the triangles covered by a specific viewpoint, such that their union equals the set of visible triangles. The objective is to find the smallest subset of these subsets that fully covers all visible triangles. The decision variables, sets, parameters, and mathematical model of the SCP are presented as follows:

Sets:

U: The set of visible triangles that need to be covered

S_j: The set of elements covered by viewpoint $\mathscr{V}^j$. Each $\mathscr{V}^j$ corresponds to a subset of U.

$$S_j \subseteq U \text{ for } j \in [1,\ m_v] \tag{16}$$

Parameters:

a_{ij}: A binary parameter indicating whether element i is covered by viewpoint j.

$$a_{ij} = \begin{cases} 1 & , i \in S_j \\ 0 & , o/w \end{cases} \tag{17}$$

Decision Variables:

x_j: A binary decision variable indicating whether viewpoint j is selected or not.

$$x_j = \begin{cases} 1 & , \mathscr{V}^j \text{ is selected} \\ 0 & , o/w \end{cases} \tag{18}$$

Mathematical Model:

$$min \sum_{j=1}^{m_v} x_j \tag{19}$$

$$\sum_{j=1}^{m_v} a_{ij} x_j \geq 1 \ \forall i \ \in U \tag{20}$$

$$x_j \in \{0,1\}\ \forall j\ \in \{1,2,\ \cdots,\ m_v\} \tag{21}$$

The objective function (18) is to minimize the number of selected viewpoints. Constraint (20) emphasizes that each triangle in the set U is covered by at least one selected viewpoint. Constraint (21) indicates that x_j is a binary decision variable. Thus, by solving the SCP for the reduction of the number of viewpoints to be visited, the scan viewpoints $\mathscr{V}_s^j$, $j \in [1, m_s]$ are obtained from (18).

4. Coverage path planning

The solution to the SCP identifies a set of viewpoints outside the structure that collectively cover all visible triangles. Once these viewpoints are established, one can compute numerous possible routes to visit them by the depth camera mounted onto a UAV. However, minimizing the route length reduces the UAV's overall flight time. In the proposed method, we prefer to handle the Travelling Salesman Problem (TSP) to determine the shortest path for the UAV to visit all viewpoints outside the structure. The goal of the TSP is to find the shortest possible route that a traveling salesman can take to visit each city exactly once and return to the starting city [40].

In this study, the mathematical model of the TSP was employed to determine the shortest path that visits each viewpoint exactly once and returns to the starting viewpoint. We note here that the inputs to the TSP consist of the set of viewpoints that need to be visited and a distance matrix D, where d_{ij} represents the distance between viewpoint i and viewpoint j. To compute the distance between viewpoints, we first examine whether the line segment connecting the viewpoints intersects with the 3D triangle mesh model, given that the viewpoints are positioned outside the structure. If an intersection occurs, a large value is assigned to the corresponding distance item to prevent the UAV from colliding with the model. Conversely, if no intersection is detected, the Euclidean distance between the 3D viewpoints is used for the corresponding distance value. The decision variable, parameters, and mathematical model of the TSP are presented as follows:

Parameters:

d_{ij}: The distance between viewpoints i and j.

Decision Variables:

x_{ij}: A binary decision variable where:

$$x_{ij} = \begin{cases} 1 & ,if\ \mathscr{V}_s^i\ is\ followed\ by\ \mathscr{V}_s^j \\ 0 & ,o/w \end{cases} \tag{22}$$

Mathematical Model:

$$min \sum_{i=1}^{m_s} \sum_{j=1}^{m_s} d_{ij} x_{ij} \tag{23}$$

Subject to:

$$\sum_{j=1, j\neq i}^{m_s} x_{ij} = 1\ \forall i\ \in \{1,\ 2, \cdots, m_s\} \tag{24}$$

$$\sum_{j=1, j\neq i}^{m_s} x_{ji} = 1\ \forall i\ \in \{1,\ 2, \cdots,\ m_s\} \tag{25}$$

$$u_i - u_j + n.x_{ij} \leq (n-1)\ \forall i\ \neq j,\ i,j \in \{2, \cdots,\ m_s\} \tag{26}$$

$$x_{ij} \in \{0,1\}\ \forall i,j\ \in \{1,\ 2, \cdots,\ m_s\} \tag{27}$$

Table 1
Hyperparameter optimization values for relative alpha and offset.

Parameters	Values
Relative Alpha	10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 25, 30, 35, 40
Offset	0.5, 0.6, 0.7, 0.8, 0.9, 1.0, 1.1, 1.2, 1.3, 1.4, 1.5

$$u_i \geq 0\ \forall i\ \in \{2,\ 3, \cdots,\ m_s\} \tag{28}$$

The objective function (23) aims to minimize the total travelled distance required to visit all viewpoints. Constraint (24) guarantees that each viewpoint has exactly one incoming and one outgoing edge. Constraint (25) ensures that each viewpoint is reached from exactly one other viewpoint. Additionally, Constraint (26), known as the Miller-Tucker-Zemlin (MTZ) constraint, prevents the subtours. Here, u_i denotes a real-valued variable associated with each viewpoint i, used to define the order of the viewpoints in the tour.

5. Experimental results

In order to assess the performance of the proposed method, experiments were conducted in two distinct cases. In the first case, the effects of some parameters on the coverage path lengths and the number of viewpoints were analyzed using three surface boundary mesh models representing structures with varying volumes and geometric variations. In the second case, simulations were performed within the realistic robotic simulation environment, Gazebo. The results from these simulations demonstrate the coverage performance achieved by the method.

Three surface boundary mesh models were generated using the 3D Alpha Wrapping method [41] provided by the Computational Geometry Algorithms Library (CGAL). Subsequently, GUROBI 11.0.2 [42] was employed to solve the SCP and TSP. GUROBI is a cutting-edge mathematical optimization solver widely utilized in academia, research, and industry to address complex optimization problems. It is capable of solving various types of problems, including linear programming (LP), mixed-integer programming (MIP), and related problems. We set the time limit to 600 s for each problem.

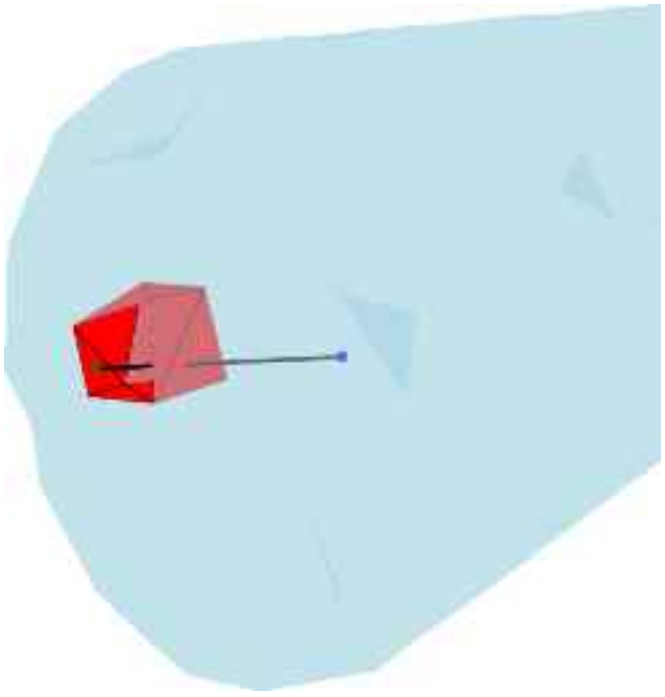

Fig. 10. Illustration of a sample concave region which causes the viewpoint to be inside of the surface mesh.

Table 2
The available relative alpha values for each offset parameter.

Offset	Models	Relative Alpha Values
0.5	Pedestrian Bridge	10,11,12,13,14,15,16,17,18,19,20,25
	Destroyer Bow	10,11,12,13,14,18
	Fuselage	10,11,12,13,14,16
0.6	Pedestrian Bridge	10,11,12,13,14,15,16,17,18,19,20,25,30
	Destroyer Bow	10,11,12,13
	Fuselage	10,11,12,13,14,15,18
0.7	Pedestrian Bridge	10,11,12,13,14,15,16,17,18,19,20,25,30,35
	Destroyer Bow	10,11,12,13,18
	Fuselage	10,11,12,13,14,15,16,17
0.8	Pedestrian Bridge	10,11,12,13,14,15,16,17,18,19,20,25,30,35,40
	Destroyer Bow	10,11,12,13,14,15,16,18,19,35
	Fuselage	10,11,12,13,14,15,17,18,20
0.9	Pedestrian Bridge	10,11,12,13,14,15,16,17,18,19,20,25,30,35,40
	Destroyer Bow	10,11,12,13,17,18,20
	Fuselage	10,11,12,13,14,15,16,17,18,19,20,25,30
1.0	Pedestrian Bridge	10,11,12,13,14,15,16,17,18,19,20,25,30,35,40
	Destroyer Bow	10,11,14,15,16,17,18,19,20,25,30
	Fuselage	10,11,12,13,14,15,16,17,18,19,20,35
1.1	Pedestrian Bridge	10,11,12,13,14,15,16,17,18,19,20,25,30,35,40
	Destroyer Bow	10,11,12,13,14,15,16,17,18,19,20,25,30
	Fuselage	10,11,12,13,14,15,16,18,19,20,25,30,35
1.2	Pedestrian Bridge	10,11,12,13,14,15,16,17,18,19,20,25,35,40
	Destroyer Bow	10,11,12,13,14,15,16,17,18,19,20,25
	Fuselage	10,11,12,13,14,15,16,17,18,19,20,25,30,35,40
1.3	Pedestrian Bridge	10,11,12,13,14,15,16,17,18,19,20,25,30,35,40
	Destroyer Bow	10,11,12,13,14,15,16,17,18,19,20,25,35
	Fuselage	10,11,12,13,14,15,16,17,18,19,20,25,35,40
1.4	Pedestrian Bridge	10,11,12,13,14,15,16,17,18,19,20,25,30,35,40
	Destroyer Bow	10,11,12,13,14,15,16,17,18,19,20,25,30,35,40
	Fuselage	10,11,12,13,14,15,16,17,18,19,20,25,40
1.5	Pedestrian Bridge	10,11,12,13,14,15,16,17,18,19,20,25,30,35,40
	Destroyer Bow	10,11,12,13,14,15,16,17,18,19,20,25,30,35,40
	Fuselage	10,11,12,13,14,15,16,17,18,19,20,25,30,40

