Imperial College London

APSS Seminar 1

# Adversarial Sensor Attack on LIDAR-based Perception

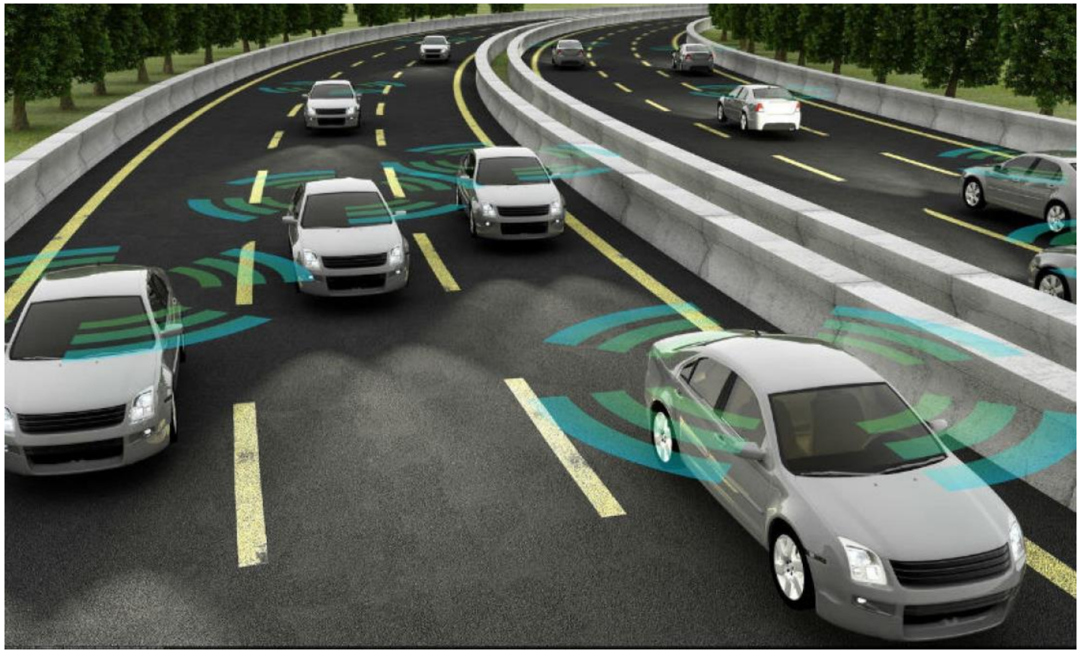
In Autonomous Driving

(Charles) Chengzeng You
Connected and autonomous vehicles
Department of computing
Email: cy19@ic.ac.uk

Imperial College London

## 1. Introduction

Autonomous vehicles rely on perception, which leverages sensors like cameras and LiDARs (Light Detection and Ranging) to understand the surrounding driving environment.



## 1. Introduction

### 1.1 Sensor-level attacks

- camera blinding [42], physical-world camera attacks [28, 29], trojan attacks on the neural networks for AV camera input [37].
- A few works demonstrated the feasibility of injecting spoofed points into the sensor input from the LiDAR[42, 44].

**The first study to explore the security of LiDAR-based perception in AV settings**

Imperial College London

# 1. Introduction

## 1.2 Existing spoofing techniques

- Machine learning output can be maliciously altered by carefully-crafted perturbations to the input [18, 20, 29, 41, 57].
- Formulating the attack task as an optimization problem has been proven effective in previous machine learning security studies [21, 24, 26, 50, 51, 53].

**Strategically controlling the spoofed points to fool the machine learning model**

Imperial College London

# 1. Introduction

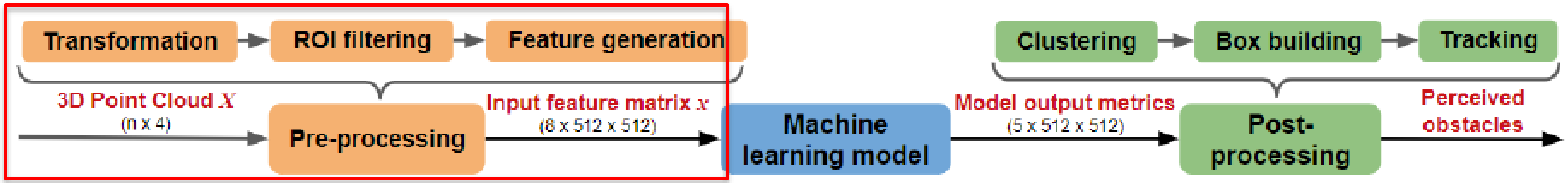
## 1.3 Contributions

- The first security study of LiDAR-based perception for AV systems.
- Two methodology-level contributions: conduct experiments to analyze the LiDAR spoofing attack capability and design a global spatial transformation-based method to model such capability in mathematical forms; design an algorithm that combines optimization and global sampling.
- Case study: construct two potential attack scenarios: emergency brake attack and AV freezing attack.

