



Research on Ground Object Echo Simulation of Avian Lidar

Zhigang Su ^{1,*} , Le Sang ¹, Jingtang Hao ¹, Bing Han ¹ , Yue Wang ² and Peng Ge ³

¹ Sino-European Institute of Aviation Engineering, Civil Aviation University of China, Tianjin 300300, China; 2021122034@cauc.edu.cn (L.S.); jthao@cauc.edu.cn (J.H.); b-han@cauc.edu.cn (B.H.)

² Information Countermeasure Technology Laboratory, Beijing Research Institute of Telemetry, Beijing 100076, China; wangyue@brit.com.cn

³ The 38th Research Institute of China Electronics Technology Group Corporation, Hefei 230093, China; gepeng09@gmail.com

* Correspondence: srsu@vip.sina.com

Abstract: The clutter suppression effect of ground objects significantly impacts the detection and tracking performance of avian lidar on low-altitude bird flock targets. It is imperative to simulate the point cloud data of ground objects in lidar to explore effective methods for suppressing clutter caused by ground objects in avian lidar. The traditional ray-tracing method is enhanced in this paper to efficiently obtain the point cloud simulation results of ground objects. By incorporating a beam constraint and a light-energy constraint, the screening efficiency of effective rays is improved, making them more suitable for simulating large scenes with narrow lidar beams. In this paper, a collision detection scheme is proposed based on beam constraints, aiming to significantly enhance the efficiency of ray-tracing collision detection. The simulation and experimental results demonstrate that, in comparison with other conventional simulation methods, the proposed method yields the point cloud results of ground objects that exhibit greater conformity to the actual lidar-collected point cloud results in terms of shape characteristics and intensity features. Additionally, the simulation speed is significantly enhanced.

Keywords: lidar; point cloud; simulation; collision detection; ground clutter



Citation: Su, Z.; Sang, L.; Hao, J.; Han, B.; Wang Y.; Ge, P. Research on Ground Object Echo Simulation of Avian Lidar. *Photonics* **2024**, *11*, 153. <https://doi.org/10.3390/photronics11020153>

Received: 22 December 2023

Revised: 25 January 2024

Accepted: 1 February 2024

Published: 5 February 2024



Copyright: © 2024 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

The lidar system enables the rapid and precise acquisition of accurate positional information regarding obstacles or targets within its environment, making it a valuable tool in environmental perception, intelligent navigation, and other related fields [1,2]. However, when monitoring bird activity in the low-altitude airspace near airports, the presence of ground object echoes significantly interferes with the detection and tracking of bird flocks by the lidar system. Methods based on deep learning, such as GACNet [3], SSTNet [4], CVANet [5], etc., are commonly employed for point cloud target segmentation, which involves the separation of ground object targets from lidar-received point clouds and the subsequent suppression of the ground object echo. However, deep learning methods necessitate an ample number of labeled samples to train their networks, a requirement that cannot be met by the limited quantity of actual collected data. Therefore, it becomes imperative to simulate lidar's ground object echoes and generate sufficient annotated samples in order to effectively train relevant networks and explore effective approaches for suppressing ground object echoes.

The main methods for simulating lidar target echoes include ray tracing [6–8], physical optics simulation [9–11], radar scattering field simulation [7,12,13], and other approaches. The ray-tracing method is capable of accurately simulating the propagation and scattering process of light, thereby generating high-precision echo signals. However, due to its computational complexity and slow execution speed, ray-tracing methods are generally more suitable for scenarios involving small targets and simple scenes. The physical optical

simulation method is based on a precise physical optical model to accurately simulate the optical characteristics of the target and generate high-precision echo signals. However, this method is highly dependent on the accuracy of the model and optical parameters, making it challenging to simulate complex scenes. The radar scattering field simulation method utilizes numerical computing technology to accurately simulate the interaction between the target and its surrounding environment, thereby generating efficient and precise echo signals. This approach necessitates the detailed modeling of the electromagnetic characteristics of the target, which entails substantial data and computational resources.

The limited storage capacity and reading and writing speed of the system restrict avian lidar from simultaneously improving the monitoring range and spatial resolution. Therefore, in order to ensure the coverage of the monitored airspace, avian lidar typically sacrifices spatial resolution performance and adopts a lower spatial resolution. Therefore, it is unnecessary to excessively focus on the detailed information of ground objects when simulating the echo of a ground object in avian lidar. In essence, a simplified model can be employed to describe the ground object in the echo simulation, thus rendering the ray-tracing method suitable for effectively simulating the ground object echo detected by avian lidar.

In ray-tracing methods, the rays emitted by and received by lidar are considered effective rays. Clearly, when tracking the rays emitted by lidar, the light path becomes more complicated due to reflections generated upon collision with object surfaces; however, the proportion of effective rays remains relatively low. Conversely, non-effective rays consume significant resources and impede the efficiency of ray-tracing methods. Common approaches to addressing the issue of extensive computational complexity in ray-tracing methods involve enhancing hardware computing capabilities or improving effective ray-screening techniques. Typically, GPUs serve as prominent hardware devices for boosting computing performance [14,15]. The parallel computing prowess of GPUs can significantly enhance computational efficiency. Furthermore, devices incorporating the Compute Unified Device Architecture (CUDA) can further optimize the GPU's computing power utilization. Enhanced methods for effective ray screening primarily involve reverse ray tracing or bidirectional ray tracing [6]. Reverse ray tracing is based on backward path tracing from the receiving end of the lidar to the transmitting end, ensuring that all rays are effective. However, this method does not accurately account for light refraction. Bidirectional ray tracing involves simultaneously calculating the light paths in both directions from both ends of the lidar, thereby improving the proportion of effective rays and enhancing performance.

In order to enhance the efficiency of ground object echo simulation for avian lidar and reduce the hardware calculation requirements of the simulation system, an improved ray-tracing method is proposed in this paper. The method incorporates the bidirectional ray-tracing concept, enabling the efficient screening of effective rays through a beam constraint and light-energy constraint, thereby avoiding the tracking of non-effective rays. Additionally, the method employs a batch processing approach for the batch collision detection of ray clusters, thus improving the collision detection efficiency in ground object echo simulation. The key contributions of this paper are as follows:

1. The beam constraint and light-energy constraint are determined based on the narrow beam of lidar and the lowest responsive light-energy level. In ground object echo simulation, these constraints can significantly enhance the efficiency of effective light screening and reduce the hardware calculation performance required by the simulation method.
2. The proposed collision detection scheme enables the simultaneous detection of all rays within the narrow lidar beam, thereby significantly reducing the time required for collision detection in the ray-tracing process.

The remaining sections of this paper are structured as follows: Section 2 provides an in-depth description of the models involved in laser pulse propagation. In Section 3, we introduce the batch collision detection scheme. Subsequently, we summarize the concrete implementation steps of the simulation method proposed in this paper in Section 4.

The simulation results are presented in Section 5, while a comprehensive discussion is provided in Section 6. Finally, the conclusion of this paper is presented in Section 7.

2. Associated Models of Ray-Tracing Method

A typical lidar structure comprises a base, a rotor, and a laser mounted on the rotor, as depicted in Figure 1a. The laser consists of a vertically and pitch-adjustable multi-channel transmitter/receiver. To avoid mutual interference between different channels during scanning, the laser transmitter/receiver is activated in a specific staggered sequence, as illustrated in Figure 1b. Consequently, the channel interference is not considered in this paper when simulating ground object echoes.

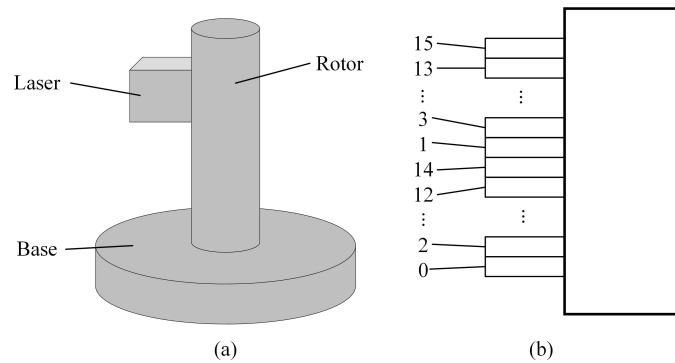


Figure 1. A schematic diagram of lidar: (a) Typical structure. (b) Activation sequence of the channels.

The entire lidar detection process encompasses the emission of laser pulses from the transmitter, their reflection off the object’s surface, and their reception by the receiver. Hence, this process can be divided into five stages: emission, forward propagation, reflection, backward propagation, and reception, as depicted in Figure 2. The lidar ground object echo simulation aims to replicate the trajectory alteration and energy attenuation of laser pulses emitted by the lidar system across five distinct stages. By conducting a comprehensive scene scan, the simulation achieves a faithful representation of ground object echoes throughout the entire scene. During this process, the Monte Carlo beamwidth model is employed to simulate the transmitted beam of the lidar, while an atmospheric attenuation model analyzes light-energy loss during laser pulse propagation in the air. Additionally, a surface light energy reflectance model describes interactions between the ray and object surfaces, and a receiver noise model further characterizes the noise impact on ground object echoes.

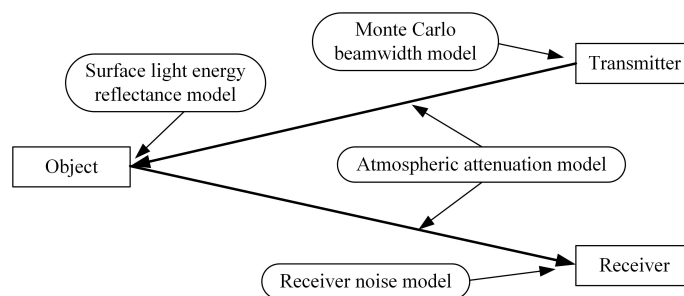


Figure 2. The process of laser pulse propagation.

