Cam-PC: A Novel Method for Camouflaging Point Clouds to Counter Adversarial Deception in Remote Sensing

Bo Wei, Teng Huang, Xi Zhang, Jiaming Liang, Yunhao Li, Cong Cao, Dan Li, Yongfeng Chen, Huagang Xiong, Feng Jiang, and Xiqiu Zhang

***Abstract*—Synthetic aperture LiDAR can generate point cloud data, which is widely used in 3-D scene reconstruction. However, existing point cloud object recognition methods are vulnerable to adversarial attacks, and such attacks are difficult to transfer to the physical world. Even if adversarial perturbations are added to physical objects, they are easily detectable by other sensors. Our proposed method includes two modules, R-D and D-R, which generate more concealed adversarial point cloud samples by modifying digital and physical features. The R-D module maps real-world entities to point cloud data in the digital world and generates adversarial samples by modifying signal amplitude values. The D-R module constructs adversarial objects by modifying the surface diffuse reflectance of the target object based on ray tracing and correspondences between digital and physical features. Our method is evaluated through experiments on attack effectiveness, robustness after subsampling and transferability, demonstrating its effectiveness, and achieving new state-of-the-art performance.**

***Index Terms*—Adversarial attack, physical simulation, point cloud, remote sensing image.**

I. Introduction

A POINT cloud is a collection of vectors that store a series of point information about an object's surface in the 3-D coordinate system of the world. This point information captures the object's 3-D key information and avoids the continuous combination of irregular and complex images on the object's surface [1]. Different sensors capture various key information, such as color (RGB) and reflection intensity information (Intensity). Compared to the regular format of 3-D data, point cloud data is less affected by lighting and image quality, and it contains less redundant data, thus significantly reducing computing and storage costs. Point cloud data is widely used in various fields, including 3-D matching [2], [3], multiview 3-D reconstruction [4], [5], object detection and recognition [6], [7], [8], semantic segmentation [9], [10], [11], graph tasks [12], [13], etc. In the field of remote sensing, synthetic aperture radar generates the point cloud data of a target by utilizing interferometric measurement technology to image the same area of the target object twice and forming a certain geometric relationship between the interferograms and the height, wavelength, and beam direction parameters of the sensor [14], [15]. Consequently, point cloud-based object recognition methods based on deep learning have become an essential perception tool in the field of remote sensing. Recent studies [16], [17] have found that existing 3-D point cloud recognition algorithms are vulnerable to adversarial attacks, where attackers use a set of artificially generated point cloud data that is difficult for humans to distinguish to mislead deep learning network models for point cloud recognition, resulting in significantly reduced recognition performance. Currently, point cloud adversarial samples are mainly generated by moving the spatial coordinates of points or adding new points [18]. For example, Liu et al. [19] borrowed the idea of 2-D adversarial attacks and proposed a 3-D adversarial attack method. This method perturbs the spatial coordinates of points instead of pixel values and evaluates the perturbations under the infinity norm and the L2 norm. In contrast, Xiang et al. [20] generated adversarial samples by adding independent adversarial points or point clusters. These attack methods are mainly based on the digital world and assume that attackers can not only move some point data in the point cloud data coordinates, but also accurately add point data at a certain coordinate. They are not easily transferable to physical world attack testing [21]. For systems that operate in the physical world, such as mapping physical entities to the digital world and constructing corresponding adversarial samples, and then capturing them through sensors to mislead the model's recognition and decision-making [22], is a very challenging problem.

The substantial differences between remote sensing [23] and autonomous driving point clouds [24] necessitate

Manuscript received 21 April 2023; revised 2 August 2023 and 5 September 2023; accepted 8 October 2023. Date of publication 13 October 2023; date of current version 23 November 2023. This work was supported in part by the National Natural Science Foundation of China under Grant 62002074. *(Bo Wei and Teng Huang contributed equally to this work.) (Corresponding authors: Xiqiu Zhang; Xi Zhang.)*

Bo Wei and Huagang Xiong are with the School of Electronic and Information Engineering, Beihang University, Beijing 100191, China (e-mail: weibo@buaa.edu.cn; hgxiong@buaa.edu.cn).

Teng Huang, Jiaming Liang, Yunhao Li, Cong Cao, and Dan Li are with the Institute of Artificial Intelligence and Blockchain, Guangzhou University, Guangzhou 511370, China (e-mail: huangteng1220@e.gzhu.edu.cn; jiaming.liang@e.gzhu.edu.cn; yunhaoli@e.gzhu.edu.cn; congcao@e.gzhu.edu.cn; danli@e.gzhu.edu.cn).

Xi Zhang and Xiqiu Zhang are with the School of Art, Sun Yat-sen University, Guangzhou 510275, China (e-mail: xizhangpiano@gmail.com; zhangxq77@mail.sysu.edu.cn).

Yongfeng Chen is with the Institute of Artificial Intelligence, University of Science and Technology Beijing, Beijing 100083, China (e-mail: m202211483@xs.ustb.edu.cn).

Feng Jiang is with the Department of Computer Sciences, Metropolitan State University of Denver, Denver, CO 80204 USA (e-mail: fjiang@msudenver.edu).

Digital Object Identifier 10.1109/JSTARS.2023.3324483

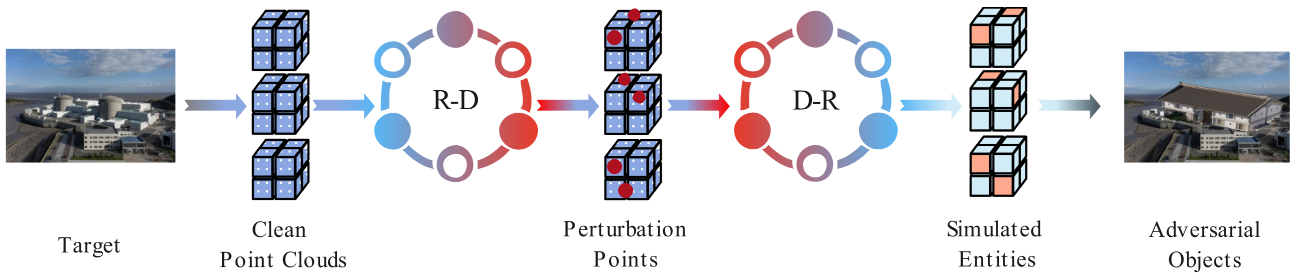

Fig. 1. Architecture of the Cam-PC. The Cam-PC consists of two basic modules, R-D and D-R, to bridge the real and digital worlds. The details of both modules are zoomed in Figs. 2 and 3.

tailored adversarial attack methods for each domain. Remote sensing point cloud data necessitates human intervention and fusion with images to reconstruct geospatial information three-dimensionally [25]. The quality of this data critically influences the structural reconstruction of target objects. Despite sharing preprocessing steps with autonomous driving point clouds, like positional correction, remote sensing point clouds uniquely require manual control points, or image-referenced positional adjustments. These distinctive attributes pose challenges for directly employing certain adversarial attack methods—such as altering point cloud positions—developed for autonomous driving scenarios [26]. Recognizing these complexities, our proposed Cam-PC method opts for a subtler strategy of modifying point cloud data's reflectance intensity to create adversarial perturbations, thus effectively addressing the intricacies associated with remote sensing data.

In response to the aforementioned issues, we propose a novel adversarial point cloud attack framework. Unlike the aforementioned methods that directly interfere with the coordinate information of point cloud data by adding perturbation points or modifying 3-D coordinates, the proposed method does not directly modify the coordinate information of point cloud data. Instead, it modifies specific additional attribute information and attempts to associate the modified data with a simulated physical environment for attack testing. As illustrated in Fig. 1, the proposed method consists of two important components: 1) the R-D module; and 2) the D-R module, which aim to establish a relationship between the digital world and the physical world. The R-D module is designed to convert physical world entities into point cloud data in the digital world. The key design of this module is to generate corresponding perturbation points by modifying the signal amplitude of the data points while keeping their 3-D position coordinates unchanged, based on the acquired point cloud data from the physical world, i.e., to generate point cloud data adversarial samples for the corresponding entities. The D-R module is designed to convert adversarial point cloud data in the digital world into target physical entities in the physical world. The key design of this module is to use the ray tracing principle to find the correspondence between the data points in the digital world and the target physical object in the real world, i.e., to construct the adversarial samples of point cloud data migrated to the simulated physical environment.

The main innovations of this article are summarized as follows.