5.1. Hyperparameter optimization for alpha and offset parameters

The surface boundary mesh plays a critical role in determining the viewpoints. The selection of the alpha and offset parameters significantly influences the generated surface boundary mesh. To optimize these parameters, we performed a hyperparameter optimization using the grid search method. Table 1 presents the parameter values tested for various structures. For the offset parameter, we employed exact values, as they represent the distance between the structure model and the resulting surface mesh. Accordingly, the offset parameter was varied from 0.5 to 1.5 meters. Conversely, the alpha parameter was defined in relative terms. Specifically, we calculated the diagonal length of the bounding box of each structure and divided this length by the values listed in Table 1, enabling us to adjust the alpha parameter based on the structure's size.

After generating the 3D triangular meshes for the structures' boundary surfaces, we assessed these meshes with a focus on the viewpoints located inside the 3D triangular meshes, particularly in concave regions. Fig. 10 illustrates a viewpoint inside the 3D triangular mesh, where triangles in concave regions are highlighted in red. As the relative alpha value increases, the mesh wraps more closely to the structure's surface, leading to the formation of concave regions within the 3D triangular mesh. Consequently, certain triangles within these concave regions exhibit surface angles inside the mesh.

Table 2 indicates the available relative alpha values for each offset parameter according to the given criteria in which the viewpoints are not inside the surface mesh. As shown in the table, the number of available relative alpha values increases with the offset for each model. However, the availability of these alpha values is significantly influenced by the complexity of the model. For instance, the pedestrian bridge, with its relatively simple form compared to the destroyer bow and fuselage, is capable of generating 3D triangular meshes without concave regions for almost all relative alpha values when using low offset values. In contrast, for more complex models such as destroyer bow and fuselage, concave regions tend to form as the relative alpha values increase, particularly when the offset is low. The reason for that is the mesh wraps more tightly to the surface as the relative alpha increases with a low offset, and the mesh does not extend adequately, leading to the formation of concave regions.

The performance of the proposed method is systematically assessed using the relative alpha-offset pairs presented in Table 2. First, the effect of the relative alpha parameter is examined. Subsequently, the best results according to path length for all offset values of each model are analyzed.

5.2. Analysis of alpha and offset parameters regarding the effects on coverage path planning

In this section, an analysis is conducted to reveal the impact of the alpha and offset parameters on the bounding surface mesh and the coverage path. To achieve this, two experiments were performed by applying the framework shown in Fig. 2 to three models—the pedestrian bridge, destroyer bow, and fuselage—for each pair of alpha and offset values. Through these experiments, three metrics were identified for evaluation: (i) the number of triangles on the boundary surface mesh, (ii) the number of scan viewpoints ($\mathscr{V}_s^j$), and (iii) the coverage path length. In the first experiment, the offset parameter was held constant while the alpha parameter was varied. In the second experiment, the same procedure in the first experiment was applied using the best offset and relative alpha pairs for each model, determined based on the path length and the coverage of all visible triangles by the viewpoints.

Fig. 11 presents the results of the first experiment for three models with an offset value of 1.2 and various alpha values. Although the analysis is based on the offset value of 1.2, the findings exhibit similar characteristics across all tested offset values.

As shown in Fig. 11, the number of triangles in the boundary surface mesh is directly influenced by the relative alpha parameter. This occurs because, as the relative alpha value increases, the method produces a tighter wrap around the structure, capturing finer details. Consequently, the mesh consists of smaller triangles, resulting in an overall increase in their number. The overall trend reveals that the number of viewpoints decreases as the relative alpha parameter increases. Initially, this may seem counterintuitive, as one might expect the number of viewpoints to rise with the increasing number of triangles. However, as the relative alpha parameter grows, the triangles become smaller, allowing each viewpoint to cover a greater number of triangles. Therefore, fewer viewpoints are required to observe all visible triangles as the relative alpha value increases. Although a decrease in the number of viewpoints is generally expected to lead to a shorter path, the spatial arrangement of the viewpoints in 3D space plays a critical role in determining the path length. For instance, the SCP may select a more distant viewpoint to observe a larger number of triangles, which can result in an increase in the path length. Additionally, the line segments between viewpoints are checked to ensure collision-free paths, which may further contribute to an increase in path length. As a result, multiple factors influence the path length, and no consistent trend can be observed across all models. Our findings indicate that mid-range relative alpha values yield better path lengths for the pedestrian bridge and destroyer bow models, while larger relative alpha values are more effective for the fuselage model.

In the second experiment, we evaluated the best offset and relative alpha pairs for each model by using the best alpha value for each offset based on the path length and the coverage of all visible triangles by the viewpoints. The results are presented in Fig. 12. The corresponding relative alpha values for these best results are indicated in the bars. GUROBI has demonstrated the ability to solve the TSP optimally for the chosen viewpoints in 600 s time limit. Similarly, the SCP results are optimal for both the pedestrian bridge and destroyer bow models. However, as illustrated in Fig. 12, the number of triangles exceeds 2500,

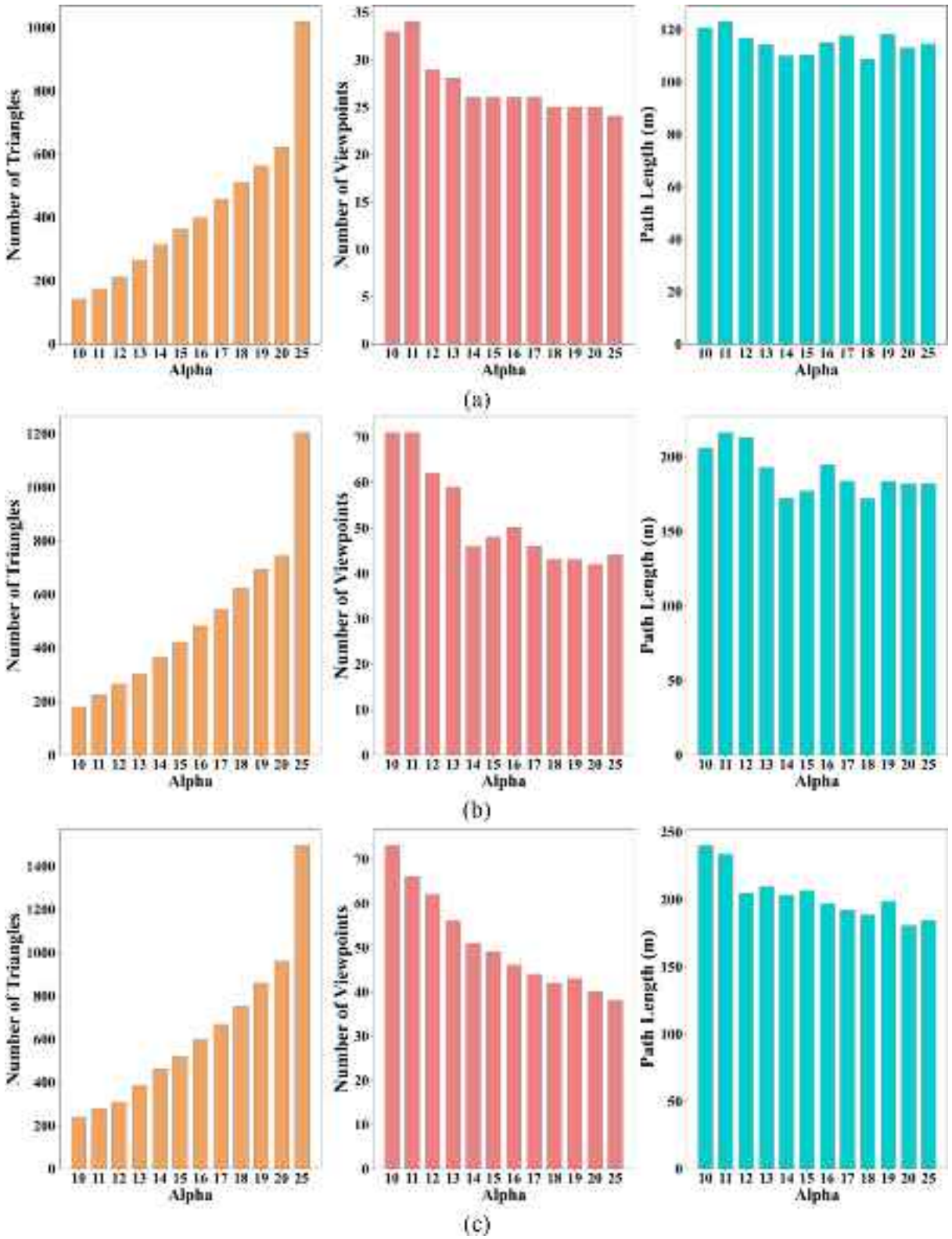

Fig. 11. Analysis results obtained with the offset 1.2 and various alpha (relative) values for three models: (a) Pedestrian Bridge, (b) Destroyer Bow, and (c) Fuselage.