The simulation of ground object echoes involves constructing a three-dimensional (3-D) virtual scene that represents the lidar working environment through computer modeling. The world origin, located at the center of this virtual scene, serves as the coordinate origin for establishing an East–North–Up coordinate system within the Cartesian coordinate framework. The object mesh grid is imported into the scene for further scanning. Each object in the scene is defined by its position and rotation information. The position

information is represented using a 3-D coordinate vector, while the rotation information can be expressed either through an orientation vector or a rotation matrix. By altering the lidar’s direction, we are able to achieve the scanning of the 3-D scene and obtain point cloud results that simulate echoes from all objects within this scene.

2.1. Monte Carlo Beamwidth Model

The laser pulse emitted by lidar propagates over a certain distance with a specific beamwidth. In the ray-tracing method, the propagation of the laser pulse is described using rays; however, this representation fails to fully capture the beam characteristics of laser pulse radiation. To address this limitation, a Monte Carlo beamwidth model is employed to describe the transmitting end of the lidar system. This model simulates a beam with a predetermined beamwidth [6,7,11] by utilizing multiple randomly distributed rays within that range, as illustrated in Figure 3.

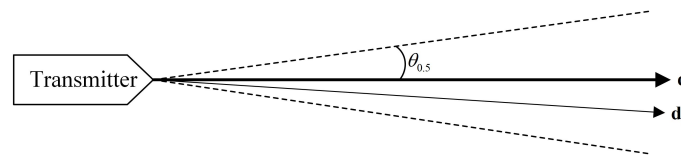


Figure 3. Monte Carlo beamwidth model.

The laser pulse is assumed to be a rectangular pulse with a pulse width of τ_e and an energy of Q_s . The total energy of N random rays within the beamwidth is denoted as

$$Q_s = \sum_{n=1}^N Q(\theta_n) \tag{1}$$

where the offset angle, θ_n , represents the angular difference between the n -th random ray and the direction vector \mathbf{d} of the beam center. This can be specifically expressed as

$$\theta_n = \widehat{\mathbf{d}, \mathbf{d}_n} \tag{2}$$

with the direction vector \mathbf{d}_n representing the direction of the n -th random ray. If the half-beamwidth is denoted by $\theta_{0.5}$, $\theta_n \in [-\theta_{0.5}, \theta_{0.5}]$. In Equation (1), the function $Q(\theta_n)$ denotes the energy of the ray at the offset angle θ_n , which describes how the energy of the emitted ray is distributed with respect to this offset angle. Assuming that the radiant energy density is uniform across the beamwidth, we have

$$Q(\theta_n) = \frac{1}{N} Q_s \tag{3}$$

2.2. Surface Light Energy Reflectance Model

After the ray is reflected by the surface of the object, there exists a relationship between the reflected light energy Q_{out} and the incident light energy Q_{in} :

$$Q_{out} = Q_{in} f_{sur}(\theta_i, \theta_r) \tag{4}$$

where $f_{sur}(\theta_i, \theta_r)$ is the surface light energy reflectance function, θ_i is the incidence angle, and θ_r is the reflected angle. The value range of θ_i is $[0, \pi/2]$, while the value range of θ_r is $[-\pi/2, \pi/2]$. When the reflected ray and the incident ray are located on the same side of the normal vector of the object surface, θ_r is negative, and vice versa. The surface light energy reflectance function ($f_{sur}(\theta_i, \theta_r)$) depends not only on the incidence angle and reflected angle but also on the roughness of the material on the object’s reflecting surface. The surface light energy reflectance function ($f_{sur}(\theta_i, \theta_r)$) comprises a diffuse reflectance component and a specular reflectance component:

$$f_{sur}(\theta_i, \theta_r) = f_d(\theta_i) + f_s(\theta_i, \theta_r) \tag{5}$$

where $f_d(\theta_i)$ is the diffuse reflectance component, while $f_s(\theta_i, \theta_r)$ is the specular reflectance component. The schematic diagrams in Figure 4a,b illustrate the phenomena of diffuse reflection and specular reflection of a ray on an object's surface, respectively. Within Figure 4, the vectors \mathbf{d}_i , \mathbf{d}_r , and \mathbf{d}_t represent the direction vectors for the incident ray, reflected ray, and specular reflection exit directions, respectively.

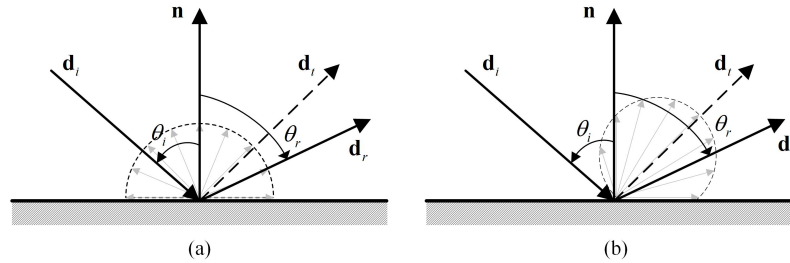


Figure 4. Ray reflection on surface: (a) Diffuse reflection. (b) Specular reflection.

According to the Lambert Cosine Law, the diffuse reflectance component can be described as [16]

$$f_d(\theta_i) = k_d \cos \theta_i \tag{6}$$

where k_d is the diffuse coefficient. Equation (6) demonstrates that the diffuse reflectance component is contingent upon the incident angle θ_i . As the incident angle increases, the diffuse reflectance component diminishes.

In the case where the surface of an object behaves like an ideal mirror, the ray will exit in a direction known as \mathbf{d}_t . However, due to imperfections on the object's surface, the exiting ray will reflect diffusion, resulting in a region centered around \mathbf{d}_t , as illustrated in Figure 4b. According to the Blinn–Phong model, the specular reflectance component is defined as [16]

$$f_s(\theta_i, \theta_r) = k_s \max \left\{ 0, \cos^{k_p}(\theta_r - \theta_i) \right\} \tag{7}$$

where k_s is the specular coefficient, while k_p is the glossiness coefficient. The specular reflectance component, as indicated by Equation (7), achieves its maximum value k_s in the direction of \mathbf{d}_t and gradually decreases with increasing offset angle relative to \mathbf{d}_t . The glossiness coefficient k_p determines the effective distribution range of reflected light during specular reflection; a larger value corresponds to a smaller effective distribution range.

Substituting Equations (6) and (7) into Equation (5) yields

$$f_{sur}(\theta_i, \theta_r) = k_d \cos \theta_i + k_s \max \left\{ 0, \cos^{k_p}(\theta_r - \theta_i) \right\} \tag{8}$$

The parameters k_d , k_s , and k_p in Equation (8) for the surface material of the object can be determined by fitting the reflectance data from various material surfaces in the MERL BRDF material database [17]. Given these surface material parameters, the surface light energy reflectance function $f_{sur}(\theta_i, \theta_r)$ is dependent on both the incidence angle θ_i and the reflected angle θ_r . When a ray is reflected off the object's surface, the reflected ray exists within a range of $[-\pi/2, \pi/2]$ centered around the normal direction of the surface.

In the simulation of ground object echoes, our focus is solely on the ray that is reflected by the target and received by the lidar or adjacent targets of interest. Therefore, the direction (\mathbf{d}_r) of a reflected ray is determined based on the positional relationship between the lidar or target of interest and the reflecting surface. Subsequently, the corresponding reflected angle (θ_r) is obtained. By utilizing Equation (8), the value of $f_{sur}(\theta_i, \theta_r)$ is calculated, which represents the reflectance function for light energy. Finally, substituting $f_{sur}(\theta_i, \theta_r)$ into Equation (4) allows us to determine Q_{out} , representing the reflected light energy.

2.3. Atmospheric Attenuation Model

The attenuation effect of light propagating in the atmospheric medium cannot be disregarded. Previous studies [18,19] have all examined the form of attenuation for light propagation in this medium. The atmosphere is modeled as an isotropic uniform medium, and the attenuation coefficient (μ) is assumed to remain constant, resulting in an exponential model for atmospheric attenuation.

$$Q_a(l) = Q_0 \exp(-\mu l) \tag{9}$$

where Q_0 is the initial energy of the light, while $Q_a(l)$ represents the energy of the laser pulse after traveling a distance of l .

2.4. Receiver Noise Model

The Monte Carlo beamwidth model employed by lidar at the transmitting end utilizes N random rays to simulate the transmission beam. Consequently, it is imperative that N random rays are also reflected back to the receiver at the receiving end for unified processing, ensuring that the light energy received by the receiver is

$$Q_{rec}[m] = \sum_{n=1}^N Q_r(\theta_n, \tau_n) + Q_n[m] \tag{10}$$

where m corresponds to a specific range unit within lidar, measured in terms of sampling time intervals (τ_s); $Q_{rec}[m]$ signifies the amount of light energy detected by the m -th range unit; $Q_r(\theta_n, \tau_n)$ quantifies how much light energy is captured when an emitted laser pulse is reflected by an object's surface via the n -th random ray; θ_n and τ_n represent, respectively, both an offset angle for the n -th random ray and a temporal delay for receiving its corresponding laser pulse; and $Q_n[m]$ stands for noise energy picked up in the m -th range unit, where it is defined as

$$Q_n[m] = b[m]\tau_s \cdot h\nu \tag{11}$$

where $b[m]$ represents the quantity of noise photons detected in the m -th range unit, h denotes the Planck constant, and ν corresponds to the frequency of the laser. Generally, the distribution of $b[m]$ follows a Poisson distribution with the parameter $\bar{b} \cdot \tau_s$, where \bar{b} represents the noise photon rate of the environment in which the lidar receiver is situated.

The dynamic range of the lidar receiver is typically limited, meaning that the receiver has acceptable upper and lower thresholds for receiving optical signals. These thresholds are recorded as Q_{high} (upper energy limit) and Q_{low} (lower energy limit), respectively. As a result, the received light energy can only fluctuate within the range from Q_{low} to Q_{high} . Any light energy ($Q_{rec}[m]$) exceeding the upper limit will be clamped, while $Q_{rec}[m]$ below the lower limit will not be perceived by the receiver. Consequently, after entering the receiver, the light energy becomes

$$Q_{en}[m] = \begin{cases} Q_{high}, & Q_{rec}[m] \geq Q_{high} \\ Q_{rec}[m], & Q_{low} < Q_{rec}[m] < Q_{high} \\ 0, & Q_{rec}[m] \leq Q_{low} \end{cases} \tag{12}$$

2.5. Received Light Energy Presentation Model

When conducting ground object echo simulation, it is necessary to visually represent the intensity information of the laser signal received by the receiver in color. To achieve a more enhanced rendering effect, the Hue-Saturation-Brightness (HSB) color mode, which closely emulates the human visual experience, is employed to accurately represent the intensity information of point clouds. The saturation and brightness of the color should be set to 1 while mapping the dynamic range of the received light energy [Q_{low}, Q_{high}] to the hue range from blue to red, as illustrated in Figure 5.