1) Our proposed method, Cam-PC, bridges the real and digital worlds by enabling the generation of point cloud adversarial samples that can be used to attack both LiDAR and remote sensing radar.
2) In the digital world, point cloud adversarial samples are commonly generated by modifying the point's 3-D position coordinates or adding perturbation points. Thus, we propose a new module, R-D, to generate adversarial samples by altering the signal amplitude values of point cloud data.
3) In the real world, generating adversarial point cloud examples often involves modifying the target physical entity or sensor directly or injecting false information into the real point cloud signal, which is complicated and uncontrollable. Thus, we propose a new module, D-R, to generate adversarial objects by modifying the surface diffuse reflectance of the target object.
4) We conduct a series of experiments to evaluate the performance of our proposed Cam-PC algorithm. The results demonstrate that Cam-PC achieves new state-of-the-art performance in terms of attack effectiveness, robustness after subsampling, and transferability.

II. Related Works

A. Object Detection for Point Cloud

Deep learning-based point cloud object recognition methods refer to the use of multilayer neural networks to automatically extract object feature information, thus improving recognition performance. Currently, deep learning-based point cloud object recognition methods can be divided into three categories [27], [28]: 1) projection-based methods; 2) voxel-based methods; and 3) point-based methods. Projection-based methods usually use projection to obtain multiview data of objects in 2-D space, and then reproject the features to 3-D space for aggregation to achieve point cloud object recognition, such as [29] and [30]. However, the projection of point cloud data onto 2-D images will lose the feature information of objects in other spaces, making it difficult to further improve recognition performance. In order to fully retain the shape information of 3-D data, voxel-based methods divide the point cloud space into multiple regular subspaces called voxels and classify objects in these subspaces. For example, the Voxnet model [31] generates a volumetric occupancy grid in the voxel space using sliding boxes. In the voxel space, a 3-D convolutional neural network is used to extract the recognition features of the object. Voxelnet [32] divides the point cloud space into equally spaced voxels, and then uses multiple voxel feature encoding layers [Voxel Feature Encoding (VFE)] to extract features for each nonempty voxel space. Finally, voxel features are aggregated through 3-D convolutional layers to obtain a high-dimensional feature representation of the object. However, higher resolutions will increase computation and processing time, leading to poor real-time performance, while lower resolutions will bring quantization errors and limit recognition performance. Compared to projection-based and voxel-based methods, point-based methods design a series of models with stronger feature expression capabilities and directly learn features on point cloud data, achieving end-to-end object recognition. For example, PointNet [33] integrates models to achieve spatial transformation and feature alignment on point cloud data, and then uses maximum pooling symmetric functions to fuse the most significant features in local features. However, the PointNet model only learns the spatial encoding of isolated points and cannot effectively capture local structural information of point clouds. PointNet++ [34] is an improved model that can achieve spatial transformation invariance of point cloud data, and has achieved good results in performance and accuracy. Currently, point-based methods have become a mainstream point cloud object recognition method, and this article will use two classic models, PointNet and PointNet++, to conduct experiments. Among them, the PointNet model will be used to evaluate the adversarial point cloud generation algorithm, and the PointNet++ model will be used to test the transferability of adversarial samples.

B. Adversarial attacks for Point Cloud

Adversarial attacks on point cloud data refer to the generation of adversarial samples that add false signals to real data in order to disrupt model detection results. In physical attacks, adversarial point cloud samples are generated by perceiving adversarial physical objects or injecting false point clouds [35], mainly targeting LiDARs. Adversarial physical objects are generated by attackers based on digital point cloud samples using 3-D printing technology, and can effectively evade detection by LiDARs. Adversarial object generation methods such as [36] and [37] have achieved similar results. However, the attack method of physical objects requires placing 3-D printed adversarial objects in the scene, which is easily noticed and can lead to attack failure. Adversarial point cloud samples are generated by attackers intercepting and tampering with part of the laser signal in the process of scattering back to the radar based on the ranging principle of LiDARs, i.e., injecting false node generation. Attack methods such as [38], [39], and [40] can cause the vehicle control system to execute emergency braking instructions when driving, or freeze in place when encountering traffic lights. Black-box attack methods such as [41] take advantage of the blind spots of LiDARs when sensing vehicles that are partially obstructed, reducing the number of injected false point cloud signals and attacking multiple detection systems. However, this method requires the use of high-precision equipment to capture and emit LiDAR signals in real-time, which is difficult to operate in complex real-world environments.

III. Methods

We introduce a novel method for generating adversarial examples tailored to remote sensing point clouds, as depicted in Fig. 1. This approach seamlessly generates adversarial entities in the physical domain through the utilization of two crucial modules, R-D and D-R, without necessitating the addition of perturbation points or alterations to 3-D coordinates. The R-D module enables the transformation from physical entities to digital point cloud data by leveraging gradient-based techniques and a binary search algorithm for adaptive balancing factor adjustment, thereby producing perturbations for target samples in accordance with their distinctive features. Conversely, the D-R module facilitates the conversion of digital adversarial point cloud data into physical domain target entities by employing ray tracing principles to ascertain correspondences between data points in the digital sphere and real-world target objects, thus providing a foundation for migrating digitally crafted adversarial perturbations into the physical environment. All the notations related to this article are summarized in Table I.

A. R-D Module: From Real World to Digital

Remote sensing-based point cloud data represents the physical world as sensed by radar sensors through electromagnetic waves. Each point within the point cloud corresponds to an echo signal received by the sensor, encompassing both the signal's phase center coordinates and amplitude. In the remote sensing domain, point cloud data containing n points is denoted as $X \subset R^{n\times 4}$, with each point $x \in R^4$ represented as a vector $[x_s, r_s, s_s, A_s]$. Here, x_s, r_s, s_s signify the signal's azimuth, range, and elevation, respectively, while A_s indicates the signal amplitude, reflecting the imaging target's characteristics. Leveraging this property, we propose a method for generating adversarial samples by altering the point cloud signal amplitude. This approach introduces a meticulously designed perturbation $p \in R^{n\times 1}$ to the fourth dimension of the pristine point cloud $x \in X \subset R^{n\times 4}$, resulting in the adversarial sample x', denoted as $x' \leftarrow x \oplus p$.

As illustrated in Fig. 2, the process of creating digital-world adversarial point clouds involves three key steps: 1) collecting clean point clouds from the target using interferometric measurement techniques; 2) applying gradient-based optimization in conjunction with binary search algorithms for adaptive adjustment of the balancing factor λ, iteratively deriving minute perturbations for the signal amplitude of each point in the point cloud data; 3) employing threshold filtering operations to retain only significant perturbation points, preserving the adversarial efficacy of the point cloud attack while reducing the number of perturbation points, thus establishing a foundation for generating adversarial samples in physical environments.

1) Adversarial Perturbation: In the adversarial sample generation process, we adopt the Euclidean distance as a metric to discretely control and generate subtle, yet effective perturbations. The perturbation magnitude can be denoted as $\|p\|_2 = \sqrt{\sum_{i=1}^{n} p_i^2}$. The objective functions for targeted and untargeted attacks are formulated as follows:

TABLE I
GENERAL NOTATIONS

Notations	Description	Notations	Description
X	Point cloud data	x	Data point
x_s,r_s,s_s	Signal's azimuth, range, and elevation	A_s	Signal amplitude
p	Perturbation	$\|p\|_2$	Perturbation magnitude
$x^{'}$	Adversarial samples	$t^{'}$	Target label
t	True label	$F(\bullet)$	Trained classifier
λ	Balancing factor	$L_{adv}(\bullet)$	Adversarial loss
$Z(x)_i$	Classifier output	D_{opt}	Metrics against interference
N	Number of iterations	eps	Number of binary searches
$\#1$,A	Intersection point 1 on object A	F_d	Diffuse reflection coefficient
I_{sig}	Intensity of the incident signal	$\vec{N}$	Surface normal vector
$\vec{L}$	Normalized vector	F_b	Surface brightness factor
$\vec{H}$	Halfway vector	F_r	Roughness factor
I_d	Diffuse signal amplitude	I_s	Signal amplitude of specular reflection
b_s	Signal reflection type	f_s	Flag for marking specular reflections
X_i,Y_i,Z_i	Intersection coordinates	ξ	Perturbation threshold

