Adversarial 3D Virtual Patches using Integrated Gradients

Chengzeng You
Department of Computing
Imperial College London
chengzeng.you19@imperial.ac.uk

Zhongyuan Hau
Department of Computing
Imperial College London
zy.hau17@imperial.ac.uk

Binbin Xu
Robotics Institute
University of Toronto
binbin.xu@utoronto.ca

Soteris Demetriou
Department of Computing
Imperial College London
s.demetriou@imperial.ac.uk

Abstract—**LiDAR sensors are widely used in autonomous vehicles to better perceive the environment. However, prior works have shown that LiDAR signals can be spoofed to hide real objects from 3D object detectors. This study explores the feasibility of reducing the required spoofing area through a novel object-hiding strategy based on virtual patches (VPs). We first manually design VPs (MVPs) and show that VP-focused attacks can achieve similar success rates with prior work but with a fraction of the required spoofing area. Then we design a framework Saliency-LiDAR (*SALL*), which can identify critical regions for LiDAR objects using Integrated Gradients. VPs crafted on critical regions (CVPs) reduce object detection recall by at least 15% compared to our baseline with an approximate 50% reduction in the spoofing area for vehicles of average size.**

I. Introduction

Connected and Autonomous Vehicles (CAVs) leverage various sensing modalities to improve situation awareness. One of those modalities is near-infrared laser light which is leveraged by Light Detection and Ranging (LiDAR) sensors to provide high-precision 3D measurements. These measurements are stored in point clouds, as collections of 3D points. Several CAV manufacturers already leverage LiDARs and there is an array of 3D object detectors which can recognise vehicles, pedestrians and cyclists based on LiDAR measurements. However, prior works have demonstrated the feasibility of LiDAR spoofing attacks which can be controlled to both inject ghost objects [1]–[3] and hide real objects [4]–[7]. These works have progressively improved the adversarial capability in both software and hardware, focusing on increasing the adversarial budget (the number of points that can be reliably spoofed) and the adversary's success rate against 3D object detectors.

Nonetheless, no prior study has focused on reducing the area that the adversary needs to apply their spoofing capability. Prior works [4], [5], [8] considered the area inside a bounding box surrounding the target object or even larger areas as the region of interest. In this work, we are the first to explore whether it is possible to hide 3D objects from detectors by concentrating the attack on a sub-region of the bounding box. This comes with reduced attack complexity and increased stealthiness benefits for the adversary: it reduces the number of signals needed to be reliably spoofed for a successful attack and it reduces the attack's footprint.

Inspired by prior works on adversarial patches in computer vision [9]–[13], we introduce the concept of 3D *virtual patches* (VPs), a region in a point cloud on which an attack strategy can be applied. We then introduce VP-LiDAR, a methodology for analyzing and perturbing measurements in VPs in the digital domain with the goal of bypassing 3D object detection.

We apply VP-LiDAR in two settings: (a) with manually crafted VPs (MVPs) and (b) with critical VPs (CVPs) designed using a novel framework for identifying critical regions in point clouds. In the first setting, we design four MVPs based on common shapes covering different parts of the target object. Applying VP-LiDAR in the second setting is non-trivial as we first need to identify critical regions. Toward this, we design a novel method we call Saliency-LiDAR or *SALL*. *SALL* computes point-level contributions to object detection using Integrated Gradients (IG), an explainability-aware approach [14]. *SALL* can aggregate contributions at the voxel level and across several 3D scenes and objects into a universal saliency map. Based on *SALL*'s universal saliency map, we define three critical VPs (CVPs).

To evaluate VP-LiDAR, we conducted LiDAR relay attacks simulating the physics of LiDAR operations. Our attacks were applied on MVPs and CVPs and empirically evaluated on their ability to hide vehicle objects from popular object detectors. We found that VP-LiDAR with MVPs can achieve similar success rates with an effective object removal attack (ORA-Random) [4] but while attacking a significantly smaller (visually shown) region of interest. We also found that VP-LiDAR attacks with *SALL*-based CVPs are at least 15% more effective than MVP attacks and require focusing the LiDAR relay attack on a CVP area which scales better with the size of target objects (analytically shown) compared to prior work [4].

II. Background and Related Work

LiDAR Spoofing Attacks. LiDAR measurements can be spoofed by replaying LiDAR pulses to create fake points in the sensed environment. Such an attack is challenging for the LiDAR system to recognize as it doesn't require any physical contact with the LiDAR sensor or interference with the processing of sensor measurements. To perform realistic attacks, researchers have been improving the hardware and software of LiDAR spoofers [2], [5], [6], [15]. A common attack strategy is to capture LiDAR signals from the victim LiDAR, then add a time delay and fire fake laser beams back to the victim LiDAR. Fake points are shown to be reliably injected to fool 3D object detectors to output erroneous predictions. As a result, real objects can be hidden while ghost objects can be injected. Real object hiding is regarded as a more dangerous type of attack than ghost object injection, as it is more likely to cause fatal collisions. Object hiding attacks can be achieved through synchronized methods [2], [3], [6], [7] and asynchronized methods such as relay attacks [1], saturating attacks [15] and high-frequency removal attacks [5]. In our work, we consider an adversary with the ability of mounting relay attacks to hide real objects from 3D object detectors.

