3DHacker: Spectrum-based Decision Boundary Generation for Hard-label 3D Point Cloud Attack

Yunbo Tao[1*]   Daizong Liu[2*]   Pan Zhou[1]   Yulai Xie[1†]   Wei Du[1]   Wei Hu[2†]

[1]Hubei Key Laboratory of Distributed System Security,
Hubei Engineering Research Center on Big Data Security,
School of Cyber Science and Engineering, Huazhong University of Science and Technology
[2]Wangxuan Institute of Computer Technology, Peking University

{tyb666, panzhou, ylxie, weidu666}@hust.edu.cn   dzliu@stu.pku.edu.cn   forhuwei@pku.edu.cn

Abstract

With the maturity of depth sensors, the vulnerability of 3D point cloud models has received increasing attention in various applications such as autonomous driving and robot navigation. Previous 3D adversarial attackers either follow the white-box setting to iteratively update the coordinate perturbations based on gradients, or utilize the output model logits to estimate noisy gradients in the black-box setting. However, these attack methods are hard to be deployed in real-world scenarios since realistic 3D applications will not share any model details to users. Therefore, we explore a more challenging yet practical 3D attack setting, i.e., attacking point clouds with black-box hard labels, in which the attacker can only have access to the prediction label of the input. To tackle this setting, we propose a novel 3D attack method, termed 3D Hard-label attacker (3DHacker), based on the developed decision boundary algorithm to generate adversarial samples solely with the knowledge of class labels. Specifically, to construct the class-aware model decision boundary, 3DHacker first randomly fuses two point clouds of different classes in the spectral domain to craft their intermediate sample with high imperceptibility, then projects it onto the decision boundary via binary search. To restrict the final perturbation size, 3DHacker further introduces an iterative optimization strategy to move the intermediate sample along the decision boundary for generating adversarial point clouds with smallest trivial perturbations. Extensive evaluations show that, even in the challenging hard-label setting, 3DHacker still competitively outperforms existing 3D attacks regarding the attack performance as well as adversary quality.

1. Introduction

Deep Neural Networks (DNNs) are known to be vulnerable to adversarial examples [37, 10], which are indistin-
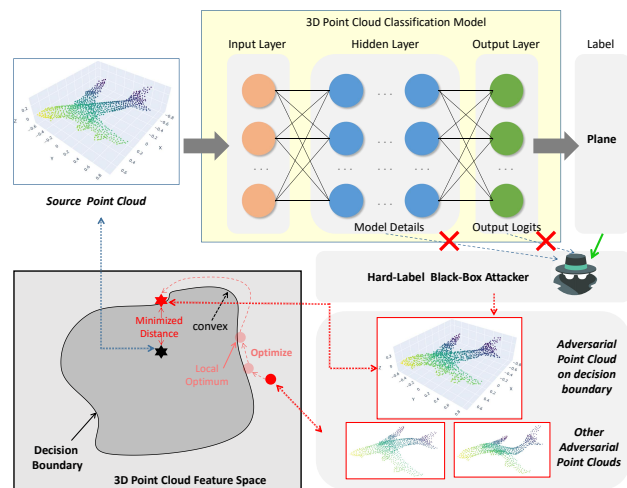

Figure 1. Illustration of our motivation. Our 3D hard-label setting cannot access any model details of the hidden layer or the logits of the output layer. To tackle this setting, we develop a spectrum-based decision boundary algorithm to iteratively optimize the best adversarial sample only with the knowledge of the predicted label.

guishable from legitimate ones by adding trivial perturbations but often lead to incorrect model prediction. Many efforts have been made into attacks on the 2D image field [7, 30, 21, 39]. However, in addition to image-based 2D attacks, 3D point cloud attacks are also crucial in various safety-critical applications such as autonomous driving [5, 49, 14], robotic grasping [41, 56], and face challenges in realistic scenarios.

Existing 3D point cloud attack methods [38, 54, 57, 43] generally delve into achieving high attack success rate and maintaining the imperceptibility of adversarial examples. However, these 3D attack methods still suffer from two main limitations: (1) Most of them are deployed in white-box setting where the attackers have the full knowledge of victim models including network structure and weights.

---

*Equal contributions.   †Corresponding authors.

This setting makes the attacks less practical since most real-world 3D applications will not share their model details to users. (2) The quality of their adversarial examples are limited. Since most previous works simply utilize geometric distance losses or additional shape knowledge to implicitly constrain the perturbations according to gradient search or gradient optimization, their adversarial examples fail to properly keep the original 3D object shape and easily have irregular surface or outliers. Although some attackers [51, 46, 43, 16] try to modify few points or design geometry-aware perturbations, their adversarial samples are hard to achieve optimal due to the greedy search in the optimization process.

Based on the above observations, in this paper, we make the attempt to explore a more challenging yet practical 3D attack setting, *i.e.*, attacking 3D point clouds with black-box hard labels, in which attackers can only have access to the final classification results of the model instead of the model details. Compared to previous general 3D black-box setting [16] that still has the knowledge of predicted logits scores of the input, our hard-label setting is more difficult since we only access the prediction labels of the input and cannot rely on the changes of the final predicted logits for updating the perturbations. To tackle this new attack setting while improving the quality of adversarial examples compared to previous works, we aim to exploit the decision boundary mechanism [24, 26, 23] as the core idea to generate adversarial point clouds by determining the class-aware model decision boundary for searching the mispredicted boundary cloud with small distance to the source cloud, as shown in Figure 1. Based on the concave-convex decision boundary, it can easily optimize samples with lowest perturbations.

However, directly adapting previous 2D decision boundary mechanisms [36, 18, 26, 24, 23] into the 3D point cloud field may face several challenges: (1) Previous 2D decision-boundary attackers [24, 23] generally generate adversarial images on decision boundary by calculating the weighted average of each pixel value between source and target images. However, 3D point cloud data mainly contain coordinates with optional point-wise values, which is unordered and irregularly sampled. Therefore, directly calculating the weighted average of the point coordinates between two point clouds would make no sense, since it will disarrange the relations of neighboring points and deform the 3D object shape. Although one solution would be to solely add coordinate-wise perturbations to generate the decision boundary, this operation is not geometry-aware and easily leads to outliers and uneven point distribution (validated in our Experiment Section). (2) In the 2D field, in addition to achieving high imperceptibility, the averaged image also preserves the semantic features of the original image since their pixel relations are implicitly maintained. However, the structural representations of 3D point clouds

in the latent space will be easily broken when modifying points in the data domain. Therefore, it is critically challenging to maintain both smooth appearance and geometric characteristics for generating high-quality 3D adversarial examples. (3) Almost all 2D decision boundary mechanisms solely utilize pixel-wise iterative walking to move the image along the decision boundary for optimization. However, directly applying the point-wise walking on 3D point clouds may stuck into the local optimum, since the optimized cloud on the decision boundary may not have the smallest perturbation and fail to overcome the large convex area without additional guidance as shown in Figure 1.