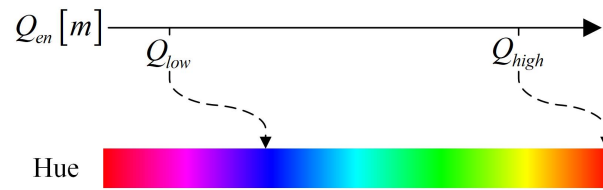


Figure 5. The mapping relationship between the received light energy and hue in the HSB mode.

As can be seen from Figure 5, the intensity of the received light energy is represented by the color tone. Because the hue values for cool blue and warm red are 0° and 240° , respectively, the hue value corresponding to the incoming laser energy at the receiver is

$$h[m] = \frac{Q_{high} - Q_{en}[m]}{Q_{high} - Q_{low}} 240^\circ \tag{13}$$

3. Collision Detection Scheme

The collision detection between the ray and the object surface in the scene poses a significant challenge when employing the ray-tracing method to simulate ground object echoes. The efficiency of the ray-tracing method is directly influenced by the quality of the collision detection scheme.

The term “collision detection” refers to the process of determining whether the emitted ray intersects with any objects in the scene based on its starting point and direction. If a collision occurs, it indicates that the ray illuminates an object along its propagation path within the scene; otherwise, the ray will not illuminate any objects and will continue infinitely. In terms of collision detection results, the closest point to the ray’s starting point represents where it shines onto an object’s surface.

After the incidence of a ray on an object’s surface, it undergoes reflection. As per the surface light energy reflectance model outlined in Section 2.2, the reflected rays are distributed within the range $[-\pi/2, \pi/2]$ centered around the normal vector \mathbf{n} of the object’s surface. A portion of these reflected rays directly return to the lidar receiver, while some no longer intersect and vanish into infinity. Additionally, certain portions collide with objects in the scene once again and are reflected. The aforementioned processes continue until all corresponding light energy is depleted. Since each emitted ray can undergo multiple reflections, conducting multiple collision detections for each ray becomes necessary. Optimizing the collision detection scheme directly enhances efficiency in ray-tracing methods.

3.1. Construction of Collision Detection Tree

The objects within a 3-D scene can be perceived as an arrangement of triangular surfaces. Collision detection between the ray and the object surface in the scene essentially involves detecting collisions between the ray and these triangular surfaces. A straightforward approach to collision detection would involve traversing all triangular surfaces within the scene to identify any collisions with the ray, but this method is computationally demanding and impractical. To enhance collision detection efficiency, it becomes necessary to construct bounding volumes of varying sizes based on the triangular planes present in the scene, utilizing these bounding volumes to establish a collision detection tree that enables the rapid identification of collisions between the ray and triangular planes.

The bounding volume is a simple geometric structure containing multiple triangular surfaces, which enables the ray to initially assess the intersection with the bounding volume and subsequently determine the precise points of potential intersections within its respective scope. The bounding volume serves as an efficient means for rapid area screening. Commonly employed types of bounding volumes include the Axis-Aligned Bounding Box (AABB) [20], Oriented Bounding Box (OBB) [21], and Sphere Bounding Volume (SBV) [22].

The collision detection tree is a tree structure based on bounding volumes, designed to optimize the calculation of collision detection. Commonly used tree structures include the Bounding Volume Hierarchy (BVH) tree [23], Binary Space Partitioning (BSP) tree [24], and k-dimensional (k-D) tree [25]. The collision detection tree consists of a root node, intermediate nodes, and leaf nodes. Each node in the collision detection tree is denoted as

$$node_i = \{B_i, F_i, p_{i,1}, p_{i,2}\} \quad (14)$$

where B_i is the bounding volume corresponding to this node, while F_i represents the triangular plane inside the leaf node. The numbers $p_{i,1}$ and $p_{i,2}$ represent the two child nodes of this node and are excluded in the case of a leaf node. The set $S_{node} = \{node_i\}$ comprises all nodes that form the collision detection tree.

The construction process of the collision detection tree involves determining each node of the tree sequentially, as depicted in Algorithm 1. Steps 1 and 2 in Algorithm 1 serve as initialization processes for the relevant parameters. Firstly, it is essential to construct a bounding volume B_i for each element F_i within the set S_{flat} and incorporate it into the bounding volume set S_1 . Secondly, the variables involved in constructing the collision detection tree are initialized. Steps 3 to 18 outline the detailed construction process of the collision detection tree. When the set S_{layer} is not empty, nodes need to be constructed from its elements. For each element L_p in the set S_{layer} , if it contains more than one bounding volume, a corresponding root node or intermediate node should be created; if it contains only one bounding volume, a leaf node needs to be constructed. In the case of a root node or intermediate node, all bounding volumes within L_p should be merged into a larger bounding volume B_{j0} , which will serve as the bounding volume for this particular node. Additionally, all bounding volumes within L_p must be divided into two subsets based on the construction rules of the respective tree structure and temporarily stored in the set S_{temp} . The BVH tree construction in step 9 allows for the direct division of the set into its first and second parts. On the other hand, when constructing a k-D tree, it is necessary to iteratively select dimensions as axes for partitioning and split the set into two approximately equal parts along these dimensions. Node $node_{j0}$ integrates B_{j0} with child node numbers j_1 and $j_1 + 1$, which are then included in the collection S_{node} . For leaf nodes, both the last bounding volume of the node and its corresponding triangular plane are integrated into node $node_{j0}$ and added to the set S_{node} . The set S_{layer} is updated with the set S_{temp} until all elements have been traversed, resulting in an empty set S_{layer} . Ultimately, the set S_{node} encompasses all nodes of the collision detection tree, thus completing its construction.

The collision detection between a ray and the object surface in the scene is achieved by utilizing the intersection of the ray and the bounding volume. The collision detection tree enables the efficient retrieval of the leaf node that intersects with the ray, ultimately providing information about the triangular surface where the collision occurs.

3.2. Intersection of Beam and Bounding Volume

The Monte Carlo beamwidth model is employed to describe the emitted beam of lidar, resulting in the confinement of the corresponding N random rays within a narrow beamwidth. During the collision detection between rays and surface elements, there exists a high probability that these N random rays will collide with the bounding volume in either the same leaf node or adjacent leaf nodes. Consequently, this inevitably leads to an increased number of identical intermediate nodes among these N random rays. Therefore, utilizing beams for retrieving leaf nodes can effectively reduce the computation required for each random ray and enhance the retrieval efficiency.

When retrieving leaf nodes using a beam, it is essential to judge the intersection between the conical beam and the bounding volume. The judgment only needs to be whether the beam and the bounding volume intersect, without necessitating the identification of the specific point or line of intersection.

Algorithm 1 Construction of the collision detection tree.**Input:** The collection of triangular surfaces $S_{flat} = \{F_i\}$ **Output:** The set of nodes in the collision detection tree $S_{node} = \{node_i\}$

- 1: The minimum bounding volume B_i is constructed for each element F_i in set S_{flat} . The type of bounding volume can be selected based on the requirements, such as the AABB, OBB, or SBV. All the generated bounding volumes constitute the set $L_1 = \{B_i\}$.
- 2: The parameters of the collision detection tree are initialized. The current node j_0 is set to 1 and the child node j_1 is set accordingly. Complex sets $S_{layer} = \{L_p\}$ and $S_{temp} = \{L_q\}$ are constructed, which consist of bounding volume sets. It is ensured that S_{layer} contains only one element, namely, L_1 , while S_{temp} and S_{node} are set as empty sets. The number of elements in the set S_{layer} is assigned to the variable P .
- 3: **while** $P > 0$ **do**
- 4: Let $q = 1$ and $j_1 = j_0 + P$
- 5: **for** $p = 1$ **to** P **do**
- 6: The variable Q is assigned the number of bounding volumes in the p -th element L_p of the set S_{layer} .
- 7: **if** $Q > 1$ **then**
- 8: The smallest bounding volume B_{j_0} that can surround all bounding volumes in L_p is built.
- 9: According to the construction rules of tree structures, such as a BVH tree or k-D tree, the elements in L_p are partitioned into two subsets, namely, L_q and L_{q+1} , which are then added to the set S_{temp} as the q -th and $q + 1$ -th elements, respectively.
- 10: The j_0 -th node $node_{j_0} = \{B_{j_0}, j_1, j_1 + 1\}$ is constructed. As this node is not a leaf node, it does not contain any triangular plane information. The node $node_{j_0}$ is included in the collection S_{node} as the j_0 -th element.
- 11: Let $q = q + 2$, $j_1 = j_1 + 2$
- 12: **else**
- 13: The leaf node $node_{j_0} = \{B_{j_0}, F_{j_0}, j_1, j_1 + 1\}$ is constructed by assigning the unique bounding volume in L_p to B_{j_0} and associating the corresponding triangular surface with F_{j_0} . The leaf node will no longer possess child node numbers. The node $node_{j_0}$ is included in the collection S_{node} as the j_0 -th element.
- 14: **end if**
- 15: Let $j_0 = j_0 + 1$
- 16: **end for**
- 17: Let $S_{layer} = S_{temp}$ and $S_{temp} = \emptyset$
- 18: **end while**
- 19: **return** S_{node}

The intersection assessment between the beam and the bounding volume involves assessing the intersection between two geometric structures in 3-D space. This process can be decomposed into evaluating the intersection between the beam angle and the rectangle on each of the three orthogonal faces in a two-dimensional (2-D) plane, as illustrated in Figure 6. If there are overlaps between the projected beams in all three planes and the projected rectangle of the bounding volume, then it can be concluded that there is an intersection between the beam and the bounding volume.