Adversarial Patches. Our approach of using virtual patches in point clouds to reduce the spoofing region of interest is inspired by prior works on adversarial patches in the 2D image domain. Previous studies have presented strong adversarial patches [9] for several downstream tasks such as person detection [10], face recognition [11], [12] and semantic segmentation [13]. There has been very little exploration of adversarial patches applied in the 3D domain. Chen et al. [16] leveraged the information of 3D adversaries and added perturbations on 2D planes managing to attack optical image sensors. Xiao et al. [17] generated adversarial meshes successfully misleading classifiers and 2D object detectors.

Critical Points. One of the main challenges we tackled in this work is identifying critical regions. It has been shown that critical points [18] contribute to features of max-pooling layers and summarize skeleton shapes of input objects [14]. Based on critical points, researchers further studied the model robustness by perturbing or dropping critical point set identified through monitoring the max-pooling layer or accumulating losses of gradients [19]–[21]. However, capturing the output of the max-pooling layer struggled to identify discrepancies between key points, and simultaneously, saliency maps based on raw gradients have been proven to be defective [22], [23]. To overcome these issues, Tan et al. [14] introduced Integrated Gradient (IG) [24] which are oriented on generating saliency maps of inputs by calculating gradients during propagation, to investigate the sensitivity of model robustness to the critical point sets and successfully fooling target classifiers with very few point perturbations. However, this study identified critical points specific to object point clouds without further summarizing richer 3D scenes. As a result, for every instance of an object, the attacker needs to run a separate iterative optimization process. In our work, we improve on the Integrated Gradient (IG) [24] approach in two main ways. First, we adapt the proposed framework to the task of 3D object detection in autonomous driving. Secondly, we integrate IG into an end-to-end framework (*SALL*) which aggregates the saliency maps across several 3D scenes and objects to derive a *universal* saliency map across all instances of an object type which we can use to construct critical virtual patches (CVPs).

III. Threat Model

We perform all our attacks digitally simulating an adversary ($\mathcal{A}$) which we assume has the ability to physically realize the attacks. We based our simulation assumptions on prior works whenever possible. In particular, we assume $\mathcal{A}$ is equipped with a state-of-the-art LiDAR spoofer capable of spoofing LiDAR return signals [1], [2], [5], [7], [15], [25].

$\mathcal{A}$ can use her spoofing capability to displace a 3D point. The displacement can be achieved along the victim LiDAR's ray direction, such that the fake point can appear either further [4], [25] or nearer [15] than the genuine point relative to the victim vehicle, within a range of 4m [-2m, 2m] and at the granularity of 1m. $\mathcal{A}$ should also be able to perform the displacements on a number of points (e.g. 1–200) and reliably as the victim vehicle moves [2], [5], [25]. Similarly with Hau et al. [4], we assume $\mathcal{A}$ can predict the bounding boxes of the victim's 3D object detector but does not have knowledge of the internals of the victim's 3D object detector. To achieve this, $\mathcal{A}$ detects target objects and transforms their 3D coordinates according to its position's relativity to the victim LiDAR.

The goal of $\mathcal{A}$ is to leverage the above capabilities to lower the confidence level of 3D object detectors causing them to miss real objects.

IV. Virtual Patches and VP-LiDAR Methodology

We first define virtual 3D patches (VPs) and then introduce our framework (VP-LiDAR) for simulating LiDAR spoofing attacks using VPs.

Virtual Patches. A *Virtual 3D Patch* or simply *VP* is a subspace within a 3D object's point cloud on which $\mathcal{A}$ can apply her perturbations. More formally, a 3D scene is a point cloud $S \in \mathbb{R}^{n \times d}$, where n is the number of 3D points in the scene. In each scene, there can be a collection of bounding boxes, one for each detected object. A bounding box B, is $B \in \mathbb{R}^{n_b \times d}$, where $n_b < n$. Then, a virtual patch can be defined as a sub-region $V \in \mathbb{R}^{n_v \times d}$, where $n_v \leq n_b < n$. The goal of $\mathcal{A}$ is to come up with a perturbed V', $V' \in \mathbb{R}^{n_{v'} \times d}$ where $n_{v'} \leq n_v$ because some points might be displaced or shifted outside the VP area.

VP-LiDAR. VP-LiDAR is a 3D adversarial VP analysis framework that aims to facilitate experimentation with VP-based attack strategies and defenses. VP-LiDAR consists of five phases, taking in raw LiDAR point cloud ($\mathcal{S}$) of the scene, performing perturbation of target objects, and producing the adversarial point cloud (S') as its output.

Phase 1: Extraction. VP-LiDAR detects objects from $\mathcal{S}$. Then it separates S into background points G ($G \in \mathbb{R}^{n_g \times d}$) and a set of target point clouds $T = \{T^1, T^2, ..., T^m\}$. There is exactly one target point cloud T^i for each of the m detected objects.

Phase 2: 2D Indexing. Each target point cloud T^i is further discretized. We use the approach by Lang et al. [26] to find the corresponding indices of each point in pillar format. 2D indexing is more efficient than voxelisation methods, because it does not need to convert points to voxels. Also, the corresponding voxel size is customized and can be set to near point level where each voxel only contains a few points or even one point.