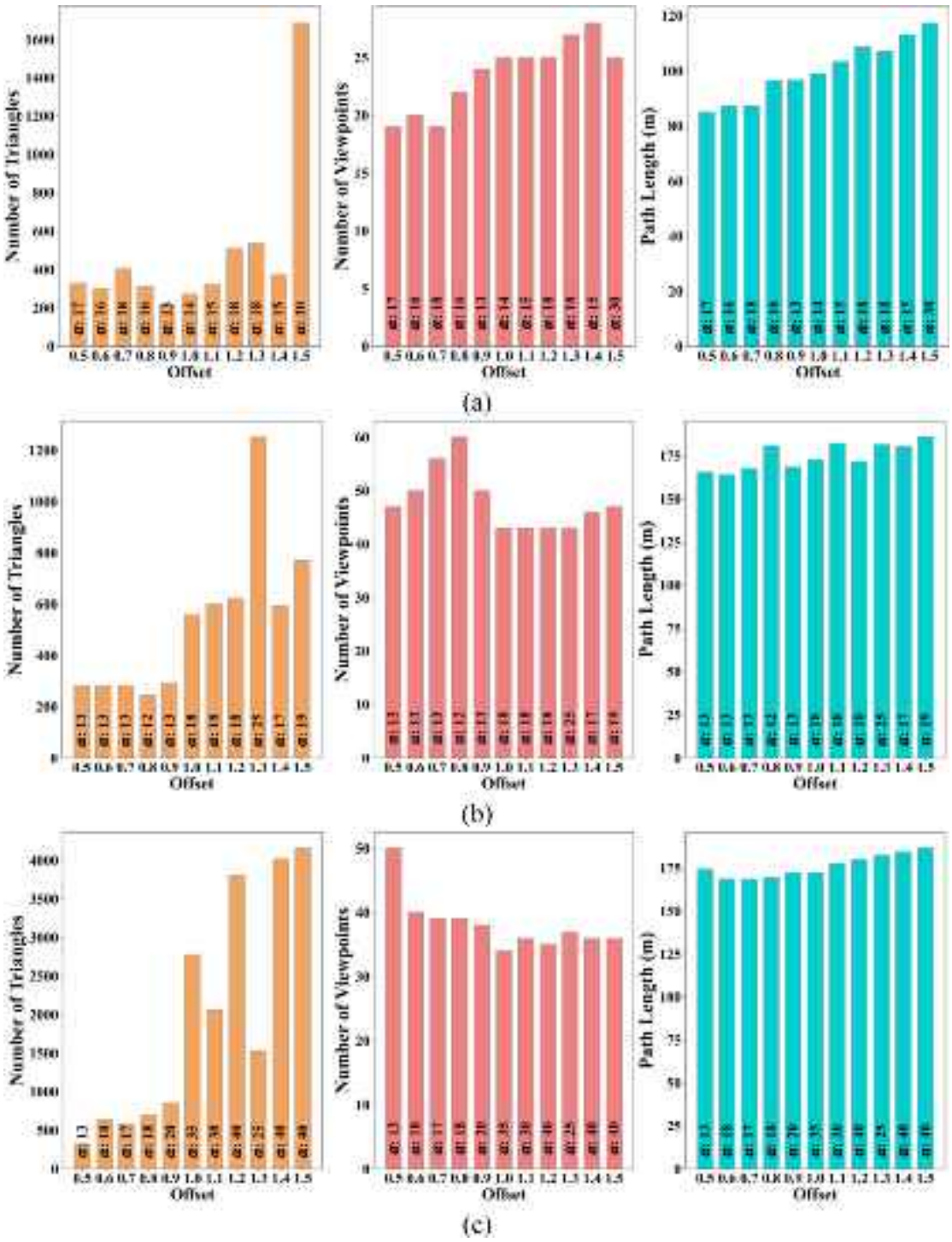

Fig. 12. Analysis results obtained with the best offset-alpha pairs based on the path length and the coverage of all visible triangles by the viewpoints (Black numbers in bars are the relative alpha values): (a) Pedestrian Bridge, (b) Destroyer Bow, and (c) Fuselage.

particularly for offset values greater than 1.0, and the alpha values surpass 25 for the fuselage model. Due to the imposed 600-second time limit, gaps ranging from 2 % to 9 % are observed for these parameter settings in the fuselage model.

As illustrated in Fig. 12, there is a general trend where the path length increases with higher offset values for all models. This is due to the growing distance between the model and the 3D triangle mesh as the offset increases. However, the number of triangles, the number of viewpoints, and the path length exhibit distinct behaviors for each model.

The pedestrian bridge is a simple and small model with a rectangular prism-like shape. For this model, lower relative alpha values result in larger triangles, and to cover these larger triangles, the SCP selects more viewpoints to cover these larger triangles. With the same offset, particularly at lower offset values, increasing the relative alpha value reduces triangle sizes and increases the number of triangles. However, this increase is minimal due to the model's small size. Consequently, beyond a certain offset, the number of viewpoints increases again to cover all visible triangles. As the offset increases, the 3D triangle mesh expands, leading to a more significant rise in the number of triangles and

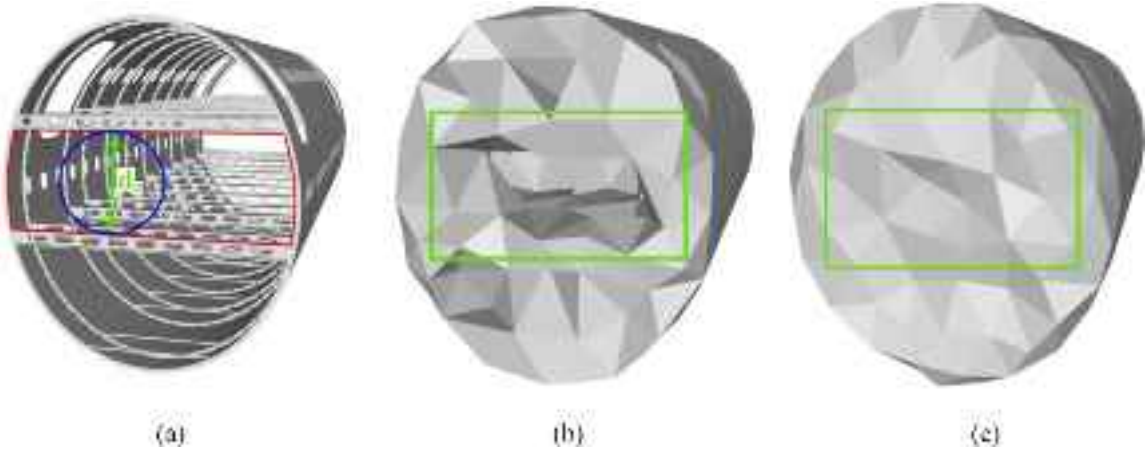

Fig. 13. Presentation of the relation between the cavities and the constraint of offset and alpha: (a) The parameter d_g, (b) the boundary surface mesh with $d_{offset} = 0.5m$ and r_α=1 m, (c) the boundary surface mesh with $d_{offset} = 1m$ and r_α=1 m.

a decrease in triangle size. At this stage, when the number of triangles surpasses a certain threshold, SCP may not find the optimal solution in a limited time duration, and lower relative alpha values can provide the best results, as seen with an offset value of 1.4. Therefore, mid-range relative alpha values, corresponding to medium-sized triangles, offer better path lengths at low and mid-offset values, whereas higher alpha values yield better results at larger offsets. In conclusion, for simple and small models, low offset values combined with mid-range relative alpha values tend to yield the most favorable path length outcomes.

The destroyer bow model is the most complex of the three models, distinguished by its small components and segmented structure. For this model, particularly at low offset values, increasing the relative alpha parameter causes the method to form a tighter wrap around the surface, capturing finer details and often leading to the formation of concave regions. As a result, the range of usable relative alpha values is limited at lower offsets. Similar to the pedestrian bridge model, as the offset increases, the 3D triangle mesh expands, leading to a significant increase in the number of triangles and a decrease in triangle size. Consequently, the range of available relative alpha values broadens, allowing for better coverage without the formation of concave regions. The best path length results are achieved with mid-range relative alpha values at both low and mid-range offsets. In contrast to the other two models, as the offset value further increases, although concave regions no longer form, the complexity of the model results in many triangles that are not fully covered by viewpoints. Since we focused on selecting the best results for visible triangles only, mid-range alpha values provide superior results at

Algorithm 1
Obtaining the parameters alpha and offset.