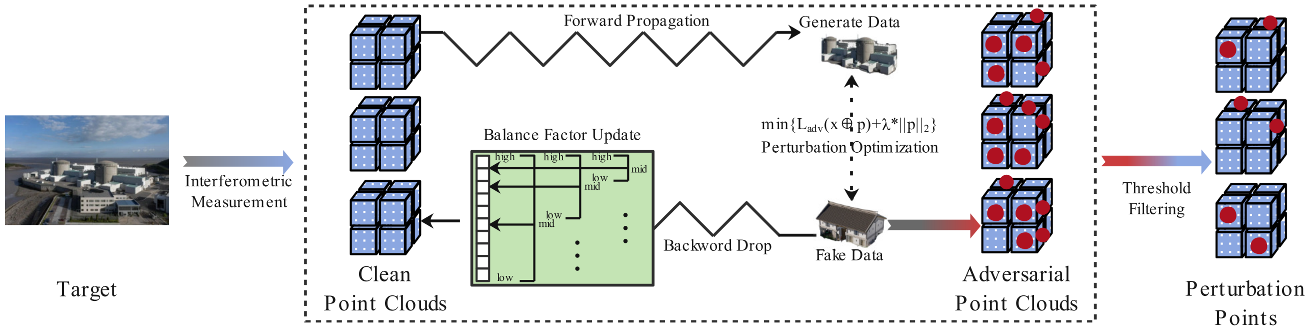

Fig. 2. Overview of R-D module.

$$\min \|p\|_2, s.t. F(x \oplus p) = t' \tag{1}$$

$$\min \|p\|_2, s.t. F(x \oplus p) \neq t \tag{2}$$

where t' denote the target label for targeted attacks, t represent the true label of the clean sample, and $F(\bullet)$ signify the trained classifier. The optimization problems for these two objective functions are challenging to solve directly. Therefore, we transform them into gradient-based optimization algorithms

$$\min \{L_{adv}(x \oplus p) + \lambda * \|p\|_2\} \tag{3}$$

In terms of the target attacks, $L_{adv}(x') = \text{Max}\{\text{Max}_{i \neq t'}\{\mathbb{Z}(x')_i\} - \mathbb{Z}(x')_{t'}, 0\}$. For the nontargeted attacks, $L_{adv}(x') = \text{Max}\{\mathbb{Z}(x')_t - \text{Max}_{i \neq t}\{\mathbb{Z}(x')_i\}, 0\}$. In both cases, the $\mathbb{Z}(x)_i$ denotes the ith element of the logit output produced by the classifier.

2) Balance Factor Update: The selection of the balance factor λ in the R-D module is crucial as it affects both the success rate of the attack and the magnitude of the added perturbation. To generate a minimal perturbation with high attack effectiveness, we propose an adaptive binary search algorithm to adjust the balance factor in each iteration. The algorithm initializes the balance factor λ and its upper and lower bounds (λ_{upper} and λ_{lower}), and sets the number of search iterations to eps. In each iteration, the algorithm adjusts the balance factor based on the size of the perturbation generated by the optimization process. If the current balance factor results in a smaller perturbation, the lower bound of the balance factor is updated to the current value λ, and the average of the upper and lower bounds is used as the next balance factor. If the current balance factor does not result in a smaller perturbation, the upper bound of the balance factor is updated to the current value λ, and the average of the upper and lower bounds is used as the next balance factor. The number of iterations is determined based on the attack results, and it will automatically stop once successful. The purpose of doing this is to ensure that the added perturbation is minimal.

Algorithm 1: Generate Adversarial Point Cloud Samples in Remote Sensing Applications.

```
Input  : Clean Samples $x$, Model $f(\cdot, n, \theta)$, Balance factor
         $\lambda$, Iteration $N$, Binary search times $eps$
Output: Tiny adversarial point cloud for remote sensing
         imaging $x'_{opt} = 0$
1  Initialization: perturbation $p$
2  for $epoch$ in $\{1,...,eps\}$ do
3      Generate Perturbation $p$
4      for $i$ in $\{1,...,N\}$ do
5          Generate Fake Data $x'$
6          Attack the Target Model $f(\cdot, n, \theta)$
7          Optimize Perturbation $p$
8      end
9      if $x'_{best}$ is not None and $D_{dest} < D_p$ then
10         $x'_{opt} \leftarrow x'_{best}$, $D_p \leftarrow D_{best}$
11         $\lambda \leftarrow (\max(\lambda_{lower},\lambda) + \lambda_{upper})$ / 2
12     else
13         $\lambda \leftarrow (\lambda_{lower}+\min(\lambda_{upper}, \lambda))$ / 2
14     end
15 end
16 Return $x'_{opt}$
```

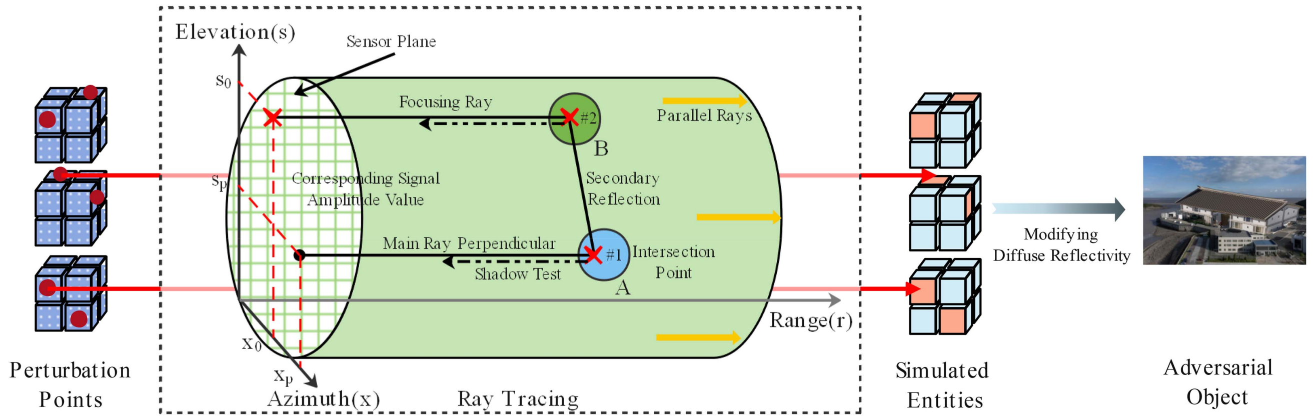

Fig. 3. Overview of D-R module.

The complete algorithm for generating adversarial point clouds for remote sensing imaging is presented in Algorithm 1.

B. *D-R Module: From Digital Back to Real World*

In order to convert our adversarial example generation to physical scenarios, we introduce the principle of ray tracing to construct adversarial objects by directly modifying the diffuse reflectance of the target object surface. The specific process is shown in Fig. 3.

Ray tracing is the process of tracking the path of rays in a scene to obtain multiple reflection signals of the rays interacting with objects in the scene, thus enabling the determination of the correspondence between each point in the point cloud data and the corresponding position on the object surface. Since tracing electromagnetic wave rays emitted by radar is still challenging, it is necessary to conduct tracing in a physical coordinate system. The details of the process are as follows.

Step 1: A physical coordinate system is established for the modeling of the scene. The 3-D scene where the target is located is created as shown in Fig. 3, where the x axis represents the azimuth, the s axis represents the elevation, and the r axis represents the distance. Second, an orthogonal projection camera is defined to simulate the radar sensor, which is located on a plane formed by the x and s axes. At the same time, the plane where the sensor is located is also the imaging plane of the signal. Finally, the sensor does not need to move to obtain information about the target in the azimuth direction. Instead, a static system is used to emit multiple parallel rays in the target direction, directly obtaining simulated radar data.

Step 2: Trace the scene objects. In the 3-D scene, two spherical objects were placed as shown in Fig. 3. The ray tracing process performs the following operations for each pixel on the sensor plane. First, a main ray perpendicular to the plane is created for the pixel center with azimuthal position x_p and height position s_p. Next, the ray intersects with object A along the path, producing intersection point #1. In this case, if there are multiple intersection points, the closest point to the ray origin is selected with the coordinates (x_p, r_1, s_p). Next, a shadow test is performed. A ray is cast from intersection object #1 towards the signal source. If the ray is interrupted by another object, the intersection point is in shadow; otherwise, it is illuminated. Finally, based on the reflection type, specular or diffuse, of the point, the corresponding signal amplitude value is calculated in (4) and (5) as

$$I_d = F_d \cdot I_{sig} \cdot (\vec{N} \cdot \vec{L})^{F_b} \tag{4}$$

$$I_s = F_s \cdot (\vec{N} \cdot \vec{H})^{\frac{1}{F_b}} \tag{5}$$

where F_d is the diffuse reflection coefficient, I_{sig} is the intensity of the incident signal, $\vec{N}$ is the surface normal vector, $\vec{L}$ is the normalized vector pointing to the signal source, and F_b is the surface brightness factor with a default value of 1. F_s is the specular reflection coefficient. In terms of SAR sensors, $\vec{H}$ is the halfway vector between the vector $\vec{L}$ pointing to the signal source and the surface normal vector $\vec{N}$. F_r is the roughness factor that defines the sharpness of the specular highlight.