Phase 3: Virtual Patch Simulation. Based on the indices, we can apply a 2D virtual patch V. Virtual patches can be defined manually (see § V) or using our *SALL* method (see § VI).

Phase 4: Perturbation. Different selection strategies can be applied to select points from $\mathcal{V}$ under an adversarial point budget. For example, VP-LiDAR supports the random selection strategy similarly to ORA-Random [4] which randomly selects points within a target bounding box. VP-LiDAR also supports selecting points according to their criticality - we can calculate such criticalities using our *SALL* method (§ VI). Due to VP-LiDAR's modular architecture, other novel strategies can be easily incorporated.

To obey the physics of LiDAR, VP-LiDAR shifts points in accordance with the rays that the LiDAR points fall on. Each point in the cartesian coordinate system is first transformed to the spherical coordinates with the radius $\mathcal{R}$ and the firing angle relative to the LiDAR origin. Then a distance R_d is added to the radius ($\mathcal{R}$). The shifted radius $\hat{\mathcal{R}} = \mathcal{R} + R_d$ along with the firing angle is then transformed back to the cartesian coordinate. The result is a perturbed virtual patch V' with $n_{v'}$ perturbed points.

Phase 5: Merge. All V's are then merged with G to output the final adversarial 3D LiDAR scene $S' = G \bigoplus V'$. S' is in the same format as the original LiDAR scene S, and can be fed into any LiDAR-based detectors for evaluations.

V. VP-LiDAR with Manual Virtual Patches

To study the feasibility of using virtual patches to reduce the spoofing area of 3D objects, we first manually defined patches, and used VP-LiDAR to evaluate their effectiveness.

A. Manual Virtual Patches

We designed four MVPs. All MVPs were defined based on the bottom surface (`Rec`) of the target object.

- *Edges*. This patch is defined as 4 edges of `Rec`. The thickness of each edge is 3 voxels.
- *Nearest-Corner*. This patch is defined as `Rec`'s corner nearest to the LiDAR unit of the ego vehicle. The dimension of the patch is 8 voxels $\times$ 8 voxels.
- *Center*. This patch shares the same center with `Rec` but in a smaller size. We define the patch width as $3/4$ of `Rec`'s width and length as $3/4$ of `Rec`'s length.
- *X*. This patch contains all voxels around the diagonal lines of `Rec`. The maximum distance from the voxel to either diagonal line is set to 1.5 voxels.

B. Experimental Setup.

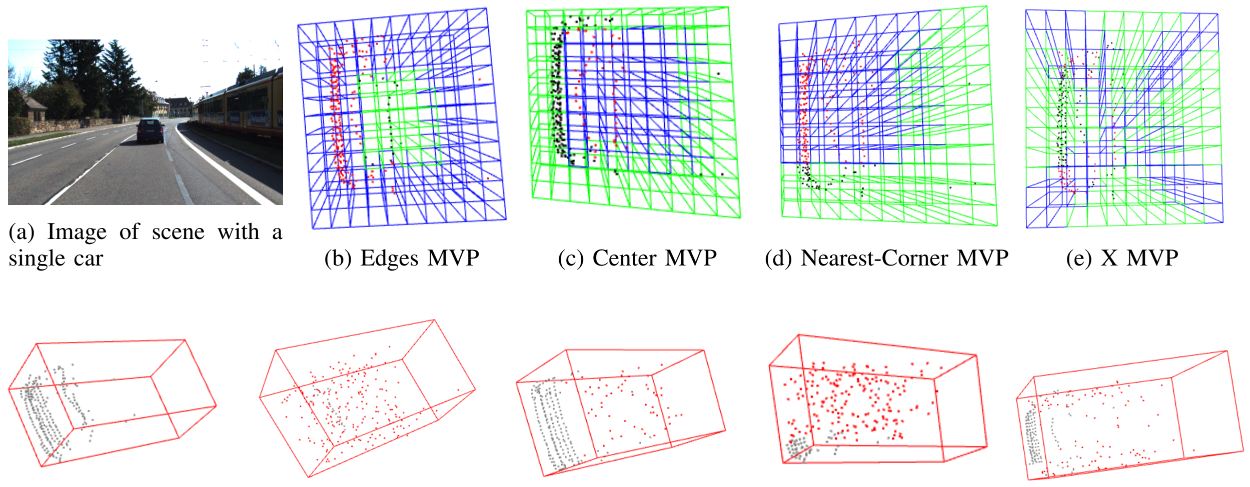
For our dataset, we randomly selected 300 autonomous driving scenes from the KITTI dataset [27] and for our evaluation metric, we used the Attack Success Rate (ASR) as the ratio of the number of hidden objects out of all targeted objects. We used ground-truth labels from KITTI as the object detection results. In practice, $\mathcal{A}$ can choose any state-of-the-art 3D object detector to obtain detection results in the target scene. We focused on *Car* objects in the front-near region which refers to the region directly in front of the ego-vehicle up to a distance of 10m from the LiDAR unit. *Car* objects are more dense than *Cyclist* and *Pedestrian* objects and are more challenging to attack. For VP-LiDAR's 2D Indexing, we set the corresponding voxel size as per Pointpillars [26] to $0.16m \times 0.16m$. For the VP simulation, we used the 4 MVPs we defined above. For point shifting, we configured VP-LiDAR to select target points within an MVP using a random strategy as ORA-Random [4] with various point budgets from 1 to 400 (step size = 40) for each object. Lastly, after point shifting, all perturbed data was fed into a target model. For our evaluation, we chose PointPillars [26] which is used in an industry-grade AD system Baidu Apollo 6.0 [28].