To address the above challenges, we propose a novel spectrum-based decision boundary attack method, called **3D H**ard-label att**acker** (**3DHacker**). Instead of fusing the point cloud via coordinate-wise average operation, we propose to fuse point clouds in the spectral domain, which not only preserves the geometric characteristics of both point clouds in coordinate-aware data domain, but also produces the smooth structure of the generated sample for achieving high imperceptibility. Specifically, our 3DHacker consists of two parts——*boundary-cloud generation* with spectrum-fusion and *boundary-cloud optimization* along the decision boundary. (1) During the *boundary-cloud generation*, we aim to attack a benign source-cloud into the class of target-clouds at their decision boundary with trivial perturbations. To be specific, we first perform Graph Fourier Transform (GFT) on each point cloud to obtain its graph spectrum, then fuse the spectrum of source-cloud and target-clouds with proper fusion rate with further inverse GFT to generate corresponding candidate point clouds. In this manner, we can obtain the decision boundary between the point clouds of different class labels. We select the best candidate with slightest distortion and project it on the decision boundary to obtain the boundary-cloud. (2) In *boundary-cloud optimization* stage, we perform iterative walking strategy on the boundary-cloud along the decision boundary aiming at further minimizing the distance between boundary-cloud and source-cloud. Each iteration starts from a perturbation generated by gradient estimation and then reduces the distortion through binary searching coordinates in the adversarial region. To jump out of the local optimum during optimization, we further design a spectrum-wise walking strategy in addition to the point-wise one. The boundary cloud optimized by the above two walking strategies will be taken as the final adversarial sample of the source cloud with high imperceptibility.

To sum up, our main contributions are three-fold:

- We achieve 3D point cloud attack in a more challenging yet practical black-box hard-label setting, and introduce a novel method 3DHacker based on decision boundary algorithm with point cloud spectrum.

- We propose spectrum-based decision boundary gener-

ation for preserving high-quality point cloud samples. Moreover, we introduce a new generation pipeline for boundary point clouds, and propose a joint coordinate- and spectrum-aware iterative walking strategy to alleviate the local optimum problem.

- Experimental results show that the proposed 3DHacker achieves 100% of attack success rates with the competitive perturbations compared to existing attack methods, even in the more challenging hard-label setting.

2. Related Work

Adversarial attack on 3D point cloud. Following previous studies on the 2D image field, many works [46, 44, 51, 55, 38, 54, 57] adapt adversarial attacks into the 3D vision community[11, 53, 48, 50, 42, 40, 3, 33, 34, 45]. [46] propose point generation attacks by adding a limited number of synthesized points/clusters/objects to a point cloud. [51] utilize gradient-guided attack methods to explore point modification, addition, and deletion attacks. Their goal is to add or delete key points, which can be identified by calculating the label-dependent importance score referring to the calculated gradient. Recently, more works [29, 52, 38] adopt point-wise perturbation by changing their xyz coordinates, which are more effective and efficient. [29] modify the FGSM strategy to iteratively search the desired pixel-wise perturbation. [38] adapt the C&W strategy to generate adversarial examples on point clouds and proposed a perturbation-constrained regularization in the overall loss function. Besides, some works [22, 20, 35, 8] attack point clouds in the feature space and target at perturbing less points with lighter distortions for an imperceptible adversarial attack. However, the generated adversarial point clouds of all above methods often result in messy distribution or outliers, which is easily perceivable by humans and defended by previous adversarial defense methods [58, 32, 25]. Although [43] improves the imperceptibility of adversarial attacks via a geometry-aware constraint, the adversarial samples sometimes also deform local surfaces and are still noticeable.

Decision boundary based 2D attacks. Decision boundary based attack method [1] is widely used in 2D filed, which is an efficient framework that uses final decision results of a classification model to implement hard-label black-box attack. Decision boundary is the boundary of classification model that separates different classes in the input space. In 2D field, the decision boundary attack process starts with two origin images called *source-image* and *target-image* with different labels. Next, it performs a binary search to obtain a *boundary-image* on the decision boundary between *source-image* and *target-image*. Then, the random walking algorithm is conducted on *boundary-image* with the goal of minimizing the distance towards *target-image* and maintaining the label same as *source-image*. Based on

this general attack framework, various 2D decision boundary based attacks are proposed committed to improve the random walking performance and query efficiency. [2, 36] propose to choose more efficient random perturbation including Perlin noise and DCT in random walking step instead of Gaussian perturbation. [4] conduct a gradient estimation method using Monte-carlo sampling strategy instead of random perturbation. [24, 23, 26] improve the gradient estimation strategy through sampling from representative low-dimensional subspace. However, to our best knowledge, there is no decision boundary based attack in 3D vision community so far, and directly adapting these 2D methods into 3D field may face many challenges.

3. Methodology

3.1. Notations and Problem Definition

3D point cloud task. Generally, a point cloud consists of an unordered set of points $\boldsymbol{P} = \{\boldsymbol{p}_i\}_{i=1}^n \in \mathbb{R}^{n \times 3}$ sampled from the surface of a 3D object or scene, where each point $\boldsymbol{p}_i \in \mathbb{R}^3$ is a vector that contains the coordinates (x, y, z) of point i, and n is the number of points. In this paper, we mainly focus on the basic point cloud classification task. Given a point cloud $\boldsymbol{P}$ as input, a learned classifier $f(\cdot)$ predicts a vector of confidence scores $f(\boldsymbol{P}) \in \mathbb{R}^C$. The final predicted label is $y = F(\boldsymbol{P}) = argmax_{i \in [C]} f(\boldsymbol{P})_i \in Y, Y = \{1, 2, 3, ..., C\}$ that represents the class of the original 3D object underlying the point cloud, where C is the number of classes.

Hard-label attack setting on 3D point cloud. In the hard-label attack setting, the attackers can only access the final predicted label $y = F(\boldsymbol{P})$ instead of the model parameters $f(\cdot)$ and the confidence scores (logits) $f(\boldsymbol{P})$ for gradient optimization. Therefore, we aim to search the decision boundary of the classification model to determine the difference between 3D object classes, and utilize the decision boundary to estimate the positive gradient for updating the perturbation. The perturbed adversarial sample $\boldsymbol{P}_{adv}$ is slightly generated from correctly classified source sample $\boldsymbol{P}_s$ labeled as y_{true} such that the predicted label $y_{adv} = F(\boldsymbol{P}_{adv}) \neq y_{true}$. We define an indicator function $\varphi(\cdot)$ of a successful attack as:

$$\varphi(\boldsymbol{P}_{adv}) \equiv \begin{cases} 1, & \text{if } y_{adv} \neq y_{true}, \\ -1, & \text{if } y_{adv} = y_{true}, \end{cases} \quad (1)$$

where y_s is the true label of source point cloud $\boldsymbol{P}_s$, $y_{adv} = F(\boldsymbol{P}_{adv})$ is the label of $\boldsymbol{P}_{adv}$ predicted by the 3D model.

