*remote sensing*

MDPI

*Article*

# BIM-Based Trajectory Planning for Unmanned Aerial Vehicle-Enabled Box Girder Bridge Inspection

**Jiangpeng Shu [1,2,*], Zhe Xia [1] and Yifan Gao [1]**

[1] College of Civil Engineering and Architecture, Zhejiang University, Anzhong Building, 886 Yuhangtang Road, Hangzhou 310058, China; 12312114@zju.edu.cn (Z.X.); yfgao91@zju.edu.cn (Y.G.)
[2] Innovation Center of Yangtze River Delta, Zhejiang University, Jiaxing 314100, China
* Correspondence: jpeshu@zju.edu.cn

**Abstract:** Inspection is essential for bridge maintenance and has been supplemented by the use of unmanned aerial vehicle (UAV) photogrammetry. However, a review of the literature reveals that existing approaches require the intervention of a human operator to select waypoints in digital twin environments. Thus, existing studies are limited by either manual or semi-automatic control restrictions on UAV navigation settings, which is the main motivation for our work. This research developed a building information modelling (BIM)-based trajectory planning approach to enable fully autonomous UAV navigation for box girder bridge inspection, where the UAV follows a predetermined sequence to traverse the environmental space of a box girder bridge. The approach reflects the geometry properties of box girders as inspection characteristics and is designed with algorithms to determine a mathematical relationship between the box girders' coordinates and inspection sequences. Field testing of the approach shows that it has satisfactory performance in terms of (1) navigation efficiency, (2) reasonableness of the sequence determined for trajectory feature points, (3) trajectory closeness and deviation, and (4) trajectory smoothness. Its satisfactory performance supports the practicability of fully autonomous UAV-enabled bridge inspection.

**Keywords:** building information modelling; robotic inspection; box girder; trajectory planning; UAV

check for updates

Academic Editor: Massimiliano Pepe

Received: 17 January 2025
Revised: 14 February 2025
Accepted: 14 February 2025
Published: 17 February 2025

**Citation:** Shu, J.; Xia, Z.; Gao, Y. BIM-Based Trajectory Planning for Unmanned Aerial Vehicle-Enabled Box Girder Bridge Inspection. *Remote Sens.* **2025**, *17*, 682. https://doi.org/10.3390/rs17040682

**Copyright:** © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

## 1. Introduction

Visual inspections serve as the basis for determining the need and schedule for the maintenance and remediation of bridge infrastructure [1]. The traditional method relies on human inspection, which is susceptible to subjective errors and is limited by safety, access, and cost considerations [2]. Unmanned aerial vehicles (UAVs), in conjunction with computer vision technologies, have shown great potential for detecting bridge structure defects [3]. UAVs are capable of accessing places where human reach is difficult [4]. Computer vision technologies (e.g., image processing algorithms) then focus on acquiring high-level understanding from collected images for decision-making, such as evaluating structure conditions [5].

In recent years, UAV-enabled bridge inspections have received considerable attention from the scientific community. Jeong et al. [6] developed a CNN-based machine learning model that enhances UAV-captured image visibility by automatically optimising brightness, contrast, and sharpness parameters, thereby improving the accuracy of structural damage detection in bridge inspection workflows. Their framework demonstrated the integration of adaptive image pre-processing with deep learning to address low-visibility

challenges in field-acquired UAV data. Mahama et al. [7] tested the performance of different UAV-equipped cameras in bridge inspection under different illumination conditions (i.e., dark, intermediate, and bright) and found that UAVs with better-performing cameras could capture images with higher level sharpness. Kao et al. [8] employed a camera and laser ranging unit together on a UAV to develop an image processing method for bridge inspection, where the laser ranging unit is used to calculate a captured image's projection plane to overcome the limitation of vertical photography. Chandra et al. [9] developed an aerial mapping approach utilising UAV photogrammetry to generate detailed 3D models of bridges, facilitating comprehensive structural inspections. Their methodology demonstrated the potential for enhanced accuracy and efficiency in identifying structural deficiencies using advanced imaging techniques. Kim et al. [10] proposed a bridge component recognition method by segmenting point clouds collected by UAV cameras, utilising geometric and spatial features to classify structural elements. Their approach demonstrated significant accuracy in identifying bridge components, thereby providing a robust foundation for automated structural health monitoring. Xu and Zhang [11] developed a deep learning-based object detection method for bridge geometric shape measurement and component detection using images collected by UAV cameras. Ni et al. [12] proposed a generative adversarial learning network specifically designed for UAV-based crack detection in highway bridges, leveraging an adversarial training framework to enhance the accuracy and robustness of crack identification in complex structural environments. Their approach demonstrated significant improvements in handling noise and varying lighting conditions, which are common challenges in real-world bridge inspections. Bono et al. [13] and Shanthakumar et al. [14] developed BIM-based digital twin representations of bridges, serving as reference environments for planning UAV inspection trajectories, thereby enhancing precision and efficiency in structural assessments. These models facilitate the integration of autonomous navigation systems by providing accurate spatial data for trajectory optimization and obstacle avoidance.

As can be seen, seven of the existing studies on UAV bridge inspection focus on developing computer vision technologies for image and point cloud processing, and lack trajectory planning for UAV navigation. The piloting of UAVs in their studies relies on manual inputs. The other two studies by Bono et al. [13] and Shanthakumar et al. [14] focus on trajectory planning for UAV navigation. However, their approaches also require the intervention of a human operator to select waypoints in digital twin environments. Thus, existing studies are limited by either manual or semi-automatic control restrictions on UAV navigation settings, which is the main motivation for our work.

In addition to the construction sector, researchers in other fields (e.g., computer science) have developed algorithms to assist the navigation of UAVs. For example, Zhang et al. [15] and Zhang et al. [16] utilised the A-star algorithm for UAV trajectory planning in a 3D dynamic environment. A-star is a graph search method that samples nodes in a graph-represented environment and connects the start and end points to form a trajectory. Guo et al. [17] developed a feedback rapidly exploring random tree star algorithm (FRRT*), which can use spatial information extracted from situation data to constrain the growth of the random tree (i.e., the formed trajectory) and make biased adjustments to improve planning efficiency. Fan et al. [18] and Fan et al. [19] proposed artificial potential field (APF)-based methods to guide the growth of the random tree in RRT* towards the target point by adding random point attraction, target point attraction and obstacle repulsion. The nature of the A-star and RRT* methods are both based on the feature of randomly sampling trajectory nodes and connecting the viable ones, which could lead to extended convergence time [20]. Given that BIM contains georeferenced properties that can be used to determine inspection coordinates and establish automatic control signals for UAV navigation, we

aim to utilise the BIM-contained geometry properties as trajectory planning constraints to address the "randomly sampling" problem in the existing methods.

In this paper, we describe our work on developing a fully autonomous UAV approach for box girder bridge inspection. The scientific question of the present study is how to expand the digital blueprint of a box girder bridge in BIM into UAV robotic control instructions. That is, how to determine a mathematical relationship between the coordinates of box girder units in BIM and an inspection sequence (considering the vertexes of the box girder). Therefore, this research aims to address this scientific question and provide a UAV-based approach that consists of the following:

(1) A task planning algorithm that can determine a reasonable sequence of vertex coordinates of box girder units in BIM for bridge inspection.
(2) A trajectory planning algorithm that can analyse the determined inspection sequence and coordinates and generate the UAV's kinematic states for performing the inspection of box girder bridges.

It is worth mentioning that the developed approach is not confined solely to the facilitation of autonomous UAV inspections for bridges. Rather, it possesses the versatility to be extended to a broader array of infrastructure types that necessitate periodic inspections, including but not limited to buildings and dams. The rest of the paper is organised as follows: Section 2 provides an extended literature review on trajectory planning-related studies in construction, Section 3 demonstrates the methodology of the study, Section 4 presents the overall architecture of the proposed approach and interprets the self-contained task and trajectory planning algorithms, Section 5 demonstrates the testing results of the developed approach, Section 6 discusses the theoretical and practical implications, and Section 7 summarises the findings, notes the limitations, and recommends future research directions.

## 2. Literature Review

In addition to UAV-related studies, as mentioned in Section 1, other studies on trajectory planning in construction engineering fields (e.g., robotics and mobile cranes) were also reviewed to clarify the theoretical underpinnings of this work.

Terada and Murata [21] developed a gradient field-based Bucket Brigade algorithm to identify optimised trajectories for concrete block assembly. King et al. [22] designed a Rhinoceros Grasshopper-based approach for task planning in tile mosaic pattern installation and utilised ABB RobotStudio to generate placement trajectories for each tile. Davtalab et al. [23] used the Lin-Kernighan algorithm to solve toolpath optimisation problems in the concrete contour crafting process. Kontovourkis et al. [24] adapted the Bidirectional Evolutionary Structural Optimisation algorithm for nozzle trajectory planning in additive manufacturing. Ding et al. [25] developed a deep convolutional neural network for the visual identification of brick patterns and utilised the ABB RobotStudio software for brick assembly trajectory planning. Using ABB RobotStudio, the trajectory can be determined, provided the start and goal (assembly) coordinates of each brick are provided. Wiranugeraha et al. [26] utilised a fifth-degree polynomial to generate cable-driven parallel robot pick-and-place operations for the evacuation process in post-disaster sites. The fifth-degree polynomial approach can establish position, velocity, and acceleration profiles for the robot's end effector. Zhu et al. [27] developed an artificial neural network (ANN)-based trajectory planning approach to control the swing of the load during the transportation of a rotating crane within a certain range. The approach is time-optimisation-based and can detect obstacles in the vicinity of the crane boom for detouring. Safouhi et al. [28] developed an outside boundary limit (OSBL)-based approach to determine the feasible mobile crane position area (FMCPA) and the mobile crane body workable area (MCBWA) in the two-dimensional space of computer-aided design (CAD) drawings. Lu et al. [29]

developed a time and spatial differential-based trajectory planning method for offshore boom cranes, where the hoisted cargoes can be delivered to designated sites by following its generated crane motions (e.g., boom luffing rotating, rope hoisting).

Gao et al. [30] and Shu et al. [31] developed BIM-based prototypes to assist in the robotic assembly of lightweight steel structures such as COVID-19 flatpack hospitalisation facilities. Their prototypes can determine both the assembly coordinates and sequence of prefabricated components for a given BIM model and generate robots' kinematic parameters for performing the assembly of the prefabricated components. Given that BIM contains georeferenced properties that could be used to determine inspection coordinates and establish automatic control signals for UAV navigation, we aim to utilise the BIM-contained geometry properties to generate feature points for bridge inspection trajectory planning in our approach.

## 3. Methodology

Figure 1 presents a schematic representation of a box girder bridge. This is the classic configuration of a simply supported girder bridge. It can be seen that box girder bridges structurally consist of two types of components: box girder beams and piers, which form multiple continuous segments of the bridge deck (Figure 1b,c). The component properties of the schematic model are provided in Table 1. Note that this research focuses on automating the bridge inspection procedure using UAV trajectory planning. A UAV is intended to replace human labour in the inspection process, where the technical challenges lie in letting the UAV and its mounted camera autonomously follow a predetermined sequence to traverse the environmental space of the box girder bridge while passing through the coordinates where they are needed in space without human intervention.

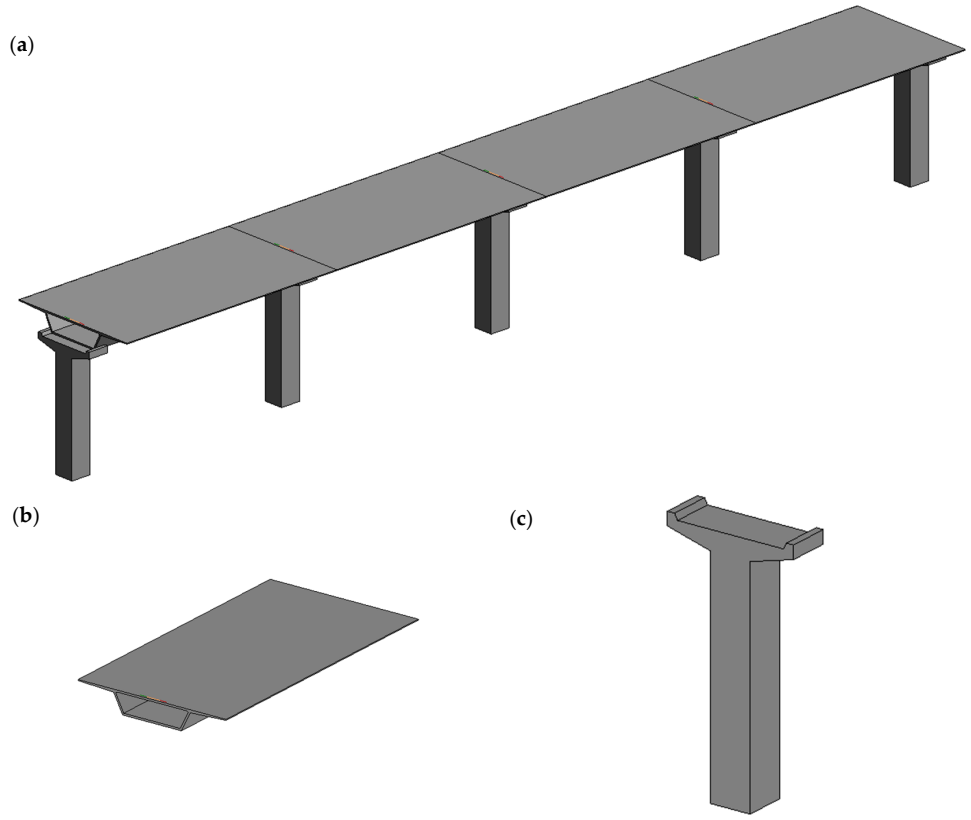



**Figure 1.** Schematic representation of a box girder bridge (**a**) and its components (**b**) a box girder beam unit and (**c**) a pier.

**Table 1.** Component properties of the bridge model used in this study.

| Components | Materials | Component Properties | Number |
|---|---|---|---|
| Box Girder Beam | Concrete | (1) Shape: octagon<br>(2) Number of sides: 8<br>(3) Number of vertices (corners): 8 | 4 |
| Pier | Concrete | (1) Shape: dodecagon<br>(2) Number of sides: 12<br>(3) Number of vertices (corners): 12 | 5 |

This research explores how autonomous bridge inspections can be achieved using UAV trajectory planning. As discussed, the motion control of a UAV consists of letting it follow a predetermined "sequence" of "coordinates". Thus, the challenge of such a solution lies in the generation of a reasonable inspection "sequence" for the UAV to follow such that it can traverse the transverse environmental space of the box girder bridge while passing through the coordinates where they are needed in space. To provide an approach that can overcome this challenge, this research considers expanding the digital blueprint of the box girder bridge in BIM into UAV robotic control instructions. The process generates a task planning algorithm and a trajectory planning algorithm. As Ding et al. [25] pointed out, BIM models contain large quantities of spatial information that can be used for construction and maintenance activities. For example, BIM models are composed of loadable families (known as "library components"), which are graphical representations of prefabricated building components [24] (Figure 2). To create a BIM project, the families are spatially integrated (Figure 2). The process of integrating families into a unifying BIM project creates georeferenced properties that are useful for determining the inspection coordinates and the sequence of building components.