Input: S_{mesh}: structure mesh, d_g: the diameter of the largest circle with an equidistant center in the biggest gate of the structure, d_{offset}^{step}: step size of offset for iteration, r_α^{step}: step size of alpha for iteration, d_{max}: the maximum measurable (or the range for the best accuracy) depth distance of the sensor, d_s: distance calculated by the Eq. (1), θ_v and θ_h: the vertical and horizontal FOV of the depth camera,

Output: d_{offset}: value of the offset parameter, r_α: value of the alpha parameter

Definition:

- $\delta > 0$: a positive constant

$d_{offset} = 0$
if $\theta_v > \theta_h$ **then**
    $r_s = d_s \tan\theta_h$
else
    $r_s = d_s \tan\theta_v$
endif
while $d_{offset} < (d_{max} - d_s)$ **do**
    $d_{offset} = d_{offset} + d_{offset}^{step}$
    $r_\alpha = d_g - 2d_{offset} + \delta$
    $\overline{r_\alpha} = 0$
    while *true* **do**
        Implement the alpha-wrapping algorithm for d_{offset} and r_α using S_{mesh}
        Calculate the maximum circumradius(r_c) of facets on the boundary surface mesh
        if $r_c < r_s$ **then**
            $\overline{r_\alpha} = r_\alpha$
            $r_\alpha = r_\alpha + r_\alpha^{step}$
        else
            break
        endif
    endwhile
    if $\overline{r_\alpha} > 0$ **then**
        $r_\alpha = \overline{r_\alpha}$
        break
    endif
endwhile

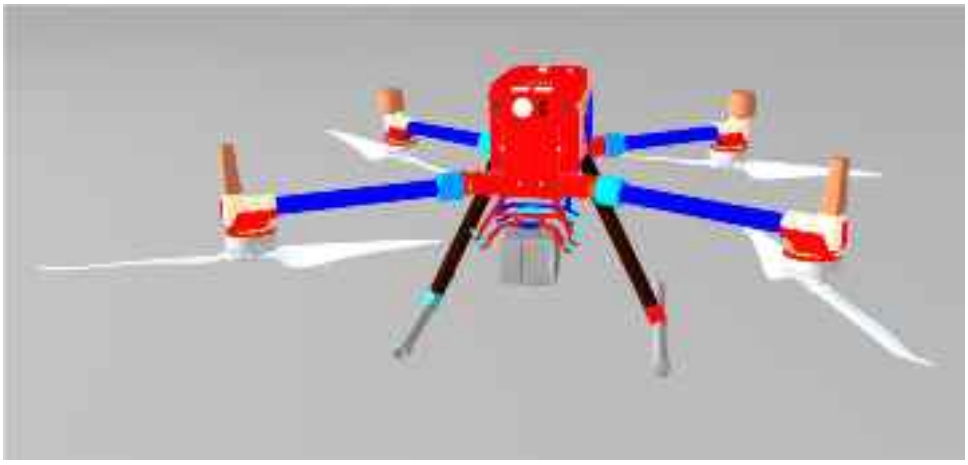

Fig. 14. The UAV and sensor model in Gazebo simulation environment.

larger offset values. In summary, for complex models, mid-range offset values paired with mid-range relative alpha values generally produce better path length results.

The fuselage model exhibits an intermediate level of complexity compared to the other two models, characterized by surface holes on a cylindrical structure and a largely hollow interior. Due to the existing surface of the model, the number of available relative alpha values increases rapidly with higher offset values. Additionally, higher relative alpha values generally produce the best results across all offset values, considering given the surface structure of the model. However, an exception arises when the number of triangles exceeds a certain threshold, at which point the SCP may fail to find an optimal solution within a limited time frame. In such cases, lower relative alpha values may provide better outcomes, as observed with an offset value of 1.3. An exception arises when the number of triangles exceeds a certain threshold, where SCP may not find an optimal solution within a limited time frame, and lower relative alpha values can yield better outcomes, as observed with an offset of 1.3. In summary, for models with pre-existing surfaces, mid-range offset values paired with higher relative alpha values typically result in the most favorable path length outcomes.

5.3. Determination of alpha and offset parameters

In the alpha wrapping algorithm, the offset parameter controls the distance between the output mesh and the input geometry. Thus, the offset value becomes the distance between the structure and the boundary surface mesh. Since viewpoints are determined to be at a distance d_s from the boundary surface mesh, the sensor viewpoints move away from the structure as the distance increases. The structure may not be within the sensor's field of view. In this case, the constraints for the offset can be defined by the sensor model. The offset d_{offset} can be chosen $0 < d_{offset} < (d_{max} - d_s)$. It is recommended to select d_{offset} close to zero, since a wider volume can be scanned within the interior of the structure, and the accuracy of the depth sensors increases as the distance decreases. However, with a smaller d_{offset}, the lower limit of the alpha parameter r_α becomes larger by causing a larger size of facets on the boundary surface mesh. Thus, the size of the facet may be wider than the sensor's field of view. Furthermore, there is a constraint on d_{offset} and r_α to prevent cavities on the boundary surface mesh, which poses a significant problem for the proposed method, as shown in Fig. 10. This constraint can be determined with respect to the gates on the structure, which have the potential to cause cavities. Let us define a parameter d_g, which is the diameter of the largest circle with an equidistant center in the biggest gate of the structure. Thus, the constraint $r_\alpha \geq d_g - 2d_{offset}$ ensures that the boundary surface mesh has no cavities. An example is shown in Fig. 13. In Fig. 13(a), the red lines show the biggest gate on the fuselage, the blue circle represents the largest circle with an equidistant center in the gate of the structure, and the green arrow shows the diameter of the circle, which is $d_g = 2.464m$. If the parameters are chosen as $d_{offset} = 0.5m$ and $r_\alpha = 1$ m, then the constraint is not met. And, the bounding surface mesh includes cavities as seen in Fig. 13(b). If the parameters are chosen as $d_{offset} = 1m$ and $r_\alpha = 1$ m, then the constraint is met. And, the bounding surface mesh does not include any cavities as seen in Fig. 13(c). The increase in the parameter alpha r_α directly results in the increase in the size of the circumradius of the facets. It will be reasonable to ensure the circumcircles of facets are covered by the field of view of the sensor at the distance d_s. The feasible r_α should obey this case.

In light of the explanations above, a feasible solution for alpha and offset values can be obtained with Algorithm 1. The algorithm searches for the pair (r_α, d_{offset}) with the minimum d_{offset} and maximum r_α, which obeys the constraints. The constraints are $r_\alpha \geq d_g - 2d_{offset}$, $0 < d_{offset} < (d_{max} - d_s)$, and the circumcircles of facets are covered by the field of view of the sensor at the distance d_s.

5.4. Simulation results

The framework for the proposed method, as shown in Fig. 2, generates coverage path plans. The actual performance of the coverage is demonstrated through a series of simulation experiments in which the generated path plans are executed by the UAV. These simulations were conducted using the Gazebo [43] simulation environment, in conjunction with ROS [44] packages that integrate the UAV, structure, and sensor models to perform the coverage task. These ROS packages facilitate the control of the UAV, enabling it to follow the planned path and collect sensor data within the simulation environment.

5.4.1. Experimental setup

The UAV model is derived from the DJI Matrice 300 RTK, while the sensor model is based on the Sick Visionary T-Mini with $\theta_v = 60^\circ$, $\theta_h = 70^\circ$, $d_n = 1m$, and $d_f = 7m$ as depicted in Fig. 14. A pose controller is used to manage the UAV's motion, receiving position input and generating twist output with the assistance of a PID controller. The position input includes the UAV's position (in Cartesian coordinates) and orientation (in quaternion). The output encompasses the linear velocities of the UAV along the x, y, and z axes, as well as the angular velocity for rotation (yaw) about the vertical (z) axis. Accordingly, the position and orientation of the UAV at each viewpoint are provided by the planned path. In fact, by utilizing the sensor position provided by the planned path at each viewpoint and applying the transformation between the UAV and the sensor, the UAV's position can be calculated and subsequently provided as an input to the controller. However, because

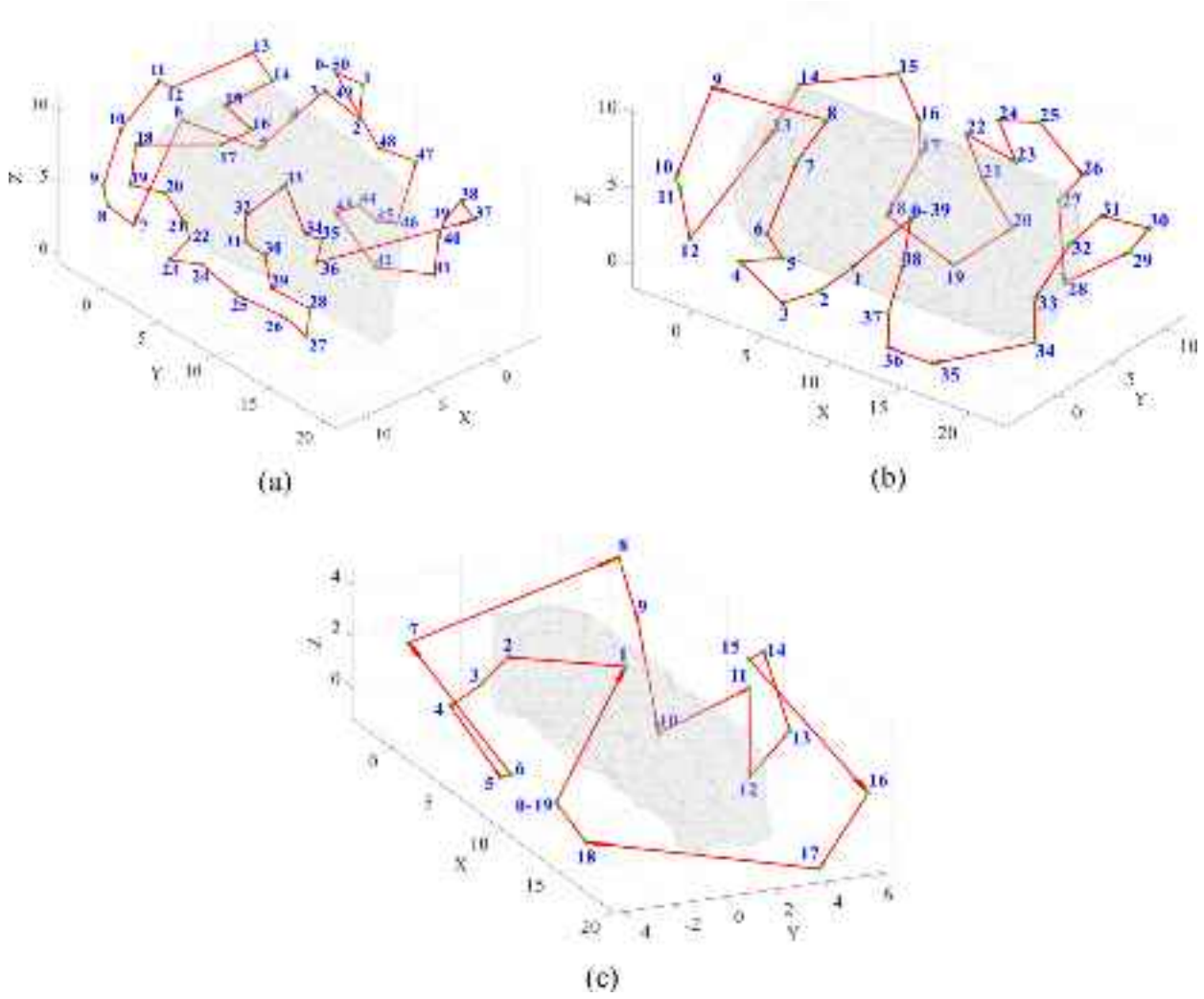

Fig. 15. The coverage path plans for three structures: (a) Destroyer bow, (b) Fuselage, (c) Pedestrian bridge.