If a secondary reflection occurs at object A, the reflection direction is obtained based on both the primary ray and the normal vector at point #1 on object A. Then, a new ray is shot in this direction and intersects with object B at point #2. Another shadow test is performed to determine if this point is illuminated. To obtain the coordinates of the secondary reflection, a focusing ray parallel to the primary ray is created starting from point #2. This ray intersects with the sensor plane at point (x_0, s_0), and the coordinates of the second reflection signal can be calculated in (6)

$$x_s = \frac{x_0 + x_p}{2} \quad r_s = \frac{r_1 + r_2 + r_3}{2} \quad s_s = \frac{s_0 + s_p}{2}. \tag{6}$$

The amplitude of the signal for the second reflection can be weighted according to the reflection coefficient of object A. To continue searching for the third reflection, a new secondary ray is defined at intersection point #2. If it intersects with object B, another focusing ray is used to search for the coordinates of the third reflection signal. This process is repeated until the set upper limit for the number of reflections is reached.

In addition, each point of the point cloud can approximately represent the information of a small rectangular area on the surface of the object, determined by the sensor's resolution and its relative position to the target object. Second, the surface's diffuse reflectivity is modified by using the mapping relationship between the signal amplitude of the point cloud data and the diffuse reflectivity of the object's surface, i.e., $Diffuse = f(Amplitude)$. Therefore, the diffuse reflectivity of the adversarial object's surface region corresponding to the sensitive point can be obtained by the signal amplitude. Furthermore, for objects with different materials, specific materials can be used to cover the surface to set the modified area's diffuse reflectivity to the desired value. Thus, an adversarial object can be obtained by modifying the surface's diffuse reflectivity.

IV. Materials

A. Simulation Tool

The RaySAR [42] simulation tool is utilized to construct various 3-D object models, such as buildings, and generate point cloud data for these objects using the included simulation radar sensors. Additionally, based on the world coordinate system of the constructed 3-D objects, the software can establish a correspondence between the 3-D objects and the point cloud data locations. When rendering the 3-D objects, the simulation tool can output multiple sets of data. Each set of data contains not only information related to the radar signal, but also the coordinates of the intersection point between the signal and the 3-D object. Each set of data contains nine elements, which can be represented as a vector

$$S_s = [x_s, r_s, s_s, A_s, b_s, f_s, X_i, Y_i, Z_i]\,. \tag{7}$$

When rendering the modeled 3-D object, the simulation tool can output multiple sets of data. Each set of data includes nine elements represented by a vector, comprising information related to the radar signal and the coordinate information of the intersection point between the signal and the corresponding 3-D object. The first six dimensions represent the information of the radar signal detected by the sensor. x_s, r_s, and s_s represent the azimuth, range, and elevation of the radar signal phase center, respectively. A_s indicates the amplitude of the detected signal, b_s represents the bounce level corresponding to the detected signal, and f_s indicates the flag for marking specular reflections. The last three dimensions, X_i, Y_i, and Z_i, represent the coordinates of the intersection point between the 3-D object corresponding to the radar signal in the world coordinate system.

B. Data and Classifier

The choice to utilize remote sensing point cloud data for experimental validation of our proposed approach was motivated by its primary function in geospatial information extraction and analysis. Applications such as terrain modeling and vegetation classification demand a high level of detail and precision in data representation, contrasted with autonomous driving data that primarily serves perception and decision-making tasks like obstacle detection and path planning. Remote sensing applications necessitate larger ground coverage and thus, entail the reconstruction of objects of interest–such as buildings and trees–at varying levels of detail. This level of detail is critical in comprehending the structure, functionality, and interrelation of these objects. Therefore, to emulate the inherent complexity and detail orientation of remote sensing data, we employed a simplified model of a building for our experimental verification. This strategic choice has enhanced the relevance and applicability of our findings in practical, real-world scenarios.

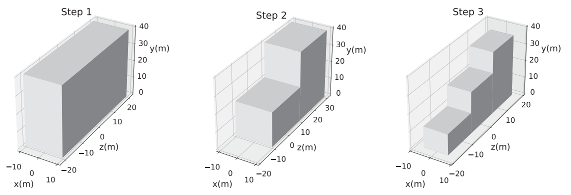

Fig. 4. Three steps of simulated entities.

TABLE II
Parameter Setting of Simulated Step Entities

True Label	Reflection	Ambient	Diffuse	Specular	Roughness
Model	0.5	0	0.3	0.3	0.005
Ground	0.5	0	–	0.1	–

In order to generate the experimental dataset, three types of simulated entities were first created using RaySAR. These entities consisted of rectangular prisms and were arranged in different quantities (1, 2, or 3). Specifically, one entity was a rectangular prism with dimensions of 40× 20× 40, denoted as "Step1;" two entities were constructed by combining a rectangular prism with dimensions of 20× 20× 20 and another with dimensions of 20× 20× 40, denoted as "Step2;" and three entities were constructed by combining rectangular prisms with dimensions of 12× 12× 12, 12× 12× 24, and 12× 12× 36, respectively, denoted as "Step3." These entities were positioned on a horizontal plane, as shown in Fig. 4. The reflection parameters, environmental conditions, and material information needed to model the entities and the ground are shown in Table II. These parameters affect the signal amplitude of the point cloud data collected by the sensor.

During rendering, the simulated radar sensor was placed on a circular path centered at the origin (0, 0, 0) with a radius of 60 and a 45° angle with respect to the horizontal plane. The sensor's position remained fixed while the models were rotated around the vertical axis passing through the origin, generating a new model every 1° of rotation. This process was repeated 360 times for each of the three simulated entities, resulting in 1 sample for each angle with respect to the sensor. Finally, RaySAR was used to render the models and collect data, and the first four dimensions of the RaySAR output were extracted to generate point cloud data samples.

The resulting point cloud data samples contained not only information on the radar signals reflected from the target objects, but also information on the signals reflected from the nontarget objects. It should be noted that when electromagnetic waves are reflected from the nontarget objects multiple times before reaching the radar target of interest, they carry less information of interest. To effectively generate adversarial samples, only the point cloud data with a single reflection was retained, and the remaining data was filtered out. After preprocessing, the resulting dataset consisted of 360×3 samples. These samples were used to train a PointNet classifier for the experiments.

For training the PointNet classifier for target recognition and attack, the dataset was divided into training and testing sets with a ratio of 80: 20, respectively. The PointNet classifier's sampling parameter was set to 512 with random and furthest distance sampling. The classifier was then trained, resulting in a training accuracy of 98.1% and a testing accuracy of 99.5%. In addition, each type of sample was sorted in increasing order according to the rotation angle, and every fourth sample was selected, resulting in a total of 90 samples used to generate adversarial samples (which accounts for 1/4 of the same type of dataset), with an average of 2564, 2188, and 1317 points for each category, respectively.

V. Results

In this section, the evaluation of the algorithm covers the attack performance of the generated adversarial samples, the robustness of the algorithm under different sampling conditions, and its transferability.

A. Attack Performance

To evaluate the effectiveness of the attack algorithm, the L2 norm was utilized as the metric for perturbation. A total of 90×3 adversarial samples were generated to attack the classifier under targeted and untargeted settings. The binary search algorithm was set to perform 10 iterations, while the optimization process was set to run for 200 iterations.

In remote sensing point cloud applications, adversarial examples refer to intentionally designed small perturbations that can mislead or deceive machine learning models based on point clouds, causing them to make incorrect predictions or classifications. These perturbations are often difficult to detect but are sufficient to generate significant errors in the model's output. Cam-PC generates adversarial samples for attacking 3-D point cloud recognition algorithms by modifying the diffuse reflectance of object surfaces, thereby altering the signal amplitude values of the point cloud data. In this process, the 3-D point cloud recognition algorithm misclassifies the target object as either an incorrect object or a specified object. For example, in practical scenario applications, specific materials are used to occlude surfaces at specific positions of a core building, changing the surface's diffuse reflectance and generating perturbed point clouds as adversarial samples. This manipulation causes the recognition algorithm to identify the core building as a common scene. The attacker's objective is to generate adversarial samples by modifying the diffuse reflectance of object surfaces, with the capability to induce erroneous outputs or targeted outputs from the attacked algorithm, corresponding to nontargeted attacks and targeted attacks, respectively. The attacker possesses knowledge of the target object's point cloud data, has sufficient computational power to compute sensitive points corresponding to the target point cloud, and has access to the target object to obtain the material with the desired diffuse reflectance. However, the attacker lacks knowledge of the internal information and parameters of the attacked model and instead employs training models to generate adversarial samples and carry out attacks against the model.