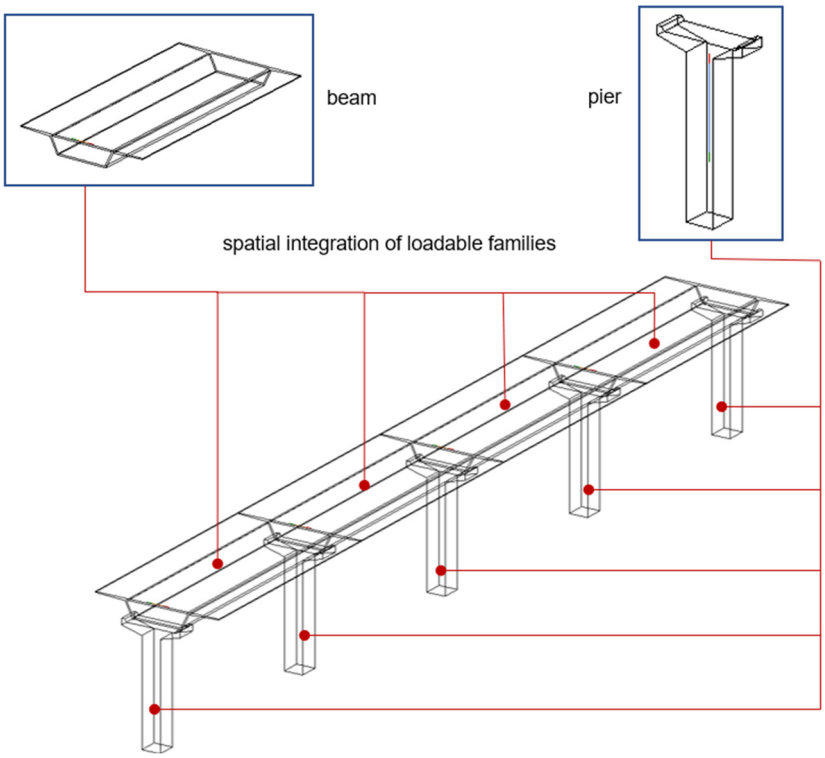



**Figure 2.** Spatial integration of loadable families into building information modelling (BIM) from Revit 3D Viewer.

BIM design tools host informative databases for their projects, where a graphical programming interface can be utilised to couple with and democratise the database for end-users to access data and retrieve desired features [32]. In this research, BIM and the Robot Operating System (ROS) were utilised to develop the UAV robotic approach for inspecting the box girder bridge, where BIM's role is to provide the required data input for ROS. A BIM-based task planning algorithm was developed to determine the mathematical relationship between the coordinates of box girder units and the inspection sequence, considering their vertexes, for box girder bridge inspection. The inspection sequence data constitute the information required for UAV inspection trajectory planning. Then, the ROS executes the trajectory planning algorithm to analyse the inspection sequence data and generate the UAV's kinematic parameters for performing the inspection of the box girder bridge. A communication interface was also established to operate the data transmission between the task and trajectory planning layers. An overview of the architecture of the proposed approach is shown in Figure 3. The detailed steps are discussed in the following sections.

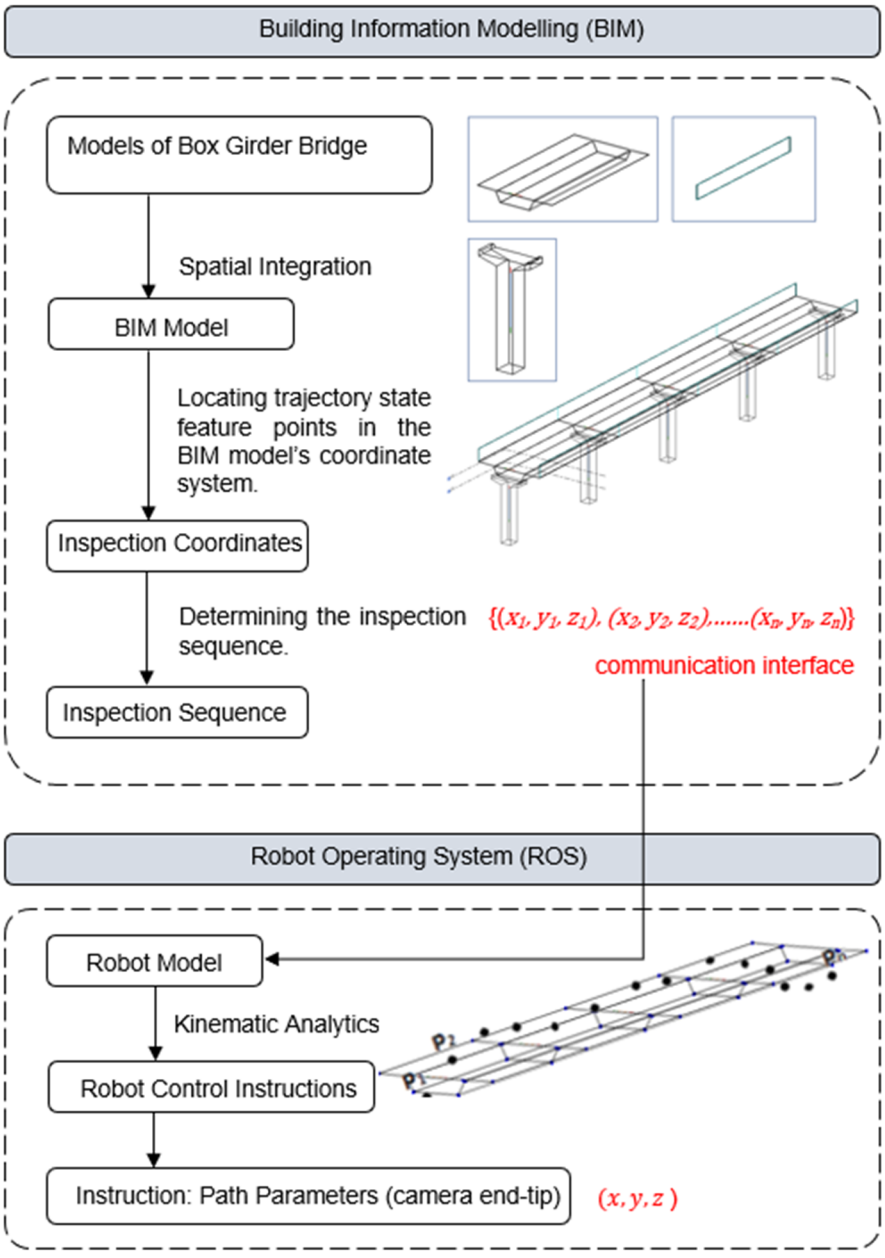
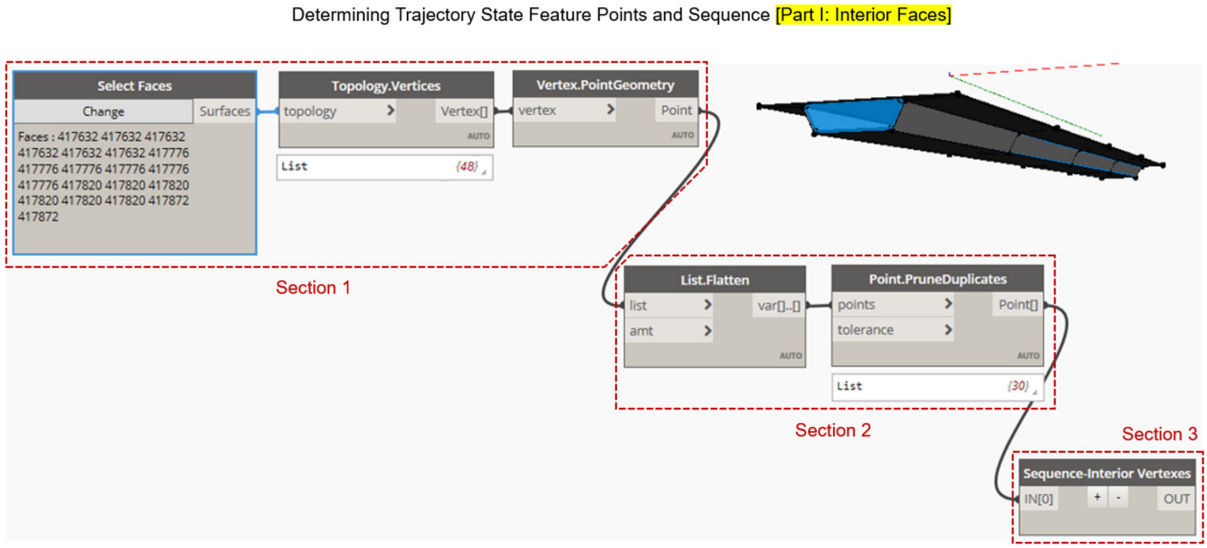



**Figure 3.** An overview of the architecture of the proposed approach.

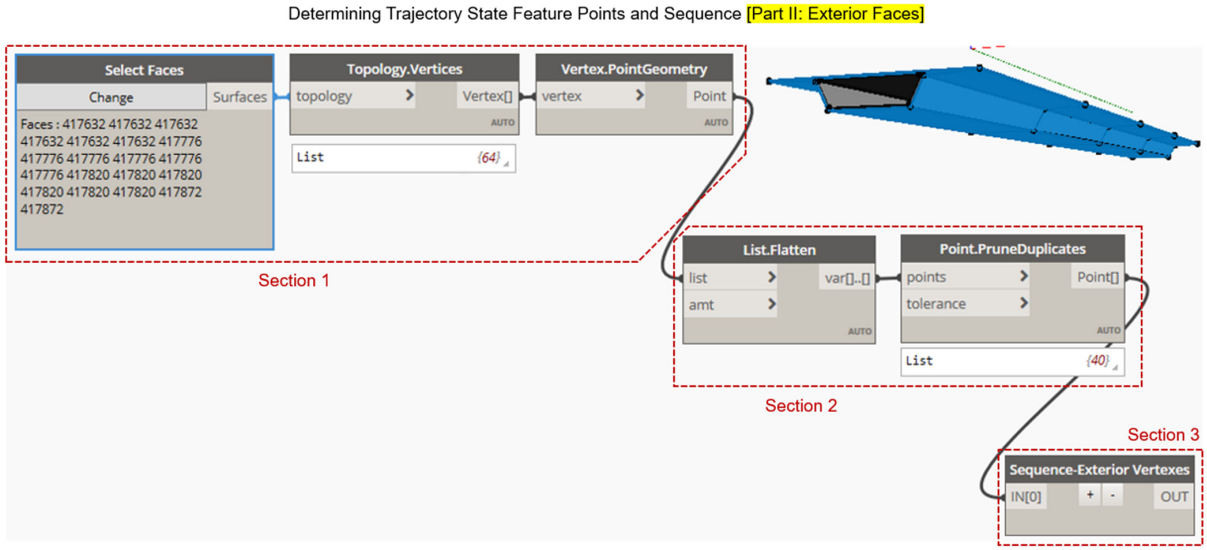
## 4. System

### *4.1. Task Planning Algorithm: Determining Trajectory State Feature Points and Sequence*

In this research, the task planning logic includes two steps: (1) retrieving feature points that can describe the geometry of the box girder beam (as well as the geometry of the trajectory) from the BIM model, and (2) sorting the feature points in a reasonable order as the inspection trajectory states. The authors utilised the BIM graphical programming interface to develop the task planning algorithm, which is named the Trajectory State Feature Points and Sequence Determination (TSFPSD) algorithm. The analytic capability of the BIM graphical programming interface is enabled through functional nodes, which are composed of input and output ports, and are connected in sequence to form a complete logic [33] (see Figure 4). Users can program in Python to create nodes for specific functions [32] (e.g., *Inspection Sequence ()* in Figure 5). The composition of the TSFPSD algorithm is presented in Figure 4 and Algorithm 1. As can be seen, the algorithm consists of three sections, which are discussed in greater detail in the following paragraphs.




(**a**)

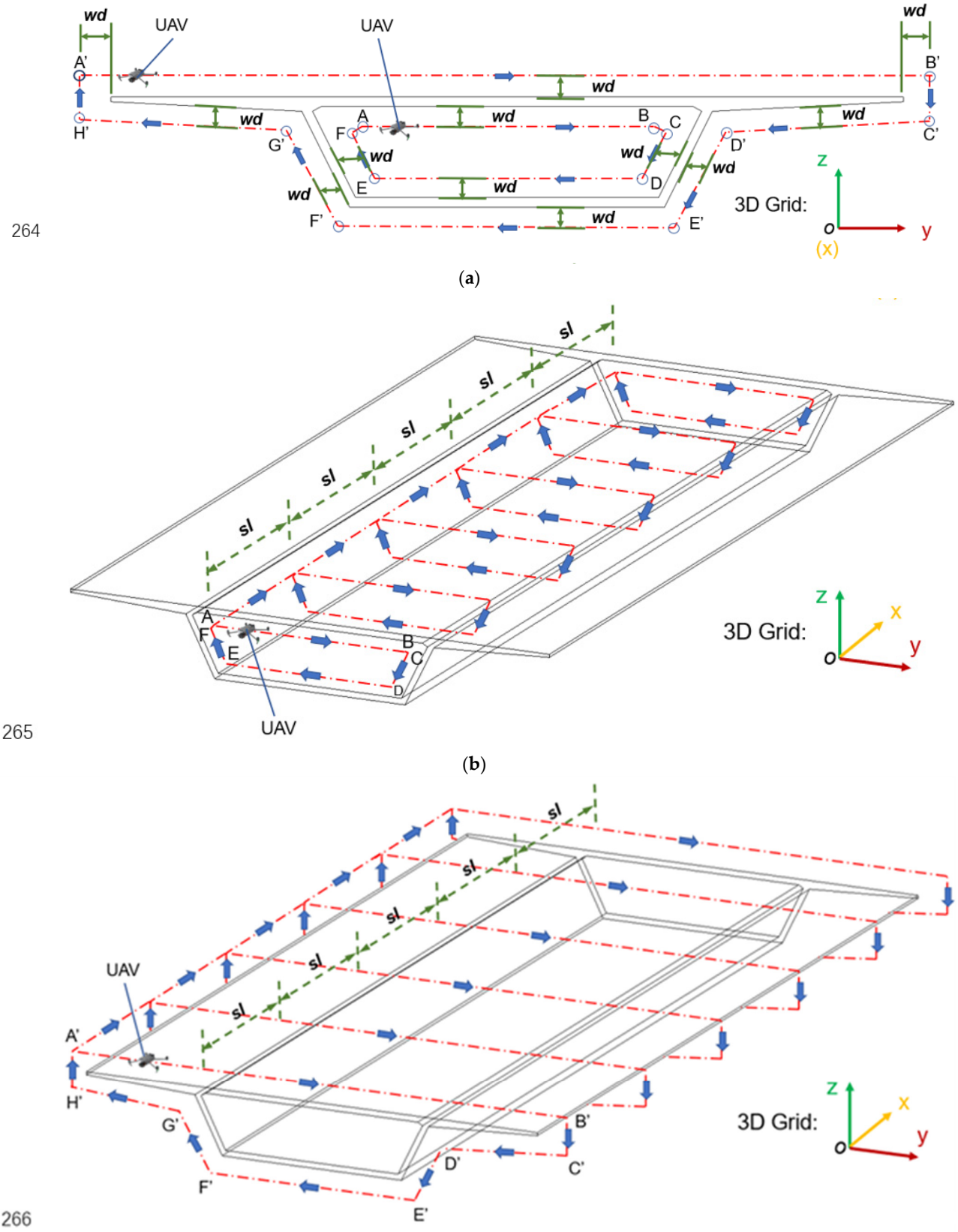



(**b**)

**Figure 4.** Trajectory State Feature Points and Sequence Determination (TSFPSD) algorithm: (**a**) determining trajectory state feature points and sequence for interior faces (Part I); (**b**) determining trajectory state feature points and sequence for exterior faces (Part II).