C. Results

MVP simulations and spoofing area. To better understand how well VP-LiDAR can simulate attacks on a VP, we use a visualization approach. We chose a *Car* object as the target (Figure 1a and 1f). MVPs are applied on 3D point clouds as shown in Figures 1b$\sim$1e while perturbed objects are shown in Figure 1g$\sim$1j. Our results show that VP-LiDAR can precisely select all points inside the VP and shift points according to the physics of LiDAR operation. It is also evident from Figures 1b$\sim$1e that MVPs occupied a small fraction of the entire region of interest.

Effectiveness of VP-LiDAR Attack with MVPs. As shown in Figure 2, the best ASR VP-LiDAR can achieve is 91.67% when performing *X Shifting* with a point budget of 400 points. When shifting more than 170 points per bounding box, the effectiveness of *X-Shifting* $>$ *Center-Shifting* $>$ *Edge-Shifting* $>$ *Corner-Shifting*. On the other hand, *X-Shifting* shares a similar trend with *Center-Shifting* if the attacker shifts less than 170 points per object. Notably, shifting only 1 point per object can also achieve an ASR of 3% using *Edge-Shifting* and *Corner-Shifting*.

Comparison with ORA-Random [4]. We compare VP-LiDAR attack with MVPs, with ORA-Random [4]. ORA-Random is a model-level object hiding attack. Note that recently Cao et al [7] introduced a new object hiding attack based on LiDAR spoofing which relaxes some of the assumptions of ORA such as the dependency on the bounding box calculations and increases the adversarial budget of the adversary. However, for this work, we selected ORA-Random for its simplicity of implementation and high performance. For this experiment, we selected 3681 autonomous driving scenes from the KITTI dataset [27] containing target *Car* objects in front of the ego-vehicle up to a distance of 20m. For this experiment, we evaluate the attacks on a pretrained PointRCNN [29] model which was the model targetted by ORA-Random by Hau et al [4]. We performed object hiding attacks on front-near *Car* objects using (a) our *X-Shifting* which was our best performing MVP and (b) ORA-Random. We also parameterize the adversarial point budget and evaluate the attacks under 10, 40, 60, 100 and 200 points. The effectiveness of both methods was measured using *Recall* which is the ratio of the number of all predicted objects out of all targeted objects. Recall is a


(a) Image of scene with a single car (b) Edges MVP (c) Center MVP (d) Nearest-Corner MVP (e) X MVP

(f) Benign Car Object (g) Edge Shifting (h) Center Shifting (i) Corner Shifting (j) X Shifting

Fig. 1: VP-LiDAR Visualization. Top row: manual virtual patches. Blue voxels and red points are selected while green voxels and black points are not selected. Bottom row: red points denote the shifted points while grey points remain unchanged.

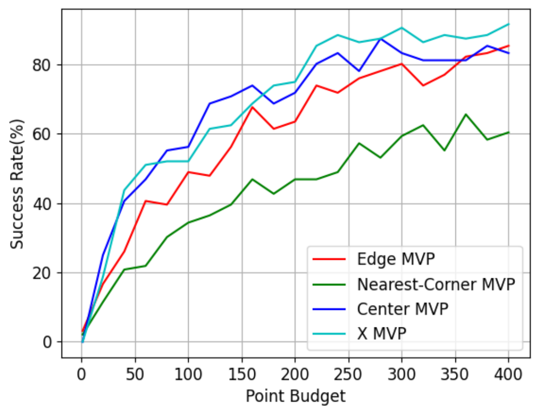


Fig. 2: ASR of MVPs for different point budgets.

metric that captures false negatives and therefore can give us an indication on how many objects are missed.

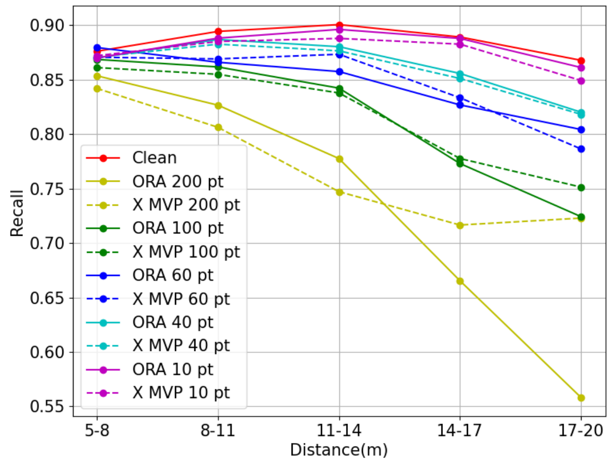


Fig. 3: Comparison of X MVP with ORA-Random.