1) Targeted Attack: This section is to evaluate the performance of the algorithm in targeted attacks. The attack target for each category was set as the other two categories, and the experimental results were presented from the perspectives of attack success rate and perturbation size. The attack results are shown in Table III. From the perspective of attack success rate, some adversarial samples were able to successfully attack under various attack settings, indicating that the proposed adversarial point cloud generation algorithm is effective in targeted attack scenarios. Particularly, the adversarial samples with the true categories of Step 1 and Step 2 achieved higher attack success rates, while the attack success rate for the true category of Step 3 was relatively lower. This result indicates that the difficulty level of generating adversarial samples varies for different attack settings. From the perspective of perturbation size, considering the large number of points in each point cloud, we can infer that the perturbations of the generated adversarial samples were relatively small by observing the three perturbation metrics in the last three columns of the table. In remote sensing point cloud applications, adversarial examples are intentionally designed small perturbations that mislead point cloud-based machine learning models, causing erroneous predictions or classifications. These imperceptible perturbations can have a significant impact on the model's output. In this article, we generate adversarial examples by modifying the signal amplitudes of some point cloud data. In remote sensing point cloud applications, adversarial examples are intentionally designed small perturbations that mislead point cloud-based machine learning models, causing erroneous predictions or classifications. These imperceptible perturbations can have a significant impact on the model's output. In this article, we generate adversarial examples by modifying the signal amplitudes of some point cloud data.

2) Nontargeted Attack: This section aims to evaluate the performance of the proposed algorithm in the context of nontargeted attacks. Similar to targeted attacks, we present experimental results from two aspects: 1) attack success rate; and 2) perturbation size. Adversarial samples were generated for each class of samples and subjected to nontargeted attacks, and the attack results are shown in Table IV. As indicated by the data in the second column of Table IV, the attack success rate of both Step 1 and Step 2 adversarial samples reached 100%, while more than half of the Step 3 adversarial samples could be successfully attacked. This demonstrates the effectiveness of the proposed adversarial point cloud generation algorithm under the nontargeted attack scenario. Moreover, overall, the success rate of nontargeted attacks was higher than that of targeted attacks. It can be observed from the last three columns of Table IV that the perturbation size of the generated adversarial samples was relatively small, considering the large number of

TABLE III
EVALUATION OF SUCCESS RATE AND PERTURBATION MAGNITUDE FOR TARGETED ATTACKS

True Label	Target Label	Success Rate	Ave. L2 Dis.	Min. L2 Dis.	Max. L2 Dis.
Step 1	Step 2	100%	0.0884	0.0445	0.1320
	Step 3	56.7%	1.0704	0.4343	2.1525
Step 2	Step 1	95.6%	1.1994	0.2610	3.5925
	Step 3	91.1%	1.0041	0.2903	2.6219
Step 3	Step 1	30.0%	1.6359	0.3284	3.8484
	Step 2	55.6%	0.2462	0.0315	1.3758

TABLE IV
EVALUATION OF SUCCESS RATE AND PERTURBATION MAGNITUDE FOR NONTARGETED ATTACKS

True label	Success rate	Ave. L2 Dis.	Min. L2 Dis.	Max. L2 Dis.
Step 1	100%	0.0899	0.0465	0.1312
Step 2	100%	0.9704	0.3772	2.9731
Step 3	56.7%	0.2969	0.0303	2.5638

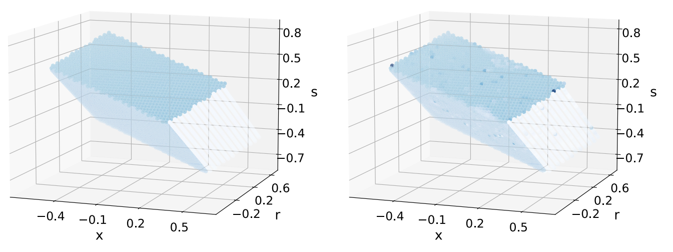

Fig. 5. Visualization of clean and adversarial samples for the attack. (Left) Clean sample. (Right) Adversarial sample.

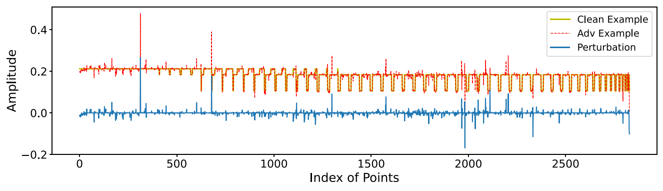

Fig. 6. Comparison of signal amplitude before and after the attack.

points in each point cloud. This further indicates that the generated perturbation by the proposed algorithm has good attack stealthiness.

3) Visualization: This section aims to provide a more intuitive understanding of the generated adversarial samples. We visualize the clean sample and the adversarial sample with Step 3 as the attack target, both of which have a rotation angle of 120° and contain 2686 points, as shown in Fig. 5 Left and Fig. 5 Right, respectively. The color depth of the points represents the magnitude of the signal amplitude. The distribution of data in the figures reveals the geometric features of the samples, while the color differences between the points in the two figures indicate the position and magnitude of the added perturbations. To accurately observe the changes in the signal amplitude before and after adding perturbations, we visualize the signal amplitude of the clean and adversarial samples in a line chart, as shown in Fig. 6. The yellow and red lines represent the signal amplitude of the clean sample and the adversarial sample, respectively, while the blue line represents the perturbations added to each point. Most of the points in the blue line have only small fluctuations, indicating that the generated perturbations are relatively small. When the perturbations represented by the blue line are added to the clean sample, the resulting adversarial sample represented by the red line almost overlaps with the clean sample represented by the yellow line, indicating that the generated adversarial sample is very close to the clean sample and is not easily detectable.

4) Robustness: In this section, the robustness of point cloud adversarial samples after sampling is evaluated using the furthest point sampling algorithm (FPS). The majority of current deep learning-based methods for point cloud recognition require preprocessing sampling of the data before classification. Thus, it is worth studying whether adversarial samples maintain their attack effectiveness after sampling. In the experiment, a sample with a true class of Step 3 and an angle of 60° was selected, and an untargeted adversarial sample was generated using Algorithm 1. Due to the binary search method used in the algorithm to search for the minimum perturbation factor, the generated untargeted adversarial sample had the minimum perturbation. However, despite the sample successfully misleading the classifier, the sample's confidence was low (49.83%). To avoid the influence of the adversarial sample's confidence on the sampling experiment's results, a high-confidence adversarial sample was also generated for the above clean sample to form a control experiment with the low-confidence adversarial sample. The high-confidence adversarial sample was generated by setting a stopping criterion for the algorithm, which stopped iterating when the algorithm produced an adversarial sample with attack effectiveness and confidence above a specific threshold (set to 90% in this article). The resulting adversarial sample had a confidence of 95.93%, and compared to the low-confidence adversarial sample, this sample had a larger perturbation.

This section presents the visualizations of the changes in the adversarial samples before and after sampling. We visualize the changes in the distribution of point cloud data of low-confidence adversarial samples with decreasing sampling rates, as shown in Fig. 7. We use PointNet to perform five identifications for both the adversarial and clean samples at different sampling rates, and obtain the number of successful attacks on the adversarial samples and the number of correct identifications on the clean samples at different sampling rates, as shown in Fig. 8. Furthermore, we obtain the average confidence score of the adversarial samples that were identified as incorrect classes and the average confidence score of the clean samples that were correctly identified, as shown in Fig. 9. To avoid uncertainty due to random sampling, we perform five identifications for each sample at the same sampling rate.