**Algorithm 1:** Trajectory State Feature Points and Sequence Determination (TSFPSD)

```
orglst1 ← face.categories(interior shell);
orglst2 ← face.categories(exterior shell);
orglst1 ← orglst1.topologyVertices();
orglst1 ← orglst1.pointGeometry();
orglst2 ← orglst2.topologyVertices();
orglst2 ← orglst2.pointGeometry();
orglst1 ← orglst1.Flatten();
orglst1 ← orglst1.pruneDuplicates();
orglst2 ← orglst2.Flatten();
orglst2 ← orglst2.pruneDuplicates();
sortPointsClockwise(yz(x), orglst1, orglst2):
        orglst1.sort(key = lambda orglst1: orglst1.X);
        orglst2.sort(key = lambda orglst2: orglst2.X);
        orglst1[i] ← [orglst1[i : i + 6] for i in range(0, len(orglst1), 6)];
        orglst2[i] ← [orglst2[i : i + 8] for i in range(0, len(orglst2), 8)];
        for point in orglst1[i]:
                y_center[i] ← statistics.mean(orglst1[i].Y);
                z_center[i] ← statistics.mean(orglst1[i].Z);
                point(y) − = y_center[i];
                point(z) − = z_center[i];
                angle = math.atan2(point(y), point(z));
                orglst1[i] ← sorted(orglst1[i], key = −angle);
                orglst1 ← append(orglst1[i]);
        end
        for point in orglst2[i]:
                y_center[i] ← statistics.mean(orglst2[i].Y);
                z_center[i] ← statistics.mean(orglst2[i].Z);
                point(y) − = y_center[i];
                point(z) − = z_center[i];
                angle = math.atan2(point(y), point(z));
                orglst2[i] ← sorted(orglst2[i], key = −angle);
                orglst2 ← append(orglst2[i]);
        end
end
OUT ← {orglst1, orglst2}
```




(**a**)

(**b**)

(**c**)

Note: *wd*: work distance; *sl*: step length

**Figure 5.** Schematic overview of the inspection trajectory: (**a**) sectional view, (**b**) side view of interior shell inspection trajectory, and (**c**) side view of exterior shell inspection trajectory.

Section 1 of the algorithm seeks to identify the vertices of all box girder beam units as trajectory state feature points from the BIM model in Figure 1a,b. This is enabled by nodes *Select Faces ()*, *Topology.Vertices ()*, and *Vertex.PointGeometry ()* (Figure 4). In BIM, the cache keeps track of the building components by attaching a unique identifier ID to each component. *Select Faces ()* returns the identifier IDs of the faces of the box girder beam units from the BIM model. Given that this research aims to plan trajectories for inspecting both the interior and exterior shells of the box girder beam units, the face-selecting task consists of two sections, namely, interior faces and exterior faces. *Topology.Vertices ()* detects the vertexes of each selected face and plots the vertexes (represented by black dots in Figure 4). *Vertex.PointGeometry ()* then takes the list of identified vertexes and retrieves the geometry as represented by the vertexes. In the list of vertexes, there are overlapping points at the spatial locations where two beam units are connected (Figure 6a). However, for the trajectory planning purpose, the overlapping state feature points only need to be considered once. Using the node *Point.PruneDuplicates ()*, Section 2 of the algorithm, is designed to trim the list of the identified vertexes and remove the repeated points. As can be seen in Figure 6b, the section profile of the example box girder beam unit has six interior vertexes and eight exterior vertexes. For the four connected beam units, there are three overlapping faces. As can be seen in Figure 4, after applying Section 2 of the TSFPSD algorithm, the total vertexes for interior and exterior faces before and after the trim are as follows: (a) 48 vertexes (6 vertexes × 8 edges) for interior faces before the trim; (b) 30 vertexes (6 vertexes × 5 edges) for interior faces after the trim; (c) 64 vertexes (8 vertexes × 8 edges) for exterior faces before the trim; and (d) 40 vertexes (8 vertexes × 5 edges) for exterior faces after the trim. The vertexes on the three overlapping faces have been successfully removed.

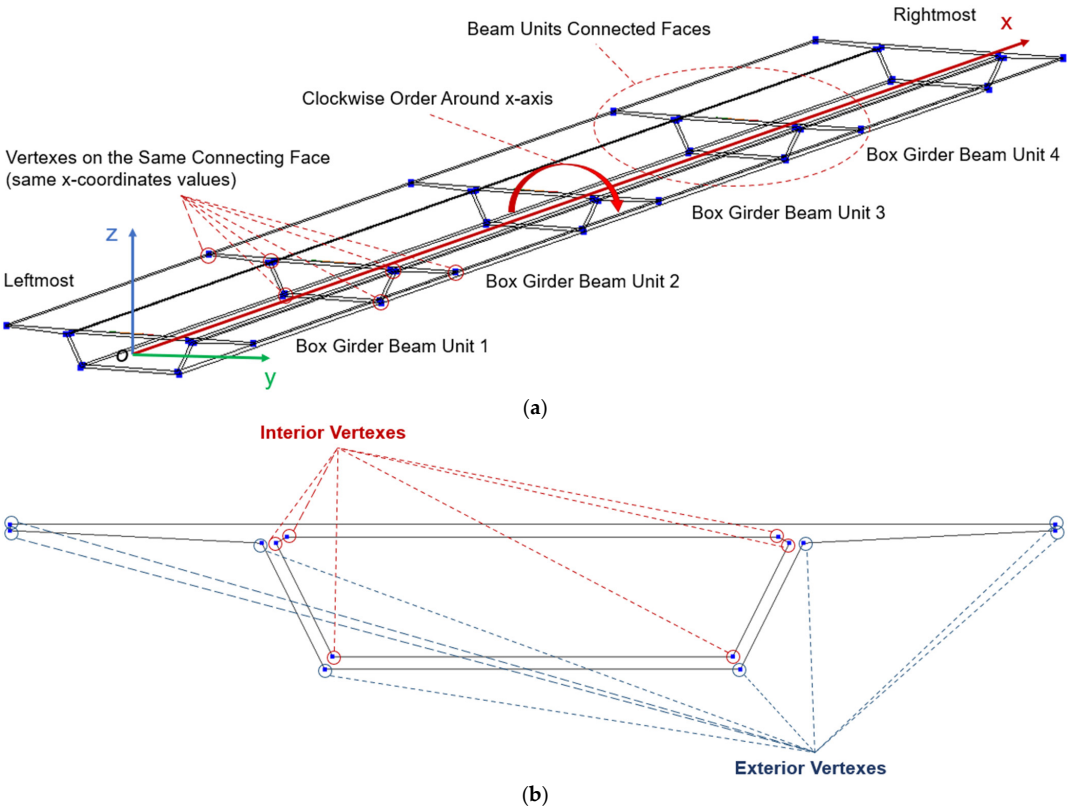



**Figure 6.** Overlapping points: (**a**) locations where two box girder beam units are connected; (**b**) interior and exterior vertexes of box girder beam unit's section profile.

The pruned lists of interior and exterior vertexes from the node *Point.PruneDuplicates ()* in Section 2 are passed to the *Sequence-Interior/Exterior Vertexes ()* node in Section 3. The

*Sequence-Interior/Exterior Vertexes ()* node is applied to determine the reasonable sequences of trajectory state feature points for inspecting the interior and exterior shells of the box girder beam units. As can be seen in Algorithm 1, the *Sequence-Interior/Exterior Vertexes ()* node functions by following a number of steps:

(1) Deriving a $\vec{x}$ vector that points along the *x*-axis, which is used to detect in sequence the x-coordinate values of the identified vertexes. This process generates a list of vertexes arranged in ascending order along the *x*-axis (from the leftmost to the rightmost). Note that the *x*-axis is along the longitudinal axis of the box girder beam. The *y*-axis is along the lateral axis. The *z*-axis points upward.
(2) Splitting the vertex list into subgroups of points with the same x-coordinate values.
(3) Sorting the points in the subgroups in clockwise order by applying the angle function: angle = math.atan2(point(y), point(z)) (Figure 6a).

The middleware interface in Figure 7 is designed for data transmission between the task and trajectory planning layers, which is enabled by functional nodes at BIM, IFC, and ROS terminals. First, the node *Export_IFC ()* at the BIM terminal takes the sorted coordinates sequence lists of interior and exterior shells and exports the lists to an IFC file. The node *IfcAxis2Placement3D ()* at the IFC terminal then organises data entries in a string form that can be parsed by the ROS system as follows: "#IFC Identifier = ifcPropertyStringValue(Parameter Label). coordinate(Parameter Content)" (see Figure 7). The node *Subscriber ()* at the ROS terminal is responsible for parsing the inspection coordinates and sequence data from the IFC tags. The kinematic parameters of the UAV for inspecting the interior and exterior shells of the box girder bridge are then generated based on the IFC data provided (see "Kinematic Interface" in Figure 7). Finally, the node *Publisher ()* at the ROS terminal publishes the generated kinematic parameters for controlling the UAV.

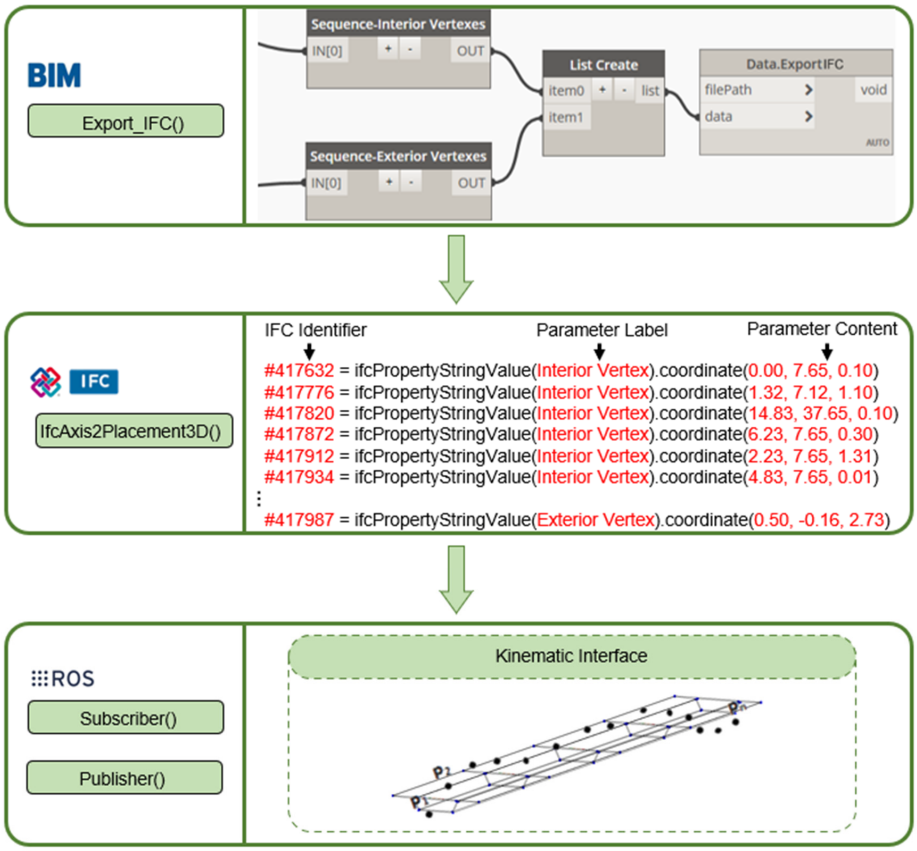



**Figure 7.** Middleware interface for data transmission between the task and trajectory planning layers.

### *4.2. Trajectory Planning Algorithm: Generating Kinematic States*

At the trajectory planning level, the algorithm analyses the sequence of the feature points as determined in the task planning layer and generates the UAV's kinematic states for performing the inspection of the interior and exterior shells of the box girder beam units. A schematic overview of the inspection trajectory is shown in Figure 5. The goal is to allow the algorithm to form a series of connected clockwise twists passing through all the feature points for the UAV to follow the trajectory to inspect. Given that a UAV can pitch, yaw, and roll and can be equipped with a 360° panorama camera to enlarge its field of view, the planning in this research mainly focuses on navigating the locations of the waypoints and smoothening the trajectory to eliminate sharp turns and jerks.

As shown in Figure 5, there are two parameters that need to be determined before designing and applying the trajectory planning algorithm, namely, *wd* and *sl*. The parameter *wd* refers to the work distance of the UAV from the faces of the interior and exterior shells (see Figure 5). As indicated in the existing literature [34,35], flying UAVs indoors should be performed in a safe manner, where UAVs should avoid flying close to and maintain a constant distance from wall surfaces. The flow of air is impeded as a UAV approaches the wall surface [35]. It is further pointed out that the UAV should maintain a minimal distance of 100 cm from the wall surface to avoid uneven airflow around the UAV [34]. Therefore, a distance of 100 cm was selected for *wd*. In addition, the parameter *sl* refers to the step length of the UAV to move forward along the *x*-axis to engage in a further clockwise twist inspection after the current loop $A \rightarrow B \rightarrow C \rightarrow D \rightarrow E \rightarrow F \rightarrow A$ or $A' \rightarrow B' \rightarrow C' \rightarrow D' \rightarrow E' \rightarrow F' \rightarrow G' \rightarrow H' \rightarrow A'$ is completed (see Figure 5). As empirically tested in the existing literature [36], for small scale patterns (e.g., cracks not wider than 0.1 mm), camera vision showed measurability within 150 cm shooting distance from the inspected surface. With this evidence, the *sl* distance was preliminarily computed to be 112 cm by applying the Pythagorean theorem as shown in Figure 8. To provide a more secure distance for the measurability of cracks, a smaller *sl* distance of 100 cm was selected.