The intersection assessment between the beam and the rectangle in a 2-D plane can be further decomposed into assessing the intersection between the two rays that form the beam and the rectangle, as illustrated in Figure 7. In a 2-D plane, a beam intersects with a rectangle only if at least one ray intersects with it or if the two rays are positioned on opposite sides of the rectangle. By considering the end point of each ray as its origin, we define the vector pointing from this origin to each vertex of the rectangle as a rectangular vertex vector. The cross-product operation is performed between the direction vector of the ray and each vertex vector of the rectangle. If the resulting cross-product vectors have opposite directions, it indicates an intersection between the particular ray and the rectangle.

Consequently, determining whether there is an intersection between a beam and bounding volume can be decomposed into evaluating whether there is an intersection among multiple rays and rectangles.

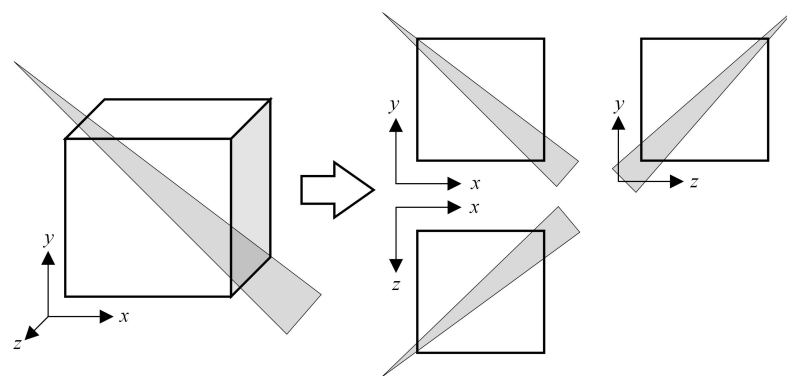


Figure 6. The intersection assessment between the beam and the bounding volume.

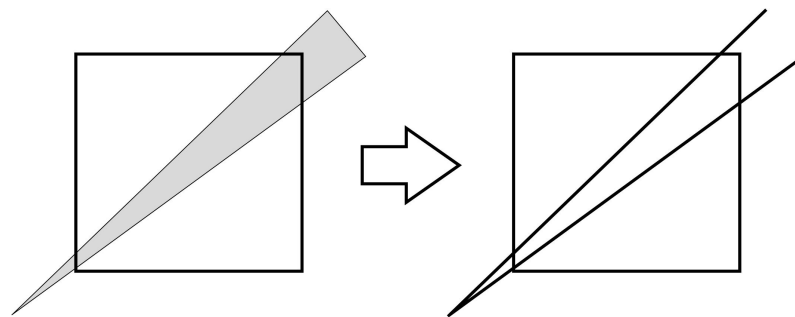


Figure 7. Intersection assessment between beam and rectangle.

The judgment of the beam-bounding volume intersection for deeper intermediate nodes within this bounding volume needs to be made if the initial judgment is true, until the leaf node that intersects the beam is retrieved.

3.3. Intersection Between Ray and Triangular Surface

By applying a beam constraint, the collision detection tree locates all leaf nodes that may intersect with random rays in the beam and constructs a set of potential collision surface elements ($S_{potential}$) from corresponding triangular surfaces. Each random ray is utilized to determine the intersection point of relevant triangular planes within the set.

The geometric configuration of the intersection assessment between a ray and a triangular surface is illustrated in Figure 8. Cartesian coordinates are established with O as the origin of the ray, where \mathbf{d} represents the direction vector of the ray. The position vectors of the three vertices A, B, and C on the triangular surface are denoted by \mathbf{p}_A , \mathbf{p}_B , and \mathbf{p}_C , respectively. Additionally, \mathbf{p}_I represents the position vector of the intersection point I between the ray and the triangular surface.

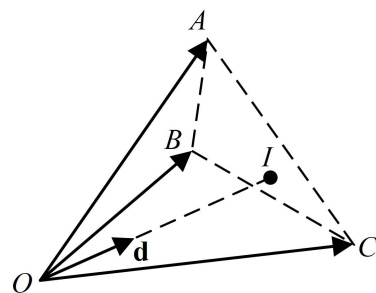


Figure 8. The intersection point between the ray and the triangular surface.

According to the geometric relationship shown in Figure 8, we can have

$$\mathbf{p}_I = w\mathbf{d} \tag{15}$$

where w is a positive real number. Meanwhile, the direction vector \mathbf{d} can also be decomposed into

$$\mathbf{d} = \alpha\mathbf{p}_A + \beta\mathbf{p}_B + \gamma\mathbf{p}_C \tag{16}$$

with the coefficients α , β , and γ being real numbers representing the decomposition of the direction vector \mathbf{d} into \mathbf{p}_A , \mathbf{p}_B , and \mathbf{p}_C , respectively. In geometry, point I being on the surface of triangle ABC is a necessary and sufficient condition, denoted by

$$w(\alpha + \beta + \gamma) = 1 \tag{17}$$

The sufficient and necessary condition for the direction vector \mathbf{d} inside the tetrahedron $OABC$ is that α , β , and γ are all non-negative and at least one of them is non-zero.

The values of α , β , and γ can be computed based on Equation (16). If the direction vector \mathbf{d} lies within the tetrahedron $OABC$, then it is necessary for the ray to intersect with the triangular surface ABC . By substituting Equation (17) into Equation (15), we can determine the position vector of the intersection point I :

$$\mathbf{p}_I = \frac{1}{\alpha + \beta + \gamma}\mathbf{d} \tag{18}$$

3.4. The Problem of Secondary Reflection

When a ray is reflected on the surface of an object, according to the surface light energy reflectance model, the reflected rays are distributed within the range of $[-\pi/2, \pi/2]$, centered around the normal vector \mathbf{n} on the object's surface. Apart from some of these reflected rays being directly reflected and received by a receiver, other portions may undergo secondary reflection upon colliding with other objects in the environment.

The reflection point of the secondary reflection must fall within the lidar receiving beam in order for the ray to enter the lidar receiver. Hence, when tracking secondary reflections, it is sufficient to consider only those arising from triangular surfaces among the set of potential collision surface elements ($S_{potential}$).

Each reflected ray undergoes significant attenuation compared to the incident ray. The increase in the number of reflections further amplifies the attenuation of light energy. A portion of the rays, after undergoing secondary reflection and reaching the receiver, fail to reach the level of sensitivity that allows for perception. Consequently, based on the sensitivity of the lidar receiver, backward precomputation is employed to calculate the minimum energy at each reflection point. If the energy of the reflected ray falls below this minimum threshold, it will no longer be tracked. This approach effectively avoids extensive, non-effective ray tracing.

After secondary reflection, there arises the issue of subsequent ray reflection once again, necessitating the repetition of the secondary reflection process. However, due to the attenuation of light energy during multiple reflections, the lidar receiver typically fails to perceive the laser pulse after undergoing multiple reflections. Therefore, in simulation processes, only the problem of secondary reflection is usually taken into consideration.

4. Proposed Simulation Method

On the basis of the aforementioned models of laser propagation and the collision detection scheme, the implementation process of the improved ray-tracing method for simulating ground object echoes can be explained, as depicted in Algorithm 2.

In the simulation method proposed in this paper, the collision detection tree is constructed using Algorithm 1 based on 3-D virtual scene information and radar parameters. Subsequently, during the simulation process of ground object echoes, the corresponding

points' 3-D position and intensity distribution information are acquired according to the beam. For a given beam, N random rays are generated using the Monte Carlo beamwidth model. Afterward, the collision information between these random rays and the triangular surface of the ground object is obtained through batch collision detection. The reflection direction of each ray is determined based on the triangular surface information of the ground object within the beam irradiation range and the position of the lidar receiver. If the reflected ray encounters another triangular surface, the information within the beam irradiation range and the position of the lidar receiver are reused to determine its reflection direction again, until it reaches the lidar or collides with a triangular surface outside of the beam irradiation range. For the ray received by lidar after one or more reflections, the light energy of the laser pulse transmitted along this ray is calculated at each crucial node. The received ray is then screened by the light-energy constraint. By summing up the energy corresponding to all the rays received by lidar within the beam and incorporating additive noise, the received light energy is transformed into $Q_{rec}[m]$. Finally, based on the received light energy presentation model, it is compared with the upper and lower limits to generate $Q_{en}[m]$, which is then mapped to points of different colors. The simulation of the interior object echo of a single beam is now complete. Upon sweeping through the entire 3-D scene, we can obtain the point cloud distribution, where distinct colors indicate varying intensities of ground object echoes at corresponding locations.

Algorithm 2 Proposed simulation method

Input: A 3-D virtual scene, parameter values for various materials and lidar

Output: The simulated point cloud of the 3-D virtual scene

- 1: Based on the information from the 3D virtual scene and lidar parameters, the collision detection tree is constructed using Algorithm 1.
 - 2: **for** traversing all beam directions in the ground object echo simulation **do**
 - 3: Generate N random rays according to Monte Carlo beamwidth model.
 - 4: By utilizing the collision detection tree, a batch collision detection is performed for N random rays in the beam.
 - 5: **for** traversing N random rays **do**
 - 6: **while** the ray collides with the triangular surface of the ground object within the beam irradiation range **do**
 - 7: The effective ray is filtered by applying the beam constraint. The direction of the reflected ray is determined based on the triangular surface information of the ground object within the illuminating range of the beam and the position of the lidar receiver.
 - 9: **end while**
 - 10: Specifically for the ray whose light path terminates in the lidar receiver, the models in Section 2 are utilized to compute the light energy of the laser pulse emitted by lidar at each crucial node along the propagation path.
 - 11: **if** the light of the laser pulse at each crucial note satisfies the light-energy constraint, **then**
 - 12: The lidar is capable of receiving the laser pulse propagated along this light path while preserving its light-energy information $Q_r(\theta_n, \tau_n)$.
 - 14: **else**
 - 15: The laser pulse propagating along the light path is insufficient to activate the lidar receiver, thus necessitating the disregard of this particular ray.
 - 16: **end if**
 - 17: **end for**
 - 18: All retained energy is added up, and additive noise is introduced to form the received light energy $Q_{rec}[m]$ in lidar. Based on the upper and lower limits of receivable light energy, the light energy $Q_{en}[m]$ is formed upon entering the receiver.
 - 19: According to the received light energy presentation model, the received energy is mapped to points of varying colors.
 - 20: **end for**
-

5. Simulation Results

The reliability and high efficiency of the ground object echo simulation method proposed in this paper are demonstrated through a series of simulation experiments conducted in this section.

