Poster: 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

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 and hide real objects. 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 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, 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. *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) 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.

II. 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. 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 or using our *SALL* method (see § III).

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 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 (§ III). 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.

III. 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 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. Below we describe *SALL*'s overall architecture and 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 $\mathcal{P}$ 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 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^p_i is generated per target object. Lastly, an integrator function integrates all C^p_i 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.

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.

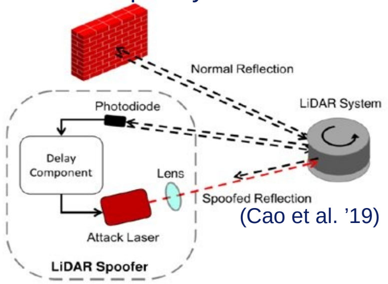
Imperial College London

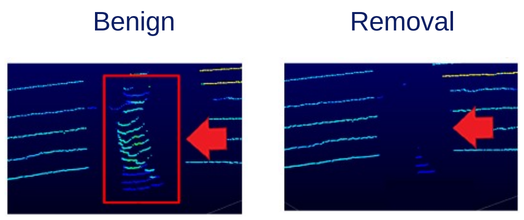
Adversarial 3D Virtual Patches using Integrated Gradients

Chengzeng You
cy19@ic.ac.uk

1. Spoofing LiDAR Sensors

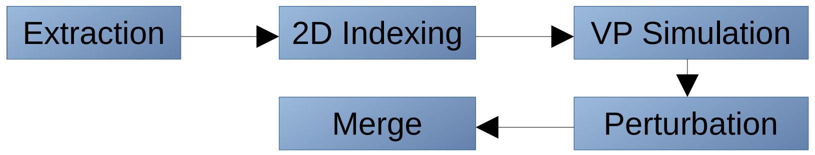
LiDAR signals can be spoofed to hide real objects from autonomous vehicle's perception causing fatal collisions. This study takes the first step to reduce the required spoofing area based on virtual patches (VPs) which can further reduce attack complexity and increase stealthiness.





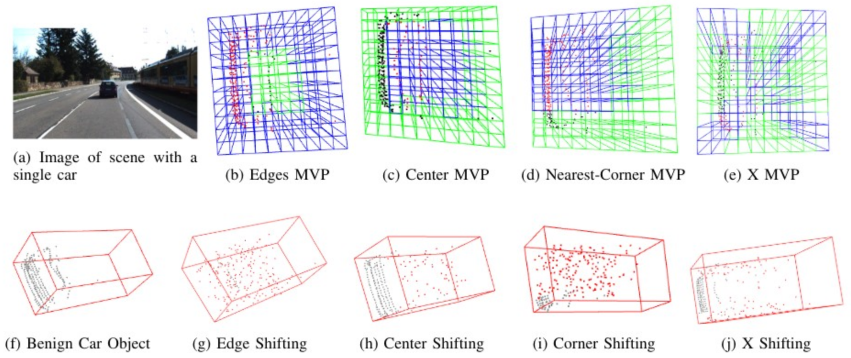
2. LiDAR Spoofing Attacks using Virtual Patches

We introduce VP-LiDAR for simulating LiDAR spoofing attacks using VPs. 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 as shown below:



3. VP-LiDAR with Manual Virtural Patches

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.


(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.

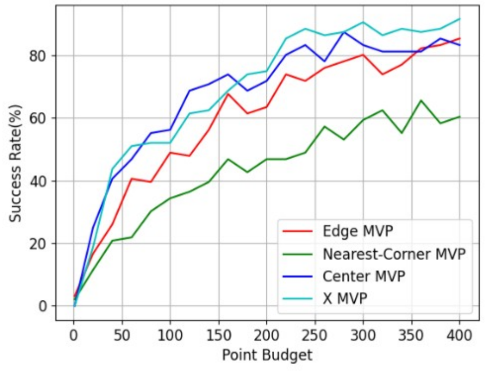


Fig. 2: ASR of MVPs for different point budgets.

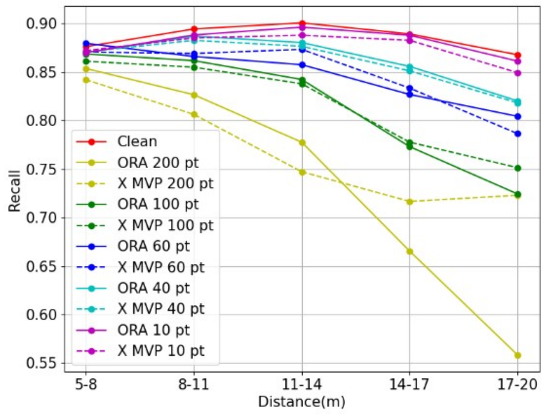
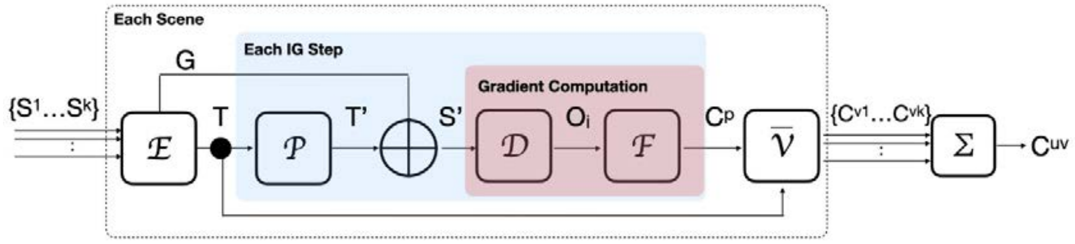


Fig. 3: Comparison of X MVP with ORA-Random.

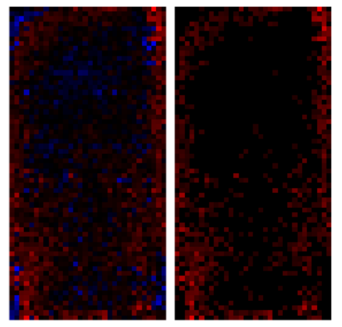
4. Identify Critical Regions for LiDAR Objects

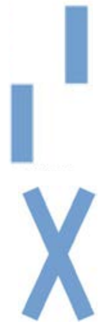
We design a framework Saliency-LiDAR (SALL), which can identify critical regions for LiDAR objects. SALL can aggregate saliency maps across instances of an object type within and across scenes to generate a universal saliency map.

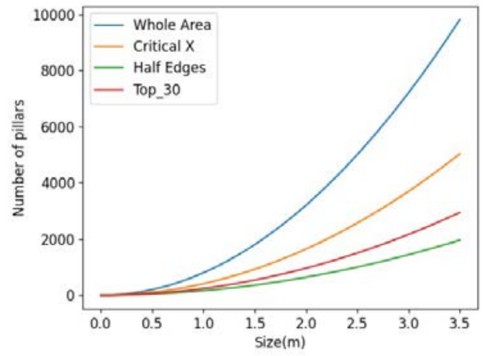


5. Evaluation of Critical Virtual Patches

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. CVPs are more effective than current attacks and require focusing the LiDAR spoofing attack on a small area which scales better with the size of target objects.






Number of pillars
Size(m)
Whole Area
Critical X
Half Edges
Top_30