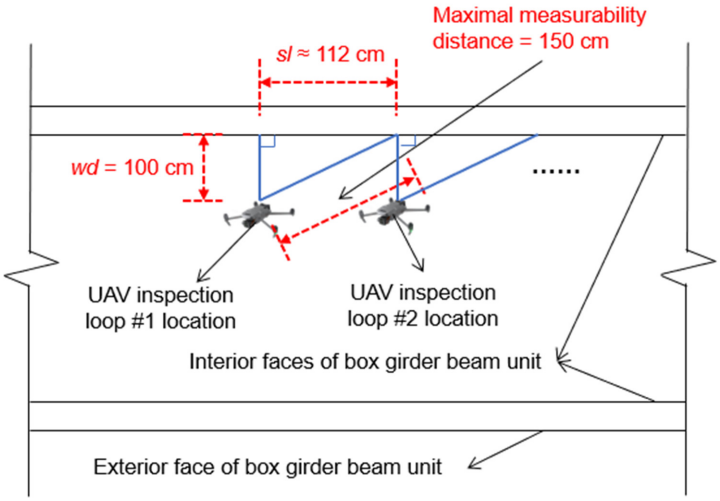



**Figure 8.** Applying the Pythagorean theorem to calculate *sl* values.

To generate the UAV's kinematic states for autonomously manoeuvring across the inspection trajectory shown in Figure 5, a trajectory planning algorithm was applied. For the trajectory planning in this case, a sampling-based algorithm, i.e., Rapidly Exploring Random Tree Star (RRT*), was varied based on a previous work [37].

To justify the use of RRT*, we investigated the categorisation of trajectory planning algorithms and compared the identified algorithms. It was found that trajectory planning can take a classical or advanced approach [38]. The classical approach is used for trajectory

planning in a predictable environment, where the obstacles are static and there is no need for real-time re-planning. The advanced approach is used to deal with trajectory planning in an unpredictable environment with dynamic obstacles, which entails making real-time modifications to trajectories so that dynamic obstacles are avoided. Given that UAV navigation is usually applied in dynamic environments, the category of advanced approaches is thus our major focus. The advanced approach can be further classified into optimisation- and sampling-based categories [39]. The basic idea of optimisation-based algorithms is to establish a set of cost functions that define some considerations as to what an expected optimum trajectory might look like (e.g., minimal trajectory completion time, obstacle constraints), and then identify the optimum trajectory that can minimise the cost functions. However, one of the concerns with optimisation-based algorithms is that they depend on the minimisation of cost functions, which are not always successful in finding the global minimum and can become trapped in a local minimum. The basic idea of sampling-based algorithms is to sample random points in space and reject points that overlap obstacles to form a collision-free trajectory. Rapidly Exploring Random Tree-Star (RRT*) is a widely used algorithm in this category, which has the following advantages: (1) inherent asymptotical convergence to the global optimality, (2) not requiring prior knowledge of the environment, (3) running in real time, and (4) planning trajectories in unpredictable environments with dynamic obstacles [40]. Given the investigation findings, RRT* was selected as the planning approach, and we varied the algorithm to enable the query of the BIM terminal and the clone of feature point coordinates and sequences into the ROS terminal.

The detailed process of the varied RRT* algorithm is presented in Algorithm 2 and is explained below:

**Algorithm 2:** Varied RRT*. Note : $wd = 100$ cm, $sl = 100$ cm, $d = 0.1$ cm.

orglst $\leftarrow$ subscribe(BIM);
$\eta \leftarrow$ initialiseTree();
$x_{startConfig} \leftarrow$ orglst$[i]$;
$x_{goalConfig} \leftarrow$ orglst$[i+1]$;
$\eta \leftarrow$ insert$\left(x_{startConfig}\right)$;
for $i = 1, 2, \ldots, N$ do
    $x_{rand} \leftarrow$ sample$(\chi_{free}, N)$;
    $x_{nearest} \leftarrow$ nearest$(\eta, x_{rand})$;
    $x_{new} \leftarrow$ steer$(x_{nearest}, x_{rand}, \Delta_d)$;
    $X_{near} \leftarrow$ near$(\eta, x_{new}, r)$;
    $x_{mid} \leftarrow \min\left(x_{new}, X_{near}, x_{startConfig}\right)$;
    if $0 \notin \chi_{uav} \ominus \chi_{env} = \{a - b | a \in \chi_{uav}, b \in \chi_{env}, x_{new} \leftarrow x_{mid}\}$ then
        $\eta \leftarrow$ insert$(x_{new}, x_{mid})$;
        $\eta \leftarrow$ edge$(x_{new}, x_{mid})$;
        $\eta \leftarrow$ side$(-wd, \eta)$;
        $\eta \leftarrow$ kinodynamicRewiring$(\eta)$;
        $\eta \leftarrow$ interpolate$(sl, \eta)$;
    end
    if $\left\| x_{new} - x_{goalConfig} \right\| < d$ *then*
        Break;
    end
end
OUT $\leftarrow \{\eta\}$

- First, the node subscribe(BIM) queries the BIM terminal and clones the orglst data (feature points coordinates and sequence) as determined in the task planning layer into the ROS terminal.
- Second, the function initialiseTree() initialises an empty tree set $\eta$, which will be assigned with waypoints to form a trajectory for the UAV.
- Third, the $x_{startConfig}$ and $x_{goalConfig}$ parameters register, respectively, the $i$th and $i+1$th elements in the orglst list as the start and goal configurations for the $i$th loop of trajectory planning. For example, the "A" and "B" feature points in Figure 5b are the 1st and 2nd elements in the orglst list for the 1st loop of trajectory planning.
- Fourth, the function insert() adds the start point $x_{startConfig}$ to $\eta$ as the root vertex in each planning loop.
- Fifth, the function sample() generates a random state $x_{rand}$ in the collision-free space $\chi_{free}$ based on a uniform distribution. After the function nearest() finds the state $x_{nearest}$ in the tree $\eta$ that is closest to the random state $x_{rand}$ in the collision-free space $\chi_{free}$, the function steer() generates a new state $x_{new}$ along the direction from $x_{nearest}$ to $x_{rand}$ at a foot step $\Delta_d$ for the state progression of the ROS system. The function near() collects a set of states $X_{near}$ in $\eta$, which is located in a spherical space with $x_{new}$ as the centre and $r$ as the radius. The function min() evaluates the travel cost from $x_{start}$ to $x_{new}$ through each state in $X_{near}$ and identifies the state $x_{mid}$ with the lowest cost.
- Sixth, the function if() checks whether the UAV $\chi_{uav}$ and the environmental objects $\chi_{env}$ share points in common along the edge between the $x_{new}$ and $x_{mid}$ states. If not, the UAV $\chi_{uav}$ is in the collision-free space $\chi_{free}$.
- Seventh, if the if() condition is met, the functions insert() and edge(), respectively, add the states $x_{mid}$ and $x_{new}$ to the trajectory tree $\eta$ and link the edge between the states $x_{mid}$ and $x_{new}$. The function side() contracts the trajectory tree $\eta$ with a predetermined variable $-wd$ as shown in Figure 5.
- Eighth, the function kinodynamicRewiring() smoothens the edges in the trajectory tree $\eta$ to eliminate sharp turns and jerks. The function interpolate() further enriches the trajectory tree $\eta$ by adding more intermediate states and loops at a predetermined distance $sl$ as shown in Figure 5.
- Ninth, the statement if $\left\| x_{new} - x_{goalConfig} \right\| < d$ checks whether the state $x_{new}$ is as close as possible to the goal statement $x_{goalConfig}$, and finalises the current loop in the planning if the distance value between $x_{new}$ and $x_{goalConfig}$ is smaller than the predefined threshold $d$.

## 5. Testing of the Approach

Field tests were performed to evaluate the developed approach in terms of (1) navigation efficiency, (2) reasonableness of the sequence determined for trajectory feature points, (3) trajectory closeness and deviation, and (4) trajectory smoothness.

### *5.1. Navigation Efficiency*

The authors carried out a field experiment to validate the navigation efficiency of the developed approach. The field project is a two-way viaduct bridge, which is the 12th bridge of the Hetianxi section and is located at the 616 County Highway, Quzhou City, Zhejiang Province in China (Figure 9a,b). The model of the UAV used in the field test is DJI Phantom 4 equipped with Zenmuse X5S (Figure 9c). The results showed that the UAV could adhere to the trajectory planned by the developed approach with no major deviation (Figure 9d–f). Following the planned trajectory, the UAV's mounted camera successfully captured several structural cracks from the field project, where an example is presented in Figure 9g. In the field test, the UAV navigation time for bridge inspection applying both the developed

autonomous approach and the traditional human control method was compared. The results are presented in Table 2. As can be seen, the developed approach exhibited better performance in terms of navigation efficiency than the traditional method, yielding shorter UAV navigation times for interior and exterior inspection loops. This is because during the experiment, the traditional method emerged with the following drawback:

- Human error: Our pilot had to control the UAV's direction while the device was flying. There were several times when the UAV's flying direction slightly deviated from the expected orientation and went into undesired places. The adjustment time extended the total navigation time.

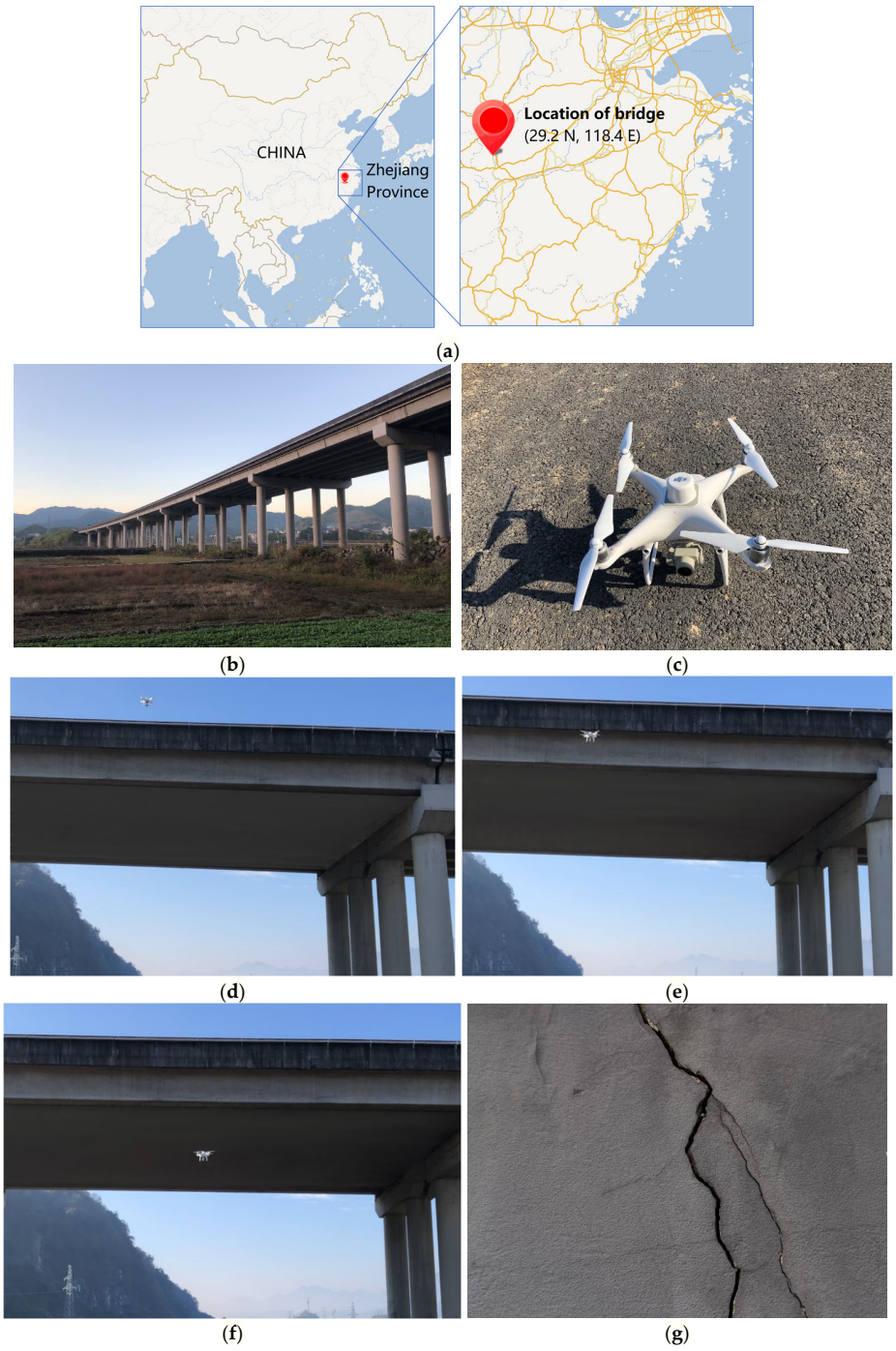


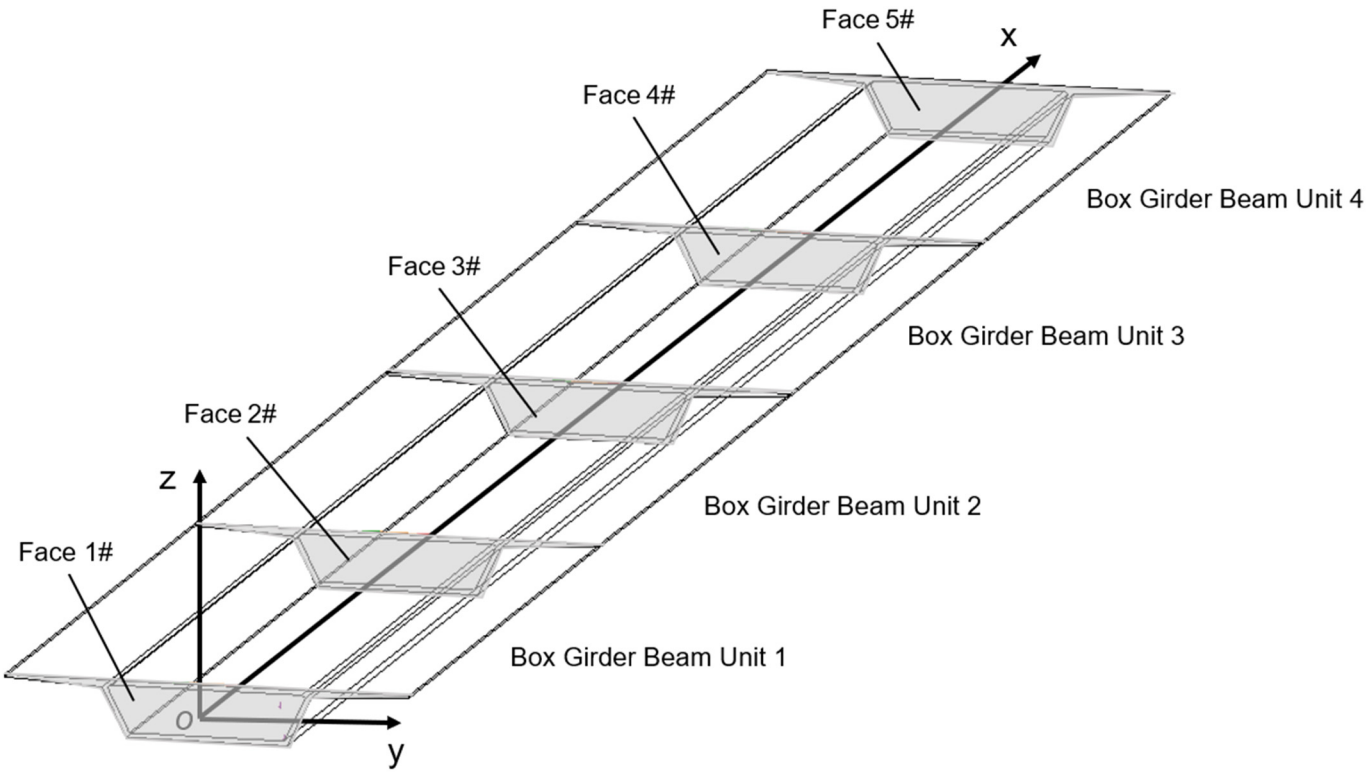

**Figure 9.** Field experiment: (**a**) location of the project; (**b**) project—12th bridge of the Hetianxi section; (**c**) DJI Phantom 4; (**d**–**f**) UAV inspection trajectory; (**g**) structural crack.