3.2. Overview of Our 3DHacker Framework

To tackle the challenging 3D hard-label attack setting, in this section, we introduce the overall pipeline of our proposed 3DHacker, which starts with one source-cloud $\boldsymbol{P}_s$ of
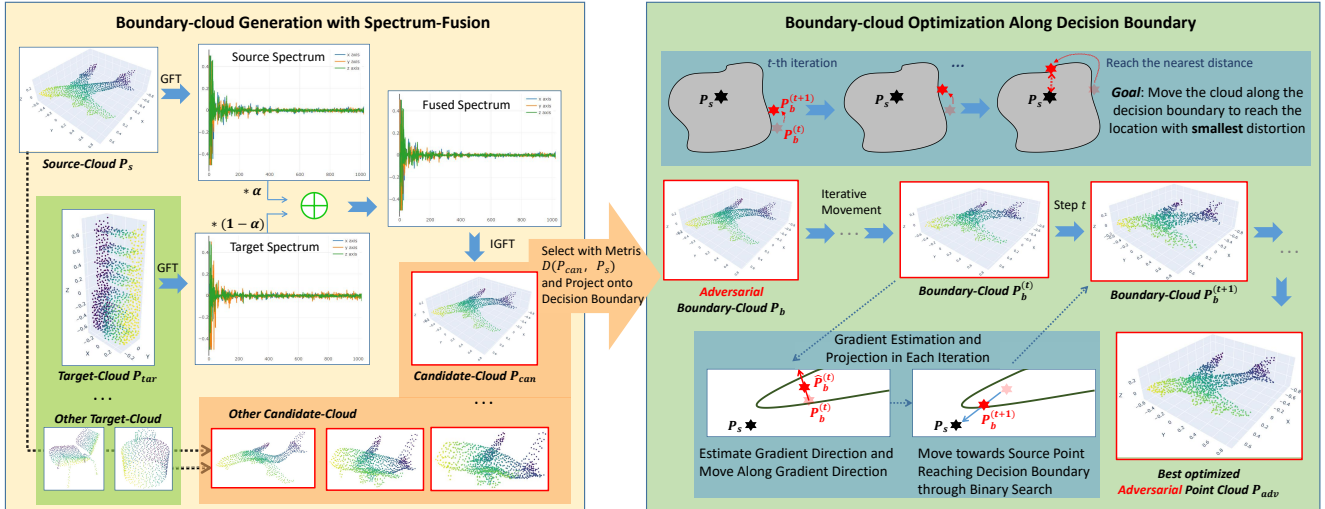
Figure 2. Overall pipeline of our proposed 3DHacker. The boundary-cloud generation module first fuses the source cloud with a set of target clouds in the spectral domain to construct candidate clouds, then selects the best one and projects it onto the decision boundary to obtain the boundary cloud via binary search. After that, the boundary-cloud optimization module iteratively moves the boundary cloud along the decision boundary via a joint coordinate-spectrum iterative walking strategy to achieve the best place with smallest distortion.

benign label y_{true} that is correctly classed by model $f(\cdot)$, and a group of target-clouds $\{\boldsymbol{P}_{tar}\}$ of class labels $\{y_{tar}\}$, $y_{tar} \neq y_{true}$. 3DHacker aims to generate a boundary cloud on the model decision boundary between the source and target point clouds as the adversarial sample, which has a different class label of the source cloud while sharing the same object shape with it. As shown in Figure 2, our 3DHacker consists of two main parts —— A **boundary-cloud generation** module is first utilized to produce a high-quality adversarial point cloud on the decision boundary, then a **boundary-cloud optimization** module is exploited to further optimize the adversarial point cloud along the decision boundary for achieving smallest perturbations.

First step: Boundary-cloud generation with spectrum-fusion. In this step, we first perform the fusion method between a source cloud $\boldsymbol{P}_s$ and multiple target clouds $\boldsymbol{P}_{tar}$ to obtain candidate point clouds $\boldsymbol{P}_{can}$ to construct the decision boundary. Then we select a best adversarial $\boldsymbol{P}_{can}$ such that $\varphi(\boldsymbol{P}_{can}) = 1$ and $D(\boldsymbol{P}_s, \boldsymbol{P}_{can})$ is the smallest, where D is the distance metric. After that, we project the best $\boldsymbol{P}_{can}$ onto the decision boundary via binary search algorithm to obtain the boundary cloud $\boldsymbol{P}_b$.

Previous 2D decision boundary mechanisms generally fuse the source and target samples via the pixel-wise average operation. However, directly applying this strategy into 3D domain is ineffective since 3D coordinate-wise fusion between point clouds will result in outlier problem and destroy the 3D object geometric shape, leading to poor imperceptibility. To address this issue, we introduce a new spectrum-fusion method, which fuses point clouds in the spectral domain to preserve the geometric topology. Instead of fusing point-wise coordinations, the spectrum-fusion method first transforms both source and target point

clouds into the spectral domain for representing their geometric characteristics, then fuses their spectral contexts and transforms the fused geometric characteristics back to the data domain as the adversarial sample $\boldsymbol{P}_{can}$. In this manner, the fused point cloud not only preserves the specific characteristics of the original point clouds, but also has more smooth geometric surface due to the spectrum advantages [15, 13], leading to high imperceptibility. Although with a slight and smooth perturbation, $\boldsymbol{P}_{can}$ still possesses a strong potential to confuse the model $f(\cdot)$ with a structure distortion learned from spectral characteristics of the $\boldsymbol{P}_{tar}$.

Second step: Boundary-cloud optimization along the decision boundary. Although we can obtain the boundary cloud $\boldsymbol{P}_b$ in the first step, this $\boldsymbol{P}_b$ is not the best optimal one due to the specific concave-convex structure of the decision boundary. Therefore, we propose to update $\boldsymbol{P}_b$ along the decision boundary via the iterative walking strategy to search for a high-quality $\boldsymbol{P}_b$ that has the lowest geometric distance to the source cloud.

Previous 2D decision boundary mechanisms generally conduct the iterative walking strategy with pixel-wise offsets, which can not be directly applied into the 3D domain since point-wise coordinate walking will lead to local optimum object shape. The reason is, although this boundary cloud is well optimized and still remains adversarial, it may get stuck in the local concave area with a large neighboring convex area, and the estimated gradients towards this convex will lead to a larger geometric distance to the source cloud. To alleviate this issue, we extend the point-wise coordinate walking with a new spectrum walking as additional guidance for jumping out of such local optimum. That is, we update the boundary cloud $\boldsymbol{P}_b$ along the decision boundary with coordinate modification for shape re-

finement and spectrum modification for maintaining the geometric smoothness and high imperceptibility. At last, the updated $\boldsymbol{P}_b$ on the decision boundary is our final generated adversarial sample. In the following, we will provide more details about each step.