As shown in Figure 3, *X-Shifting* performs similarly with *ORA-Random*, even though it attacks points within a much smaller area. For objects nearer than 15 meters, *X-Shifting* can achieve even better performance (marginally) than ORA-Random for all budgets. For objects further away which are sparser, a random strategy on the entire area might still be the better option, although attacks on those objects are less impactful. Overall, we can see that MVPs can be effective in attacking near-front objects with a fraction of the spoofing area compared to ORA-Random. The reason might be that some MVPs happen to contain certain regions that are critical to the object detector. In the following part, we propose a new approach to identify critical regions and help with the design of even smaller critical VPs (CVPs).

VI. Saliency-LiDAR and Critical Virtual Patches

A. *Saliency-LiDAR Method*

To identify critical regions, we develop a method we call Saliency-LiDAR (*SALL*). *SALL*, inspired by Tan et al [14] leverages the Integrated Gradient approach to generate saliency maps of inputs. *SALL* adapts IG the task of object detection in autonmous driving scenarios, and can aggregate saliency maps across instances of an object type within and across scenes to generate a universal saliency map. *SALL*'s overall architecture is shown on Figure 4 and below we explain each component.

Preprocessing. *SALL* takes a raw 3D scene S as input. Before IG computation, it preprocesses the scene through an extraction module ($\mathcal{E}$) which identifies regions of interest R, one per target object. It then extracts target objects T and background points G. Subsequently, the target objects T are fed into the IG component to compute point-level contributions.

Integrated Gradient Computation. For each IG step, points in a T^i are perturbed by a perturbation module ($\mathcal{P}$) which works similarly to VP-LiDAR's $\mathcal{P}$(§ V) and outputs $T^{i'}$. All $T^{i'}$ in T' are then merged with the background points (G) to produce the perturbed 3D scene (S'): $S' = T' \bigoplus G$. S' is used for the gradient computation. It passes through a

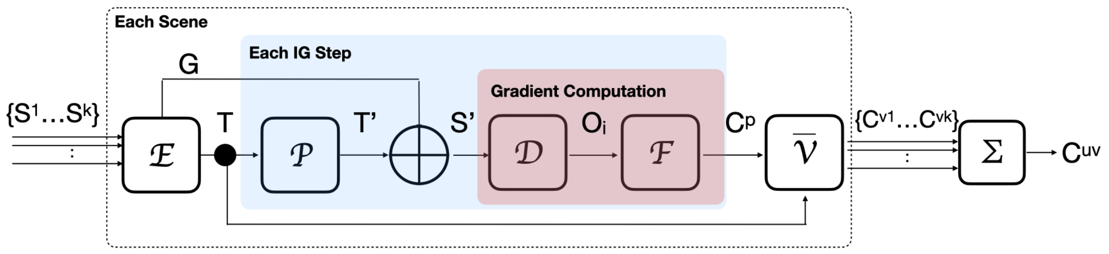


Fig. 4: Overview of Universal Saliency Map Generation for LiDAR Objects with *SALL*.

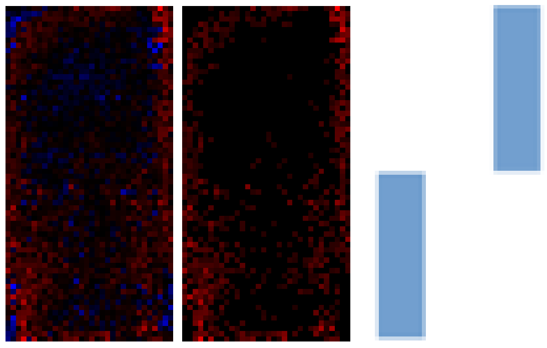
3D object detector ($\mathcal{D}$) which outputs a set of logits O^i for each target object i. To focus on target objects instead of the whole LiDAR scene, Intersection of Unions(IOUs) between the focus regions R and predicted bounding boxes are first computed to identify the best predictions. Gradients of the best predictions are saved while other gradients are filtered out in the filter module ($\mathcal{F}$). Finally, a point-level contribution map C_i^p is generated per target object. Lastly, an integrator function integrates all C_i^p across all IG steps, to produce a point-level saliency or contribution map C^p for objects in a single scene.

Adaptive Indexing ($\tilde{\mathcal{V}}$). Since R regions have different dimensions and rotations in different LiDAR scenes, to generate a universal saliency map, point-level saliency maps need to be downsampled to pixel-level with the same size. To achieve that, each extracted target point cloud T^i is first converted from LiDAR coordinates to bounding box coordinates. Then, given the target size of the universal saliency map (2D-pixel image), $\tilde{\mathcal{V}}$ adaptively computes the voxel size for each target object based on R's dimension. After that, indices of each point in T^i can be computed. According to the point-level saliency map C^p, the contribution of each voxel is summed up to generate a 2D-pixel matrix C^v in which each element indicates the contributions of each voxel.

Aggregation Across Scenes ($\sum$). For each scene S, we generate a contribution matrix C^v. C^vs are then aggregated across all k scenes by simple matrix additions to generate the universal saliency map C^{uv} for the target object type. Figure 5a shows the saliency map for *Car* objects at 5-8m. Most positive pixels fall in edges with some less critical and negative pixels falling in the center of the bounding box. This is true for LiDAR objects where most points appear on the surfaces.