**Table 2.** UAV navigation time in the field test (developed approach versus traditional human control).

| Methods | Interior Loop Navigation Time (s) | Exterior Loop Navigation Time (s) |
|---|---|---|
| The developed approach | 35 | 50 |
| Traditional human control | 55 | 62 |

### *5.2. Reasonableness of Sequence Determined for Trajectory Feature Points*

In the test, the TSFPSD algorithm of the proposed approach was applied to a BIM bridge model. The results are plotted in Figure 10. As can be seen, the TSFPSD algorithm can identify the vertices of box girder beam units as the trajectory state feature points and determine reasonable sequences of feature points for inspecting the interior and exterior shells of the box girder beam units. First, the algorithm identifies the faces of the box girder beam units from the BIM model (e.g., face 1# in Figure 10a) and detects the vertexes of each face as the feature points. Then, the feature points are split into subgroups with the same x-coordinate values (e.g., group $\{A_1, B_1, C_1, D_1, E_1, F_1\}$ in Figure 10b). Next, the points in the subgroups are sorted in a clockwise order to form local loops (e.g., loop "$A_1 \rightarrow B_1 \rightarrow C_1 \rightarrow D_1 \rightarrow E_1 \rightarrow F_1 \rightarrow A_1$" in Figure 10b). Finally, the local loops are connected end-to-end to form sequences of feature points for inspecting the box girder beam units (e.g., the connected $A_1$ and $A_2$ points in Figure 10b). The connected loops direct the UAV to move forward along the $x$-axis to engage in a further clockwise twist for inspecting both the interior and exterior shells of the box girder beam units after the current loop "$A_i \rightarrow B_i \rightarrow C_i \rightarrow D_i \rightarrow E_i \rightarrow F_i \rightarrow A_i$" is completed (see Figure 10b).




(**a**)

**Figure 10.** *Cont.*

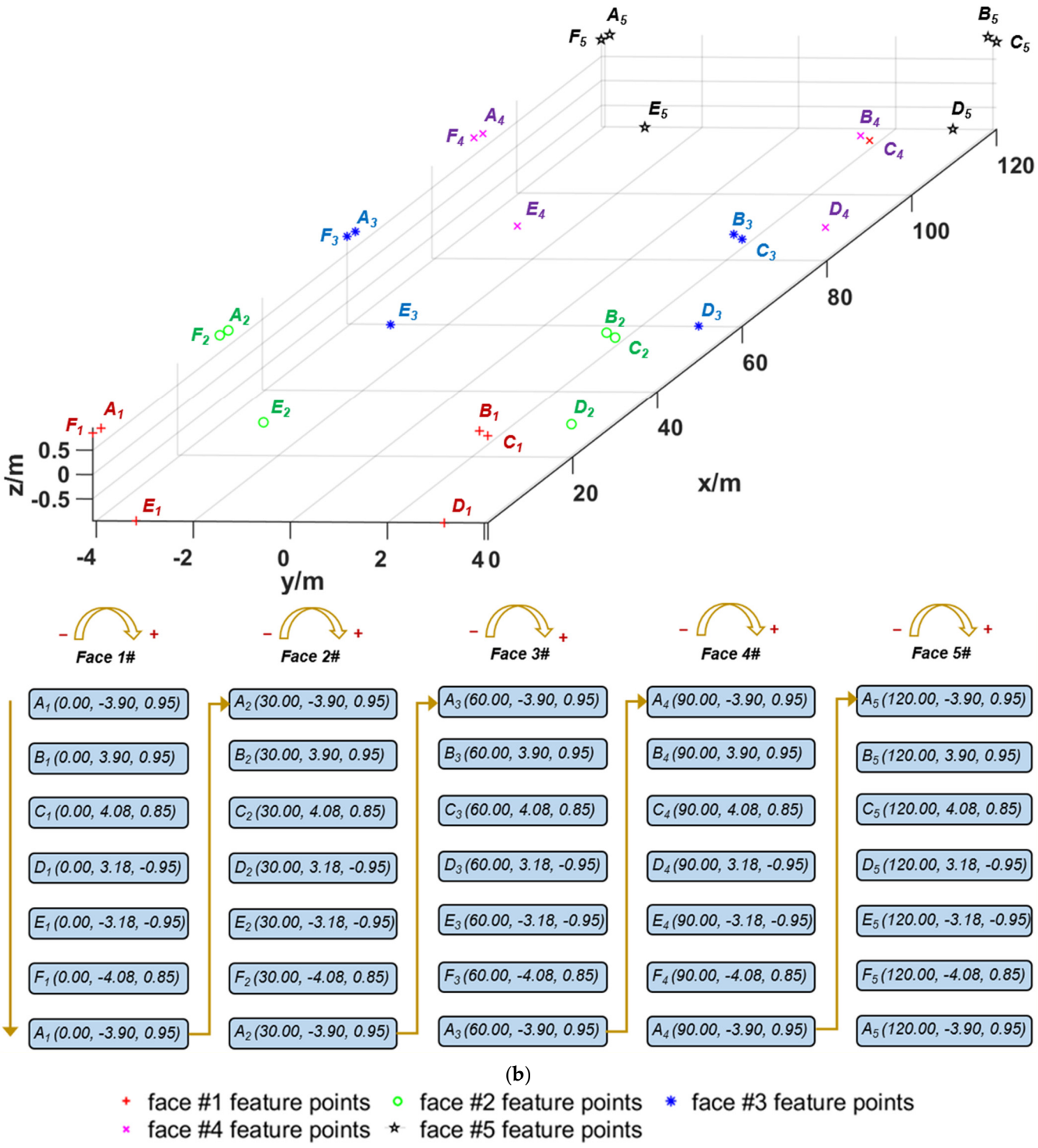
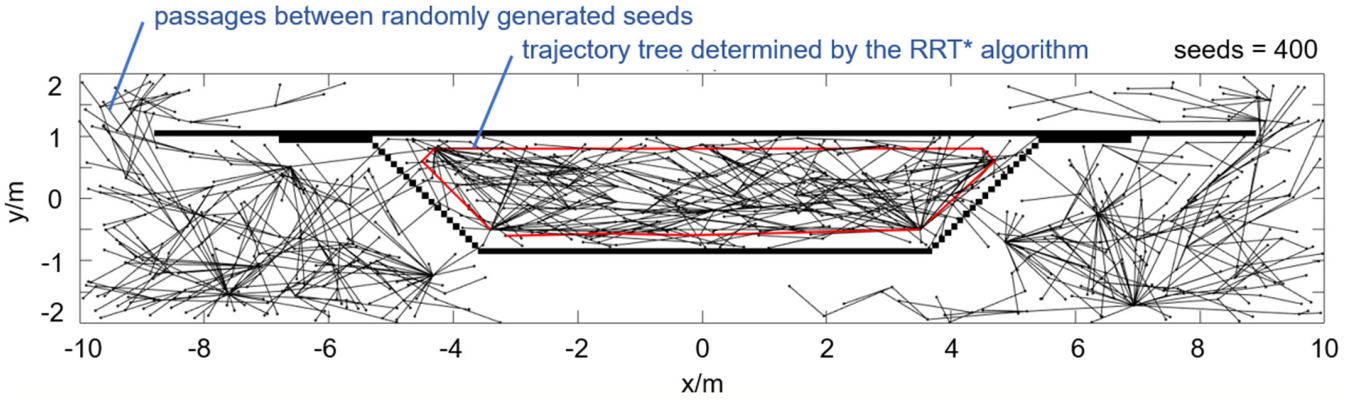



**Figure 10.** Trajectory feature points sequence determined by the TSFPSD algorithm: (**a**) BIM model sectional view; (**b**) sequence for interior shell feature points.

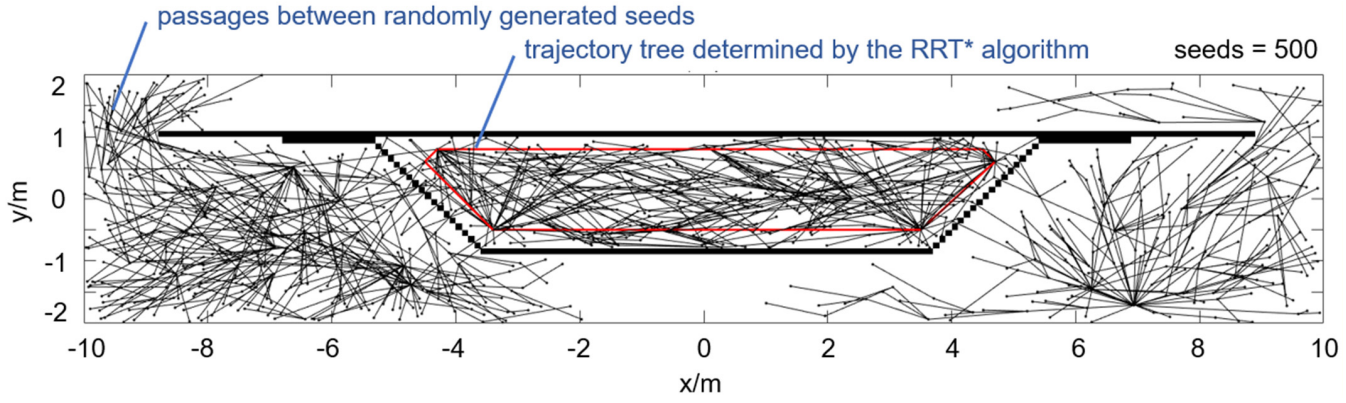
### *5.3. Trajectory Closeness and Deviation*

The varied RRT* algorithm of the approach was applied to analyse the determined inspection coordinates and generate UAV's kinematic states for performing the inspection of box girder beam units. The varied RRT* algorithm strived to search the space for a feasible path by growing a space-filling tree. This tree was rooted at the starting point (e.g., point '$A_1$' in Figure 10). At each incremental expansion of the tree growth, a random seed was generated. Then, the already-existed seeds were inspected, and the tree grew from its current

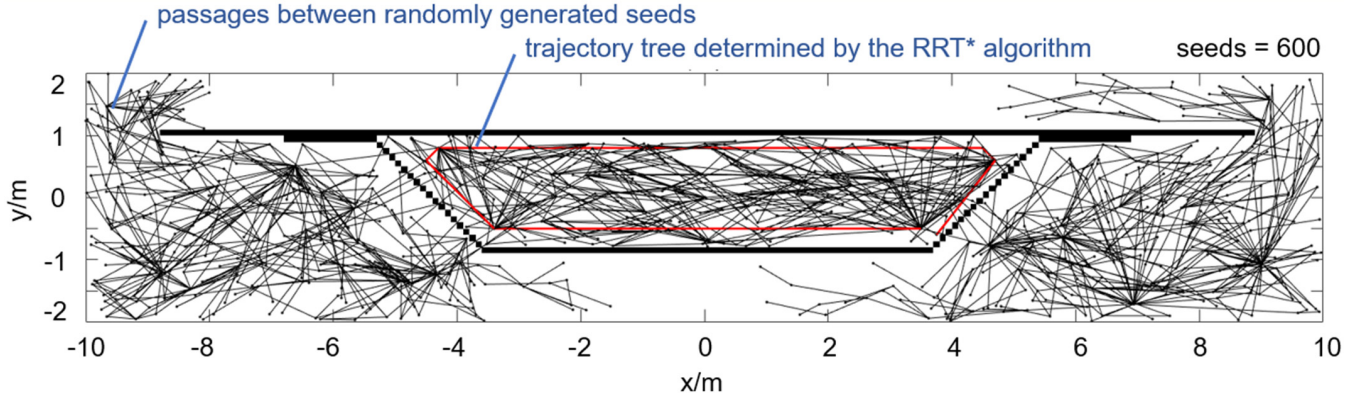
form by adding reasonable seeds until the tree's far end reached the goal state. The results of the trajectory planning for an interior loop "$A_i \rightarrow B_i \rightarrow C_i \rightarrow D_i \rightarrow E_i \rightarrow F_i \rightarrow A_i$" are plotted in Figure 11. The black lines in Figure 11 refer to computer-generated passages between seeds that can potentially be added as trajectory nodes. The red lines in Figure 11 refer to the trajectory tree expansion by passing through passages between certain seeds, as determined by the varied RRT* algorithm. As can be seen, when the number of seeds reached 500, the planned trajectory indicated the best performance (Figure 11b): (1) a fully end-to-end connection with no breaches and (2) no major deviation from the expected trajectory (see Figure 5a). On the contrary, the planning with more and fewer seeds showed obvious breaches on the left and right bottom (see Figure 11a,c) and deviation from the expected trajectory (see Figure 11d). Specifically, the maximum deviation from the expected trajectory in Figure 11d was 70 cm.