5.1. Selection of Bounding Volume Type and Collision Detection Tree Structure

The BVH tree [23] or k-D tree [25] is constructed using the AABB [20] or SBV [22] bounding volumes to investigate the efficiency of collision detection for different combinations. The total time required for constructing the collision detection tree and performing collision detection under various combinations varies with the number of triangular surfaces, as depicted in Figure 9. It can be observed from Figure 9 that as the number of triangular surfaces increases, both the construction time of the collision detection tree and the total time spent on collision detection increase proportionally. Notably, when adopting the SBV, there is a significantly higher growth rate in total time compared to the AABB, which remains consistently high throughout. Under identical bounding volume conditions, the k-D tree exhibits higher construction and detection efficiency. Therefore, in this paper, the AABB is utilized to construct the k-D tree in ground object echo simulation.

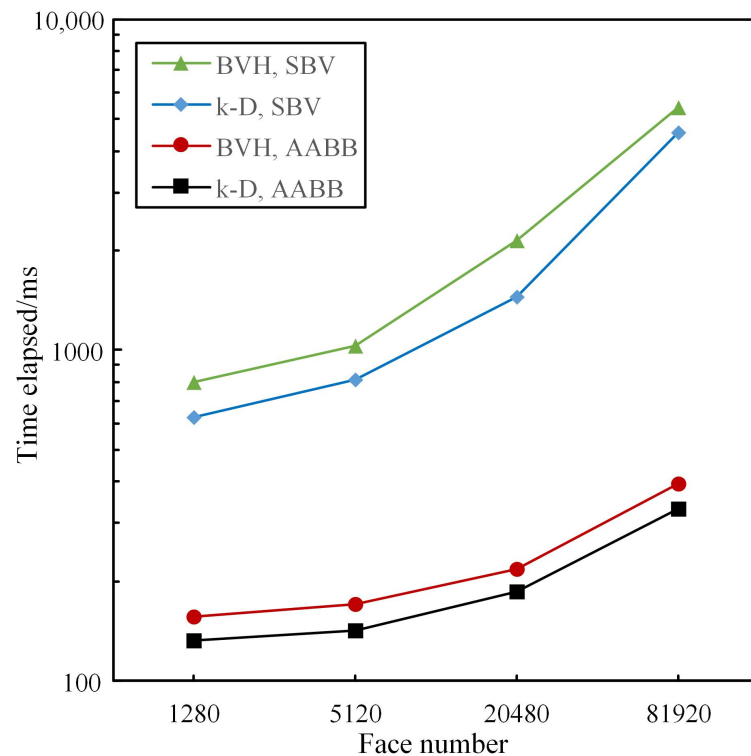


Figure 9. The time elapsed for constructing the collision detection tree and performing collision detection under various combinations.

The efficiency of the point cloud simulation method proposed in this paper is further analyzed here. In the simulation experiment, a spherical surface composed of a specific number of triangular surfaces was used as the simulation target. Figure 10 illustrates the point cloud simulation time for the proposed method, the method based on the bidirectional BRDF model [10], and the method based on the Phong model [11] with varying numbers of triangular surfaces. From Figure 10, it can be observed that when there are fewer triangular surfaces, the proposed method takes more time compared to the other two methods. However, as the number of triangular surfaces increases, the rate at which time consumption grows for simulating point clouds using our proposed method is significantly lower than those of both alternative methods. Consequently, as we increase the number of

triangular surfaces, our proposed method exhibits an increasingly advantageous reduction in time consumption compared to both alternative methods. When dealing with point cloud simulations in complex environments, our proposed method demonstrates superior advantages over these two alternatives.

The present study introduces a beam constraint, a light-energy constraint, and a batch collision detection scheme to expedite the screening of effective rays in the ray-tracing process and reduce the computational complexity of collision detection throughout the simulation. To further analyze the impact of these enhancements on simulation efficiency, it is beneficial to draw upon ablation experiments, commonly employed in neural networks, for discussion. In this experiment, a spherical surface composed of a specific number of triangular surfaces remains as the simulation target. Various combinations of the beam constraint, light-energy constraint, and batch collision detection schemes are incorporated based on the original ray-tracing method to assess time consumption during the point cloud simulation under different numbers of triangular surfaces. The specific results are illustrated in Figure 11. The combination of the original ray-tracing method with the beam constraint, the light-energy constraint, or the batch collision detection scheme resulted in a noticeable decrease in simulation time, as depicted in Figure 11. When the number of triangular surfaces is low, the combination of the original ray-tracing method and batch collision detection scheme exhibits the shortest execution time. However, when dealing with a high number of triangular surfaces, employing the combination of the original ray-tracing method and the beam constraint proves to be more efficient. Furthermore, incorporating any two aspects with the original ray-tracing method further reduces the simulation time. Notably, among these combinations, utilizing both the beam constraint and the batch collision detection scheme yields superior performance.

After combining the original ray-tracing method with the aforementioned three aspects to form the enhanced approach proposed in this paper, it is evident that the corresponding simulation time exhibits optimal performance. In conclusion, by addressing three key aspects of improvement within the original ray-tracing method, the proposed approach effectively enhances computational efficiency in simulations.

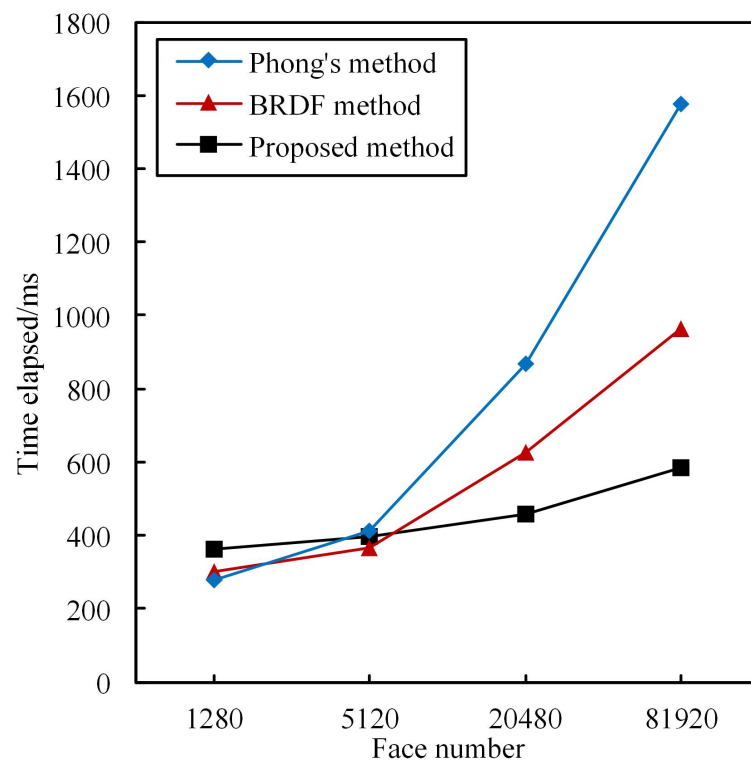


Figure 10. Time elapsed for different methods.

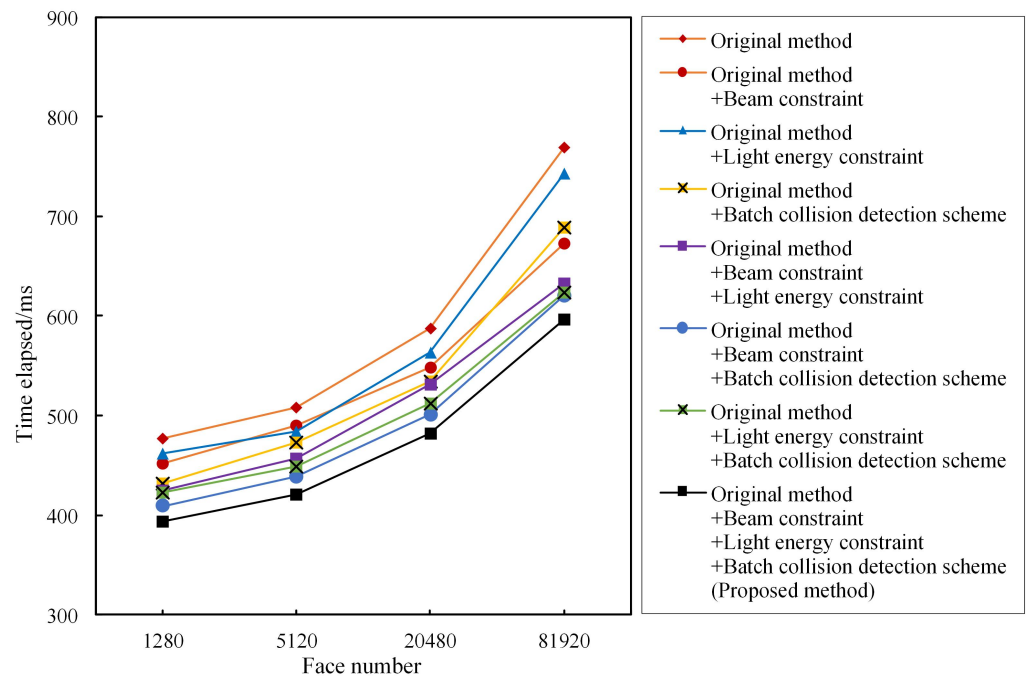


Figure 11. Results of ablation experiments.

5.2. Simulation Results of Simple Small Scene

The proposed method was utilized for simulating the point cloud of a simple small scene and validating its effectiveness by comparing it with the actual lidar-collected data. The experimental area selected is the corridor section within the laboratory, encompassing various elements, such as windows, doors, stairwells, walls, and long corridors, as depicted in Figure 12a. Based on the real scene information, a 3-D virtual scene was constructed, as shown in Figure 12b. The diffuse coefficient (k_d), specular coefficient (k_s), and glossiness coefficient (k_p) of different materials in the 3-D virtual scene were determined by fitting data from the MERL BRDF material library [17], with the specific values presented in Table 1.