3.3. Boundary-Cloud Generation with Spectrum-Fusion

Given the source cloud $\boldsymbol{P}_s$ and T number of target clouds $\boldsymbol{P}_{tar}$, we aim to generate a desired boundary-cloud $\boldsymbol{P}_b$ of label $y_b \neq y_{true}$ with a minimized distance $D(\boldsymbol{P}_s, \boldsymbol{P}_b)$. Specifically, for each pair of $\boldsymbol{P}_s$ and $\boldsymbol{P}_{tar}$, we first conduct Graph Fourier Transform (GFT) [15] to obtain their corresponding spectrum coefficient vector $\hat{\boldsymbol{x}}_s$ and $\hat{\boldsymbol{x}}_{tar}$. Considering that low-frequency components mainly represent the basic 3D object shape while high-frequency components encode find-grained details [13], we then fuse their spectrum coefficient vectors with different fusion weights for low and high frequencies, and perform inverse GFT (IGFT) to transform the fused spectrum coefficient vector back to data domain for obtaining the candidate-cloud $\boldsymbol{P}_{can}$. In this way, we can get T candidate-clouds $\boldsymbol{P}_{can}$ for T target clouds $\boldsymbol{P}_{tar}$. The above process can be formulated as:

$$
\begin{aligned}
\hat{\boldsymbol{x}}_{low} &= \alpha_{low}\text{GFT}(\boldsymbol{P}_s)_L + (1 - \alpha_{low})\text{GFT}(\boldsymbol{P}_{tar})_L, \\
\hat{\boldsymbol{x}}_{high} &= \alpha_{high}\text{GFT}(\boldsymbol{P}_s)_H + (1 - \alpha_{high})\text{GFT}(\boldsymbol{P}_{tar})_H, \\
\boldsymbol{P}_{can} &= \text{IGFT}(\hat{\boldsymbol{x}}_{low} \boxplus \hat{\boldsymbol{x}}_{high}),
\end{aligned}
\tag{2}
$$

where $GFT(\boldsymbol{P}_s)_L, GFT(\boldsymbol{P}_s)_H$ denotes conducting GFT on $\boldsymbol{P}_s$ and splitting its spectrum coefficient into the low L and high H frequencies. $\hat{\boldsymbol{x}}_{low} \boxplus \hat{\boldsymbol{x}}_{high}$ denotes piecing two coefficient vectors into a complete one. For spectrum-fusion, we use different fusion weights α_{low} and α_{high} for low and high spectrum frequencies since the lowest 32 frequencies account for almost 90% of energy and the other 992 spectrum frequencies account for 10% of energy [28], which lead to point clouds more sensitive to modifications on the lowest 32 frequencies than higher frequencies.

After obtaining T candidate clouds, we conduct adversarial evaluation on each candidate cloud to select the best one as the boundary cloud $\boldsymbol{P}_b$. To be specific, we select the best boundary-cloud $\boldsymbol{P}_{can}^b$ from the adversarial candidate clouds with the slightest distortion measured by distance metric $D(\boldsymbol{P}_s, \boldsymbol{P}_{can})$:

$$
\begin{aligned}
\boldsymbol{P}_{can}^b &= argmin_{\boldsymbol{P}_{can}^i} D(\boldsymbol{P}_s, \boldsymbol{P}_{can}^i)(i = 1, 2, ..., T), \\
s.t. \ &\varphi(\boldsymbol{P}_{can}^i) = 1, max_j(\left\|\boldsymbol{p}_{can,j}^i - \boldsymbol{p}_{s,j}\right\|_2) \leq \varepsilon,
\end{aligned}
\tag{3}
$$

where $\boldsymbol{P}_{can}^i$ denotes the i-th candidate-cloud, $\boldsymbol{p}_{can,j}^i$ denotes the j-th points in $\boldsymbol{P}_{can}^i$, $\boldsymbol{p}_{s,j} \in \boldsymbol{P}_s$ denotes the point with lowest distance from $\boldsymbol{p}_{can,j}^i$. The distance measure-

ment function $D(\cdot)$ is formulated as:

$$
\begin{aligned}
D(\boldsymbol{P}_s, \boldsymbol{P}_{can}) &= D_{Chamfer}(\boldsymbol{P}_s, \boldsymbol{P}_{can}) \\
&+ \gamma_1 D_{Hausdorff}(\boldsymbol{P}_s, \boldsymbol{P}_{can}) + \gamma_2 D_{Variance}(\boldsymbol{P}_s, \boldsymbol{P}_{can}),
\end{aligned}
\tag{4}
$$

where $D_{Chamfer}$ and $D_{Hausdorff}$ measure the distance between two point clouds following [43]. $D_{Variance}$ is the variance of distance between two point clouds. γ_1 and γ_2 are penalty parameters. We then project the $\boldsymbol{P}_{can}^b$ onto the decision boundary to obtain the boundary-cloud $\boldsymbol{P}_b$. Specifically, we conduct a binary search strategy [31] to move $\boldsymbol{P}_{can}^b$ towards $\boldsymbol{P}_s$ until reaching the decision boundary as:

$$
\boldsymbol{P}_b = \beta\boldsymbol{P}_s + (1 - \beta)\boldsymbol{P}_{can}^b,
\tag{5}
$$

where β is the moving ratio in binary search.

3.4. Boundary-Cloud Optimization Along Decision Boundary

The goal of boundary-cloud optimization is to further minimize the distance between the boundary cloud $\boldsymbol{P}_b$ and the source cloud $\boldsymbol{P}_s$ by moving $\boldsymbol{P}_b$ along the decision boundary to the optimal place so that the perturbation is smallest yet imperceptible while $\boldsymbol{P}_b$ keeps adversarial. Specifically, we utilize the iterative walking algorithm to optimize the point cloud $\boldsymbol{P}_b$ by two steps: *1. Estimate updation gradient and move the point cloud along it*, *2. Project the moved point cloud back to the decision boundary*.

1. Estimate updation gradient and move the point cloud along it. As for initialization, we take $\boldsymbol{P}_b$ obtained in Section.3.3 as the initial boundary-cloud $\boldsymbol{P}_b^{(0)}$. For the next step, we denote $\boldsymbol{P}_b^{(t)}$ as the boundary-cloud obtained in the t-th walking iteration which is exactly on the decision boundary. We aim to estimate a gradient direction of $\boldsymbol{P}_b^{(t)}$ for improving the gap between the adversarial and the true class labels while preserving the geometric distance, so that we can make $\boldsymbol{P}_b^{(t)}$ more aggressive with small distortion and can further move it towards source cloud $\boldsymbol{P}_s$. Specifically, we utilize the *point-wise walking* to conduct Monte Carlo method [19] to obtain the estimated gradient vector $\triangledown S^{(t)}$ under the guidance of indicator function $\varphi(\cdot)$:

$$
\triangledown S^{(t)} = \frac{1}{B}\sum_{i=1}^{B}\varphi(\boldsymbol{P}_b^{(t)} + v_i)v_i,
\tag{6}
$$

where v_i is the sampled move vectors obeying a normal distribution and B is the number of vectors sampled in Monte Carlo method. Then we move $\boldsymbol{P}_b^{(t)}$ along $\triangledown S^{(t)}$ with a step size ξ by:

$$
\boldsymbol{P}_b^{temp} = \boldsymbol{P}_b^{(t)} + \xi\frac{\triangledown S^{(t)}}{\left\|\triangledown S^{(t)}\right\|_2}.
\tag{7}
$$

2. Project the moved point cloud back to decision boundary. After moving the point cloud along the estimated gradient, it will leave the decision boundary. Thus we conduct a binary search strategy to move $\boldsymbol{P}_b^{temp}$ towards $\boldsymbol{P}_s$ until reaching the decision boundary again. With multiple such walking iterations, we take the last $\boldsymbol{P}_b^{(t)}$ with the best optimization as the final adversarial point cloud.