Fig. 16. A snapshot taken during the simulation.

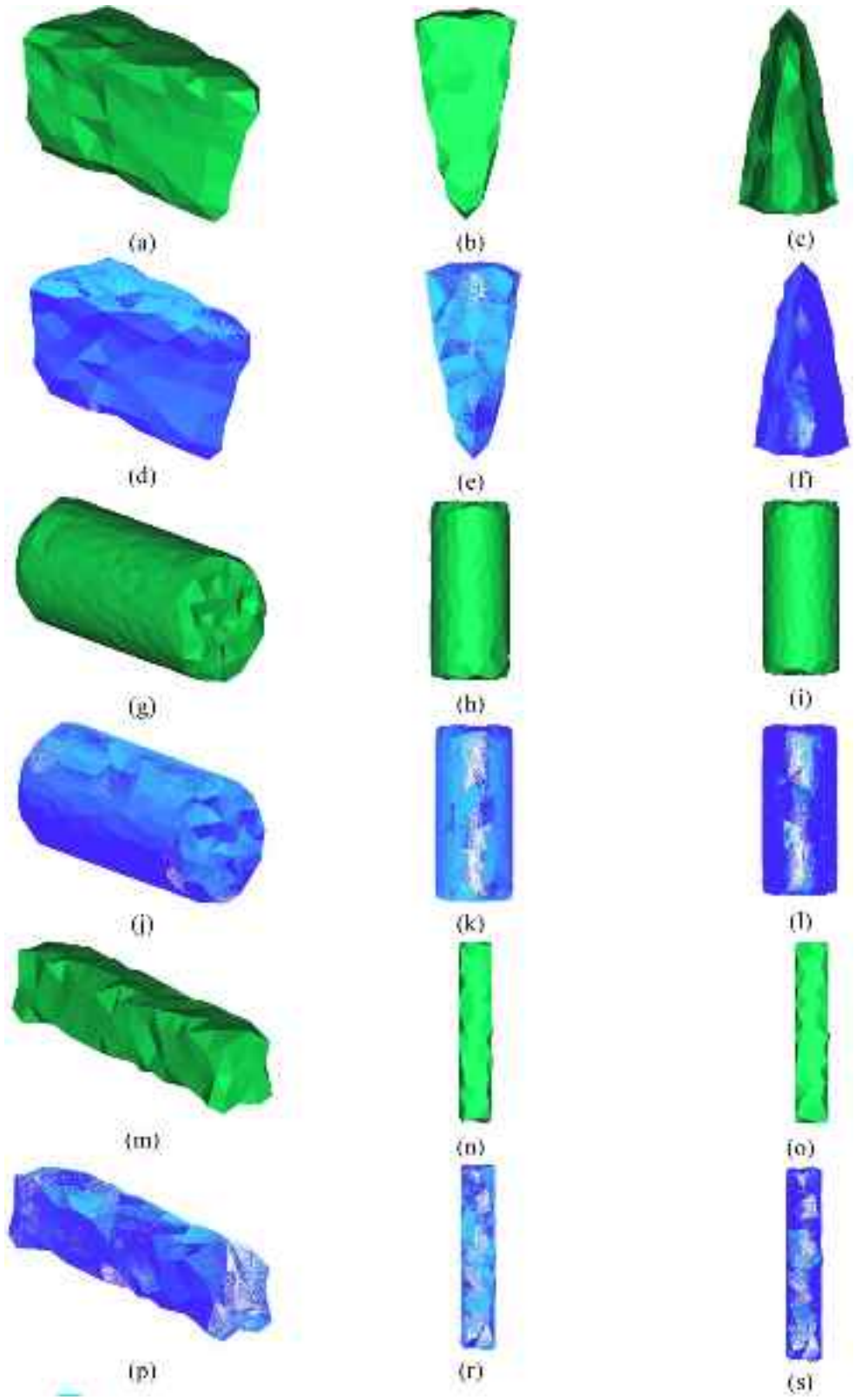

Fig. 17. The boundary surface mesh models (with green) and the point clouds obtained from the scans with blue (Side, top, and bottom views of models are given in the first, second, and third columns, respectively). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article).

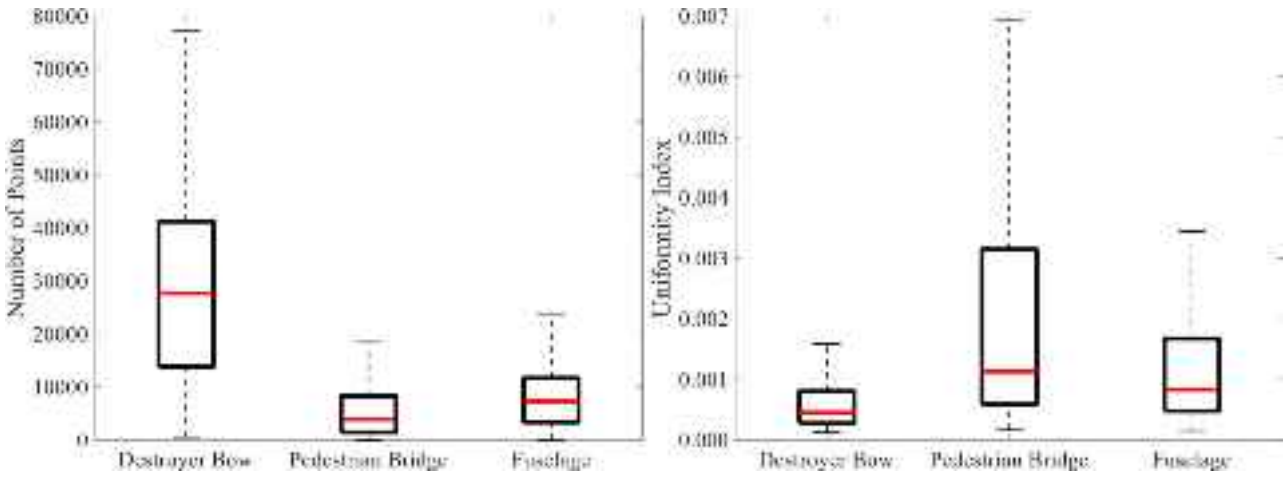

Fig. 18. The box plots for the number of points and uniformity index of captured points within each triangles of the boundary surface mesh.

the UAV's orientation is required to be represented as a quaternion, the components of the unit direction vector defining the viewing direction at each viewpoint provided by the planned path cannot be directly used. Hence, the UAV's alignment towards the target orientation at each viewpoint is achieved using the axis-angle rotation, where the rotation axis and angle are then converted into quaternion form. Once the quaternion components are obtained, the UAV's orientation at the relevant viewpoint is provided to the controller as a quaternion input. To facilitate the UAV's motion between two consecutive viewpoints, intermediate points are linearly distributed between them. The UAV's position and orientation at these intermediate points are transmitted to the controller via a ROS topic, ensuring a smooth motion through the path. The controller then provides the UAV's motion by producing a velocity output that corresponds to the specified position and orientation inputs.

5.4.2. Experimental procedure

The UAV visits all of the viewpoints by tracking the planned path, stopping at each viewpoint to capture point clouds using the depth camera. It stops at each viewpoint, and the depth camera captures a point cloud. The experiments were conducted in two stages. In the first stage, the boundary surface mesh was placed in the workspace, and the captured point clouds correspond to this model. Thus, the captured point cloud will belong to this model. Since the viewpoints, and accordingly the coverage path plan, were determined by using this model, the actual coverage of the model could be analyzed. In the second stage, the CAD model of the original structure was placed in the workspace. This time, by tracking the same path, the resulting point clouds correspond to the surface of the target structure that is actually desired to be covered.

The values of the relative alpha and offset pairs (alpha, offset) used to generate the surface boundary meshes in these experiments are (13, 0.6), (18, 0.8), and (17, 0.5) for the destroyer bow, fuselage, and pedestrian bridge, respectively. The generated coverage path plans, along with the viewpoints for each surface boundary mesh, are shown in Fig. 15. The path plans include 51, 40, and 20 viewpoints for destroyer bow, fuselage, and pedestrian bridge, respectively.