The green line in Fig. 8 shows that, despite the decreasing sampling rate, the classifier is still capable of correctly recognizing

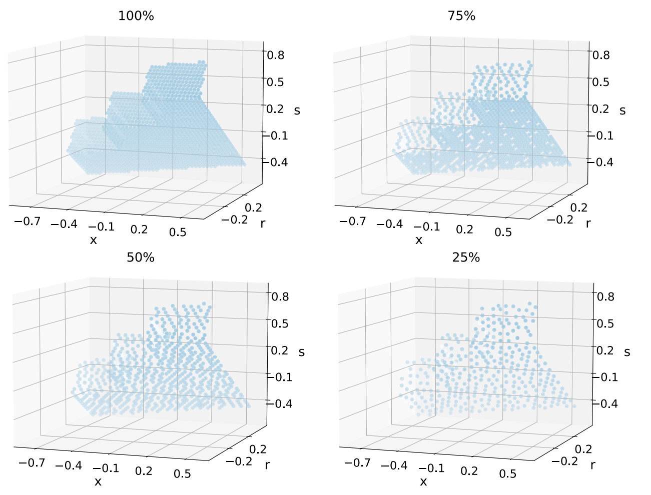

Fig. 7. Features variation of the target in point cloud data at different sampling rates. In this case, x, r, and s represent azimuth, distance, and elevation, respectively.

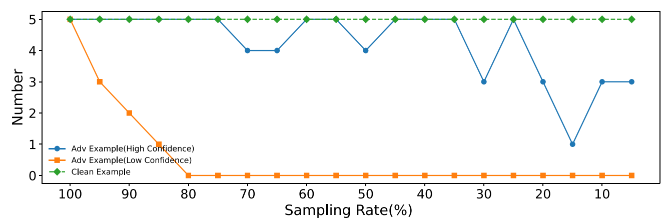

Fig. 8. Effect of sampling rates on adversarial attack success and clean sample recognition.

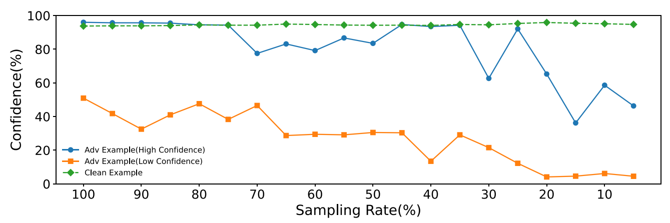

Fig. 9. Confidence levels of misidentified adversarial samples and correctly recognized clean samples under different sampling rates.

the sampled clean samples. Similarly, the confidence level of the sampled clean samples, shown in the green line of Fig. 9, remains largely unchanged. Strong robustness in identifying clean samples under sampling conditions is demonstrated. The orange line in Fig. 8 reveals that, as the sampling rate decreases below 80%, the number of successful low-confidence adversarial sample attacks gradually decreases, and such attacks cannot be successful when the sampling rate is below 80%. The blue line in Fig. 8 shows that high-confidence adversarial samples are still capable of successfully attacking the classifier under different sampling rates. The experimental results demonstrate that the attackability of adversarial samples is affected by sampling, and the lower the sampling rate, the weaker the attackability of adversarial samples. The robustness of adversarial sample attacks under sampling conditions must be considered by attackers.

TABLE V
Adversarial Sample Attack Results on PointNet++ Classifier Generated by the Adversarial Perturbation Method

True label	Target label	Predicted label			ASR
		Step 1	Step 2	Step 3	
Step 1	Step 2	97.78%	2.22%	0.00%	2.22%
	Step 3	71.11%	28.89%	0.00%	0.00%
	Untarget	95.56%	4.44%	0.00%	4.44%
Step 2	Step 1	8.89%	91.11%	0.00%	8.89%
	Step 3	7.78%	92.22%	0.00%	0.00%
	Untarget	6.67%	93.33%	0.00%	6.67%
Step 3	Step 1	0.00%	58.82%	41.18%	0.00%
	Step 2	0.00%	31.37%	68.63%	**31.37%**
	Untarget	0.00%	41.18%	58.82%	**41.18%**

Note: ASR means the success rate of attack.

B. Transferability

In this section, we verify the transferability of radar point cloud adversarial samples. In this experiment, 512 points were sampled from each targeted and nontargeted adversarial sample generated. Table V presents the results of attacking PointNet++ using the two types of sampled adversarial samples. The success rates of the first two attacks under each real label in the table indicate that most of the adversarial samples cannot induce PointNet++ to recognize them as the specified class. These findings suggest that adversarial samples generated based on PointNet are not effective for launching targeted attacks on PointNet++. Furthermore, the success rate of nontargeted attacks is only slightly higher.

In addition, the success rates of adversarial samples of different real classes show that the attack success rate of adversarial samples with real class Step 3 is significantly higher than the other two classes, implying that different types of adversarial samples have varying attack effects on PointNet++. However, overall, the success rate of adversarial samples generated based on PointNet attacking PointNet++ is low, indicating weak transferability of adversarial samples.

C. Physical Environment Attack Simulation

It can be seen from the blue line that the signal amplitude of each point in the clean sample is the same. From the green line, it can be seen that most of the added perturbations are relatively small, with only a few points having large perturbations.

In order to verify the feasibility of the proposed method for constructing adversarial objects in Section III-B, this experiment conducted nontargeted attacks on a selected sample. For targeted attacks, all steps were the same except for the generated adversarial point cloud in the digital world, and thus further elaboration is omitted. A sample was selected from the dataset for the experiment. Specifically, a clean sample with a real label of Step 1 and an angle of 0° was chosen. This sample contains 1775 points, as shown in Fig. 10 Left. The corresponding 3-D object has two features: 1) each face of the object is parallel to the coordinate plane, making it easy to determine the modification locations required to construct an adversarial object; 2) the rays emitted by the sensor only intersect with the upper and side faces of the object, and the angle between the sensor and these two faces is 45°. This ensures that the signal amplitude of each

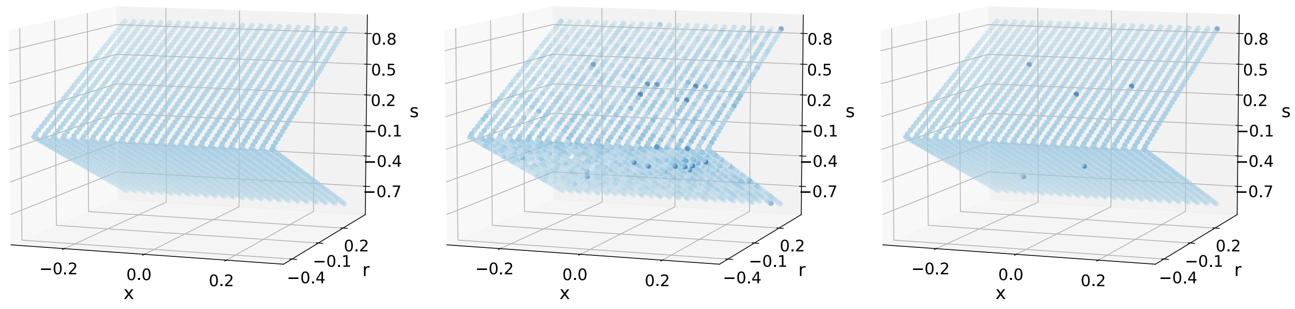

Fig. 10. Visualization of a 0° sample with a single step: (Left) Clean sample. (Middle) Adversarial sample (with full perturbation). (Right) Adversarial sample (with minimal perturbation).

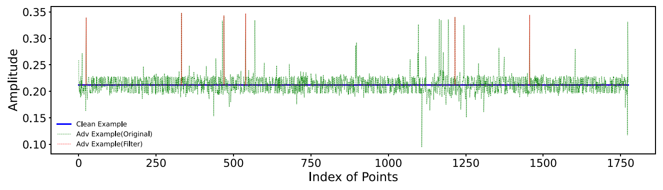

Fig. 11. Signal amplitude of the sample.

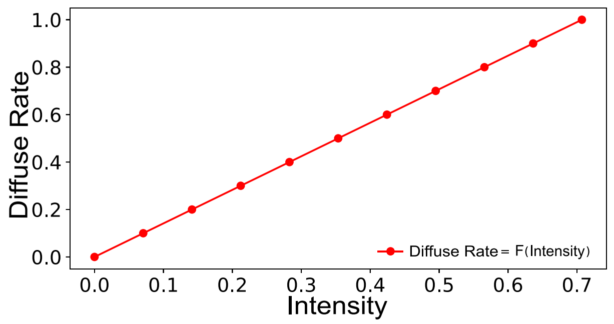

Fig. 12. Relationship regression line between diffuse reflectance and signal amplitude.

point received is the same, so the relationship between diffuse reflectance and signal amplitude only needs to be considered in one case. Next, an adversarial sample was generated from the selected sample and filtered. An adversarial sample with nontargeted attacks was generated from the clean sample, and this adversarial sample was recognized by the PointNet classifier as belonging to Step 2 with a confidence score of 97.81%, as shown in Fig. 10 Middle. In the filtering process, the threshold value of perturbation ξ was set to 0.125, and the adversarial sample was filtered accordingly. The filtered adversarial sample only retains perturbations of six sensitive points, as shown in Fig. 10 Right. This sample still maintains a strong attack effect, as it is recognized as belonging to Step 2 with a confidence score of 89.57%. It can be used to construct an adversarial object.