B. Critical Virtual Patches

Constructing Adversarial Critical Virtual Patches. With the guidance of the universal saliency map, $\mathcal{A}$ can generate adversarial CVPs by perturbing points in voxels with top contribution values. As a proof of concept, we constructed three CVPs: *Top_30* (Figure 5b), *Half-Edges* (Figure 5c) and *Critical-X*. *Top_30* uses only voxels with top 30% positive contributions of the universal saliency map. *Half-Edges* is designed to capture areas which include the most contributing voxels. *Critical-X* is a more space-efficient version of the well-performing *X-Shifting* MVP (§ V) which contains all voxels around the diagonal lines of the bounding box.



(a) Saliency Map (b) Top_30 (c) Half Edges

Fig. 5: Saliency Map Visualization. Red pixels denote positive contribution values while blue pixels denote negative contribution values.

VII. Evaluation of Critical Virtual Patches

For our experiments, we selected 500 autonomous driving scenes from the KITTI dataset [27] with 400 scenes for generating saliency maps and 100 scenes for testing the effectiveness of CVPs. We used the ground-truth labels of *Car* objects in front of the ego-vehicle between 5m and 8m as the region of interest. For integrated gradient computation, we set IG steps = 25. As for the *adaptive indexing* module, we set the output matrix to a fixed size of 64 $voxels \times 32\ voxels$. The corresponding voxel size of each target object is around $0.05m \times 0.05m$ on average. In terms of the target model for 3D object detection, we chose PointPillars [26].

A. Spoofing Area Analysis

Let the size of the target object and a voxel be determined by $(l_{tar}, w_{tar}, h_{tar})$ and (l_v, w_v) respectively, where we use l, w and h to indicate width, length and height of an area. Let also α and β be scale factors for l_{tar} and w_{tar} to control the patch thickness. For complex patches such as *Critical-X*, we calculate the spoofing area by subtracting 2 pairs of equilateral triangles from the whole area as shown in Equation 2. Given these, we calculate the number of pillars needed for each patch as shown in Equations 1–4.

$$Area_{whole} = \frac{l_{tar} * w_{tar}}{l_v * w_v} \tag{1}$$

$$Area_{critical\_x} = (1 - \frac{(0.5-\alpha)(1-2\alpha)(1-\beta)}{(1-\alpha)} - \frac{(0.5-\beta)(1-2\beta)(1-\alpha)}{(1-\beta)}) * \frac{l_{tar} * w_{tar}}{l_v * w_v} \quad (2)$$

$$Area_{half\_edges} = \frac{l_{tar} * \beta * w_{tar}}{l_v * w_v} \quad (3)$$

$$Area_{top\_n} = n\% * Area_{whole} \quad (4)$$

Assuming $h_{tar} = 1.5m$, $w_{tar} = S$, $l_{tar} = 2S$, $l_v = w_v = 0.05$, and setting $\alpha = 0.1$, and $\beta = 0.2$, we plot the number of pillars needed for different sizes (S) of the target object (Figure 6). As a baseline for comparison, we calculate the entire size of the object (e.g. its bounding box) which we call *Whole Area*. *Whole Area* corresponds to approaches like ORA-Random which target the entire object area. We observe that CVPs can drastically reduce the spoofing areas as the object size increases compared to the *Whole Area* approach. If the average vehicle length is 5m, then CVPs can reduce the spoofing areas by at least 50%.

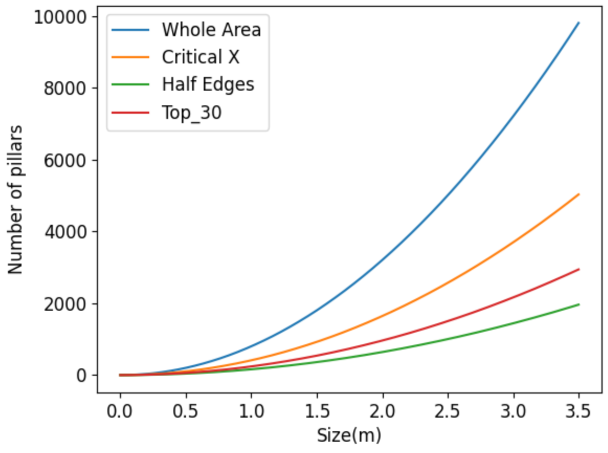


Fig. 6: Spoofing Areas of VPs.

B. Effectiveness of CVPs

Setup. We used VP-LiDAR to generate adversarial point clouds and attack the target model on the test dataset. For each CVP, we apply 2 different point selection strategies while perturbing: *Random Selection* and *Critical First*. With *Random Selection*, given a point-perturbation budget, $\mathcal{A}$ randomly selects points among all point candidates. With *Critical First*, $\mathcal{A}$ selects the most critical points inside a CVP according to the *SALL*-generated universal saliency map of the target object.

We compare our CVPs using the above strategies with ORA-Random [4]. We also design an optimized version of ORA-Random (which we call *Whole Area*) for which points can be shifted between -2m to 2m—ORA-Random uses shifting distances between 0m to 2m but prior work [15] has shown $\mathcal{A}$ can relay LiDAR signals to inject points both nearer and further than the genuine location. Also, in contrast with ORA-Random, *Whole Area* can be configured to use any of the point selection strategies above. For the rest of the CVPs, VP-LiDAR is also configured to shift points between -2m to 2m.