(**a**)

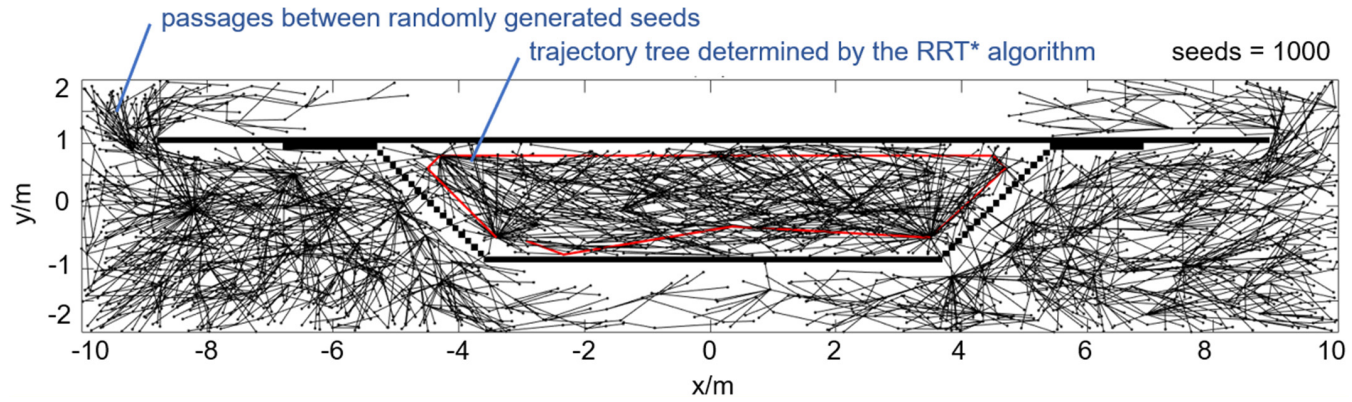



(**b**)




(**c**)

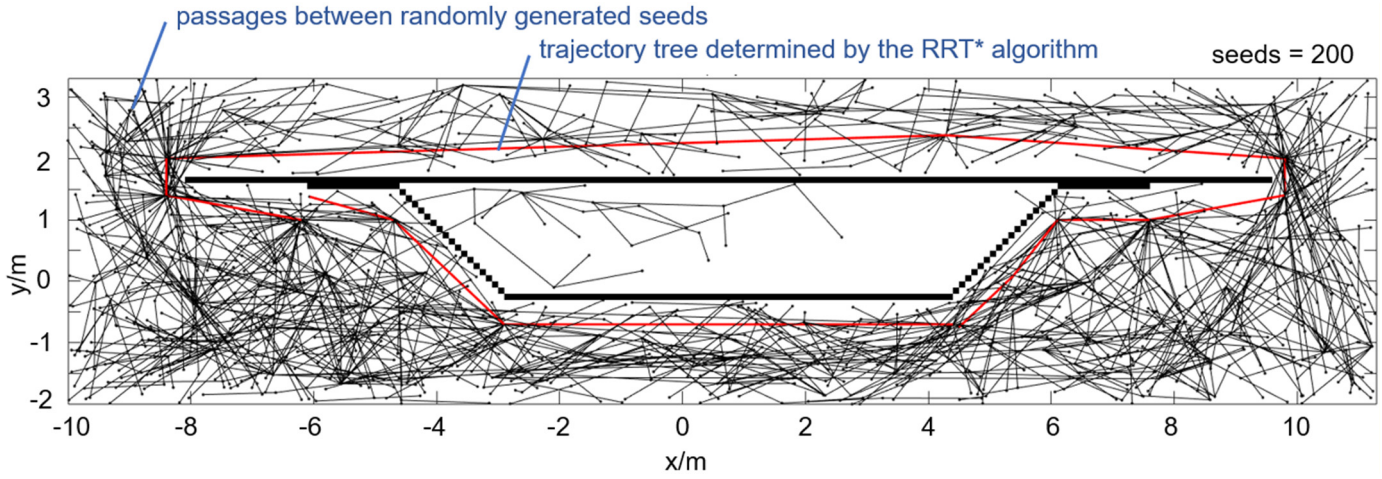
**Figure 11.** *Cont.*




(**d**)

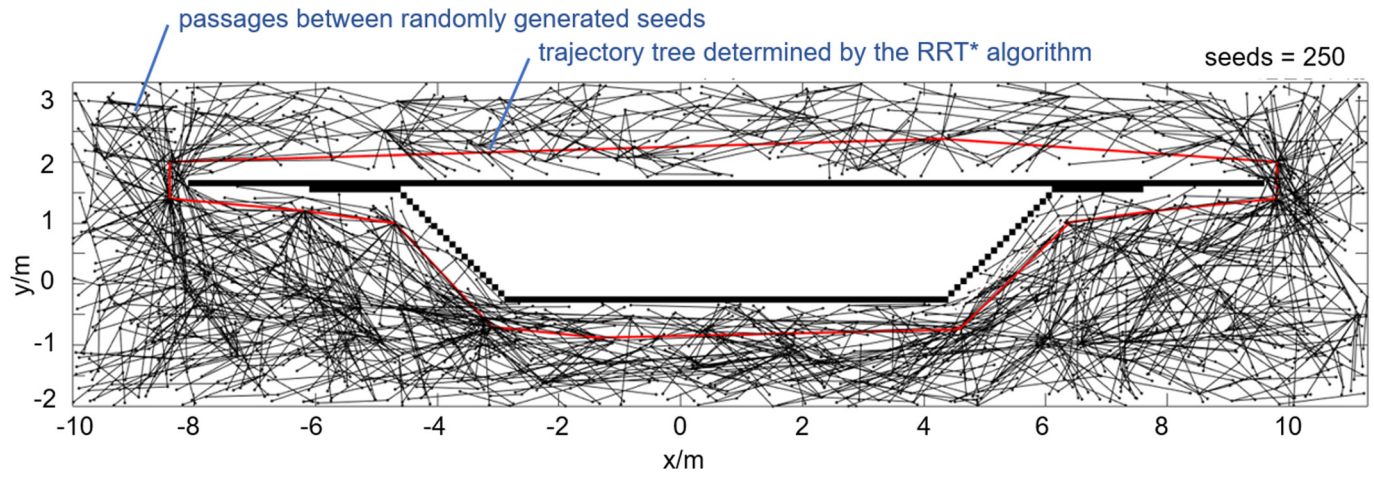
**Figure 11.** Comparison between trajectories determined by the varied RRT* algorithm for the interior loop: (**a**) seed number at 400, (**b**) seed number at 500, (**c**) seed number at 600, and (**d**) seed number at 1000.

The results of the trajectory planning for an exterior loop "$A'_i \rightarrow B'_i \rightarrow C'_i \rightarrow D'_i \rightarrow E'_i \rightarrow F'_i \rightarrow G'_i \rightarrow H'_i \rightarrow A'_i$" are plotted in Figure 12. The black lines in Figure 12 refer to computer-generated passages between seeds that can potentially be added as trajectory nodes. The red lines in Figure 12 refer to the trajectory tree expansion by passing through passages between certain seeds, as determined by the varied RRT* algorithm. As can be seen, when the number of seeds reached 250, the planned trajectory indicated the best performance (Figure 12b): (1) a fully end-to-end connection with no breaches and (2) no major deviation from the expected trajectory (see Figure 5b). On the contrary, the planning with more and fewer seeds showed obvious breaches and deviation from the expected trajectory (see Figure 12a,c,d). Specifically, the maximum deviations from the expected trajectory in Figure 12a,c,d were 80, 120, and 150 cm.

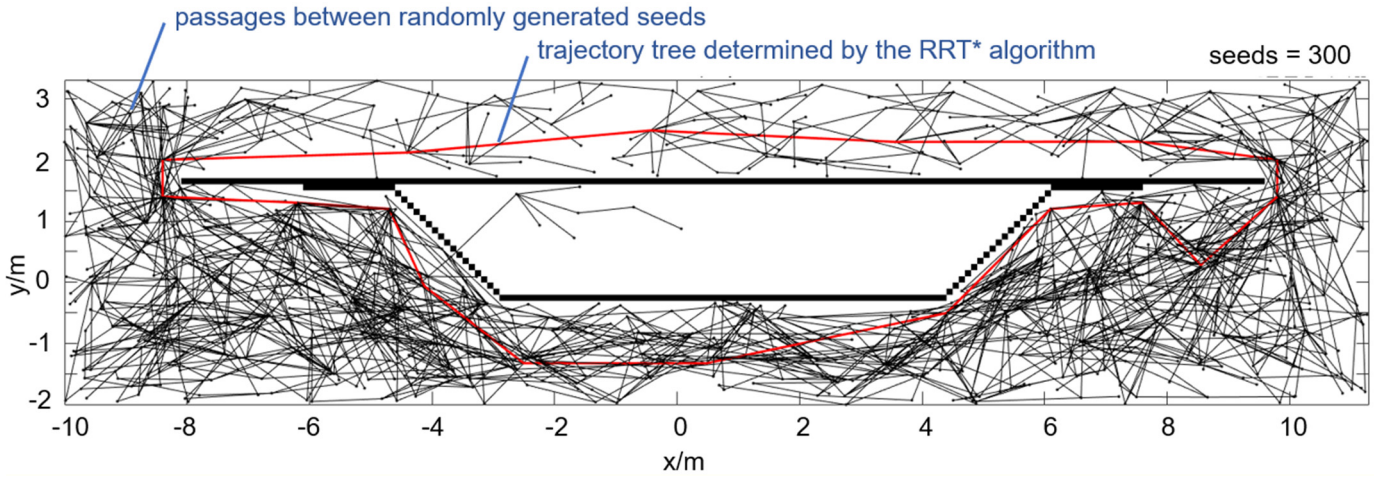



(**a**)

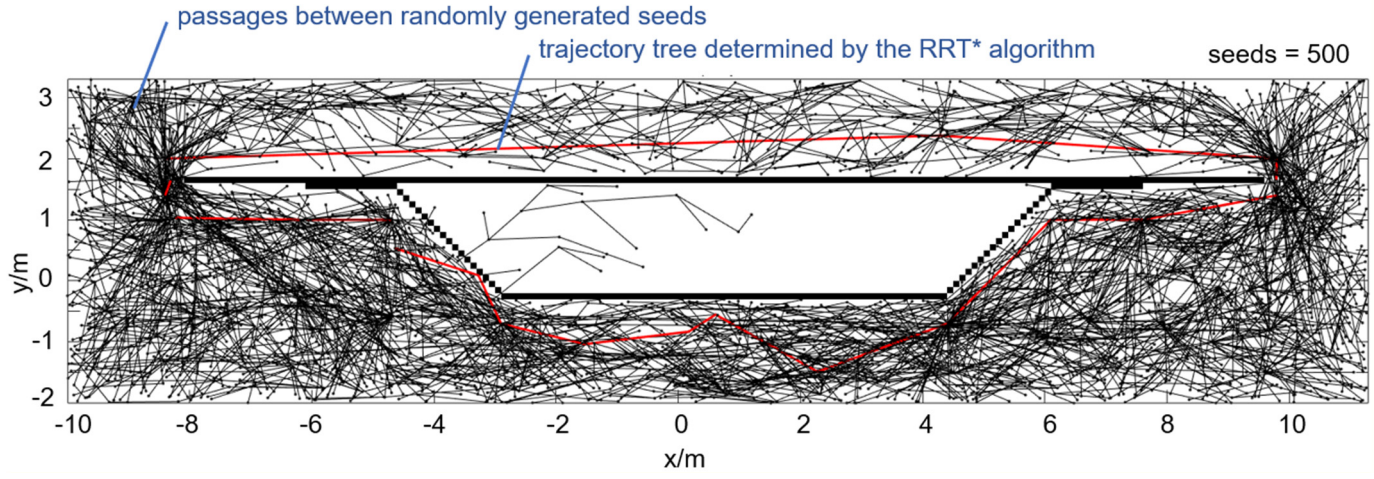
**Figure 12.** *Cont.*




(**b**)




(**c**)




(**d**)

**Figure 12.** Comparison between trajectories determined by the varied RRT* algorithm for exterior loop: (**a**) seed number at 200, (**b**) seed number at 250, (**c**) seed number at 300, and (**d**) seed number at 500.

Given the test results, the varied RRT* algorithm of the approach showed sufficient capability in planning the trajectory for box girder beam inspection with satisfactory closeness and deviation.

### *5.4. Trajectory Smoothness*

To quantitatively evaluate the trajectory smoothness performance of the developed approach, the derivatives $\dot{x}(t)$ and $\dot{y}(t)$ of the planned trajectories were examined. The parameters $\dot{x}(t)$ and $\dot{y}(t)$ refer to the flying speeds of a UAV in the horizontal and vertical planes (i.e., $x$ and $y$ directions) in real time. The criteria for a smooth trajectory are that its derivatives $\dot{x}(t)$ and $\dot{y}(t)$ are continuous and not simultaneously zero [41]. Taking the determined trajectories for interior and exterior loops in Figures 11b and 12b as the analytic objectives, the derivatives $\dot{x}(t)$ and $\dot{y}(t)$ were computed and the results were plotted in Figures 13a and 13b, respectively. To provide a visual demonstration, the UAV locations corresponding to the six-time nodes in Figure 13a (e.g., point "A") are illustrated in a virtual environment in Figure 14. As can be seen, the trajectory execution times were 35 s and 50 s for interior and exterior inspection loops, respectively. The derivatives $\dot{x}(t)$ and $\dot{y}(t)$ were continuous and not simultaneously zero for both interior and exterior loops, which indicate that the trajectories are smooth for UAV navigation.