Imperial College London

## 2. Background

### 2.1 LiDAR-based Perception in AV Systems - Apollo



**Input**

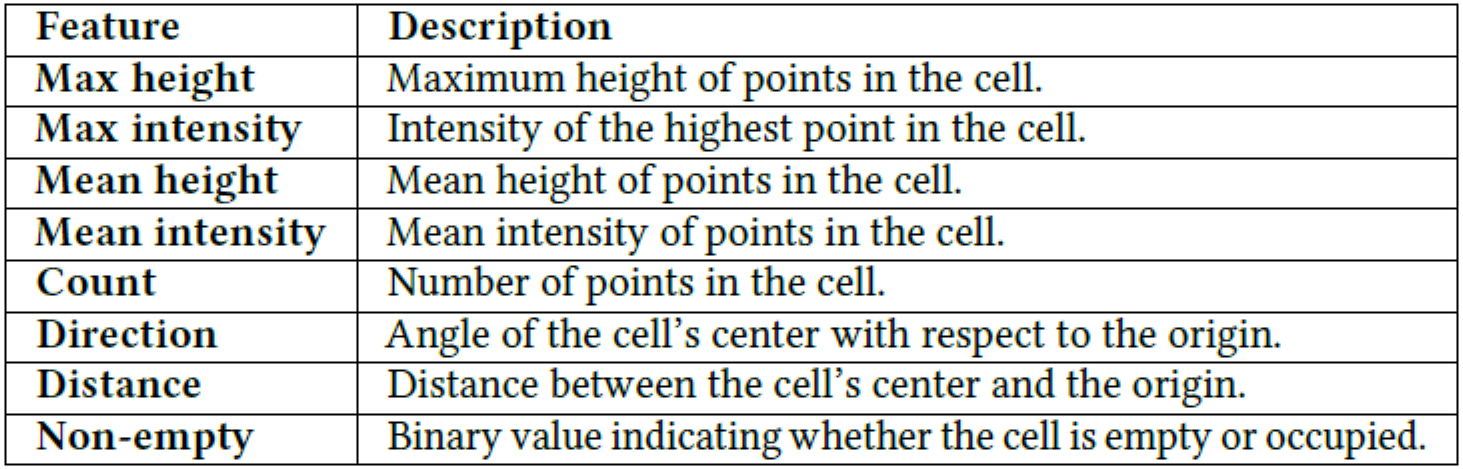
**Trans**

**ROI f**

outsic

**Featu**

cell 8

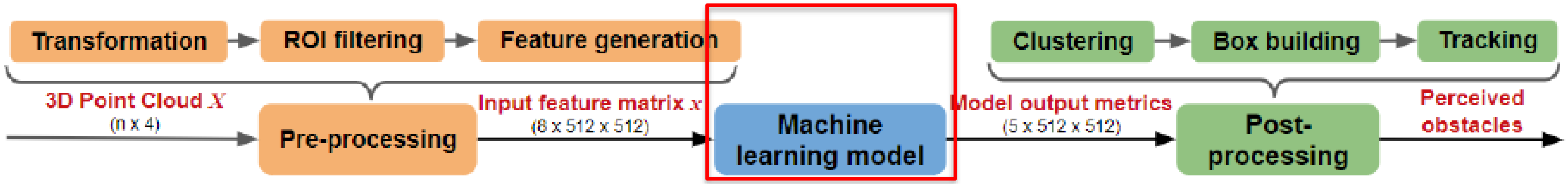


| Feature | Description |
|---|---|
| Max height | Maximum height of points in the cell. |
| Max intensity | Intensity of the highest point in the cell. |
| Mean height | Mean height of points in the cell. |
| Mean intensity | Mean intensity of points in the cell. |
| Count | Number of points in the cell. |
| Direction | Angle of the cell's center with respect to the origin. |
| Distance | Distance between the cell's center and the origin. |
| Non-empty | Binary value indicating whether the cell is empty or occupied. |