To accurately display the signal amplitude changes of the samples in Fig. 10, a line chart in Fig. 11 is used to visualize the signal amplitudes of the three samples. The blue, green, and red lines represent the signal amplitudes of the clean sample, the original adversarial sample, and the filtered adversarial sample, respectively.

From the red line, it can be seen that the filtered adversarial sample retains only a very few points with perturbations greater than the threshold value. Furthermore, the mapping relationship between diffuse reflectance and signal amplitude of the object surface was obtained. In RaySAR, different diffuse reflectance values (0.0–1.0) were set for the 3-D object model, and point cloud data was rendered to obtain the signal amplitude of each point. By fitting the data, the relationship curve between diffuse reflectance and signal amplitude was obtained, as shown in Fig. 12. It can be seen from the curve that there is a linear relationship between diffuse reflectance and signal amplitude, with $DiffuseRate = 1.41421369 \times Intensity$. Based on this mapping relationship, the diffuse reflectance corresponding to the signal amplitude of the 6 sensitive points in the adversarial sample can be calculated.

Finally, the position of the modified area was determined and its diffuse reflectance was changed to generate the adversarial object. The locations of the six sensitive points were obtained from the point cloud data based on the indices of the perturbation points in Fig. 11. Then, the surface coordinates of the sensitive points were obtained using RaySAR, as described in Section III-B. Each point on the object's surface represents a small area centered at that point. The upper and side surfaces of the 3-D object were divided into multiple regions, each centered at a point and with a size of 0.8 × 1.131371 based on the resolution of the simulated radar sensor plane and the relative position of the sensor to the target object in RaySAR. The diffuse reflectance of each region was modified according to the calculated diffuse reflectance of the corresponding sensitive point to obtain the adversarial object. Fig. 13 shows RaySAR rendering images of original 3-D objects and adversarial 3-D objects. The six bright spots in the right picture of Fig. 13 are generated by Algorithm 1, corresponding to the positions of the six points where the diffuse reflectance is modified.

To validate the effectiveness of the generated adversarial object, we first collected point cloud data of the 3-D adversarial object using the RaySAR software and visualized it, as shown in Fig. 14. Additionally, the signal amplitudes of each point

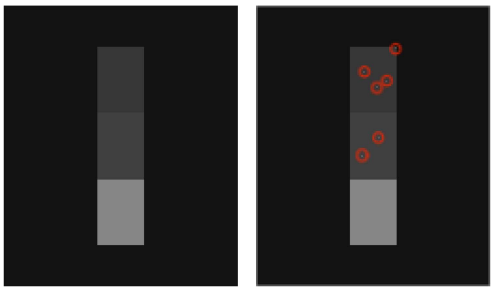

Fig. 13. Rendered 3-D object images generated by RaySAR software: (Left) Original 3-D object. (Right) Modified 3-D object.

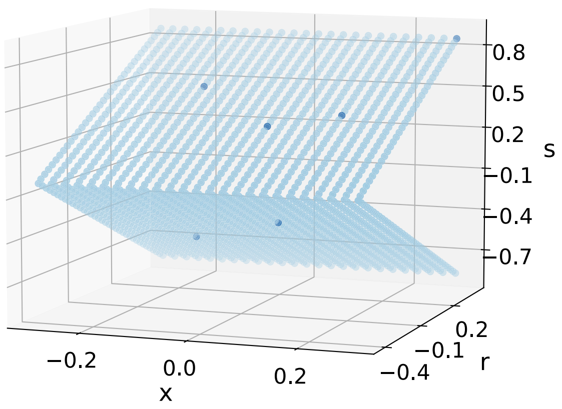

Fig. 14. Visualization of physical adversarial samples.

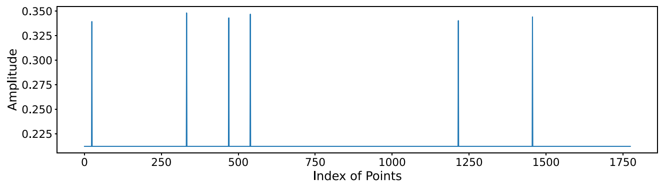

Fig. 15. Signal amplitude of physical adversarial samples.

were visualized, as shown in Fig. 15. Comparing the blue line in Fig. 15 and the red line in Fig. 11, the obtained point cloud data is almost identical to that of Fig. 10 (Right). Using the PointNet classifier to recognize the point cloud data, the classifier recognizes it as Step 2 with a confidence score of 89.57%, indicating that the adversarial object can successfully attack the classifier, demonstrating the feasibility of the proposed approach for constructing adversarial objects.

VI. Discussion

The discussion section focuses on the vulnerability of deep learning-based point cloud recognition methods to adversarial attacks and the proposed solution, Cam-PC, which bridges the real and digital worlds. Our proposed approach, Cam-PC, generates more concealed adversarial point cloud samples by modifying digital and physical features.

In the digital world, point cloud adversarial samples are commonly generated by modifying the 3-D position coordinates of the points or adding additional perturbation points. Our R-D module generates adversarial samples by modifying the signal amplitude values of the point cloud data, which offers better concealment and operability.

In the real world, generating adversarial point cloud examples often involves directly modifying the target physical entity or sensor, or injecting false information into the real point cloud signal. However, these methods are complicated and uncontrollable. To address this issue, our D-R module constructs adversarial objects by modifying the surface diffuse reflectance of the target object based on ray tracing and correspondences between digital and physical features. This approach provides good concealment and operability and can be used to extend the physical attack objects of point cloud data from the LiDAR domain to the remote sensing radar domain.

We conducted a series of experiments to evaluate the performance of our proposed Cam-PC algorithm. Our method achieved new state-of-the-art performance in terms of attack effectiveness, robustness after subsampling, and transferability. The experimental results demonstrate that the adversarial samples generated using the modified signal amplitude approach have attack effectiveness and, under certain sampling conditions, exhibit a certain degree of transferability. Moreover, we demonstrated the feasibility of creating physical adversarial samples using RaySAR software, which effectively deceived deep learning-based target recognition algorithms.

In future work, we plan to investigate the use of our method for other applications, such as autonomous driving and robotics. Additionally, we will explore the use of other physical features to generate adversarial samples and investigate the effectiveness of these features in the physical world. Another direction of future research is to investigate the robustness of our method against advanced attacks, such as black-box attacks. We believe that our proposed method can be extended to address other challenges in the field of deep learning-based point cloud recognition and has the potential to contribute to the development of more robust and reliable point cloud recognition systems.

VII. Conclusion

In conclusion, this study marks a significant advancement in the exploitation and augmentation of LiDAR point cloud data, introducing a novel Cam-PC method adept at addressing the intricate nature of artificial object rendering in 3-D reconstruction. A distinctive element of the Cam-PC method is its employment of a covert strategy designed to mislead deep learning algorithms, utilizing simplified models to authentically mimic real-world structures. This effectiveness positions the method as a valuable resource in disrupting LiDAR remote sensing target analysis, and thereby serves as a pivotal reference for enhancing LiDAR security defense protocols.

As the field of LiDAR data analysis broadens, the proposed Cam-PC method presents a unique value in informing and improving adversarial attack and defense strategies, particularly relevant to the increasingly critical sector of autonomous driving point cloud data. Furthermore, our approach proves the potential of bridging the real and digital worlds, using R-D and D-R modules to generate concealed adversarial point cloud samples with unparalleled performance in attack effectiveness, robustness after subsampling, and transferability. By leveraging ray tracing technology, our physical attack scheme effectively deceives deep learning-based target recognition algorithms in real-world scenarios. Thus, our work not only significantly contributes to bolstering the security of point cloud recognition methods, but also pioneers the extension of physical attack objects from the LiDAR domain to the remote sensing radar domain. This robust and versatile method is primed to shape the future of LiDAR data handling, advancing the scientific and technological facets of this field.