In the simulations, the UAV tracks the paths given in Fig. 15. The simulations were implemented two times for each structure by placing the original structure model and the boundary surface mesh seen in the first and second columns of Fig. 3, respectively. The UAV starts at the initial viewpoint and proceeds to traverse all designated viewpoints sequentially through the coverage path. At each viewpoint, the UAV stops to capture the 3D point cloud and record the transformation data. The transformations are utilized to represent the point clouds with respect to a global coordinate frame. This process continues until the UAV returns to the initial viewpoint, at which point the procedure is completed. Fig. 16 shows an image depicting the execution of a scanning operation in the simulation environment. In order to avoid performance degradation in the simulations, the UAV and the sensor model were represented using STL files, resulting in a white appearance due to the lack of texture and color information.

5.4.3. Experimental results

We first assessed the effectiveness of the proposed method for scanning the boundary surface mesh models. This analysis aimed to examine both the number of points per triangle and their spatial distribution. A higher point count with uniform distribution generally indicates an effective scan. To achieve this, first, the UAV scanned three different models by tracking the path and capturing the point cloud at each viewpoint. Consequently, 51 distinct point clouds were collected for the destroyer bow, while 40 and 20 point clouds were collected for the fuselage and pedestrian bridge, respectively. Fig. 17 shows the scanned boundary surface mesh models and the point clouds obtained from these scans. Then, the point cloud data obtained from viewpoints were merged and for each triangle in the boundary surface mesh, we computed both the number of points belonging to it and the uniformity of their distribution. The Uniformity Index (UI) was employed as a metric to quantify the uniformity of points within each triangle. UI serves as a quantitative measure of point distribution within an area, surface, or volume, frequently applied in fields such as 3D geometry, computer graphics, and simulations. A result close to 0 signifies a uniform distribution, while values approaching 1 indicate a non-uniform distribution.

We summarized the results for each model using box plots, which is an effective statistical visualization that summarizes the distribution of data. Each box plot features a red line indicating the median of the result, while the box itself illustrates the interquartile range, spanning from the first quartile (25th percentile) to the third quartile (75th percentile). The whiskers extending from the box depict the full range of the results.

The number of points associated with each triangle is directly correlated with the number of viewpoints at which the UAV observes the scene. The reason for that is the proposed method considers only

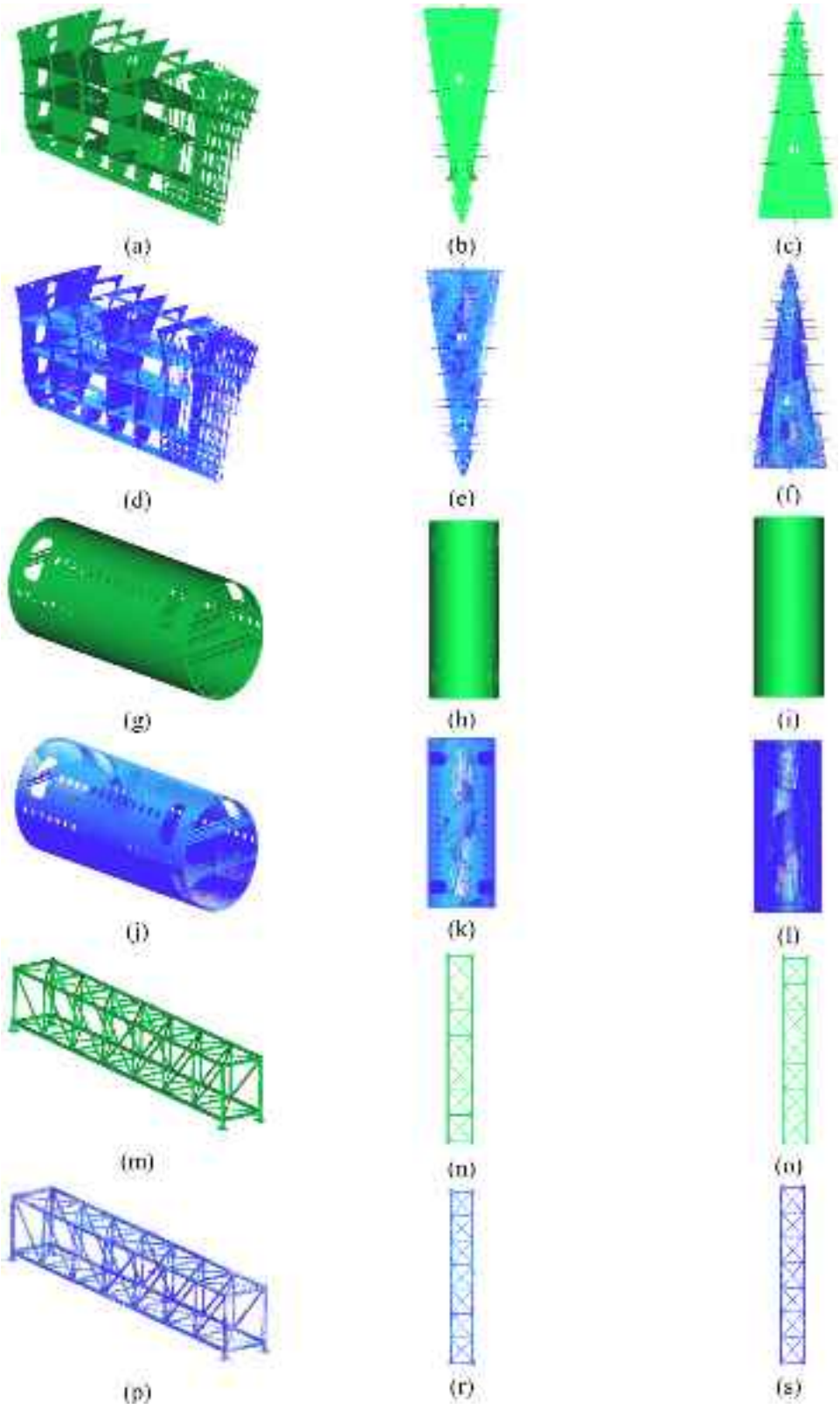

Fig. 19. The original structure models with green and the point clouds obtained from the scans with blue (Side, top, and bottom views of models are given in the first, second, and third columns, respectively).

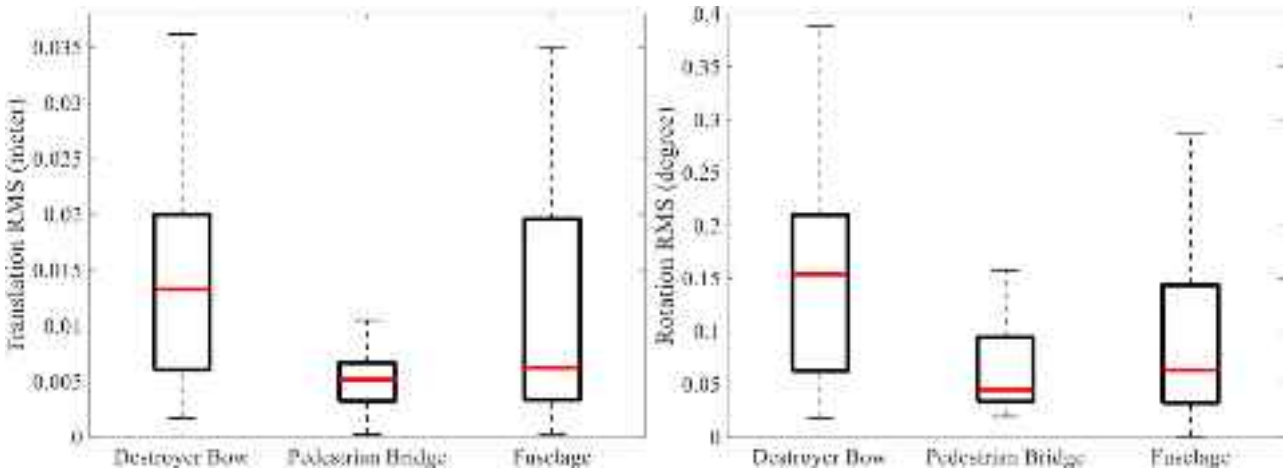

Fig. 20. The box plots for translation and rotation RMS calculated by taking into account the FOV from the viewpoints.

triangles that are completely inside the FOV in the determination of triangles covered by the viewpoints step to ensure that a visible triangle is completely covered by at least one viewpoint. Nevertheless, numerous triangles fall partially within the FOV, allowing the sensor to capture points from these partially covered triangles. Consequently, as the number of viewpoints increases, the number of points falling into triangles also rises.

Fig. 18 presents the number of points and UI metric for each model. As seen from the figure, the destroyer bow exhibits the highest median value for the number of points among all models. The triangles in this mesh are larger, reflecting the overall larger scale of the destroyer bow. Therefore, the drone needs to visit 50 viewpoints to completely cover all visible triangles at least once. At each viewpoint, the UAV gathers points from the same triangles, which increases the number of points falling into each triangle. Additionally, the spatial arrangement of viewpoints impacts the number of points falling into each triangle. Some triangles are observed from multiple viewpoints, which increases the number of points associated with them, while others are covered by only a few viewpoints, resulting in fewer points for those triangles. This variation causes longer whiskers and wider box for the destroyer bow compared to the other two models. The best alpha-offset pair for minimizing path length on the fuselage model requiring the UAV to visit 39 viewpoints to fully cover all visible triangles at least once. For the pedestrian bridge, only 19 viewpoints are necessary to achieve complete coverage. As seen from Fig. 18, a reduction in the number of viewpoints is associated with a decrease in the median value, along with shorter whiskers and narrower box plots. Although each model has different characteristics in terms of the number of points, the median values show that each triangle consists of at least approximately 5000 points, which is sufficient to accurately represent the 3D mesh model.