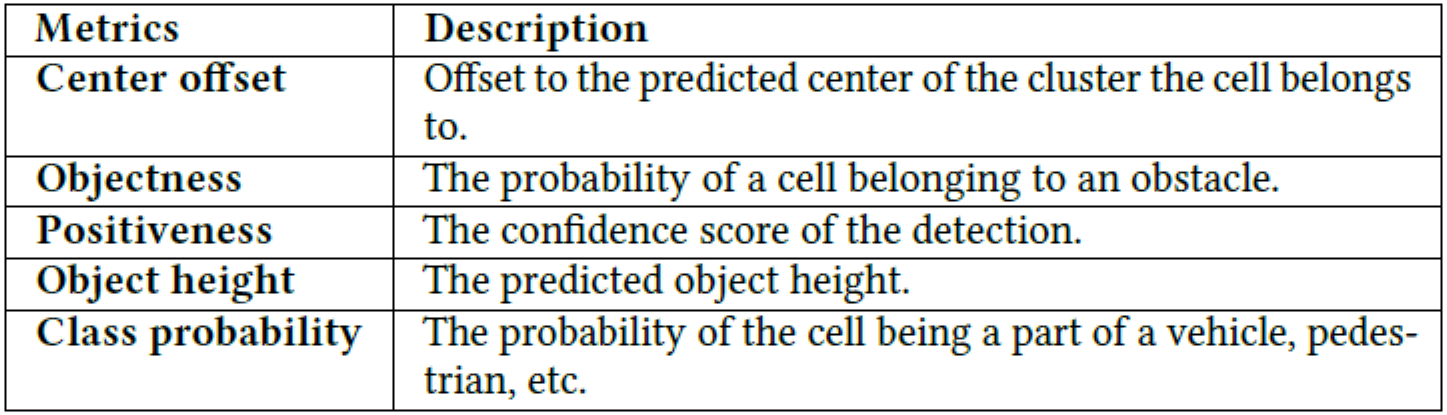
Imperial College London

## 2. Background

### 2.1 LiDAR-based Perception in AV Systems - Apollo



**Deep**
cell.
**Outp**

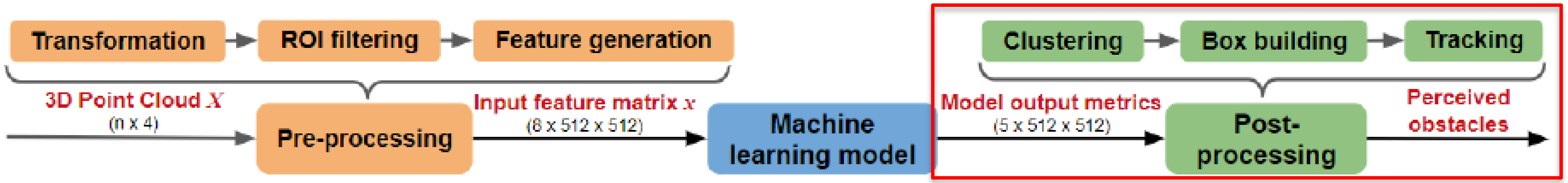


| Metrics | Description |
|---|---|
| Center offset | Offset to the predicted center of the cluster the cell belongs to. |
| Objectness | The probability of a cell belonging to an obstacle. |
| Positiveness | The confidence score of the detection. |
| Object height | The predicted object height. |
| Class probability | The probability of the cell being a part of a vehicle, pedestrian, etc. |

Imperial College London

## 2. Background

### 2.1 LiDAR-based Perception in AV Systems - Apollo



**Input :** cells with **objectness** greater than 0.5(by default)

**Clustering** : build candidate object clusters (connected graph) → select clusters greater than 0.1(by default)

**Box building**: reconstructs the bounding box including height, width, length of an obstacle candidate

**Tracking**: generate tracked obstacles

**prediction module**: predict the future trajectories of perceived obstacles

**Planning module**: plan the future driving routes and makes decisions

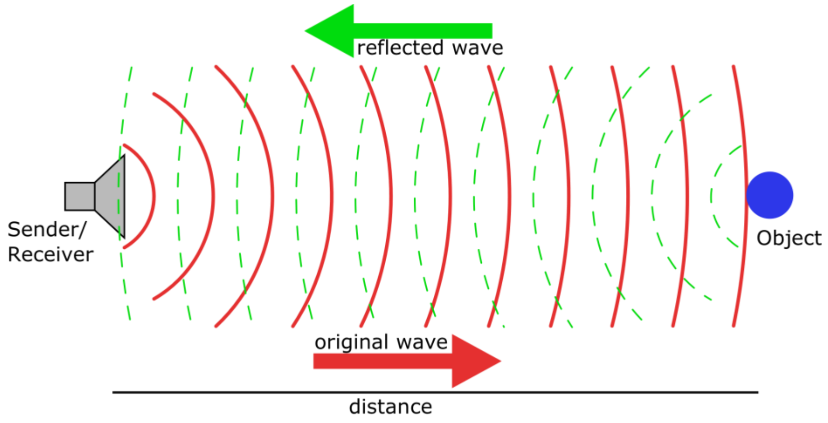
Imperial College London

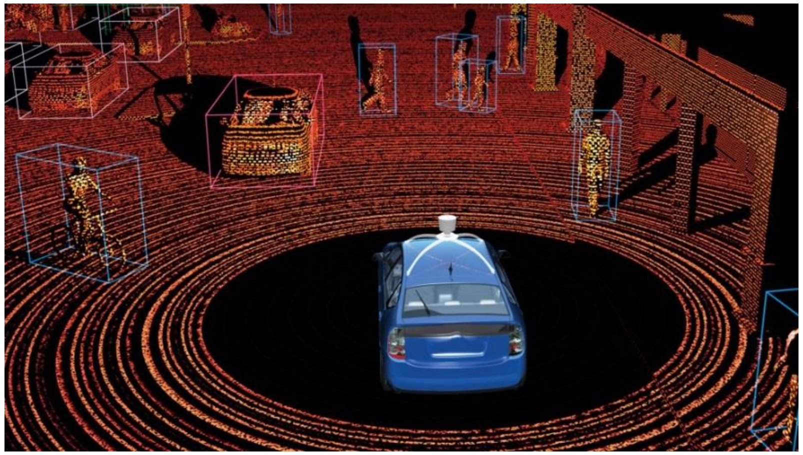
## 2. Background

### 2.2 LiDAR Sensor and Spoofing Attacks

Light Detection and Ranging (LiDAR) use laser to measure distance:

1. emitting a laser pulse on a surface
2. catching the reflected laser back to the LiDAR pulse source
3. measuring the time laser travelled
4. Distance = (Speed of light x Time elapsed) / 2





https://news.voyage.auto/

Imperial College
London

## 2. Background

### 2.2 LiDAR Sensor and Spoofing Attacks

**Sensor spoofing attacks** use the same physical channels as the targeted sensor to manipulate the sensor readings.

- place an attacking device at the roadside to shoot malicious laser pulses to AVs passing by.
- drive an attack vehicle equipped with an attacking device that shoots laser pulses to the victim AV's LiDAR.

LiDAR has been shown to be vulnerable to laser spoofing attacks

https://news.voyage.auto/

Imperial College London

## 2. Background

### 2.3 Adversarial Machine Learning

Given a machine learning model M, input x and its corresponding label y, an **adversarial attacker** aims to generate adversarial examples x ′

So that,

M(x ′) ≠ y(**untargeted attack**)

Or,

M(x′) = y′, where y′ is a target label (**targeted attack**).

Imperial College London

## 2. Background

### 2.3 Adversarial Machine Learning

Carlini and Wagner [21] proposed to generate an adversarial perturbation for a targeted attack:

$$\min \|x - x'\|_p \quad \text{s.t. } M(x') = y' \text{ and } x' \in X$$

More adversarial examples on:

segmentation [26, 53], human pose estimation [26], object detection [53], Visual Question Answer system [54], image caption translation [24], etc.

This paper also leverages an optimization-based method

Imperial College London

## 3. Attack Goal and Threat Model

### 3.1 Attack goal

Fool the LiDAR-based perception into perceiving fake obstacles in front of a victim AV.

Target: front-near fake obstacles around 5 meters

Imperial College London

## 3. Attack Goal and Threat Model

### 3.2 Threat model

Consider **LiDAR spoofing attacks** as the threat model, which is a demonstrated practical attack vector for LiDAR sensors.

Attacker has **white-box access** to the machine learning model and the perception system

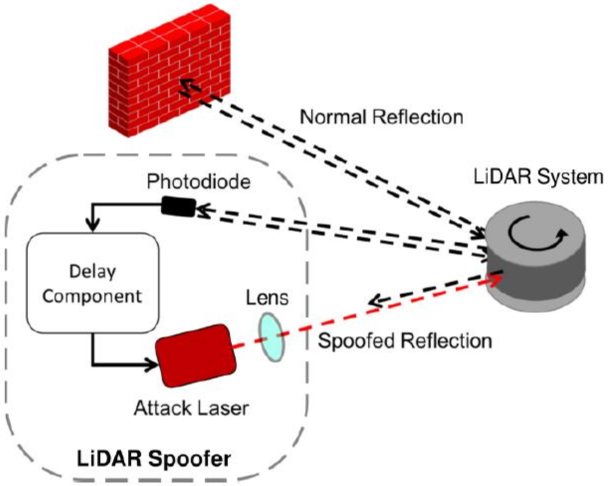
Imperial College London

# 4. Methodology

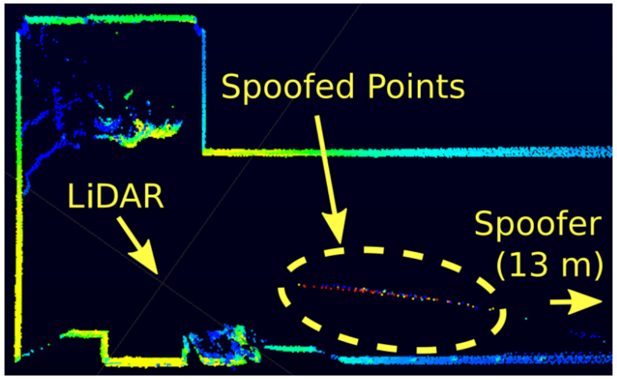
## 4.1 Reproduce the LiDAR spoofing attack

**Photodiode** : trigger the delay component whenever it captures laser pulses fired from the victim LiDAR.

**Delay component**: trigger the attack laser after a certain amount of time



**Results:** In total, around **100 dots** can be spoofed.
Among 60 points at the center 8-10 vertical lines can be stably spoofed with high intensity.