References

[1] F. Levet and J.-B. Sibarita, "Poca: A software platform for point cloud data visualization and quantification," *Nature Methods*, vol. 20, pp. 629–630, 2023.

[2] Y. Zhang, L. Chen, Z. XuanYuan, and W. Tian, "Three-dimensional cooperative mapping for connected and automated vehicles," *IEEE Trans. Ind. Electron.*, vol. 67, no. 8, pp. 6649–6658, Aug. 2020.

[3] F. Ghorbani, H. Ebadi, A. Sedaghat, and N. Pfeifer, "A novel 3-D local daisy-style descriptor to reduce the effect of point displacement error in point cloud registration," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 15, pp. 2254–2273, 2022.

[4] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "ScanNet: Richly-annotated 3D reconstructions of indoor scenes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5828–5839.

[5] Z. Wu et al., "3D shapenets: A deep representation for volumetric shapes," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 1912–1920.

[6] B. Yang, W. Luo, and R. Urtasun, "Pixor: Real-time 3D object detection from point clouds," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7652–7660.

[7] S. Shi, X. Wang, and H. Li, "PointRCNN: 3D object proposal generation and detection from point cloud," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 770–779.

[8] F. Gao, T. Huang, J. Sun, J. Wang, A. Hussain, and E. Yang, "A new algorithm for SAR image target recognition based on an improved deep convolutional neural network," *Cogn. Comput.*, vol. 11, pp. 809–824, 2019.

[9] F. J. Lawin, M. Danelljan, P. Tosteberg, G. Bhat, F. S. Khan, and M. Felsberg, "Deep projective 3D semantic segmentation," in *Proc. Int. Conf. Comput. Anal. Images Patterns*, 2017, pp. 95–107.

[10] Y. Pang et al., "Improved crop row detection with deep neural network for early-season maize stand count in UAV imagery," *Comput. Electron. Agriculture*, vol. 178, 2020, Art. no. 105766.

[11] B. Wu, A. Wan, X. Yue, and K. Keutzer, "Squeezeseg: Convolutional neural nets with recurrent CRF for real-time road-object segmentation from 3D LiDAR point cloud," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 1887–1893.

[12] Y. Pang et al., "Graph decipher: A transparent dual-attention graph neural network to understand the message-passing mechanism for the node classification," *Int. J. Intell. Syst.*, vol. 37, no. 11, pp. 8747–8769, 2022.

[13] Y. Pang et al., "Sparse-dyn: Sparse dynamic graph multirepresentation learning via event-based sparse temporal attention network," *Int. J. Intell. Syst.*, vol. 37, no. 11, pp. 8770–8789, 2022.

[14] J. Gao, B. Deng, Y. Qin, X. Li, and H. Wang, "Point cloud and 3-D surface reconstruction using cylindrical millimeter-wave holography," *IEEE Trans. Instrum. Meas.*, vol. 68, no. 12, pp. 4765–4778, Dec. 2019.

[15] F. Ma, X. Sun, F. Zhang, Y. Zhou, and H.-C. Li, "What catch your attention in SAR images: Saliency detection based on soft-superpixel lacunarity cue," *IEEE Trans. Geosci. Remote Sens.*, vol. 61, 2023, Art. no. 5200817.

[16] T. Huang, Q. Zhang, J. Liu, R. Hou, X. Wang, and Y. Li, "Adversarial attacks on deep-learning-based SAR image target recognition," *J. Netw. Comput. Appl.*, vol. 162, 2020, Art. no. 102632.

[17] T. Huang, Y. Chen, B. Yao, B. Yang, X. Wang, and Y. Li, "Adversarial attacks on deep-learning-based radar range profile target recognition," *Inf. Sci.*, vol. 531, pp. 159–176, 2020.

[18] F. He, Y. Chen, R. Chen, and W. Nie, "Point cloud adversarial perturbation generation for adversarial attacks," *IEEE Access*, vol. 11, pp. 2767–2774, 2023.

[19] D. Liu, R. Yu, and H. Su, "Extending adversarial attacks and defenses to deep 3D point cloud classifiers," in *Proc. IEEE Int. Conf. Image Process.*, 2019, pp. 2279–2283.

[20] C. Xiang, C. R. Qi, and B. Li, "Generating 3D adversarial point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 9128–9136.

[21] H. Zhang and X. Ma, "Misleading attention and classification: An adversarial attack to fool object detection models in the real world," *Comput. Secur.*, vol. 122, 2022, Art. no. 102876.

[22] Y. Liu, H. Xu, D. Liu, and L. Wang, "A digital twin-based sim-to-real transfer for deep reinforcement learning-enabled industrial robot grasping," *Robot. Comput.- Integr. Manuf.*, vol. 78, 2022, Art. no. 102365.

[23] A. Diab, R. Kashef, and A. Shaker, "Deep learning for LiDAR point cloud classification in remote sensing," *Sensors*, vol. 22, no. 20, 2022, Art. no. 7868.

[24] Y. Li et al., "Deep learning for LiDAR point clouds in autonomous driving: A review," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 32, no. 8, pp. 3412–3432, Aug. 2021.

[25] R. Wang, J. Peethambaran, and D. Chen, "LiDAR point clouds to 3-D urban models: A review," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 11, no. 2, pp. 606–627, Feb. 2018.

[26] C. Vishnu, J. Khandelwal, C. K. Mohan, and C. L. Reddy, "EVAA—Exchange vanishing adversarial attack on LiDAR point clouds in autonomous vehicles," *IEEE Trans. Geosci. Remote Sens.*, vol. 61, 2023, Art. no. 5703410.

[27] S. Qi et al., "Review of multi-view 3D object recognition methods based on deep learning," *Displays*, vol. 69, 2021, Art. no. 102053.

[28] S. Huang, L. Liu, X. Fu, J. Dong, F. Huang, and P. Lang, "Overview of LiDAR point cloud target detection methods based on deep learning," *Sensor Rev.*, vol. 42, no. 5, pp. 485–502, 2022.

[29] H. Su, S. Maji, E. Kalogerakis, and E. Learned-Miller, "Multi-view convolutional neural networks for 3D shape recognition," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 945–953.

[30] G. Pang and U. Neumann, "Fast and robust multi-view 3D object recognition in point clouds," in *Proc. Int. Conf. 3D Vis.*, 2015, pp. 171–179.

[31] D. Maturana and S. Scherer, "VoxNet: A. 3D convolutional neural network for real-time object recognition," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2015, pp. 922–928.

[32] Y. Zhou and O. Tuzel, "VoxelNet: End-to-end learning for point cloud based 3D object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 4490–4499.

[33] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 652–660.

[34] G. Qian et al., "Pointnext: Revisiting pointnet++ with improved training and scaling strategies," *Adv. Neural Inf. Process. Syst.*, vol. 35, pp. 23 192–23 204, 2022.

[35] C. Chen and T. Huang, "Camdar-adv: Generating adversarial patches on 3D object," *Int. J. Intell. Syst.*, vol. 36, no. 3, pp. 1441–1453, 2021.

[36] J. Tu et al., "Physically realizable adversarial examples for lidar object detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 13716–13725.

[37] A. Bar et al., "The vulnerability of semantic segmentation networks to adversarial attacks in autonomous driving: Enhancing extensive environment sensing," *IEEE Signal Process. Mag.*, vol. 38, no. 1, pp. 42–52, 2020.

[38] Y. Cao et al., "Adversarial sensor attack on lidar-based perception in autonomous driving," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, 2019, pp. 2267–2281.

[39] C. Gao, G. Wang, W. Shi, Z. Wang, and Y. Chen, "Autonomous driving security: State of the art and challenges," *IEEE Internet Things J.*, vol. 9, no. 10, pp. 7572–7595, May 2022.

[40] S. Parkinson, P. Ward, K. Wilson, and J. Miller, "Cyber threats facing autonomous and connected vehicles: Future challenges," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 11, pp. 2898–2915, Nov. 2017.

[41] J. Sun, Y. Cao, Q. A. Chen, and Z. M. Mao, "Towards robust lidar-based perception in autonomous driving: General black-box adversarial sensor attack and countermeasures," in *Proc. 29th USENIX Secur. Symp.*, 2020, pp. 877–894.

[42] S. Auer, R. Bamler, and P. Reinartz, "RaySAR-3D SAR simulator: Now open source," in *Proc. IEEE Int. Geosci. Remote Sens. Symp.*, 2016, pp. 6730–6733.