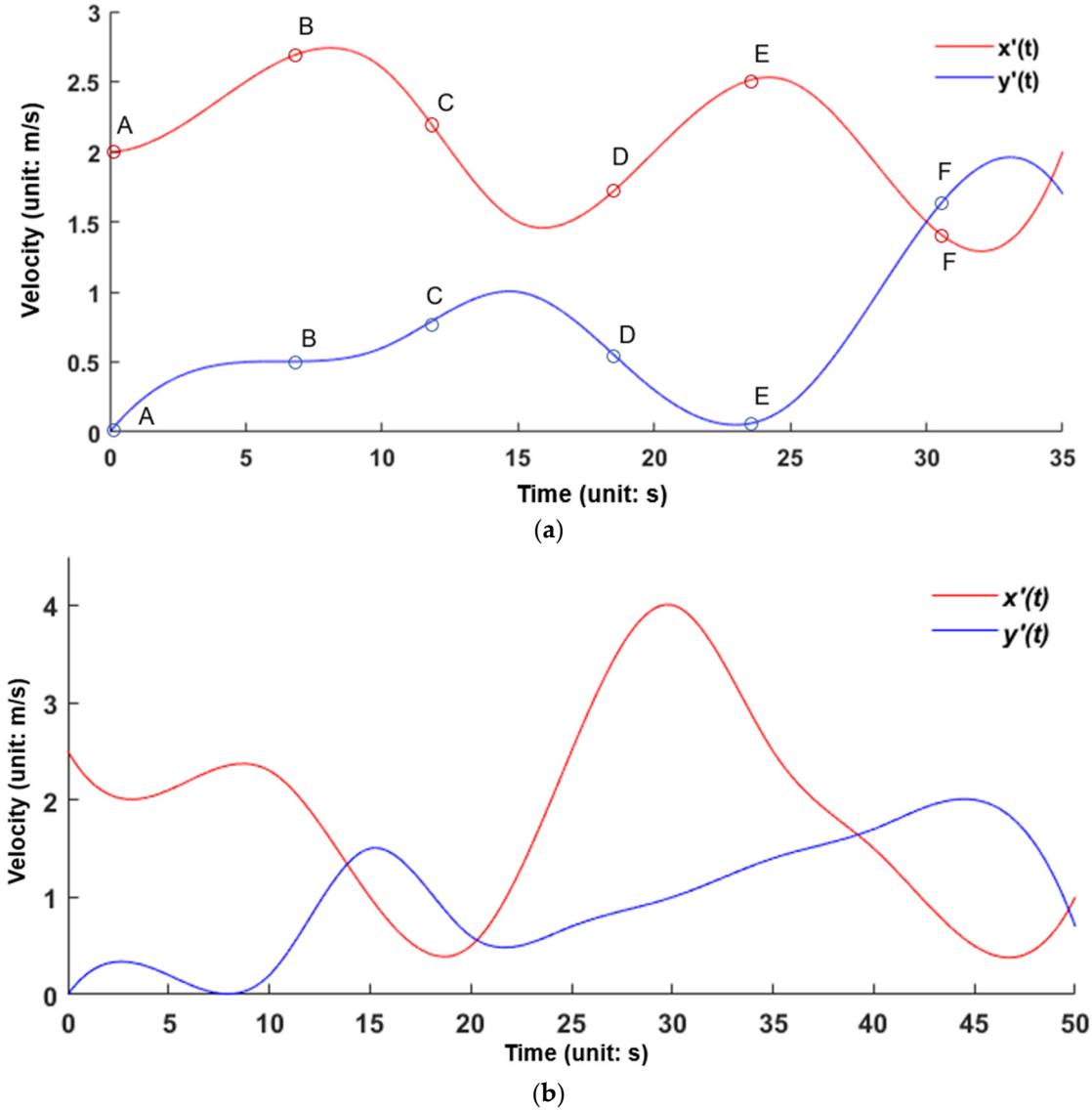



**Figure 13.** Time-velocity plots: (**a**) interior loop and (**b**) exterior loop.

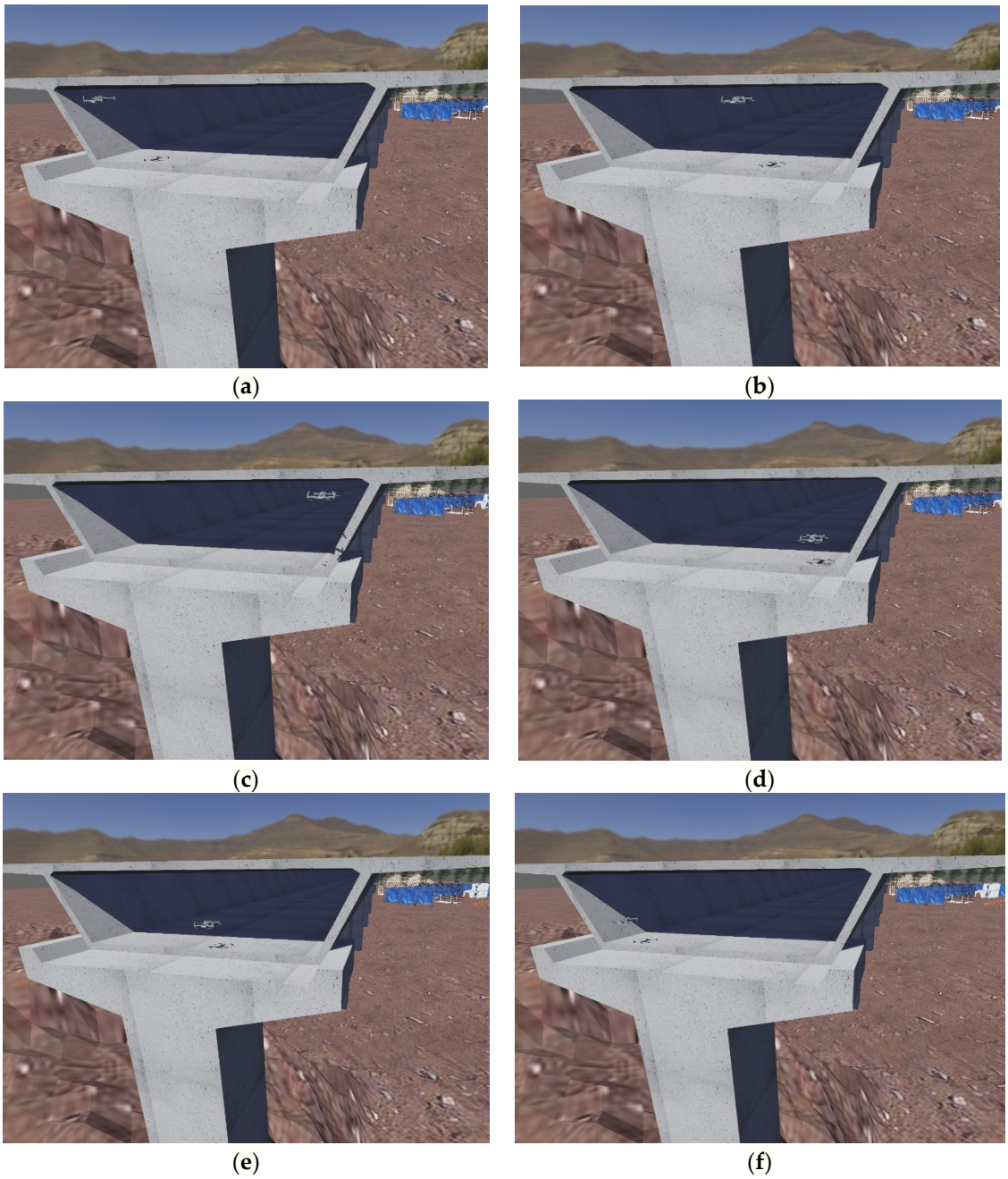


**Figure 14.** Simulation demonstration of UAV locations corresponding to the six-time nodes in Figure 11a: (**a**) top-left position; (**b**) top-middle position; (**c**) top-right position; (**d**) bottom-right position; (**e**) bottom-middle position; (**f**) bottom-left position.

## 6. Discussion

This research developed a BIM-based trajectory planning approach that can assist in UAV inspection of box girder beam units. As proposed in Section 1, the scientific question for the present study to address is how to expand the digital blueprint of the box girder bridge in BIM into UAV robotic control instructions, i.e., how to determine a mathematical relationship between the coordinates of box girder units in BIM and an inspection sequence, in consideration of the box girder's vertexes. In scrutinising the scientific question, the findings suggest that the question has been answered in this research. The approach was developed to reflect the geometry properties of box girders as inspection characteristics, and consists of a task planning algorithm and a trajectory planning algorithm that can

(1) retrieve feature points that can describe the geometry of the box girder beam (as well as the geometry of the inspection trajectory) from the BIM model, (2) mathematically arrange the feature points in a reasonable order as the inspection trajectory states, and (3) analyse the determined inspection sequence and coordinates and generate the UAV's kinematic states for performing the inspection of a box girder bridge.

As presented in Section 5, the developed approach was tested in four aspects: (1) navigation efficiency, (2) reasonableness of sequence determined for trajectory feature points, (3) trajectory closeness and deviation, and (4) trajectory smoothness. To quantitatively evaluate the navigation efficiency performance, the UAV navigation time for bridge inspection applying both the developed autonomous approach and the traditional human control method was compared (Section 5.1). The results showed that the developed approach performed better in terms of navigation efficiency than the traditional method, yielding shorter UAV navigation times for interior and exterior inspection loops. To quantitatively evaluate the sequence determination performance, the vertices of the box girder beam units in the sorted sequence list (Figure 10) were evaluated (Section 5.2). As can be seen, the coordinates of the vertices were arranged to form end-to-end connected local loops for inspecting both the interior and exterior shells of the box girder beam units. This implies a reasonable clockwise order to establish the UAV inspection logic. To quantitatively evaluate whether the planned trajectories are close and do not deviate, kinematic analysis on the sorted list of trajectory feature points was performed (Section 5.3). The results showed that when the number of randomly sampled seeds reached 500 and 250 in the planning space, the planned trajectories for both the interior and exterior inspection loops were fully end-to-end connected with no breaches and indicated no major deviation from the expected trajectory (see Figures 11b and 12b). The trajectory smoothness (Section 5.4) was further tested. The results showed that the trajectory execution time was 35 s and 50 s for interior and exterior inspection loops, respectively. The derivatives $\dot{x}(t)$ and $\dot{y}(t)$ were continuous and not simultaneously zero for both interior and exterior loops, which indicates that the trajectories are smooth for UAV navigation (see Figure 13).

Our approach plans trajectories by leveraging the geometric features of structural components in a BIM model, which uses the same pipeline for both the girder and pier. At the connection between the girder and the pier, the approach first extracts the feature points and then connects the feature points from both parts to form a trajectory in clockwise order (see Figure 15).

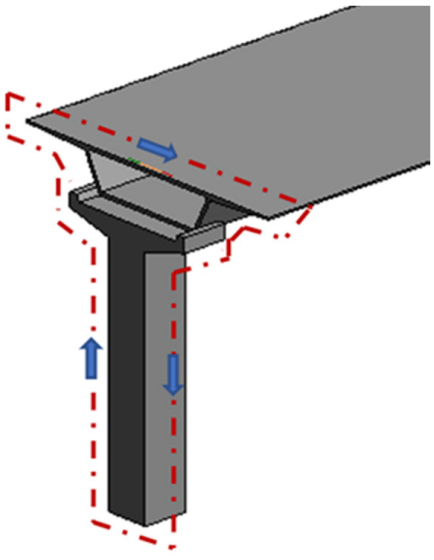


**Figure 15.** Planned trajectory at the connection between the girder and the pier.

Compared with the methods developed in previous studies [42–44], our approach extends the existing body of knowledge with BIM-enabled UAV autonomous navigation capabilities for box girder bridge inspection. The approaches all have their unique paradigm for engineering application. The approach of Hamieh et al. [42] largely exploits a BIM indoor space to sample topological points and form a navigation graph. The approach by Amir and Mani [43] divides the 3D space of a BIM model into grids and uses the unoccupied grids to form a navigation path. The approach by Wang et al. [44] takes the photogrammetry viewpoints into consideration to form a navigation path. It seems that the three studies did not provide an open source for their codes, which makes it difficult to make an experimental comparison of the approaches. However, based on the nature of these approaches (as mentioned above), we can infer that our approach is more computationally lightweight. The approaches by Hamieh et al. [42], Amir and Mani [43] and Wang et al. [44] exploit a 3D space or 2D planar to retrieve trajectory feature points as many as possible, whereas our approach considers the geometric feature of structural components and only samples the geometry vertices just enough to form an inspection trajectory by connecting them. For example, in the approach of Wang et al. [44], multiple midpoints are generated on the trajectory between two vertices in a straight-line format, where the removal of the midpoints does not seem to affect the final trajectory appearance. This is what our approach is capable of—sampling vertices that are sufficient to represent geometry and connecting them, which seems to be more computationally lightweight. In addition, compared with the mentioned approaches, our method further considers the smoothness of its generated trajectories to eliminate sharp turns and jerks.

Contemporary drone technologies enable the Pilot in Command (PIC) mode to sketch a path for the drone. However, it requires manual input to designate waypoints and lacks a BIM model as a built-in planning reference. Our work bridges this gap and provides an autonomous UAV navigation approach for box girder bridge inspection. Given this, the technical contributions of this research are as follows:

(1) The Trajectory State Feature Points and Sequence Determination (TSFPSD) algorithm. The TSFPSD algorithm is developed to determine the mathematical relationship between the coordinates of box girder beam units and the inspection sequence in consideration of their vertexes for box girder bridge beam inspection.
(2) The varied Rapidly Exploring Random Tree Star (RRT*) algorithm. We provide a variant of the RRT* algorithm, which can analyse the sequence of the feature points as determined by the TSFPSD algorithm, grow trajectory trees $\eta$, and contract/expand the trajectory trees with a predetermined variable $wd$ for performing the UAV inspection of the interior/exterior shell of box girder beam units.

The practical value of our approach is as follows: Once a UAV is set up in a real project, the approach can first receive data input from a BIM model, generate trajectory parameters, and publish useful kinematic control signals to the digital processors of the UAV for performing the inspection of box girder beam units. Then, image processing algorithms (e.g. U-Net used in [45,46]) can be further embedded to localise and measure the width of concrete cracks along with UAV photography.

## 7. Conclusions and Future Work

This research presents a BIM-based approach for UAV inspection of box girder beam units. The development of the prototype consists of a task planning algorithm and a trajectory planning algorithm. The task planning algorithm, i.e., Trajectory State Feature Points and Sequence Determination (TSFPSD) is designed to utilise the spatial information contained in a BIM model to retrieve feature points that can describe the geometry of a box girder beam. Then, the TSFPSD algorithm sorts the feature points in a clockwise order

as the inspection trajectory states. The motion planning algorithm, i.e., varied Rapidly Exploring Random Tree Star (varied RRT*), analyses the determined inspection sequence and coordinates and generates UAV's kinematic states for performing the inspection of box girder bridges.

Different tests were performed to assess the developed approach. The corresponding results demonstrated that the approach has satisfactory performance in all the tests. First, the approach was capable of creating a reasonable sequence of feature points in order to establish the trajectory for inspecting the box girder beam units. Second, the planned trajectories were fully end-to-end connected with no breaches and indicated no major deviation from the expected trajectories. Third, the trajectory derivatives $x(t)$ and $y(t)$ were continuous and not simultaneously zero, which indicated that the trajectory was smooth for UAV navigation.

The existing studies still require human pilots to navigate the UAVs for bridge inspection. Therefore, the present research contributes to the field of bridge inspection using UAV technologies. More specifically, the developed approach fills in the following knowledge gaps: (1) determining a reasonable sequence of the vertex coordinates of box girder units for bridge inspection, and (2) analysing the determined inspection sequence and coordinates and generating UAV's kinematic states for performing the inspection of box girder bridges. In addition, the outcomes of this research have practical implications. Once a UAV is set up in a real project, the approach can first receive data input from the BIM model, generate trajectory parameters, and publish useful kinematic control signals to the digital processors of the UAV for performing the inspection of the box girder beam units. Then, the image processing algorithms can be further embedded to localise and measure the width of concrete cracks along with the UAV photography. The self-contained task planning algorithm in our approach, TSFPSD, can retrieve feature points from the BIM model to describe the geometry of the box girder beam regardless of the dimensions of the box girder. Using the retrieved feature points, the TSFPSD algorithm can further sort the feature points in a reasonable order as the inspection trajectory states. Given this feature, our approach has generalisability for bridge inspection in the box girder category.