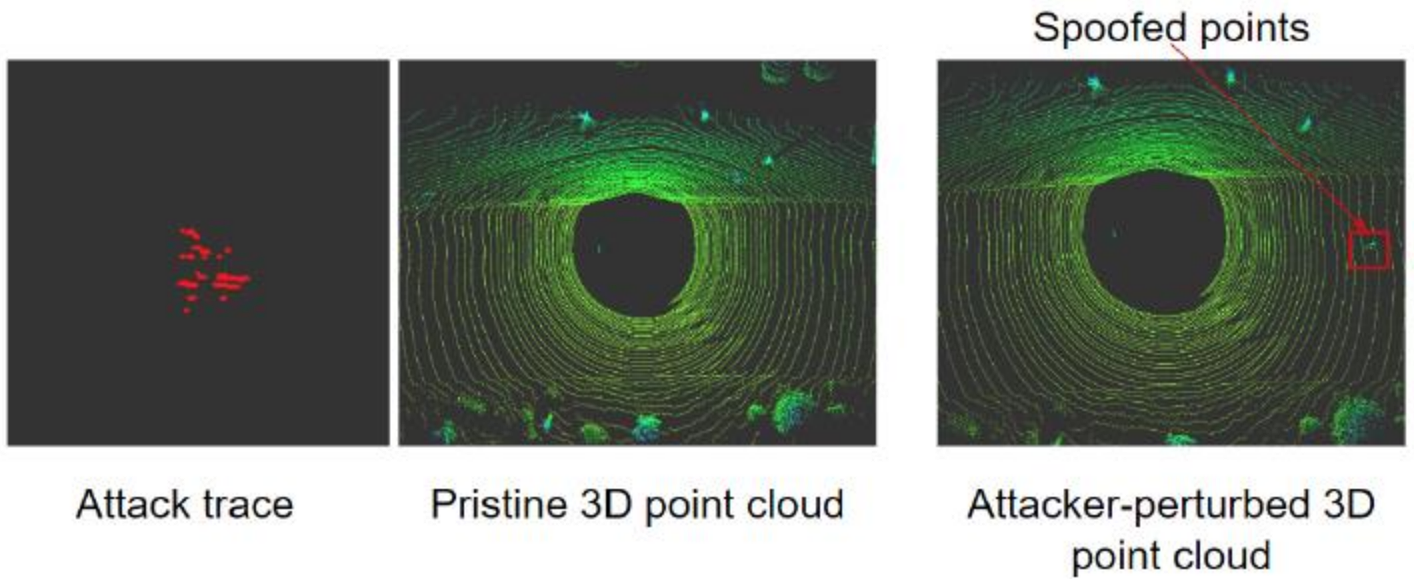


Imperial College London

## 4. Methodology

### 4.2 Blind sensor spoofing

Experiment 1: Directly apply original spoofing attack traces.
Experiment 2: Apply spoofing attack traces at different angles.
Experiment 3: Apply spoofing attack traces with different shapes

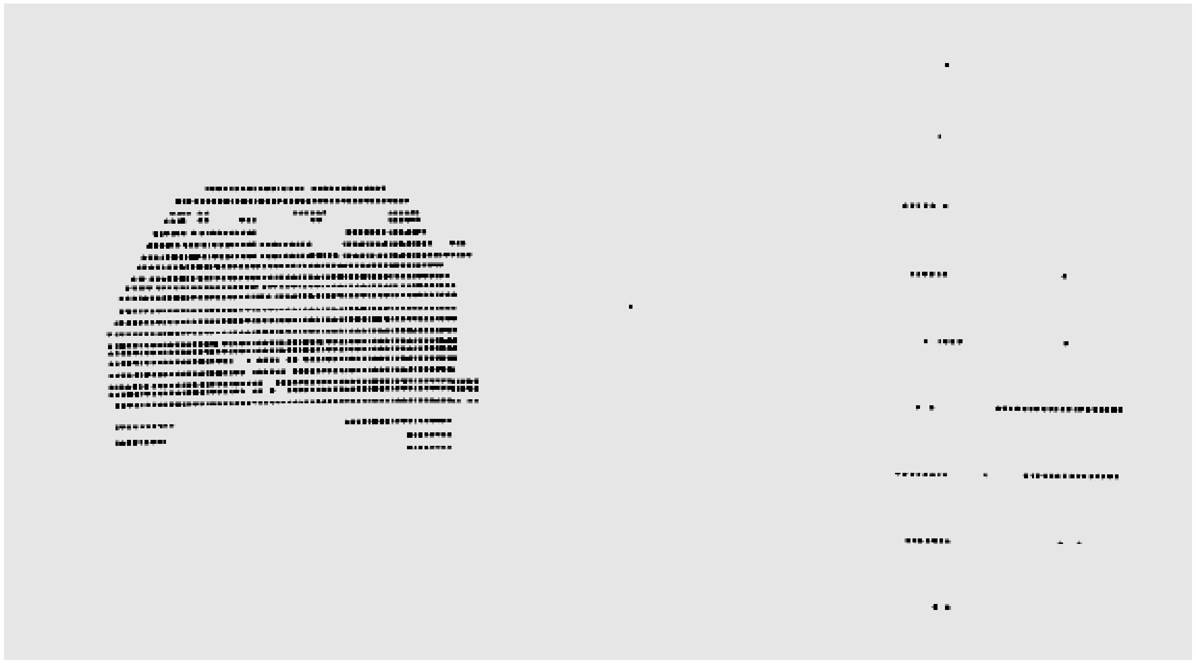


Failed to generate spoofed obstacles in the LiDAR-based perception pipeline in Baidu Apollo.