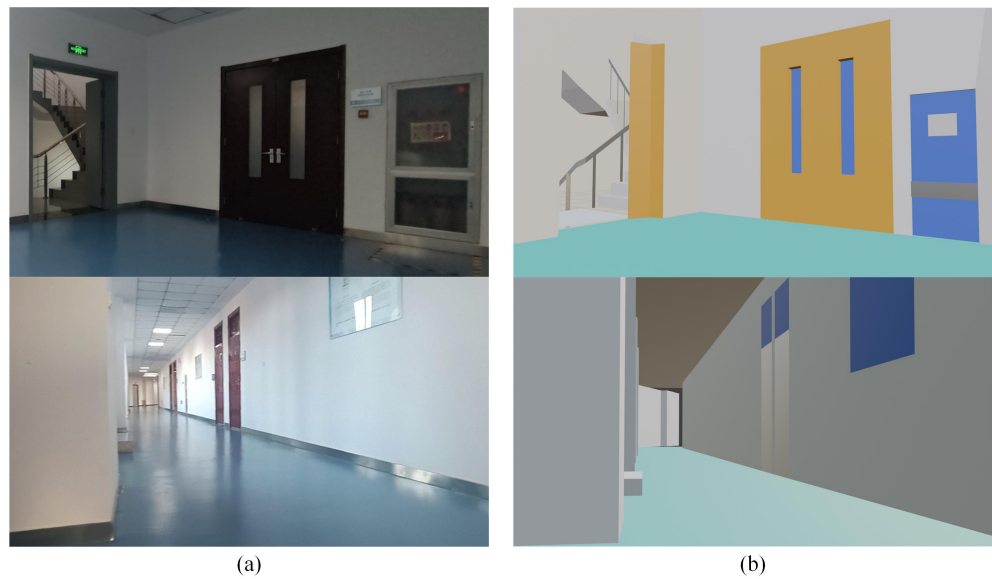


Figure 12. Experimental scene: (a) Real scene. (b) Virtual scene.

Table 1. The parameter values for various materials.

Material	k_d	k_s	p
Wall	0.146	0.054	112
Glass	0.020	0.852	8046
Metal	0.075	0.634	6803
Wood	0.107	0.100	82

The lidar utilized in this experiment is the Laser Intelligent C16 multi-line mechanical lidar. The horizontal resolution and vertical resolution of the lidar are 0.09° and 2.0° , respectively, with the vertical resolution being approximately 22 times greater than the horizontal resolution. To ensure a balanced point cloud data resolution in both the vertical and horizontal directions, the actual radar data collected during the experiment were downsampled by a factor of 22 along the horizontal direction. In order to compare the actual collected point cloud with the simulated point cloud, an object echo simulation was conducted using radar with set horizontal and vertical resolutions of 2.0° each. Other relevant parameter settings employed in this simulation can be found in Table 2. The simulation device utilizes an Intel i7-10510U CPU operating at a frequency of 1.8 GHz and is equipped with 8 GB of memory. All program codes were implemented in the Java language without utilizing GPU acceleration. Whether the point cloud was obtained through actual collection or simulated generation, the color tone represents the light-energy intensity of each corresponding point, while the diameter of the point indicates its distance distribution. The simulation method proposed in this paper takes 83 ms when simulating the simple small scene, whereas the original ray-tracing method requires approximately 130 ms. The proposed approach demonstrates a time reduction of about 36.2% compared to the original method.

Table 2. Configurations of model parameters.

Notation	Explanation	Value
$\theta_{0.5}$	Half-beamwidth	5.6×10^{-3} rad
N	Total number of random rays within beam	100
μ	Atmospheric attenuation coefficient	0.088
\bar{b}	Noise photon rate	20 kHz
h	Planck constant	6.626×10^{-34} J · s
ν	Frequency of the laser	3.313×10^{14} Hz
Q_s	Energy of the emitted laser pulse	2.5×10^{-4} J
Q_{high}	Receiver upper energy limit	6.25×10^{-5} J
Q_{low}	Receiver lower energy limit	2×10^{-6} J

Three typical areas are selected as the judgment indicators to evaluate the consistencies of the shape characteristics and intensity distribution between the real and simulated results.

The first area to consider is depicted in Figure 13a, featuring a wooden door with a window, a glass cabinet housing the fire hydrant, and adjacent walls. As indicated by the parameters in Table 1, wood and wall materials have higher diffuse coefficients (k_d), while glass materials exhibit a higher specular coefficient (k_s) and glossiness coefficient (k_p). Consequently, the reflection of wood and wall materials is predominantly diffuse, whereas glass materials primarily undergo specular reflection with weak diffuse reflection. Therefore, for a lidar system whose transmitter and receiver are located at the same position, only when ray irradiates the surface at a very small incidence angle can it receive specular reflection energy; otherwise, the received energy is mainly from diffuse reflections. Figure 13b,c depict the light intensity distribution of actual radar-acquired point clouds and simulated point clouds, respectively. In comparison with Figure 13a, it is evident that the wall echo exhibits maximum strength, followed by the wood echo, while the glass echo is

the weakest; furthermore, material boundaries are clearly discernible. Notably, the middle region of the wooden door reflects stronger echoes than its sides due to its surface being nearly perpendicular to the incident light, resulting in specular reflection.

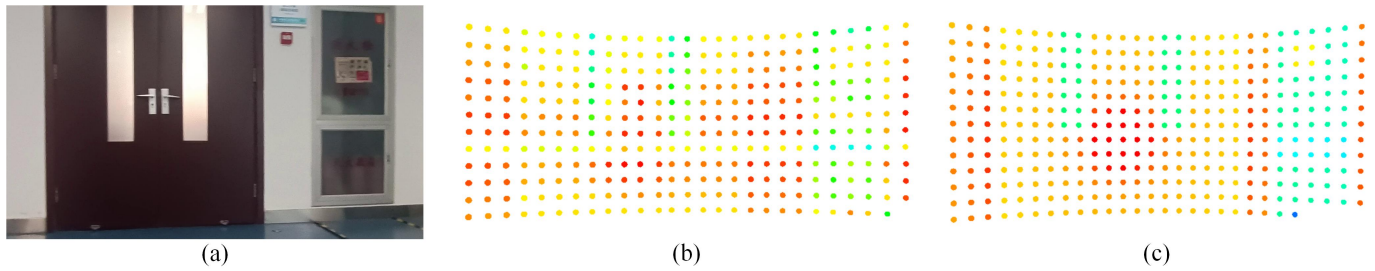


Figure 13. Door area: (a) Area photo. (b) Actual collected point clouds. (c) Simulated point clouds.

To accurately assess the distribution consistency between the actual collection point cloud and the simulated point cloud, the Hausdorff distance [26] and Root Mean Square Error (RMSE) [27] were employed as quantitative metrics. The Hausdorff distance, a commonly used measure of dissimilarity between two point clouds, considers the distances between each point in both clouds and their nearest neighbors, selecting the maximum value among them. The specific expression for calculating the Hausdorff distance is denoted by

$$D_{Hausdorff} = \max \left\{ \max_{\mathbf{p}_a} \left(\min_{\mathbf{p}_s} \|\mathbf{p}_a - \mathbf{p}_s\| \right), \max_{\mathbf{p}_s} \left(\min_{\mathbf{p}_a} \|\mathbf{p}_s - \mathbf{p}_a\| \right) \right\} \quad (19)$$

where \mathbf{p}_a and \mathbf{p}_s represent the coordinates of any point in the actual collected point cloud and simulated point cloud, respectively. Due to a discrepancy in the total number of points between the two clouds, the Root Mean Square Error (RMSE) is utilized to measure the distance between the nearest neighbors in each cloud, i.e.,

$$RMSE = \sqrt{\frac{1}{n} \sum_{a=1}^n \min_{\mathbf{p}_s} \|\mathbf{p}_a - \mathbf{p}_s\|^2} \quad (20)$$

where n represents the total number of points in the simulated point cloud. The smaller the values of the Hausdorff distance and RMSE, the higher the similarity between the two point clouds. Based on Figure 13b,c, the Hausdorff distance is measured to be 0.164, while the RMSE is calculated as 0.04.

The second selected area is the stairwell, as depicted in Figure 14a. As evident from Figure 14a, this area exhibits a relatively intricate geometric structure and comprises two different materials: metal and wall. The stair handrail is constructed of metal, while the wall and steps can be treated based on the wall material. As demonstrated by the point cloud results in Figure 14b,c, the complex structure of this region poses a challenge for fully capturing its structural information when point clouds are sparse. However, based on the distribution of point clouds, Figure 14b,c exhibit good similarity to each other. The Hausdorff distance and RMSE values are 0.908 and 0.33, respectively, indicating their strong resemblance, but it is weaker than that shown in Figure 13.

The final region analyzed is the long corridor region, as depicted in Figure 15a. As evident from Figure 15a, an acrylic bulletin board is installed in this area. The reflective characteristics of the material resemble those of glass, thus substituting the parameters of glass material for simulation purposes. Comparing Figure 15b,c, it can be observed that the intensity of point clouds reflecting the bulletin-board area is significantly weaker than that of point clouds on the surrounding wall. With increasing distance, the intensity of wall point clouds gradually diminishes. In even more distant regions, where the echo intensity falls below the receiver’s threshold and goes unnoticed, the corresponding areas are absent from the point cloud results. The Hausdorff distance and RMSE in this region

are 0.373 and 0.10, respectively. Evidently, the point cloud similarity is superior in the long corridor region compared to the stairwell region but inferior to that in the region depicted in Figure 13a.