So far, the optimization algorithm is complete, and we generate a desired adversarial point cloud $\boldsymbol{P}_b^{(t)}$. However, as mentioned in Section.3.2, there remains the local optimum problem. To address it, in addition to the *point-wise walking strategy*, we also conduct a *spectrum-wise walking* to bring a great movement for cloud features and escape the convex area. Specifically, we maintain other operations and solely replace the coordinate gradient walking in Eq.5 and Eq.6 with a spectrum gradient walking as:

$$\nabla S^{(t)} = \frac{1}{B} \sum_{i=1}^{B} \varphi(IGFT(GFT(\boldsymbol{P}_b^{(t)}) + u_i))u_i, \quad (8)$$

$$\boldsymbol{P}_b^{temp} = \boldsymbol{P}_b^{(t)} + \xi_{spe} \frac{\nabla S^{(t)}}{\left\| \nabla S^{(t)} \right\|_2}, \quad (9)$$

where ξ_{spe} is a step size for spectrum walking. By combining two walking operations for multi-step walking, the spectrum walking is able to perform large movement to jump out of local optima for a better optimization region, while the coordinate walking is able to perform slight movement to gradually fine-tune for getting the best optimization point in such region. The iterative walking algorithm is detailed in Algorithm.1. The optimized point cloud is our final adversarial sample.

Algorithm 1: Joint Spectrum&Coordinate Walking

Input: Boundary cloud $\boldsymbol{P}_b^{(0)}$, iteration step R, best optimized cloud list $Best = []$
Output: Adversarial point cloud $\boldsymbol{P}_{adv}$

1   $Best[0] = \boldsymbol{P}_b^{(0)}$;
2   **for** $i = 1 : R$ **do**
3   conduct coordinate walking: $\boldsymbol{P}_b^{(i)} \rightarrow \boldsymbol{P}_b^{(i+1)}$;
4   **if** $\boldsymbol{P}_b^{(i)} = \boldsymbol{P}_b^{(i+1)}$ **then**
5   conduct spectrum walking:
       $\boldsymbol{P}_b^{(i+1)} \rightarrow \boldsymbol{P}_b^{(i+1)}$;
6   **else**
7       $Best$.append($\boldsymbol{P}_b^{(i+1)}$);
8   **end**
9   **end**
10 Select $\boldsymbol{P}_{adv} \in Best$ with the smallest distance D;

---

4. Experiments

4.1. Dataset and 3D Models

Dataset. We use ModelNet40 [45] in our experiments to evaluate the attack performance. ModelNet40 consists of 12,311 CAD models from 40 object categories, in which 9,843 models are intended for training and the other 2,468 for testing. Following the settings of previous work [43, 27, 13], we uniformly sample 1024 points from the surface of each object and scale them into a unit ball. For the adversarial point cloud attacks, we follow [46, 13] and randomly select 25 instances for each of 10 object categories in the ModelNet40 testing set, which can be well classified by the classifiers of interest.

3D Models. We use three 3D networks as the victim models: PointNet [33], PointNet++ [34], DGCNN [42]. Experiments on other models can be found in supplementary.

4.2. Implementation Details

Attack Setup. We set K = 10 to build a K-NN graph to conduct Graph Fourier Transform (GFT). The weights of Hausdorff distance loss and variance loss e.g.γ_1 and γ_2 in Eq.4 are set to 2.0 and 0.5, respectively. For the spectrum fusion rate α in Eq.2, the low frequencies weight α_{low} are set to 0.85 and the high frequencies weight α_{high} are set to 0.2. We use $B = 50$ queries in the Monte Carlo algorithm to estimate the gradient. We conduct $R = 200$ iteration rounds during boundary-cloud Optimization stage. We set $\xi = \|\boldsymbol{P}_b^{(t)} - \boldsymbol{P}_s\|_2/\sqrt{t}$ in Eq.7 and $\xi_{spe} = 5.0$ in Eq.8.

Evaluation Metrics. To quantitatively evaluate the effectiveness of our 3DHacker, we measure the perturbation size via three metrics: (1) L2-norm distance D_{norm} [6]; (2) Chamfer distance D_c [9]; (3) Hausdorff distance D_h[17].

4.3. Evaluation on Our 3DHacker Attack

Comparison with existing methods. To investigate the effectiveness of our attack, we perform several existing white-box adversarial attacks and one black-box adversarial attack for quantitative comparison as shown in Table 1. Table 1 shows that our 3DHacker achieves smaller perturbation sizes than the black-box model and achieves very competitive results with white-box models. Since our 3DHacker conducts global perturbations to origin point clouds which possess a strong potential to confuse the victim models with a structure distortion, this global perturbations produce a higher Chamfer distance D_c compared with 3D-ADV^p and SI-Adv because D_c measures the average squared distance between each adversarial point and its nearest original point and we modify all the points leading to a large sum of displacements. Instead, attacking by modifying a few points in 3D-ADV^p and SI-Adv has advantage in D_c because most of the distance is equal to 0. However, 3DHacker performs better in D_h since we conduct relatively average perturba-

Table 1. Comparative results on the perturbation sizes of adversarial point clouds generated by different attack methods under 100% ASR.

Setting	Attack	Model Details		PointNet [33]			PointNet++ [34]			DGCNN [42]		
		Para.	Logits	D_h	D_c	D_{norm}	D_h	D_c	D_{norm}	D_h	D_c	D_{norm}
White-Box	FGSM [47]	✓	✓	0.1853	0.1326	0.7936	0.2275	0.1682	0.8357	0.2506	0.189	0.8549
	PGD [30]	✓	✓	0.1322	0.1329	0.7384	0.1623	0.1138	0.7596	0.1546	0.1421	0.7756
	AdvPC [12]	✓	✓	0.0343	0.0697	0.6509	0.0429	0.0685	0.6750	0.0148	0.0623	0.6304
	3D-ADV^p [46]	✓	✓	0.0105	0.0003	0.5506	0.0381	0.0005	0.5699	0.0475	0.0005	0.5767
	LG-GAN [57]	✓	✓	0.0362	0.0347	0.7184	0.0407	0.0238	0.6896	0.0348	0.0119	0.8527
	GeoA^3 [43]	✓	✓	0.0175	0.0064	0.6621	0.0357	0.0198	0.6909	0.0402	0.0176	0.7024
	SI-Adv^w [16]	✓	✓	0.0204	0.0002	0.7614	0.0310	0.0004	1.2830	0.0127	0.0006	1.1120
Black-Box	SI-Adv^b [16]	×	✓	0.0431	0.0003	0.9351	0.0444	0.0003	1.0857	0.0336	0.0004	0.9081
Hard-Label Black-Box	Ours	×	×	**0.0136**	**0.0017**	**0.8561**	**0.0245**	**0.0023**	**0.9324**	**0.0129**	**0.0026**	**0.9030**

Table 2. Resistance of the black-box attacks on defended point cloud models.