Imperial College London

# 4. Methodology

## 4.2 Blind sensor spoofing



The point cloud for a real vehicle has a much wider angle and many more points than the attack traces.

Imperial College London

## 4. Methodology

### 4.3 Adv-LiDAR overview

the problem is formulated as follows:

$$\begin{aligned} \min \quad & \mathcal{L}_{\text{adv}}(x \oplus t'; M) \\ \text{s.t.} \quad & t' \in \{\Phi(T') \mid T' \in \mathcal{A}\} \;\; \& \;\; x = \Phi(X) \end{aligned}$$

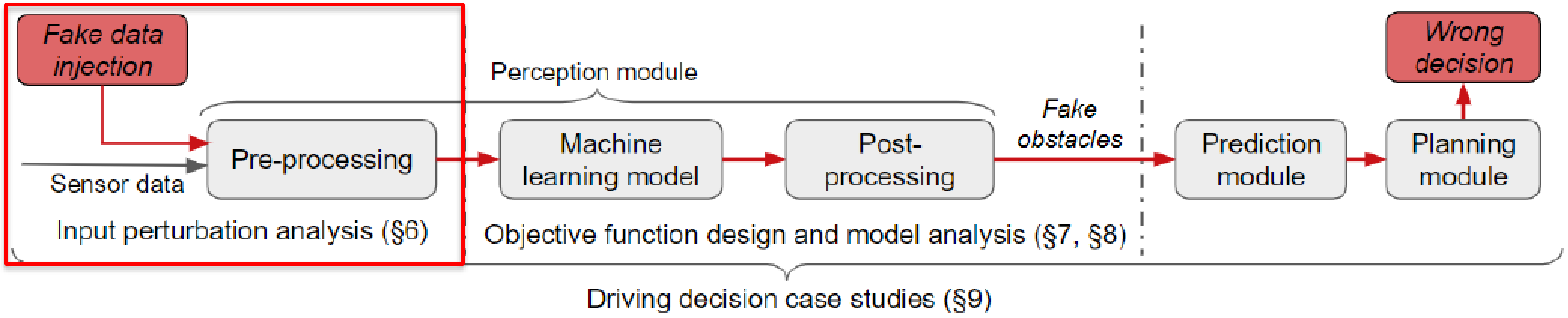
$x$ : the corresponding 2D input feature matrix
t $'$ : adversarial spoofed input feature matrix
$\oplus(\cdot)$ : merging function
$L_{\text{adv}}^{(\cdot; M)}$ : the adversarial loss, given the machine learning model

$X$ : the pristine 3D point cloud
$T'$ : adversarial spoofed 3D point cloud
A : 3D point cloud generated from LiDAR spoofing attacks
$\Phi(\cdot)$ : the pre-processing function that maps $X$ into $x$

Imperial College London

# 4. Methodology

## 4.3 Adv-LiDAR overview



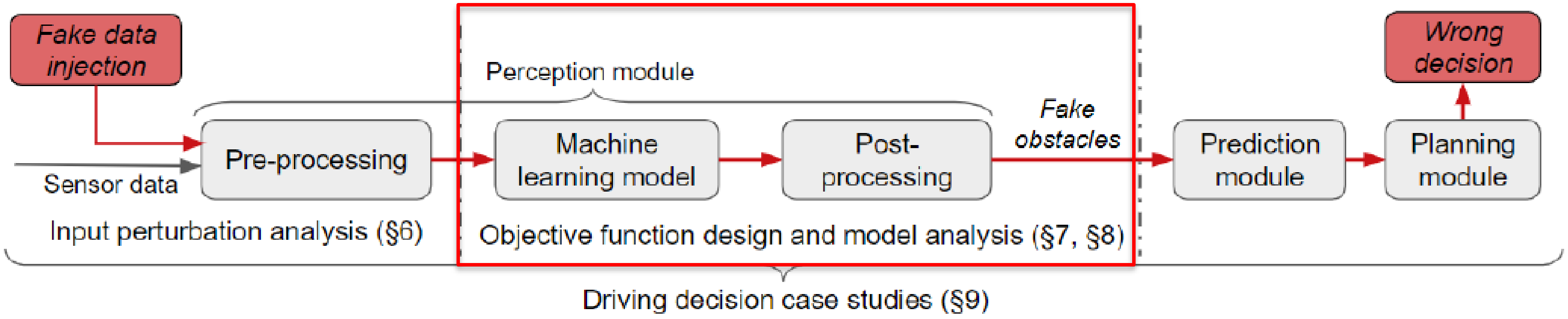
**Input perturbation analysis**:

1. conduct spoofing attacks on LiDAR to collect a set of possible spoofed 3D point cloud
2. model the spoofing attack capability **A**
3. formulate the spoofed input feature matrix into a differentiable function