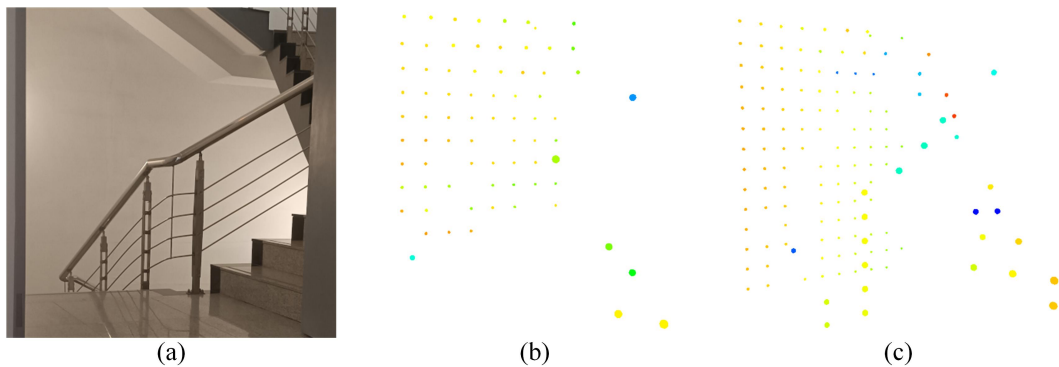


Figure 14. Stairwell area: (a) Area photo. (b) Actual collected point clouds. (c) Simulated point clouds.

The previous analysis of the point cloud consistency and intensity changes in key areas within the experimental scene confirms that the proposed point cloud simulation method in this paper effectively yields highly accurate results consistent with reality.

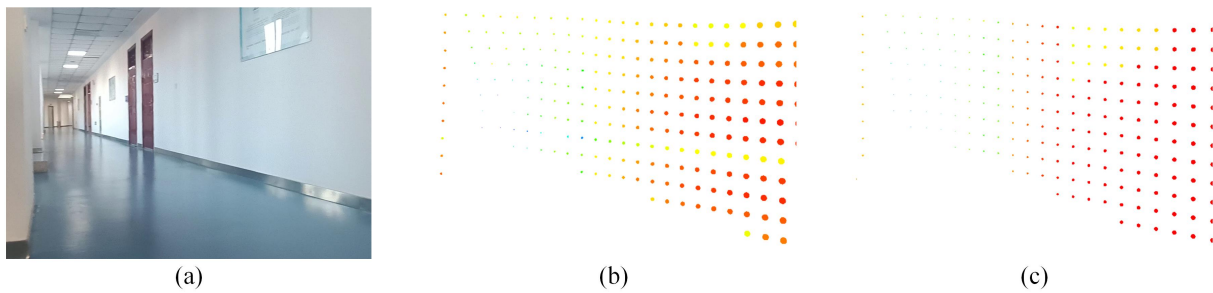


Figure 15. Corridor area: (a) Area photo. (b) Actual collected point clouds. (c) Simulated point clouds.

5.3. Simulation Results of Complex Large Scene

In this section, based on the technical parameters of avian lidar, the point cloud of the ground object echoes is simulated in a complex large scene. The selected scene is Shanghai Hongqiao International Airport. Figure 16a shows its satellite thumbnail, and the top view of the mesh grid of the main airport buildings is illustrated in Figure 16b. The lidar system is positioned near the center of the runway, indicated by a red “+” symbol in both Figure 16a,b. For this experiment, we adopted identical model parameters to those used to simulate the simple small scene, except for setting the atmospheric attenuation coefficient μ to 0.035 and transmitted pulse power Q_s to 3.6×10^{-4} J.

During the simulation of ground object echoes, the lidar’s vertical coverage range is 0 to 6 degrees, which is divided into either 16 or 32 channels. This means that the lidar has a vertical resolution of either 0.375 degrees or 0.1875 degrees, and its horizontal resolution is equal to the vertical resolution. The same hardware as described in Section 5.2 was utilized for this ground object echo simulation experiment. When using 16 channels, our proposed method completes the simulation for the entire airport in approximately 1122 ms, whereas the original ray-tracing method takes around 1439 ms. On the other hand, when employing all available 32 channels on the lidar system, our proposed method requires about 1633 ms compared to 1831 ms with the original ray-tracing approach.

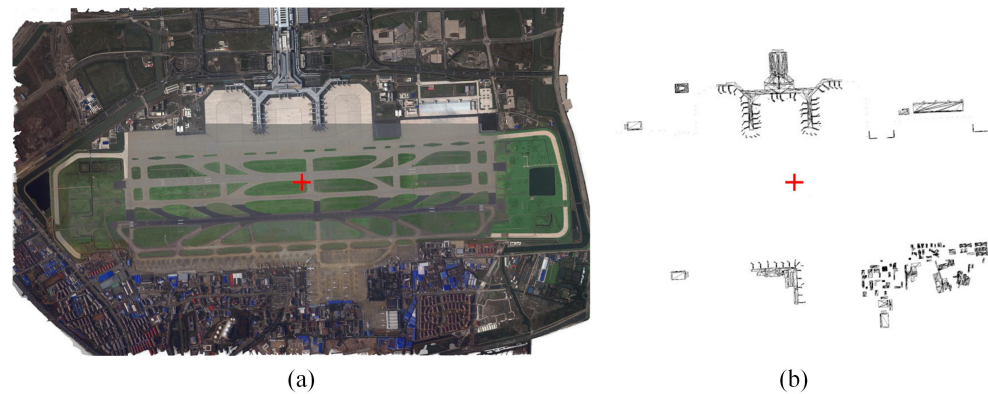


Figure 16. Shanghai Hongqiao international airport overview: (a) Satellite thumbnail. (b) Top view of mesh grid of main buildings.

Taking the T1 terminal of the airport as an example, the top views of simulated point clouds obtained by lidar using 16 channels and 32 channels are plotted in Figure 17a,b, respectively. In Figure 17, the black network represents the corresponding architectural structure based on the spatial positioning of the T1 terminal in the 3-D virtual scene. Simulated point clouds are represented by colored dots, indicating their coordinate positions within the same 3-D virtual space. Each point's coordinates within the cloud are determined by factors such as the lidar location, beam direction, and laser pulse travel distance. It can be observed from Figure 17a,b that both cases effectively capture the facade shape of the building toward the lidar. However, in Figure 17a,b, these point clouds appear blue due to their relatively low light energy compared to other colors on a color–energy mapping scale. This is primarily attributed to two factors: firstly, there is a significant distance between the lidar and the T1 terminal; secondly, considering this distance, the points on the T1 terminal are relatively close together compared to their distance from lidar. By comparing Figure 17a,b, it can be further observed that the lidar employs a greater number of channels to acquire denser point clouds, thereby enabling a more precise depiction of target details.

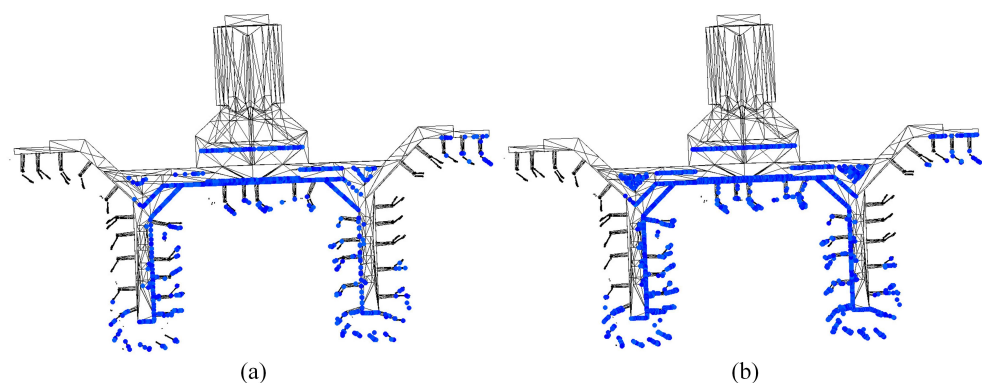


Figure 17. Top view for results of different vertical channels: (a) Sixteen channels. (b) Thirty-two channels.

By utilizing simulation techniques, a multitude of specific target point clouds can be conveniently generated under varying conditions, such as different noise levels, viewing angles, and resolutions. These generated point clouds can serve as annotated samples for training neural networks, enabling the trained network to accurately identify and eliminate ground objects from lidar-received point clouds. This ensures that the detection and tracking of bird targets are not impeded by the presence of ground object data.

6. Discussion

Avian lidar is an effective means for monitoring bird activity in low-altitude airspace. However, the presence of ground object echoes seriously affects the detection and tracking performance of the avian lidar system. To effectively suppress these ground object echoes, it is necessary to simulate them accurately in the avian lidar system. In this paper, an improved ray-tracing method is proposed to simulate ground object echoes in airport scenes. Our proposed method utilizes bidirectional ray tracing and incorporates beam constraint and light-energy constraint techniques to efficiently filter out irrelevant rays during the simulation process. Additionally, our method employs batch collision detection to enhance the collision efficiency between multiple random rays and ground objects within a beam. By optimizing specific strategies within the simulation process, our proposed method significantly improves the simulation efficiency without compromising the resolution of the simulated point cloud.

The advantage of the ray-tracing method [6–8] lies in its ability to accurately simulate the propagation and scattering process of light, as well as generate high-precision ground object echo signals. However, when dealing with complex scenes, the calculation load of ray tracing significantly increases due to multiple reflections of light, resulting in low simulation efficiency. The introduction of a beam constraint and a light-energy constraint in this method improves the selection of effective light and greatly reduces the tracking of unnecessary rays. Compared to the hardware acceleration method mentioned in the literature [14,15], our proposed method enables a complex scene simulation without relying on expensive GPU resources. A comparative analysis with the BRDF method [10] and Phong's method [11] reveals that our proposed method exhibits superior simulation efficiency for ground object echoes within environments featuring large surface element scales. In small-scale scene experiments, the simulated point cloud generated in this paper demonstrates a strong realistic effect, closely resembling actual lidar-collected point clouds both structurally and in terms of intensity distribution. Furthermore, when simulating an airport scene, the generated point cloud effectively describes the distribution characteristics of ground object surfaces toward radar; particularly noteworthy is its improved descriptive capability after increasing the number of vertical lidar channels.

The research presented in this paper successfully achieved the efficient simulation of target echoes in complex environments. However, due to the limited availability of experimental conditions, there is currently no actual lidar system for acquiring ground object echo data in large scenes. Therefore, it is not possible to directly compare the simulated point cloud data with real point cloud data collected from such scenes. Consequently, there remains a need for more robust methods to validate the rationality of the simulated ground object echo point cloud.