Defense	Attack	PointNet [33]				PointNet++ [34]				DGCNN [42]			
		ASR(%)	D_h	D_c	D_{norm}	ASR(%)	D_h	D_c	D_{norm}	ASR(%)	D_h	D_c	D_{norm}
SOR[58]	SI-Adv^b [16]	89.7	0.0420	**0.0009**	3.0193	78.9	0.0436	**0.0025**	1.3843	72.0	0.0341	**0.0009**	1.6480
	Ours	**90.4**	**0.0100**	0.0023	**1.2486**	**82.7**	**0.0218**	0.0043	**1.3759**	**85.4**	**0.0124**	0.0031	**1.2387**
Drop(30%)	SI-Adv^b [16]	96.9	0.0426	**0.0003**	1.3680	70.1	0.0473	**0.0023**	1.4538	71.2	0.0400	**0.0004**	0.8598
	Ours	**97.2**	**0.0179**	0.0016	**0.8391**	**71.3**	**0.0298**	0.0031	**1.2810**	**78.5**	**0.0195**	0.0033	1.1742
Drop(50%)	SI-Adv^b [16]	93.6	0.0420	**0.0002**	1.3844	67.6	0.0501	**0.0013**	1.9193	75.2	0.0358	**0.0004**	0.6992
	Ours	**95.4**	**0.0182**	0.0023	**0.8328**	**77.4**	**0.0285**	0.0032	**1.4735**	**76.8**	**0.0172**	0.0036	1.2914

tions to point cloud which does not count on a few outliers to confuse the victim models, leading to imperceptible and having the potential to bypass the outlier detection defense. **Visualization results.** We provide visualization on adversarial samples generated by our 3DHacker, SI-Adv^w[16] (white box attack) and SI-Adv^b[16] (black box attack) in Figure 3. Our 3DHacker can alleviate the outlier point problems and produce more imperceptible adversarial samples.
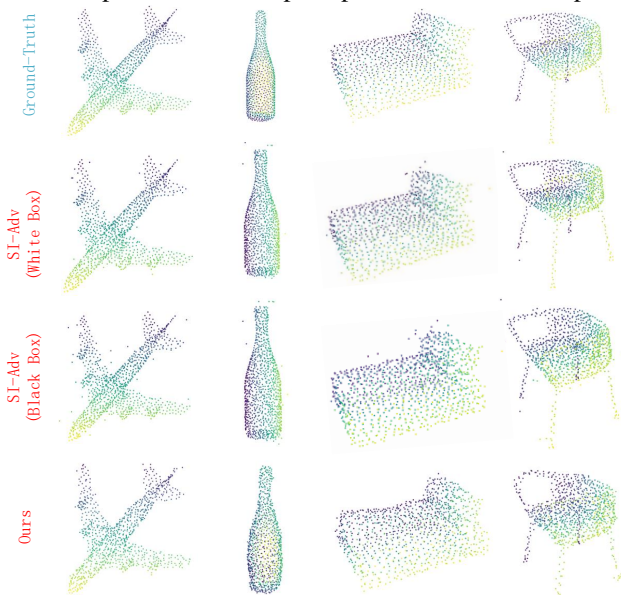

Figure 3. Visualization results of adversarial samples.

Resistance to Defenses. To evaluate the robustness of our 3DHacker against different adversarial defenses, we conduct the experiments on two widely used defense methods: Statistical Outlier Removal(SOR) [58] and Simple Random Sampling(SRS) [47]. Following the defense experiments setting on SRS in [16], we conduct SRS by randomly dropping 30% and 50% of input points respectively. As shown in Table 2, (1) Our 3DHacker can achieve a higher attack success rate than SI-Adv^b when attacking the model protected by SOR. This is because our method alleviates the outlier point problems and selects the best adversarial samples with the smallest perturbations, while SI-Adv^b still suffers from the perturbed point of outlier in the sharp component. (2) As for SRS defense, our 3DHacker still achieves a better attack than SI-Adv^b as we generate the adversarial sample with high similarity to the original one in both geometric topology and local point distributions. Overall, our 3DHacker is much more robust to existing defense strategies.

4.4. Ablation Study

Investigation on different strategies for boundary-cloud generation. To verify the effects of our spectrum fusion method in boundary-cloud generation stage, we conduct the experiments by replacing the spectrum fusion method with different strategies while maintaining the latter procedure and settings in boundary-cloud optimization stage the same. Specifically, two general strategies are compared: traditional coordinate fusion (which fuses source-cloud and target-cloud in coordinate space with proper fusion rate) and simple random perturbation (which directly add pointwise noise to source-cloud to reach the decision boundary). As shown in Table 3, our spectrum fusion achieves

Table 3. Investigation on different strategies in the boundary-cloud generation stage. Victim model: PointNet.

Generation Strategy	D_h	D_c	D_{norm}
Spectrum Fusion	**0.0136**	**0.0017**	**0.8561**
Coordinate Fusion	0.0275	0.0038	1.4379
Random Perturbation	0.0256	0.0023	0.9673

Table 4. The effect of different walking methods in the boundary-cloud optimization stage. Victim model: PointNet.

Coordinate Walking	Spectrum Walking	D_h	D_c	D_{norm}
✓	✓	**0.0136**	**0.0017**	**0.8561**
✓	✗	0.0198	0.0047	1.5827
✗	✓	0.0598	0.0109	3.1746

Table 5. Analysis on Iteration Round R. Victim model: PointNet.

Iteration Rounds R	D_h	D_c	D_{norm}
$R = 100$	0.0185	0.0024	1.1613
$R = 150$	0.0152	0.0022	0.9470
$R = 200$	0.0136	0.0017	0.8561
$R = 250$	**0.0131**	**0.0016**	**0.8487**

Table 6. Sensitivity analysis on Numbers of selected samples B in Monte Carlo algorithm. Victim model: PointNet.

selected samples B	D_h	D_c	D_{norm}
$B = 10$	0.0164	0.0023	1.3725
$B = 30$	0.0141	0.0018	0.9636
$B = 50$	**0.0136**	**0.0017**	**0.8561**
$B = 70$	0.0139	**0.0017**	0.9073

the smallest perturbations than other strategies in all metrics, this is because: (1) coordinate fusion will destroy to geometric structure by averaging different shapes of 3D objects; (2) random perturbation will lead to outliers and uneven point distribution without geometric awareness. Their visualized adversarial samples are shown in Figure 4 (a,b,c), where our samples are more imperceptible.

The effect of different walking methods for boundary-cloud optimization. In the boundary-cloud optimization stage, we design a spectrum walking method in addition to the coordinate one to jump out of the local optimum. To investigate the effect of each walking strategy, as shown in Table 4, we remove one of them to conduct the ablations for comparison. From this table, without spectrum walking, the attack process is easily trapped into the local optimum, leading to larger perturbations. Without coordinate walking, as shown in Figure 4 (d), it is hard to measure the point-wise imperceptibility for optimization, thus achieving the worst performance. By utilizing both of them, our model can preserve both high imperceptibility and geometric smoothness.