Imperial College London

# 4. Methodology

## 4.3 Adv-LiDAR overview



**Objective function design and model analysis**:

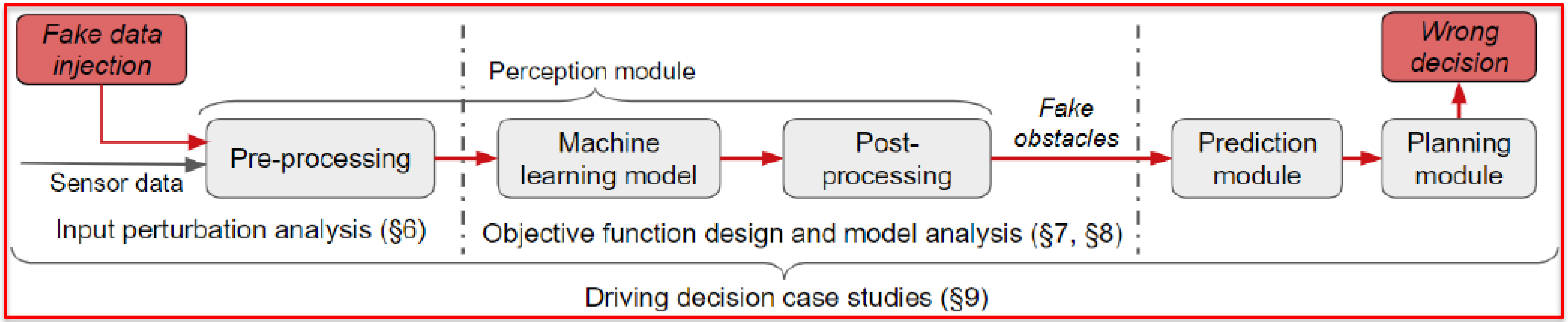
1. study the post-processing steps to extract key strategies of transforming model output into perceived obstacles, and formulate it into an **objective function**.

2. Instead of directly using existing optimization-based methods, we improve the methodology by **combining global sampling with optimization**.

Imperial College London

# 4. Methodology

## 4.3 Adv-LiDAR overview



**Driving decision case study**:

1. generate adversarial 3D point cloud that can inject spoofed obstacles at the LiDAR-based perception level

2.model the spoofing attack capability

3.formulate the spoofed input feature matrix into a differentiable function

Imperial College
London

## 5. Input perturbation analysis

### 5.1 Spoofing Attack Capability

**Number of spoofed points**:

The maximum number of spoofed points could be increased if the attacker uses more advanced attack equipment. we find that **around 60 points** can be reliably spoofed in our experiments

Imperial College London

# 5. Input perturbation analysis

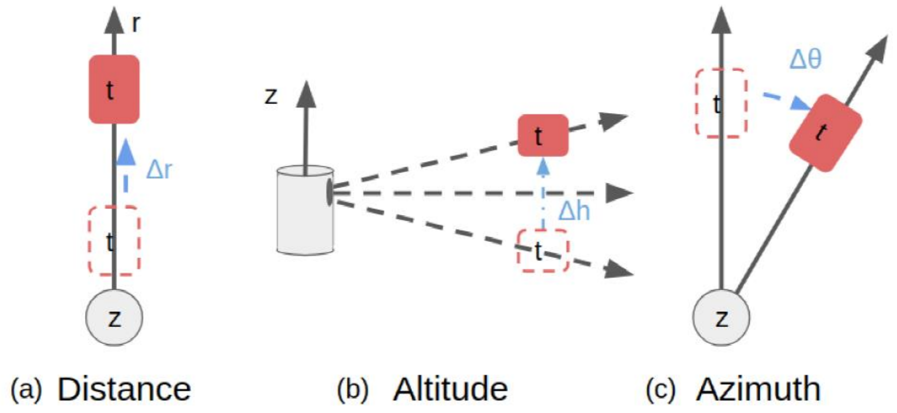
## 5.1 Spoofing Attack Capability

**Modify the Location of spoofed points**:

Distance: change the **delay of the attack laser signal pulses in small intervals** -- moving spoofed points nearer or further on the axis

Altitude: change the **delay in intervals of 2.304 μs** -- moving the position of the spoofed points from vertical line to vertical line to change the height

Azimuth: change the **delay in intervals of 55.296 μs** -- rotating the spoofed points with the LiDAR sensor as the pivot point on the horizontal plane