This research has the following limitations, and subsequent research needs to be conducted shortly. In the field test, the UAV inspection robustness was influenced by the illumination conditions. When the UAV entered less illuminated areas (e.g., underneath the bridge beam units), the visual quality of the captured images began to degrade, which affected the crack detection performance. This paper is a part of an ongoing research project. In subsequent research, the authors will investigate a method for enhancing image brightness and preserving image details under low-light conditions for UAV bridge inspection. Although affected by illumination conditions, the method proposed in this paper is still applicable to other non-illuminated sensors (e.g., LiDAR and InSAR radar). The BIM model specifically includes the bridge structure but overlooks on-site obstacles, such as surrounding trees or auxiliary pipelines on the bridge deck. In subsequent research, the authors will investigate the method for incorporating detection algorithms to recognise surrounding obstacles for the UAV's detour considerations. In addition, centimeter-level positioning deviations may occur under challenging environmental conditions, such as wind, turbulence, and atmospheric thermal gradients. Future work will implement Real-Time Kinematic (RTK) technologies to address this limitation. By analysing carrier-phase observations with millimetre-level wavelength resolution, we aim to quantify trajectory regression accuracy, particularly for mission-critical applications requiring sub-centimeter precision (e.g., sensor calibration and 3D reconstruction).

**Author Contributions:** Conceptualization, Y.G.; Methodology, Y.G.; Software, Y.G.; Validation, Y.G.; Formal analysis, Y.G.; Investigation, Z.X.; Data curation, Z.X.; Writing—original draft, Y.G.; Writing—review & editing, Z.X.; Supervision, J.S.; Project administration, J.S.; Funding acquisition, J.S. All authors have read and agreed to the published version of the manuscript.

**Funding:** The authors would like to gratefully acknowledge the support from the Key R&D Program of Zhejiang (2023C01161), the National Natural Science Foundation of China (W2412092), the Key R&D Program of Zhejiang (2024C01132) and the Key R&D Program of Ningbo (2024H013), which made the research possible.

**Data Availability Statement:** Some data supporting the findings of this study are available from the corresponding author upon reasonable request: Trajectory feature points; Trajectory execution time and velocity.

**Conflicts of Interest:** The authors declare no conflict of interest.

## References

1. Siraj, N.B.; Fayek, A.R. Risk Identification and Common Risks in Construction: Literature Review and Content Analysis. *J. Constr. Eng. Manag.* **2019**, *145*, 3119004. [CrossRef]
2. Irem, T.Z.; Chiara, I.; Francesco, G.P.; Pina, L.M. Development and Implementation of Indicators to Assess Bridge Inspection Practices. *J. Constr. Eng. Manag.* **2021**, *147*, 4021165. [CrossRef]
3. Hesam, H.; Seyedomid, S.; Brenda, M.; Martin, F. Automation of Inspection Mission Planning Using 4D BIMs and in Support of Unmanned Aerial Vehicle–Based Data Collection. *J. Constr. Eng. Manag.* **2021**, *147*, 4020179. [CrossRef]
4. Ali, H.M.; Rod, H.; Mohammad, F.; Abbas, R.; Trenton, H.; Kaye, S.D.; Trent, H. Feasibility Study of Using Nebulizer-Retrofitted UAVs at Construction Projects: The Case Study of Residential Jobsites in Utah. *J. Constr. Eng. Manag.* **2022**, *148*, 5022009. [CrossRef]
5. Muhammad, K.; Rabia, K.; Sharjeel, A.; Van-Tien, T.S.; Chansik, P. Fall Prevention from Scaffolding Using Computer Vision and IoT-Based Monitoring. *J. Constr. Eng. Manag.* **2022**, *148*, 4022051. [CrossRef]
6. Jeong, E.; Seo, J.; Wacker, J.P. UAV-Aided Bridge Inspection Protocol through Machine Learning with Improved Visibility Images. *Expert Syst. Appl.* **2022**, *197*, 116791. [CrossRef]
7. Mahama, E.; Karimoddini, A.; Khan, M.A.; Cavalline, T.L.; Hewlin, R.L.; Smith, E.; Homaifar, A. Testing and Evaluating the Impact of Illumination Levels on UAV-Assisted Bridge Inspection. In Proceedings of the 2022 IEEE Aerospace Conference (AERO), Big Sky, MT, USA, 5–12 March 2022; pp. 1–8.
8. Kao, S.-P.; Wang, F.-L.; Lin, J.-S.; Tsai, J.; Chu, Y.-D.; Hung, P.-S. Bridge Crack Inspection Efficiency of an Unmanned Aerial Vehicle System with a Laser Ranging Module. *Sensors* **2022**, *22*, 4469. [CrossRef]
9. Congress, S.S.; Escamilla, J.; Chimauriya, H.; Puppala, A.J. Challenges of 360° Inspection of Bridge Infrastructure Using Unmanned Aerial Vehicles (UAVs). *Int. Conf. Transp. Dev.* **2022**, *2023*, 96–108.
10. Kim, H.; Narazaki, Y.; Spencer, B.F., Jr. Automated Bridge Component Recognition Using Close-Range Images from Unmanned Aerial Vehicles. *Eng. Struct.* **2023**, *274*, 115184. [CrossRef]
11. Xu, Y.; Zhang, J. UAV-Based Bridge Geometric Shape Measurement Using Automatic Bridge Component Detection and Distributed Multi-View Reconstruction. *Autom. Constr.* **2022**, *140*, 104376. [CrossRef]
12. Ni, F.; He, Z.; Jiang, S.; Wang, W.; Zhang, J. A Generative Adversarial Learning Strategy for Enhanced Lightweight Crack Delineation Networks. *Adv. Eng. Inform.* **2022**, *52*, 101575. [CrossRef]
13. Bono, A.; D'Alfonso, L.; Fedele, G.; Filice, A.; Natalizio, E. Path Planning and Control of a UAV Fleet in Bridge Management Systems. *Remote Sens.* **2022**, *14*, 1858. [CrossRef]
14. Shanthakumar, P.; Yu, K.; Singh, M.; Orevillo, J.; Bianchi, E.; Hebdon, M.; Tokekar, P. *View Planning and Navigation Algorithms for Autonomous Bridge Inspection with UAVs BT—Proceedings of the 2018 International Symposium on Experimental Robotics*; Xiao, J., Kröger, T., Khatib, O., Eds.; Springer International Publishing: Cham, Switzerland, 2020; pp. 201–210.
15. Zhang, Z.; Wu, J.; Dai, J.; He, C. A Novel Real-Time Penetration Path Planning Algorithm for Stealth UAV in 3D Complex Dynamic Environment. *IEEE Access* **2020**, *8*, 122757–122771. [CrossRef]
16. Zhang, Z.; Wu, J.; Dai, J.; He, C. Optimal Path Planning with Modified A-Star Algorithm for Stealth Unmanned Aerial Vehicles in 3D Network Radar Environment. *Proc. Inst. Mech. Eng. Part G J. Aerosp. Eng.* **2021**, *236*, 72–81. [CrossRef]
17. Guo, J.; Xia, W.; Hu, X.; Ma, H. Feedback RRT* Algorithm for UAV Path Planning in a Hostile Environment. *Comput. Ind. Eng.* **2022**, *174*, 108771. [CrossRef]

18. Fan, J.; Chen, X.; Wang, Y.; Chen, X. UAV Trajectory Planning in Cluttered Environments Based on PF-RRT* Algorithm with Goal-Biased Strategy. *Eng. Appl. Artif. Intell.* **2022**, *114*, 105182. [CrossRef]
19. Fan, J.; Chen, X.; Liang, X. UAV Trajectory Planning Based on Bi-Directional APF-RRT* Algorithm with Goal-Biased. *Expert Syst. Appl.* **2023**, *213*, 119137. [CrossRef]
20. Aggarwal, S.; Kumar, N. Path Planning Techniques for Unmanned Aerial Vehicles: A Review, Solutions, and Challenges. *Comput. Commun.* **2020**, *149*, 270–299. [CrossRef]
21. Terada, Y.; Murata, S. Automatic Modular Assembly System and Its Distributed Control. *Int. J. Robot. Res.* **2008**, *27*, 445–462. [CrossRef]
22. King, N.; Bechthold, M.; Kane, A.; Michalatos, P. Robotic Tile Placement: Tools, Techniques and Feasibility. *Autom. Constr.* **2014**, *39*, 161–166. [CrossRef]
23. Davtalab, O.; Kazemian, A.; Khoshnevis, B. Perspectives on a BIM-Integrated Software Platform for Robotic Construction through Contour Crafting. *Autom. Constr.* **2018**, *89*, 13–23. [CrossRef]
24. Kontovourkis, O.; Tryfonos, G.; Georgiou, C. Robotic Additive Manufacturing (RAM) with Clay Using Topology Optimization Principles for Toolpath Planning: The Example of a Building Element. *Archit. Sci. Rev.* **2020**, *63*, 105–118. [CrossRef]
25. Ding, L.; Jiang, W.; Zhou, Y.; Zhou, C.; Liu, S. BIM-Based Task-Level Planning for Robotic Brick Assembly through Image-Based 3D Modeling. *Adv. Eng. Inform.* **2020**, *43*, 100993. [CrossRef]
26. Wiranugeraha, M.B.; Tan, H.; Nurahmi, L. *Trajectory Planning of Cable-Driven Parallel Robot with Mobile Cranes BT—Recent Advances in Mechanical Engineering*; Tolj, I., Reddy, M.V., Syaifudin, A., Eds.; Springer Nature Singapore: Singapore, 2023; pp. 310–317.
27. Zhu, H.; Ouyang, H.; Xi, H. Neural Network-Based Time Optimal Trajectory Planning Method for Rotary Cranes with Obstacle Avoidance. *Mech. Syst. Signal Process.* **2023**, *185*, 109777. [CrossRef]
28. Safouhi, H.; Mouattamid, M.; Hermann, U.; Hendi, A. An Algorithm for the Calculation of Feasible Mobile Crane Position Areas. *Autom. Constr.* **2011**, *20*, 360–367. [CrossRef]
29. Lu, B.; Lin, J.; Fang, Y.; Hao, Y.; Cao, H. Online Trajectory Planning for Three-Dimensional Offshore Boom Cranes. *Autom. Constr.* **2022**, *140*, 104372. [CrossRef]
30. Gao, Y.; Meng, J.; Shu, J.; Liu, Y. BIM-Based Task and Motion Planning Prototype for Robotic Assembly of COVID-19 Hospitalisation Units—Flatpack House. *Autom. Constr.* **2022**, *140*, 104370. [CrossRef]
31. Shu, J.; Li, W.; Meng, J.; Gao, Y. Collision-Free Trajectory Planning for Robotic Assembly of COVID-19 Healthcare Facilities. *Autom. Constr.* **2022**, *142*, 104520. [CrossRef]
32. Tang, F.; Ma, T.; Zhang, J.; Guan, Y.; Chen, L. Integrating Three-Dimensional Road Design and Pavement Structure Analysis Based on BIM. *Autom. Constr.* **2020**, *113*, 103152. [CrossRef]
33. Fernández-Rodríguez, S.; Cortés-Pérez, J.P.; Muriel, P.P.; Tormo-Molina, R.; Maya-Manzano, J.M. Environmental Impact Assessment of Pinaceae Airborne Pollen and Green Infrastructure Using BIM. *Autom. Constr.* **2018**, *96*, 494–507. [CrossRef]
34. Li, F.; Zlatanova, S.; Koopman, M.; Bai, X.; Diakité, A. Universal Path Planning for an Indoor Drone. *Autom. Constr.* **2018**, *95*, 275–283. [CrossRef]
35. Wu, K.J.; Gregory, T.S.; Moore, J.; Hooper, B.; Lewis, D.; Tse, Z.T.H. Development of an Indoor Guidance System for Unmanned Aerial Vehicles with Power Industry Applications. *IET Radar Sonar Navig.* **2017**, *11*, 212–218. [CrossRef]
36. Jeong, H.; Jeong, B.; Han, M.; Cho, D. Analysis of Fine Crack Images Using Image Processing Technique and High-Resolution Camera. *Appl. Sci.* **2021**, *11*, 9714. [CrossRef]
37. Karaman, S.; Frazzoli, E. Sampling-Based Algorithms for Optimal Motion Planning. *Int. J. Robot. Res.* **2011**, *30*, 846–894. [CrossRef]
38. Vagale, A.; Oucheikh, R.; Bye, R.T.; Osen, O.L.; Fossen, T.I. Path Planning and Collision Avoidance for Autonomous Surface Vehicles I: A Review. *J. Mar. Sci. Technol.* **2021**, *26*, 1292–1306. [CrossRef]
39. Petrović, L.; Peršić, J.; Seder, M.; Marković, I. Stochastic Optimization for Trajectory Planning with Heteroscedastic Gaussian Processes. In Proceedings of the 2019 European Conference on Mobile Robots (ECMR), Prague, Czech Republic, 4–6 September 2019; pp. 1–6.
40. Gammell, J.D.; Srinivasa, S.S.; Barfoot, T.D. Informed RRT*: Optimal Sampling-Based Path Planning Focused via Direct Sampling of an Admissible Ellipsoidal Heuristic. In Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, IL, USA, 14–18 September 2014; pp. 2997–3004.
41. Fan, X.; Guo, Y.; Liu, H.; Wei, B.; Lyu, W. Improved Artificial Potential Field Method Applied for AUV Path Planning. *Math. Probl. Eng.* **2020**, *2020*, 6523158. [CrossRef]
42. Hamieh, A.; Ben Makhlouf, A.; Louhichi, B.; Deneux, D. A BIM-Based Method to Plan Indoor Paths. *Autom. Constr.* **2020**, *113*, 103120. [CrossRef]
43. Amir, I.; Mani, G.-F. 4D BIM Based Optimal Flight Planning for Construction Monitoring Applications Using Camera-Equipped UAVs. *Comput. Civ. Eng.* **2019**, *2019*, 217–224.
44. Wang, F.; Zou, Y.; del Rey Castillo, E.; Ding, Y.; Xu, Z.; Zhao, H.; Lim, J.B.P. Automated UAV Path-Planning for High-Quality Photogrammetric 3D Bridge Reconstruction. *Struct. Infrastruct. Eng.* **2022**, *20*, 1595–1614. [CrossRef]

45. Shu, J.; Li, J.; Zhang, J.; Zhao, W.; Duan, Y.; Zhang, Z. An Active Learning Method with Difficulty Learning Mechanism for Crack Detection. *Smart Struct. Syst.* **2022**, *29*, 195–206. [CrossRef]
46. Yu, K.; Zhang, C.; Shooshtarian, M.; Zhao, W.; Shu, J. Automated Finite Element Modeling and Analysis of Cracked Reinforced Concrete Beams from Three Dimensional Point Cloud. *Struct. Concr.* **2021**, *22*, 3213–3227. [CrossRef]

**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.