Sensitivity on the iteration rounds R. As shown in Table.5, we conduct the ablation on the iteration rounds R of the boundary-cloud optimization. Our model achieves the best performance when R is set to 250. However, the model with $R = 250$ is slightly better than the model with $R = 200$, but leads to much more time consumption. To balance both the performance and time cost, we choose
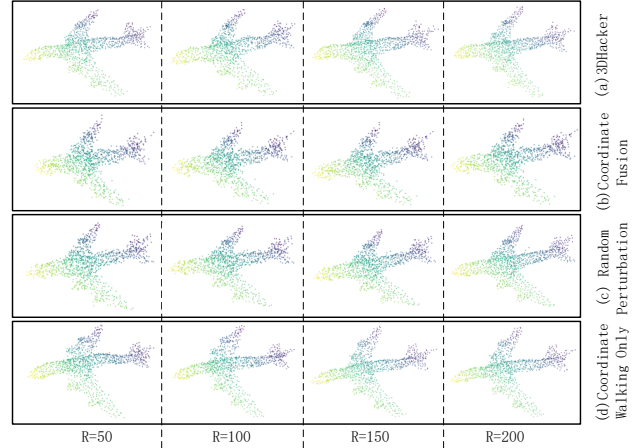

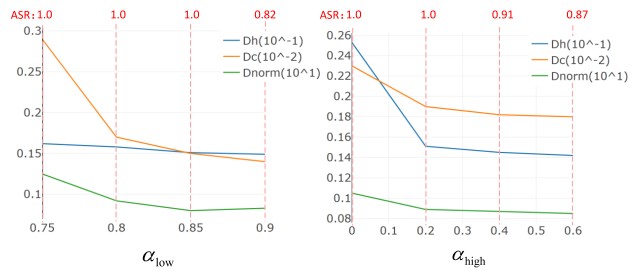Figure 4. Visualization on different ablations.

Figure 5. Influence of different spectrum fusion rates.

$R = 200$ in our all experiments. Visualization on adversarial point clouds of different R is shown in Figure 4.

Sensitivity on number B in Monte Carlo algorithm. As shown in Table 6, we conduct the ablation on the number B of selected samples in Monte Carlo algorithm. It shows that we achieve the smallest perturbation when $B = 50$.

Influence of different spectrum fusion rate. The spectrum fusion rate α_{low} and α_{high} decide the fusion weights of rough shape and fine details in Eq.2, respectively. Here, we modify α_{low} and α_{high} to analyze their influence on attack quality. In Figure 5, by balancing perturbation size and attack success rate, we set $\alpha_{low} = 0.85, \alpha_{high} = 0.2$.

5. Conclusion

We introduce a new and challenging 3D attack setting, *i.e.*, attacking point clouds with black-box hard labels. To address this practical setting, we propose a novel attack method called 3DHacker based on our decision boundary algorithm, which adopts spectrum fusion to generate boundary clouds with high imperceptibility and employs an additional spectrum walking strategy to move the boundary clouds along the decision boundary for further optimization with trivial perturbations. Experiments validate both effectiveness and robustness of our 3DHacker.

No.61972449).

References

[1] Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. *arXiv preprint arXiv:1712.04248*, 2017. 3

[2] Thomas Brunner, Frederik Diehl, Michael Truong Le, and Alois Knoll. Guessing smart: Biased sampling for efficient black-box adversarial attacks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4958–4966, 2019. 3

[3] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 3

[4] Jianbo Chen, Michael I Jordan, and Martin J Wainwright. Hopskipjumpattack: A query-efficient decision-based attack. In *2020 ieee symposium on security and privacy (sp)*, pages 1277–1294. IEEE, 2020. 3

[5] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1907–1915, 2017. 1

[6] Corinna Cortes, Mehryar Mohri, and Afshin Rostamizadeh. L2 regularization for learning kernels. *arXiv preprint arXiv:1205.2653*, 2012. 6

[7] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9185–9193, 2018. 1

[8] Yinpeng Dong, Jun Zhu, Xiao-Shan Gao, et al. Isometric 3d adversarial examples in the physical world. *Advances in Neural Information Processing Systems*, 35:19716–19731, 2022. 3

[9] Haoqiang Fan, Hao Su, and Leonidas J Guibas. A point set generation network for 3d object reconstruction from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 605–613, 2017. 6

[10] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint*, 2014. 1

[11] Meng-Hao Guo, Jun-Xiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph R Martin, and Shi-Min Hu. Pct: Point cloud transformer. *Computational Visual Media*, 7:187–199, 2021. 3

[12] Abdullah Hamdi, Sara Rojas, Ali Thabet, and Bernard Ghanem. Advpc: Transferable adversarial perturbations on 3d point clouds. In *European Conference on Computer Vision (ECCV)*, pages 241–257, 2020. 7

[13] Qianjiang Hu, Daizong Liu, and Wei Hu. Exploring the devil in graph spectral domain for 3d point cloud attacks. *arXiv preprint arXiv:2202.07261*, 2022. 4, 5, 6

[14] Qianjiang Hu, Daizong Liu, and Wei Hu. Density-insensitive unsupervised domain adaption on 3d object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17556–17566, 2023. 1

[15] Wei Hu, Jiahao Pang, Xianming Liu, Dong Tian, Chia-Wen Lin, and Anthony Vetro. Graph signal processing for geometric data and beyond: Theory and applications. *IEEE Transactions on Multimedia*, 2021. 4, 5

[16] Qidong Huang, Xiaoyi Dong, Dongdong Chen, Hang Zhou, Weiming Zhang, and Nenghai Yu. Shape-invariant 3d adversarial point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15335–15344, 2022. 2, 7

[17] Daniel P Huttenlocher, Gregory A. Klanderman, and William J Rucklidge. Comparing images using the hausdorff distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(9):850–863, 1993. 6

[18] Andrew Ilyas, Logan Engstrom, Anish Athalye, and Jessy Lin. Black-box adversarial attacks with limited queries and information. In *International Conference on Machine Learning*, pages 2137–2146. PMLR, 2018. 2

[19] Frederick James. Monte carlo theory and practice. *Reports on progress in Physics*, 43(9):1145, 1980. 5

[20] Jaeyeon Kim, Binh-Son Hua, Thanh Nguyen, and Sai-Kit Yeung. Minimal adversarial examples for deep learning on 3d point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7797–7806, 2021. 3

[21] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016. 1

[22] Kibok Lee, Zhuoyuan Chen, Xinchen Yan, Raquel Urtasun, and Ersin Yumer. Shapeadv: Generating shape-aware adversarial 3d point clouds. *arXiv preprint arXiv:2005.11626*, 2020. 3

[23] Huichen Li, Linyi Li, Xiaojun Xu, Xiaolu Zhang, Shuang Yang, and Bo Li. Nonlinear projection based gradient estimation for query efficient blackbox attacks. In *International Conference on Artificial Intelligence and Statistics*, pages 3142–3150. PMLR, 2021. 2, 3