We also analyzed the uniformity of point distribution within each triangle. Among all models, the pedestrian bridge produces the highest median UI value, which is anticipated, as the UI value tends to decrease with a reduction in viewpoints or, equivalently, a reduction in points. Furthermore, based on the spatial configuration of viewpoints, triangles positioned at the top and bottom contain fewer points, resulting in a higher median UI value, along with longer whiskers and broader box plots. Conversely, with an increase in viewpoints, the median UI value descends, accompanied by shorter whiskers and narrower box plots. Additionally, the destroyer bow's geometry gradually narrows toward its base, with a considerable number of viewpoints focused near the top (Fig. 15). Consequently, the destroyer bow yields the best results for the UI metric, as triangles located at both the top and bottom contain more points compared to those in other models. Although each model exhibits distinct characteristics in terms of the UI metric, the median UI values across all models are close to zero, indicating a generally uniform point distribution within each triangle.

We also assessed the efficacy of the proposed method for scanning original models. The same procedure is then applied to the original structure models, using the same path plans as those used for scanning the boundary surface mesh models. Fig. 19 shows the scanned original structure models and the point clouds obtained from these scans.

The evaluation process began by converting the original structure models into point cloud data, referred to as the model point cloud, while taking into account the total number of points captured by the sensor. Subsequently, for each viewpoint, we identified a corresponding point cloud, designated as the reference point cloud, that was situated within the FOV of that viewpoint in the model point cloud. We then employed the Iterative Closest Point (ICP) algorithm to compare the reference point cloud with the acquired point cloud for each viewpoint. The ICP algorithm generates a transformation matrix that encompasses both the rotation matrix and translation vector between the two point clouds. We calculated the root mean square (RMS) value for both the rotation and translation. Finally, we presented the results for each model using box plots in Fig. 20. As seen from the figure, the RMS value for both the rotation and translation is near zero for all models, which indicates that the proposed method successfully scans the models without any translational or rotational error.

To evaluate the coverage performance, additional analyses were conducted, including a distance comparison between two point clouds. The first point cloud represents the scan acquired from the simulation, as illustrated in the second, fourth, and sixth rows of Fig. 19. The second point cloud comprises points sampled from the triangular mesh model of the original structure. The results for three distinct structures are presented in Fig. 21. In the first column, the original structure's point cloud is visualized, with colors indicating the distance between the two point clouds. As the distance increases, the color changes from blue to red, highlighting regions with green, yellow, and red as unscanned areas. Furthermore, the second column provides the corresponding distance histograms. Assuming regions with a distance exceeding 10 cm are considered unscanned, the histograms enable an assessment of the coverage rate. As illustrated in Fig. 21, 99.93 % of the destroyer bow was successfully covered. Given that the structure's interior is largely visible from the exterior, substantial coverage of the internal regions was achieved through external scanning alone. Similarly, a high coverage rate of 99.99 % was obtained for the pedestrian bridge. In contrast, the fuselage

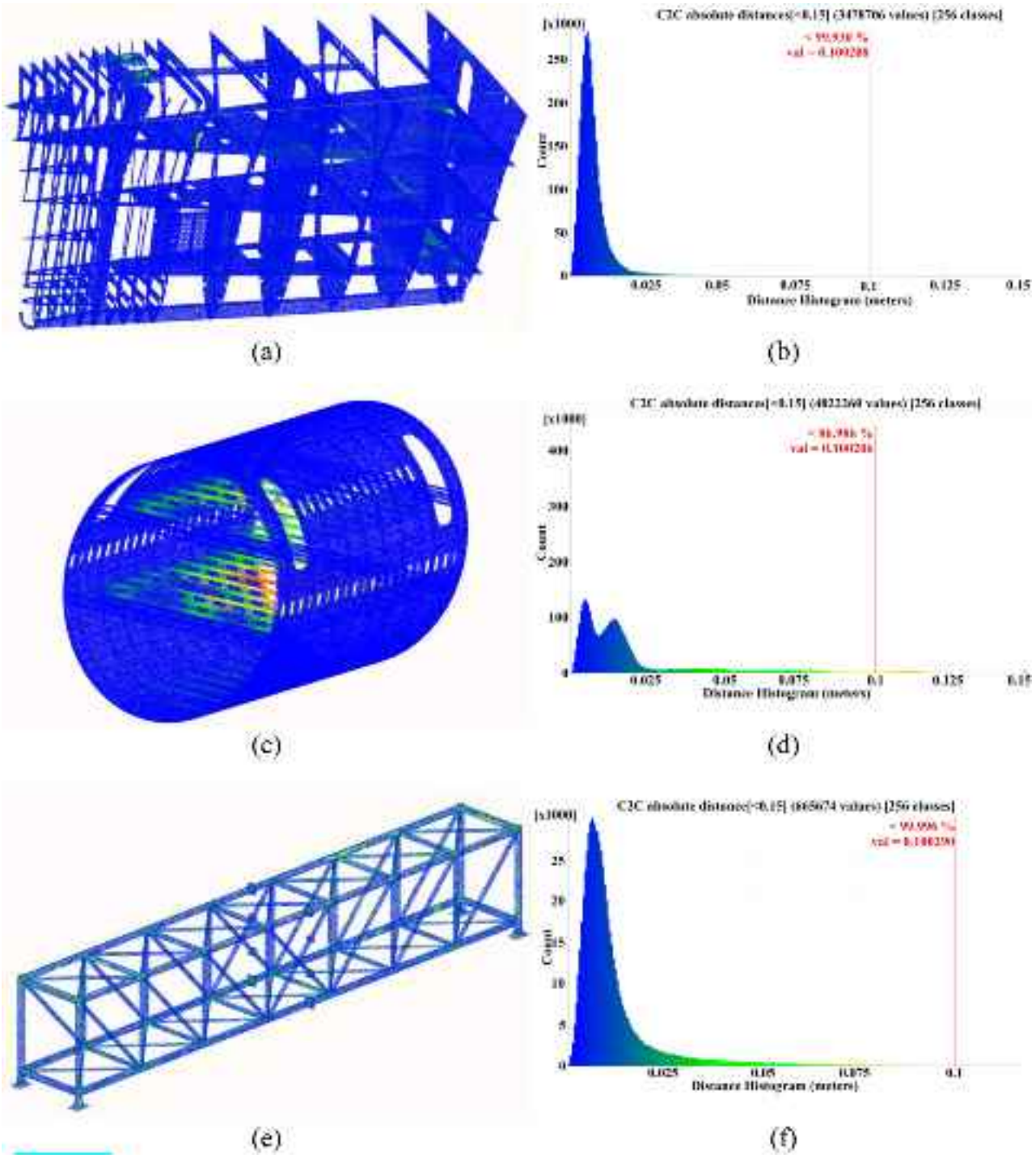

Fig. 21. The coverage analysis with distance map: (First Column) The point clouds of the original destroyer bow, fuselage, and pedestrian bridge, where the color changes from blue to red proportionally increasing distance, (Second Column) The distance histograms of the original destroyer bow, fuselage, and pedestrian bridge.

exhibited a coverage rate of 86.99 %. This lower rate is attributed to the fuselage's enclosed geometry compared to the destroyer bow. The depth camera was unable to capture points from the fuselage's internal regions, as the scan path was designed to cover only the structure's external surface.

6. Conclusion

In this paper, we presented a novel coverage path planning method for comprehensive sensor scanning of the outer surface of complex structures using a UAV equipped with a depth camera. Our aim is to generate a coverage path that minimizes flight distance while ensuring complete coverage of the structure's exterior. The proposed coverage path planning method comprises three main stages: 1) Bounding surface generation, 2) viewpoint acquisition, and 3) coverage path planning. To generate the bounding surface, the 3D alpha wrapping technique is employed, producing a 3D triangular mesh representation of the structure. Viewpoint determination involves several steps: initially, an initial viewpoint for each triangle is calculated based on the triangle's normal vector and centroid. These viewpoints are subsequently refined to ensure compatibility with the depth camera's FOV constraints and the UAV's motion limitations. Next, the triangles covered by each viewpoint

are identified, taking into account the depth camera's FOV and using a visibility graph-based approach. The minimum set of viewpoints required to cover all visible triangles is then determined using the SCP. Finally, the TSP is solved for the remaining viewpoints, optimizing a collision-free path for the UAV.

In the experimental studies, extensive evaluations were conducted on three distinct structures. First, hyperparameter optimization was performed for two critical parameters of the 3D alpha wrapping method. The effects of these parameters on different structures were thoroughly analyzed and discussed to provide readers with valuable insights. Subsequently, using the optimal alpha-offset parameter pairs that yielded the shortest path, scans were conducted on both the bounding surface mesh and the original model. The experimental results for the bounding surface mesh revealed that each triangle contained at least approximately 5000 points for all models, ensuring an accurate representation of the 3D mesh. Furthermore, the median UI values across all models were close to zero, indicating a generally uniform distribution of points within each triangle. For the original model, the experimental results demonstrated that the RMS values for both rotation and translation were near zero across all models, confirming the method's capability to scan without introducing translational or rotational errors. To assess coverage performance, additional analyses were performed, including a distance-based comparison between two point clouds. Assuming regions with a distance greater than 10 cm were unscanned, the destroyer bow and pedestrian bridge achieved coverage rates of 99.93 % and 99.99 %, respectively. In contrast, the fuselage exhibited a coverage rate of 86.99 %, attributed to its enclosed geometry and a scan path designed to target only the structure's external surface.

Future work will focus on developing a comprehensive path planning approach for both the interior and exterior surfaces of structures using the 3D alpha wrapping technique. For the interior regions, the methodology will involve employing significantly higher alpha values combined with minimal offset values, resulting in a substantial increase in the number of generated triangles. To address the computational challenges posed by this increased complexity, heuristic methods will be explored to efficiently solve the SCP and TSP for the large number of triangles.