		Point Budget				
Selection	**Patch**	**200**	**150**	**100**	**50**	**10**
Random	ORA-Random [4]	74.1%	68.5%	56.5%	42.6%	17.6%
	Whole Area	94.4%	88.0%	76.9%	56.5%	27.8%
	Critical X	91.7%	90.7%	75.0%	61.1%	25.9%
	Half Edges	92.6%	90.7%	84.3%	57.4%	24.1%
	Top_30	91.7%	88.0%	74.1%	65.7%	24.1%
Critical	Whole Area	91.7%	86.1%	82.4%	50.9%	18.5%
	Critical X	90.7%	85.2%	77.8%	54.6%	21.3%
	Half Edges	91.7%	88.0%	78.7%	55.6%	25.9%
	Top_30	93.5%	87.0%	83.3%	55.6%	22.2%

TABLE I: Effectiveness comparison of CVPs in hiding cars at 5-8m for different point budgets.

Detectors	Clean	Half Edges	Top_30
PV-RCNN	98.2%	66.7% (↓32.1%)	62.0% (↓36.9%)
SECOND	98.2%	60.2% (↓38.7%)	61.1% (↓37.8%)
Voxel R-CNN	98.2%	63.9% (↓34.9%)	70.4% (↓28.3%)

TABLE II: Recall of Different Detectors in Benign Scenarios (Clean) compared to when exposed to LiDAR spoofing attacks using CVPs (Half Edges, Top_30). ↓ indicates the percentage decrease compared to the benign scenarios.

Results. In Table I, we show the ASRs of different CVPs using different selection strategies under different point budgets. Compared with the *X-Shifting* MVP (see Figure 2), the improved *Critical-X* demonstrates over 15% ASR improvements under all point budgets. This indicates that CVPs are more effective than MVPs. Moreover, all CVPs and our optimized *Whole Area* attack can achieve significantly better performance compared to ORA-Random (15%–20% ASR improvements when shifting more than 100 points). At the same time, CVPs only use a significantly reduced spoofing area compared to *Whole Area*.

Critical First point selection strategies did not perform much better than *Random Selection*. The reason might be that within the CVPs we already capture the most critical points. We plan to analyze this further in future work.

C. Transferability

To evaluate whether our CVPs are effective against other detectors, we selected one point-based detector (PV-RCNN) and two voxel-based detectors (SECOND(one stage detector) and Voxel-RCNN(two stage detector)) as shown in Table II. While detecting objects in 5-8m, all 3 detectors achieved the same recall of 98.2% with undetected objects tending to be the same or around the same location. Although our target objects are very dense (one object normally contains thousands of points), using CVPs under the budget of 200 points, the recall of 3 detectors dropped from 98.2% to 61.1%–70.4% (with a decrease between 28.3% and 38.7%). For objects at larger distances or smaller objects (such as pedestrians and cyclists) that are sparser and more vulnerable, this approach would likely cause greater drops in recall.

VIII. Discussion & Future Work

We defined 3D virtual patches (VPs) and proposed a modular analysis framework (VP-LiDAR) which leverages VPs to digitally test 3D object-hiding attacks. We first demonstrate the potential of VPs by defining manual VPs (MVPs) and showing that they can achieve similar attack success rates compared to strong object-hiding attacks while reducing the spoofing area. We then introduce *SALL*, a method that uses integrated gradients to generate universal saliency maps for target objects and show how we can use such maps to construct critical virtual patches (CVPs). Our evaluations showed that with a point budget of 200, one can leverage CVPs to attack state-of-the-art detectors with more than a 90% success rate. This is 15-20% better compared to ORA-Random [4] while it requires targeting only a fraction of the spoofing area.

In future work, we plan to explore the effectiveness of CVP-based attacks against other object types such as pedestrians and cyclists. We expect our attacks to be more effective against these since they exhibit higher point sparsity [30]. We will also test the robustness of our attack method against point and object-level defenses such as CARLO [25], Shadow Catcher [30], 3D-TC2 [8] and ADoPT [31]. Lastly, we note that our spoofing capability simulations are based on prior works' findings on the physical capability of LiDAR spoofers. We leave it to future work to verify the feasibility of physically realizing VP-LiDAR attacks with MVPs and CVPs.

References

[1] J. Petit, B. Stottelaar, M. Feiri, and F. Kargl, "Remote attacks on automated vehicles sensors: Experiments on camera and lidar," *Black Hat Europe*, vol. 11, p. 2015, 2015.

[2] 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 *29th USENIX Security Symposium (USENIX Security 20)*, 2020, pp. 877–894.

[3] R. S. Hallyburton, Y. Liu, Y. Cao, Z. M. Mao, and M. Pajic, "Security analysis of camera-lidar fusion against black-box attacks on autonomous vehicles," in *31st USENIX Security Symposium (USENIX Security 22)*, 2022, pp. 1903–1920.

[4] Z. Hau, T. Kenneth, S. Demetriou, and E. C. Lupu, "Object removal attacks on lidar-based 3d object detectors," in *Workshop on Automotive and Autonomous Vehicle Security (AutoSec)*, vol. 2021, 2021, p. 25.