[24] Huichen Li, Xiaojun Xu, Xiaolu Zhang, Shuang Yang, and Bo Li. Qeba: Query-efficient boundary-based blackbox attack. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1221–1230, 2020. 2, 3

[25] Kaidong Li, Ziming Zhang, Cuncong Zhong, and Guanghui Wang. Robust structured declarative classifiers for 3d point clouds: Defending adversarial attacks with implicit gradients. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15294–15304, 2022. 3

[26] Xiu-Chuan Li, Xu-Yao Zhang, Fei Yin, and Cheng-Lin Liu. Decision-based adversarial attack with frequency mixup. *IEEE Transactions on Information Forensics and Security*, 17:1038–1052, 2022. 2, 3

[27] Daizong Liu and Wei Hu. Imperceptible transfer attack and defense on 3d point cloud classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. 6

[28] Daizong Liu, Wei Hu, and Xin Li. Point cloud attacks in graph spectral domain: When 3d geometry meets graph signal processing. *arXiv preprint arXiv:2207.13326*, 2022. 5

[29] Daniel Liu, Ronald Yu, and Hao Su. Extending adversarial attacks and defenses to deep 3d point cloud classifiers. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 2279–2283, 2019. 3

[30] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017. 1, 7

[31] Robert Nowak. Generalized binary search. In *2008 46th Annual Allerton Conference on Communication, Control, and Computing*, pages 568–574. IEEE, 2008. 5

[32] Juan C Pérez, Motasem Alfarra, Silvio Giancola, Bernard Ghanem, et al. 3deformrs: Certifying spatial deformations on point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15169–15179, 2022. 3

[33] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 652–660, 2017. 3, 6, 7

[34] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in Neural Information Processing Systems (NIPS)*, 2017. 3, 6, 7

[35] Zhenbo Shi, Zhi Chen, Zhenbo Xu, Wei Yang, Zhidong Yu, and Liusheng Huang. Shape prior guided attack: Sparser perturbations on 3d point clouds. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 8277–8285, 2022. 3

[36] Vignesh Srinivasan, Ercan E Kuruoglu, Klaus-Robert Müller, Wojciech Samek, and Shinichi Nakajima. Black-box decision based adversarial attack with symmetric α-stable distribution. In *2019 27th European Signal Processing Conference (EUSIPCO)*, pages 1–5. IEEE, 2019. 2, 3

[37] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013. 1

[38] Tzungyu Tsai, Kaichen Yang, Tsung-Yi Ho, and Yier Jin. Robust adversarial objects against deep learning models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 954–962, 2020. 1, 3

[39] Chun-Chen Tu, Paishun Ting, Pin-Yu Chen, Sijia Liu, Huan Zhang, Jinfeng Yi, Cho-Jui Hsieh, and Shin-Ming Cheng. Autozoom: Autoencoder-based zeroth order optimization method for attacking black-box neural networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 742–749, 2019. 1

[40] Mikaela Angelina Uy, Quang-Hieu Pham, Binh-Son Hua, Thanh Nguyen, and Sai-Kit Yeung. Revisiting point cloud classification: A new benchmark dataset and classification model on real-world data. In *Proceedings of the IEEE/CVF*

[41] Jacob Varley, Chad DeChant, Adam Richardson, Joaquín Ruales, and Peter Allen. Shape completion enabled robotic grasping. In *IEEE international Conference on Intelligent Robots and Systems (IROS)*, pages 2442–2447, 2017. 1

[42] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *ACM Transactions on Graphics (tog)*, 38(5):1–12, 2019. 3, 6, 7

[43] Yuxin Wen, Jiehong Lin, Ke Chen, CL Philip Chen, and Kui Jia. Geometry-aware generation of adversarial point clouds. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020. 1, 2, 3, 5, 6, 7

[44] Matthew Wicker and Marta Kwiatkowska. Robustness of 3d deep learning in an adversarial setting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11767–11775, 2019. 3

[45] Zhirong Wu, Shuran Song, Aditya Khosla, Fisher Yu, Linguang Zhang, Xiaoou Tang, and Jianxiong Xiao. 3d shapenets: A deep representation for volumetric shapes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1912–1920, 2015. 3, 6

[46] Chong Xiang, Charles R Qi, and Bo Li. Generating 3d adversarial point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9136–9144, 2019. 2, 3, 6, 7

[47] Jiancheng Yang, Qiang Zhang, Rongyao Fang, Bingbing Ni, Jinxian Liu, and Qi Tian. Adversarial attack and defense on point sets. *arXiv preprint arXiv:1902.10899*, 2019. 7

[48] Xumin Yu, Lulu Tang, Yongming Rao, Tiejun Huang, Jie Zhou, and Jiwen Lu. Point-bert: Pre-training 3d point cloud transformers with masked point modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19313–19322, 2022. 3

[49] Xiangyu Yue, Bichen Wu, Sanjit A Seshia, Kurt Keutzer, and Alberto L Sangiovanni-Vincentelli. A lidar point cloud generator: from a virtual world to autonomous driving. In *Proceedings of the 2018 ACM on International Conference on Multimedia Retrieval*, pages 458–464, 2018. 1

[50] Cheng Zhang, Haocheng Wan, Xinyi Shen, and Zizhao Wu. Pvt: Point-voxel transformer for point cloud learning. *International Journal of Intelligent Systems*, 37(12):11985–12008, 2022. 3

[51] Qiang Zhang, Jiancheng Yang, Rongyao Fang, Bingbing Ni, Jinxian Liu, and Qi Tian. Adversarial attack and defense on point sets. *arXiv preprint*, 2019. 2, 3

[52] Yu Zhang, Gongbo Liang, Tawfiq Salem, and Nathan Jacobs. Defense-pointnet: Protecting pointnet against adversarial attacks. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 5654–5660, 2019. 3

[53] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip HS Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 16259–16268, 2021. 3

international conference on computer vision, pages 1588–1597, 2019. 3

[54] Yue Zhao, Yuwei Wu, Caihua Chen, and Andrew Lim. On isometry robustness of deep 3d point cloud models under adversarial attacks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1201–1210, 2020. 1, 3

[55] Tianhang Zheng, Changyou Chen, Junsong Yuan, Bo Li, and Kui Ren. Pointcloud saliency maps. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1598–1606, 2019. 3

[56] Boxuan Zhong, He Huang, and Edgar Lobaton. Reliable vision-based grasping target recognition for upper limb prostheses. *IEEE Transactions on Cybernetics*, 2020. 1

[57] Hang Zhou, Dongdong Chen, Jing Liao, Kejiang Chen, Xiaoyi Dong, Kunlin Liu, Weiming Zhang, Gang Hua, and Nenghai Yu. Lg-gan: Label guided adversarial network for flexible targeted attack of point cloud based deep networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10356–10365, 2020. 1, 3, 7

[58] Hang Zhou, Kejiang Chen, Weiming Zhang, Han Fang, Wenbo Zhou, and Nenghai Yu. Dup-net: Denoiser and upsampler network for 3d adversarial point clouds defense. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1961–1970, 2019. 3, 7