Declaration of generative AI and AI-assisted technologies in the writing process

During the preparation of this work the authors used ChatGPT-4o in order to improve language and readability. After using this service, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

CRediT authorship contribution statement

Burak Kaleci: Writing – review & editing, Writing – original draft, Software, Methodology, Formal analysis, Conceptualization. **Gulin Elibol Secil:** Writing – review & editing, Visualization, Software, Formal analysis, Conceptualization. **Sezgin Secil:** Writing – review & editing, Visualization, Software, Investigation, Conceptualization. **Zühal Kartal:** Writing – review & editing, Software, Methodology, Conceptualization. **Metin Ozkan:** Writing – original draft, Supervision, Project administration, Methodology, Funding acquisition, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work was supported by the Scientific and Technological Research Council of Türkiye (TÜBİTAK) under project number 122E408 and project title "Development of Autonomous Robotic Inspection Method for Quality Inspection of Multi-Part Structures."

Data availability

Data will be made available on request.

References

[1] B.F. Spencer, V. Hoskere, Y. Narazaki, Advances in computer vision-based civil infrastructure inspection and monitoring, Engineering 5 (2) (2019) 199–222.
[2] J.Y. Park, J. Lange, O. Koc, F. Al-Bakhat, Design of an enhanced defect identification system for commercial building construction, in: 2017 Systems and Information Engineering Design Symposium (SIEDS), 2017, pp. 67–72.
[3] Pebsteel, (2024). https://pebsteel.com/en/types-of-steel-structures/, The last access: November 10, 2024.
[4] R.R. Iwańkowicz, Integration of ship hull assembly sequence planning, scheduling and budgeting. Advances in science and technology, Res. J. 9 (25) (2015) 12–19.
[5] PPG Aerospace, (2024). https://www.ppgaerospace.com/Products/Adhesives.html, The last access: November 10, 2024.
[6] Britannica, (2024). https://www.britannica.com/technology/bridge-engineering/Suspension-bridges, The last access: November 10, 2024.
[7] F. Wu, Y.J. Chen, L.Y. Yao, M. Hu, The development of hybrid ES-FE-SEA method for mid-frequency vibration analysis of complex built-up structure, Appl. Math. Model. 64 (2018) 298–319.
[8] Buckler Systems, (2024). https://bucklersystems.com/buckler-railway/, The last access: November 10, 2024.
[9] A. Ellenberg, A. Kontsos, F. Moon, I. Bartoli, Bridge related damage quantification using unmanned aerial vehicle imagery, Struct. Control Health Monit. 23 (9) (2016) 1168–1179.
[10] T. Rakha, A. Gorodetsky, Review of Unmanned Aerial System (UAS) applications in the built environment: towards automated building inspection procedures using drones, Autom. Constr. 93 (2018) 252–264.
[11] M. Tappe, D. Dose, M. Alpen, J. Horn, Autonomous surface inspection of airplanes with unmanned aerial systems, in: 2021 7th International Conference on Automation, Robotics and Applications (ICARA), 2021, pp. 135–139.
[12] L. Poggi, T. Gaggero, M. Gaiotti, E. Ravina, C.M. Rizzo, Robotic inspection of ships: inherent challenges and assessment of their effectiveness, Sh. Offshore Struct. (2020).
[13] S. Jordan, J. Moore, S. Hovet, J. Box, J. Perry, K. Kirsche, D. Lewis, Z.T.H. Tse, State-of-the-art technologies for UAV inspections, IET Radar Sonar Navig. 12 (2) (2018) 151–164.
[14] Y. Tan, S. Li, H. Liu, P. Chen, Z. Zhou, Automatic inspection data collection of building surface based on BIM and UAV, Autom. Constr. 131 (103881) (2021).
[15] R. Almadhoun, T. Taha, J. Dias, L. Seneviratne, Y. Zweiri, Coverage path planning for complex structures inspection using unmanned aerial vehicle (UAV)", Intell. Robot. Appl. (2022) 243–266.
[16] E. Galceran, M. Carreras, A survey on coverage path planning for robotics, Robot. Auton. Syst. 61 (12) (2013) 1258–1276.
[17] Z. Shang, J. Bradley, Z. Shen, A co-optimal coverage path planning method for aerial scanning of complex structures, Expert Syst. Appl. 158 (2020) 113535.
[18] F. Yan, E. Xia, Z. Li, Z. Zhou, Sampling-based path planning for high-quality aerial 3D reconstruction of urban scenes, Remote Sens. 13 (2021) 989.
[19] S. Jung, S. Song, P. Youn, H. Myung, Multi-layer coverage path planner for autonomous structural inspection of high-rise structures, in: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 2018, pp. 1–9.
[20] R. Almadhoun, T. Taha, D. Gan, J. Dias, Y. Zweiri, L. Seneviratne, Coverage path planning with adaptive viewpoint sampling to construct 3D models of complex structures for the purpose of inspection, in: 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2018.
[21] N. Bolourian, A. Hammad, LiDAR-equipped UAV path planning considering potential locations of defects for bridge inspection, Autom. Constr. 117 (2020) 103250.
[22] B. Englot, F.S. Hover, Sampling-based sweep planning to exploit local planarity in the inspection of complex 3D structures, in: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2012.
[23] C. Dornhege, A. Kleiner, A. Kolling, Coverage search in 3D, in: 2013 IEEE International Symposium on Safety, Security, and Rescue Robotics (SSRR), 2013.
[24] J. Li, P. Su, L. Qu, G. Lv, W. Qian, A CAD-based method for 3D scanning path planning and pose control, Aerospace 12 (8) (2025) 654.
[25] E. Glorieux, P. Franciosa, D. Ceglarek, Coverage path planning with targetted viewpoint sampling for robotic free-form surface inspection, Robot. Comput.-Integr. Manuf. 61 (2020) 101843.
[26] R. Almadhoun, T. Taha, J. Dias, L. Seneviratne, Y. Zweiri, Coverage path planning for complex structures inspection using unmanned aerial vehicle (UAV), intelligent robotics and applications (ICIRA 2019), Lect. Notes Comput. Sci. 11744 (2019) 243–266.
[27] Bircher, A., Alexis, K., Burri, M., Oettershagen, P., Omari, S., Mantel, T., & Siegwart, R. (2015). Structural inspection path planning via iterative viewpoint resampling with application to aerial robotics, IEEE international conference on robotics and automation (ICRA), 6423–6430.

[28] A. Bircher, M. Kamel, K. Alexis, M. Burri, P. Oettershagen, S. Omari, T. Mantel, R. Siegwart, Three-dimensional coverage path planning via viewpoint resampling and tour optimization for aerial robots, Auton. Robots 40 (2016) 1059–1078.
[29] F. Kong, F. Du, D. Zhao, Station-viewpoint joint coverage path planning towards mobile visual inspection, Robot. Comput.-Integr. Manuf. 91 (2025) 102821.
[30] M. Xi, Y. Wang, H. Liu, D. Li, H. Xiao, X. Li, X. Zhu, Inspection path planning of complex surface based on one-step inverse approach and curvature-oriented point distribution, IEEE Trans. Instrum. Meas. 71 (2022) 1–11.
[31] R.M. Claro, M.I. Pereira, F.S. Neves, A.M. Pinto, Energy efficient path planning for 3d aerial inspections, IEEE Access 11 (2023) 32152–32166.
[32] X. Huang, Y. Liu, L. Huang, S. Stikbakke, E. Onstein, BIM-supported drone path planning for building exterior surface inspection, Comput. Ind. 153 (2023) 104019.
[33] D. Applegate, W. Cook, A. Rohe, Chained Lin-Kernighan for large traveling salesman problems, INFORMS J. Comput. 15 (1) (2003) 82–92.
[34] K. Helsgaun, An effective implementation of the Lin–Kernighan traveling salesman heuristic, Eur. J. Oper. Res. 126 (1) (2000) 106–130.
[35] Y. Li, L. Zeng, K. Tang, C. Xie, Orientation-point relation based inspection path planning method for 5-axis OMI system, Robot. Comput.-Integr. Manuf. 61 (2020) 101827.
[36] M. Dorigo, L.M. Gambardella, Ant colonies for the travelling salesman problem, Biosystems 43 (2) (1997) 73–81.
[37] P.E. Hart, N.J. Nilsson, B. Raphael, A formal basis for the heuristic determination of minimum cost paths, Trans. Syst. Sci. Cybern. 4 (2) (1968) 100–107.
[38] C. Portaneri, M. Rouxel-Labbé, M. Hemmer, D. Cohen-Steiner, P. Alliez, Alpha wrapping with an offset, ACM Trans. Graph. 41 (4) (2022) 1–22.
[39] H. Choset, K.M. Lynch, S. Hutchinson, G.A. Kantor, W. Burgard, L.E. Kavraki, S. Thrun, Principles of Robot Motion: Theory, Algorithms, and Implementations, The MIT Press, 2005.
[40] R.M. Karp, Reducibility among combinatorial problems, in: R.E. Miller, J. W. Thatcher, J.D. Bohlinger (Eds.), Complexity of Computer Computations, Plenum, New York, 1972, pp. 85–103, https://doi.org/10.1007/978-1-4684-2001-2_9. ISBN 978-1-4684-2003-6.
[41] AlphaWrapping, (2024). https://doc.cgal.org/latest/Alpha_wrap_3/index.html, Access: August 19, 2024.
[42] GUROBI, (2024). https://www.gurobi.com/, The last access: October 01, 2024.
[43] Gazebo, (2024). https://gazebosim.org/, The last access: November 10, 2024.
[44] ROS, (2024). https://www.ros.org/, The last access: November 10, 2024.