[5] T. Sato, Y. Hayakawa, R. Suzuki, Y. Shiiki, K. Yoshioka, and Q. A. Chen, "Revisiting lidar spoofing attack capabilities against object detection: Improvements, measurement, and new attack," *arXiv preprint arXiv:2303.10555*, 2023.

[6] Z. Jin, X. Ji, Y. Cheng, B. Yang, C. Yan, and W. Xu, "Pla-lidar: Physical laser attacks against lidar-based 3d object detection in autonomous vehicle," in *2023 IEEE Symposium on Security and Privacy (SP)*. IEEE, 2023, pp. 1822–1839.

[7] Y. Cao, S. H. Bhupathiraju, P. Naghavi, T. Sugawara, Z. M. Mao, and S. Rampazzi, "You can't see me: Physical removal attacks on {LiDAR-based} autonomous vehicles driving frameworks," in *32nd USENIX Security Symposium (USENIX Security 23)*, 2023, pp. 2993–3010.

[8] C. You, Z. Hau, and S. Demetriou, "Temporal consistency checks to detect lidar spoofing attacks on autonomous vehicle perception," in *Proceedings of the 1st Workshop on Security and Privacy for Mobile AI*, 2021, pp. 13–18.

[9] T. B. Brown, D. Mané, A. Roy, M. Abadi, and J. Gilmer, "Adversarial patch," *arXiv preprint arXiv:1712.09665*, 2017.

[10] S. Thys, W. Van Ranst, and T. Goedemé, "Fooling automated surveillance cameras: adversarial patches to attack person detection," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, 2019, pp. 0–0.

[11] X. Yang, F. Wei, H. Zhang, and J. Zhu, "Design and interpretation of universal adversarial patches in face detection," in *European Conference on Computer Vision*. Springer, 2020, pp. 174–191.

[12] Z. Xiao, X. Gao, C. Fu, Y. Dong, W. Gao, X. Zhang, J. Zhou, and J. Zhu, "Improving transferability of adversarial patches on face recognition with generative models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 11 845–11 854.

[13] F. Nesti, G. Rossolini, S. Nair, A. Biondi, and G. Buttazzo, "Evaluating the robustness of semantic segmentation for autonomous driving against real-world adversarial patch attacks," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, pp. 2280–2289.

[14] H. Tan and H. Kotthaus, "Explainability-aware one point attack for point cloud neural networks," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023, pp. 4581–4590.

[15] H. Shin, D. Kim, Y. Kwon, and Y. Kim, "Illusion and dazzle: Adversarial optical channel exploits against lidars for automotive applications," in *International Conference on Cryptographic Hardware and Embedded Systems*. Springer, 2017, pp. 445–467.

[16] C. Chen and T. Huang, "Camdar-adv: Generating adversarial patches on 3d object," *International Journal of Intelligent Systems*, vol. 36, no. 3, pp. 1441–1453, 2021.

[17] C. Xiao, D. Yang, B. Li, J. Deng, and M. Liu, "Meshadv: Adversarial meshes for visual recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 6898–6907.

[18] C. R. Qi, H. Su, K. Mo, and L. 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*, 2017, pp. 652–660.

[19] J. Kim, B.-S. Hua, T. Nguyen, and S.-K. Yeung, "Minimal adversarial examples for deep learning on 3d point clouds," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 7797–7806.

[20] J. Yang, Q. Zhang, R. Fang, B. Ni, J. Liu, and Q. Tian, "Adversarial attack and defense on point sets," *arXiv preprint arXiv:1902.10899*, 2019.

[21] T. Zheng, C. Chen, J. Yuan, B. Li, and K. Ren, "Pointcloud saliency maps," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 1598–1606.

[22] J. Adebayo, J. Gilmer, M. Muelly, I. Goodfellow, M. Hardt, and B. Kim, "Sanity checks for saliency maps," *Advances in neural information processing systems*, vol. 31, 2018.

[23] M. Sundararajan, A. Taly, and Q. Yan, "Gradients of counterfactuals," *arXiv preprint arXiv:1611.02639*, 2016.

[24] ——, "Axiomatic attribution for deep networks," in *International conference on machine learning*. PMLR, 2017, pp. 3319–3328.

[25] Y. , C. Xiao, B. Cyr, Y. Zhou, W. Park, S. Rampazzi, Q. A. Chen, K. Fu, and Z. M. Mao, "Adversarial sensor attack on lidar-based perception in autonomous driving," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, 2019, pp. 2267–2281.

[26] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "Pointpillars: Fast encoders for object detection from point clouds," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 697–12 705.

[27] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[28] "Baidu apollo," http://apollo.auto, 2020.

[29] S. Shi, X. Wang, and H. Li, "Pointrcnn: 3d object proposal generation and detection from point cloud," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 770–779.

[30] Z. Hau, S. Demetriou, L. Muñoz-González, and E. C. Lupu, "Shadow-catcher: Looking into shadows to detect ghost objects in autonomous vehicle 3d sensing," in *Computer Security–ESORICS 2021: 26th European Symposium on Research in Computer Security, Darmstadt, Germany, October 4–8, 2021, Proceedings, Part I 26*. Springer, 2021, pp. 691–711.

[31] M. Cho, Y. Cao, Z. Zhou, and Z. M. Mao, "Adopt: Lidar spoofing attack detection based on point-level temporal consistency," 2023.