7. Conclusions

In this paper, an enhanced and efficient ray-tracing simulation method is presented for simulating ground object echoes in airport avian lidar. By optimizing the effectiveness of light screening and collision detection between light and targets, the proposed method can be applied to simulate target echoes in large and complex scenes. Simulation experiment results demonstrate that the optimized ray-tracing and collision detection methods presented in this paper exhibit superior overall performance compared to classical algorithms. Under the similar control of other parameters, the running time and simulation speed of the proposed target simulation method are notably improved when compared with other methods, particularly for scenarios involving a high number of surfaces. Furthermore, based on actual lidar-collected point cloud results, the simulated results from our model accurately reflect both the physical shape and echo intensity characteristics of objects within scanned scenes.

Author Contributions: Conceptualization, Z.S.; methodology, Z.S.; software, L.S. and J.H.; validation, Z.S. and J.H.; formal analysis, Z.S. and L.S.; investigation, L.S. and J.H.; resources, Z.S., Y.W. and P.G.; data curation, L.S., B.H., Y.W. and P.G.; writing—original draft preparation, Z.S. and L.S.; writing—review and editing, Z.S. and L.S.; supervision, Z.S.; project administration, Z.S.; funding acquisition, Z.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Tianjin Municipal Education Commission under Grant No. 2022KJ059, the Major Science and Technology Projects in Anhui Province under Grant No. 202103a13010006, and the Fundamental Research Funds for the Central Universities under Grant No. 3122017111.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

Conflicts of Interest: Author Peng Ge was employed by the company the 38th Research Institute of China Electronics Technology Group Corporation. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

GPU	Graphic Processing Unit
CUDA	Compute Unified Device Architecture
3-D	Three-dimensional
HSB	Hue–Saturation–Brightness
AABB	Axis-Aligned Bounding Box
OBB	Oriented Bounding Box
SBV	Sphere Bounding Volume
BVH	Bounding Volume Hierarchy
BSP	Binary Space Partitioning
k-D	k-dimensional
2-D	Two-dimensional

References

- Zhao, J.; Li, Y.; Zhu, B.; Deng, W.; Sun, B. Method and Applications of Lidar Modeling for Virtual Testing of Intelligent Vehicles. *IEEE Trans. Intell. Transp. Syst.* **2021**, *22*, 2990–3000. [[CrossRef](#)]
- Rougeron, G.; Garrec, J.L.; Andriot, C. Optimal positioning of terrestrial LiDAR scanner stations in complex 3D environments with a multiobjective optimization method based on GPU simulations. *ISPRS J. Photogramm. Remote Sens.* **2022**, *193*, 60–76. [[CrossRef](#)]
- Zhang, W.; Li, Z.; Li, G.; Zhuang, P.; Hou, G.; Zhang, Q.; Li, C. GACNet: Generate Adversarial-Driven Cross-Aware Network for Hyperspectral Wheat Variety Identification. *IEEE Trans. Geosci. Remote Sens.* **2023**, *62*, 5503314. [[CrossRef](#)]
- Zhang, W.; Li, Z.; Sun, H.H.; Zhang, Q.; Zhuang, P.; Li, C. SSTNet: Spatial, spectral, and texture aware attention network using hyperspectral image for corn variety identification. *IEEE Geosci. Remote Sens. Lett.* **2022**, *19*, 5. [[CrossRef](#)]
- Zhang, W.; Zhao, W.; Li, J.; Zhuang, P.; Sun, H.; Xu, Y.; Li, C. CVANet: Cascaded visual attention network for single image super-resolution. *Neural Netw.* **2024**, *170*, 622–634. [[CrossRef](#)] [[PubMed](#)]
- Yang, X.; Wang, Y.; Yin, T.; Wang, C.; Lauret, N.; Regaieg, O.; Xi, X.; Gastellu-Etchegorry, J.P. Comprehensive LiDAR simulation with efficient physically-based DART-Lux model (I): Theory, novelty, and consistency validation. *Remote Sens. Environ.* **2022**, *272*, 112952. [[CrossRef](#)]
- Gastellu-Etchegorry, J.P.; Yin, T.G.; Lauret, N.; Grau, E.; Rubio, J.; Cook, B.D.; Morton, D.C.; Sun, G.Q. Simulation of satellite, airborne and terrestrial LiDAR with DART (I): Waveform simulation with quasi-Monte Carlo ray tracing. *Remote Sens. Environ.* **2016**, *184*, 418–435. [[CrossRef](#)]
- Esmorís, A.M.; Yermo, M.; Weiser, H.; Winiwarter, L.; Höfle, B.; Rivera, F.F. Virtual LiDAR Simulation as a High Performance Computing Challenge: Toward HPC HELIOS++. *IEEE Access* **2022**, *10*, 105052–105073. [[CrossRef](#)]
- Linnhoff, C.; Rosenberger, P.; Winner, H. Refining Object-Based Lidar Sensor Modeling—Challenging Ray Tracing as the Magic Bullet. *IEEE Sens. J.* **2021**, *21*, 24238–24245. [[CrossRef](#)]
- Chai, G.; Zhang, J.; Huang, X.; Guo, B.; Tian, L. Study of the dynamic scenes model for laser imaging radar simulation. *Xi'an Dianzi Keji Daxue Xuebao/J. Xidian Univ.* **2014**, *41*, 107–113. [[CrossRef](#)]

11. Yang, J.S.; Li, T.J. Simulation of Space-Based Space Target Scene Imaging. *Laser Optoelectron. Prog.* **2022**, *59*, 253–260.
12. Neumann, T.; Kallage, F. Simulation of a Direct Time-of-Flight LiDAR-System. *IEEE Sens. J.* **2023**, *23*, 14245–14252. [[CrossRef](#)]
13. Winiwarter, L.; Esmoris Pena, A.M.; Weiser, H.; Anders, K.; Martínez Sánchez, J.; Searle, M.; Höfle, B. Virtual laser scanning with HELIOS++: A novel take on ray tracing-based simulation of topographic full-waveform 3D laser scanning. *Remote Sens. Environ.* **2022**, *269*, 112772. [[CrossRef](#)]
14. López, A.; Ogayar, C.J.; Jurado, J.M.; Feito, F.R. A GPU-Accelerated Framework for Simulating LiDAR Scanning. *IEEE Trans. Geosci. Remote Sens.* **2022**, *60*, 3000518. [[CrossRef](#)]
15. Richa, J.P.; Deschaut, J.-E.; Goulette, F.; Dalmaso, N. AdaSplats: Adaptive Splatting of Point Clouds for Accurate 3D Modeling and Real-Time High-Fidelity LiDAR Simulation. *Remote Sens.* **2022**, *14*, 6262. [[CrossRef](#)]
16. Tan, K.; Cheng, X. Specular reflection effects elimination in terrestrial laser scanning intensity data using Phong model. *Remote Sens.* **2017**, *9*, 8. [[CrossRef](#)]
17. Matusik, W.; Pfister, H.; Brand, M.; McMillan, L. A Data-Driven Reflectance Model. *ACM Trans. Graph. (TOG)* **2003**, *22*, 759–769. [[CrossRef](#)]
18. Ceolato, R.; Berg, M.J. Aerosol light extinction and backscattering: A review with a lidar perspective. *J. Quant. Spectrosc. Radiat. Transf.* **2021**, *262*, 107492. [[CrossRef](#)]
19. Liang, B.; Niu, J.; He, S.; Liu, H.; Qin, C. Tunnel lighting calculation model based on bidirectional reflectance distribution function: Considering the dynamic changes in light transmittance in road tunnels. *Tunn. Undergr. Space Technol.* **2023**, *140*, 105313. [[CrossRef](#)]
20. Xing, Y.S.; Liu, X.P.; Xu, S.P. Efficient collision detection based on AABB trees and sort algorithm. In Proceedings of the 2010 8th IEEE International Conference on Control and Automation (ICCA 2010), Xiamen, China, 9–11 June 2010; pp. 328–332. [[CrossRef](#)]
21. Zhao, Y.; Huang, J.; Li, W. Fitting oriented bounding box algorithm for accurate positioning of transformer wiring terminals. In Proceedings of the 2023 35th Chinese Control and Decision Conference (CCDC), Yichang, China, 20–22 May 2023; pp. 4058–4061. [[CrossRef](#)]
22. Huang, X.; Zhang, S.; Zhou, L.; Huang, L. A hybrid algorithm for the minimum bounding sphere problem. *Oper. Res. Lett.* **2022**, *50*, 150–154. [[CrossRef](#)]
23. Gu, Y.; He, Y.; Fatahalian, K.; Blemloch, G. Efficient BVH construction via approximate agglomerative clustering. In Proceedings of the—High-Performance Graphics 2013, HPG 2013, Anaheim, CA, USA, 19–21 July 2013; pp. 81–88. [[CrossRef](#)]
24. Ize, T.; Wald, I.; Parker, S.G. Ray tracing with the BSP tree. In Proceedings of the RT'08-IEEE/EG Symposium on Interactive Ray Tracing 2008, Proceedings, Los Angeles, CA, USA, 9–10 August 2008; pp. 159–166. [[CrossRef](#)]
25. Shan, Y.X.; Li, S.; Li, F.X.; Cui, Y.X.; Li, S.; Zhou, M.; Li, X. A Density Peaks Clustering Algorithm With Sparse Search and K-d Tree. *IEEE Access* **2022**, *10*, 74883–74901. [[CrossRef](#)]
26. Liu, K.; Ma, H.; Zhang, L.; Cai, Z.; Ma, H. Strip Adjustment of Airborne LiDAR Data in Urban Scenes Using Planar Features by the Minimum Hausdorff Distance. *Sensors* **2019**, *19*, 5131. [[CrossRef](#)]
27. Chen, J.H.; Zhao, D.; Zheng, Z.J.; Xu, C.; Pang, Y.; Zeng, Y. A clustering-based automatic registration of UAV and terrestrial LiDAR forest point clouds. *Comput. Electron. Agric.* **2024**, *217*, 108648. [[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.