Imperial College London

# 5. Input perturbation analysis

## 5.2 Input Perturbation Modeling

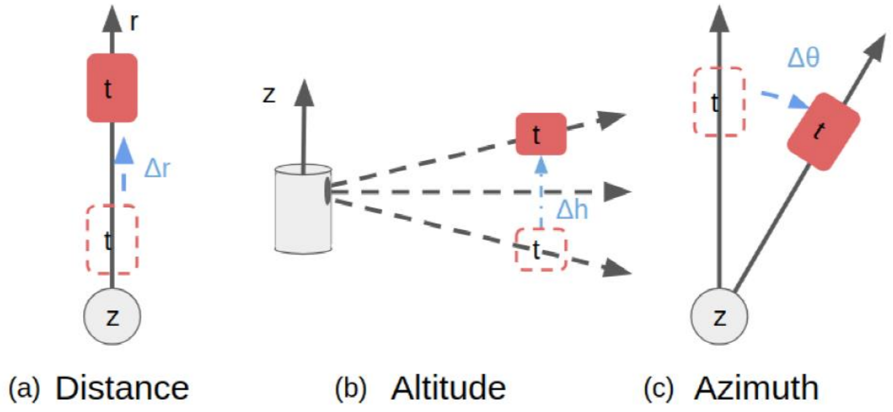
$$\min \quad \mathcal{L}_{\text{adv}}(x \oplus t'; M)$$
$$\text{s.t.} \quad t' \in \{\Phi(T') | T' \in \mathcal{A}\} \;\; \& \;\; x = \Phi(X)$$

- **Formulating merging function (⊕)**

distance: change the **delay of the attack laser signal pulses in small intervals** -- moving spoofed points nearer or further on the axis

Altitude: change the **delay in intervals of 2.304 μs** -- moving the position of the spoofed points from vertical line to vertical line to change the height

Azimuth: change the **delay in intervals of 55.296 μs** -- rotating the spoofed points with the LiDAR sensor as the pivot point on the horizontal plane

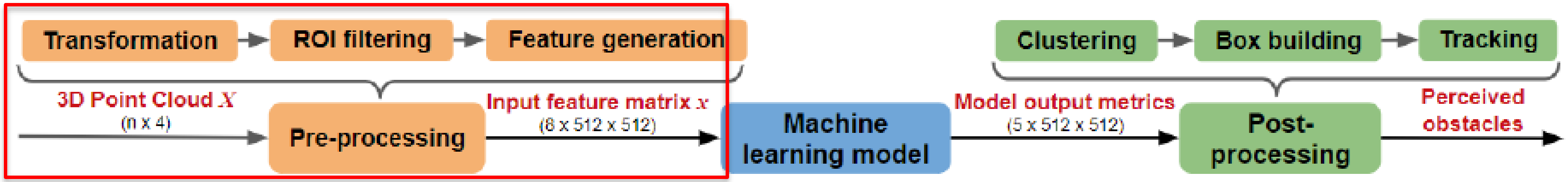


Imperial College London

# 5. Input perturbation analysis

## 5.2 Input Perturbation Modeling

$$\begin{aligned} \min \quad & \mathcal{L}_{\text{adv}}(x \oplus t'; M) \\ \text{s.t.} \quad & t' \in \{\Phi(T') | T' \in \mathcal{A}\} \ \ \& \ \ x = \Phi(X) \end{aligned}$$



➢ **Formulating merging function (⊕)**

1.the adversarial spoofed 3D point cloud T ′ will be transformed along with the 3D point cloud X. - **coordinate transformation process** has no effect

2.ROI filtering process filters out 3D point cloud located outside of the road from a bird’s-eye view - as long as spoof points T ′ on the road, the **ROI filtering process** makes no effect

3.The feature extraction process, extracts statistical features such as average height and average intensity.

Imperial College London

## 5. Input perturbation analysis

### 5.2 Input Perturbation Modeling

$$\begin{aligned} \min \quad & \mathcal{L}_{\text{adv}}(x \oplus t'; M) \\ \text{s.t.} \quad & t' \in \{\Phi(T') | T' \in \mathcal{A}\} \;\; \& \;\; x = \Phi(X) \end{aligned}$$

- **Formulating merging function (⊕)**

$$x' = x \bigoplus t'$$

$$= \begin{bmatrix} I^x_{cnt} + I^{t'}_{cnt} \\ (I^x_{avg\_h} \cdot I^x_{cnt} + I^{t'}_{avg\_h} \cdot I^{t'}_{cnt})/(I^x_{cnt} + I^{t'}_{cnt}) \\ \max(I^x_{max\_h}, I^{t'}_{max\_h}) \\ (I^x_{avg\_int} \cdot I^x_{cnt} + I^{t'}_{avg\_int} \cdot I^{t'}_{cnt})/(I^x_{cnt} + I^{t'}_{cnt}) \\ \sum I^x_{max\_int} \cdot 1\{I^x_{max\_h} = max\{I^x_{max\_h}, I^{t'}_{max\_h}\}\} \end{bmatrix}$$

Imperial College London

## 5. Input perturbation analysis

### 5.2 Input Perturbation Modeling

$$\begin{aligned} \min \quad & \mathcal{L}_{\text{adv}}(x \oplus t'; M) \\ \text{s.t.} \quad & t' \in \{\Phi(T') | T' \in \mathcal{A}\} \ \ \& \ \ x = \Phi(X) \end{aligned}$$

- **Modeling input feature matrix spoofing capability Φ(A)**

**it equals to representing adversarial input feature matrix t ′ with known spoofed input feature matrix t**.

We can use **global spatial transformations** including rotation, translation and scaling, under certain constraints to represent the input feature matrix spoofing capability.

Imperial College London

# 5. Input perturbation analysis

$$\min \quad \mathcal{L}_{\text{adv}}(x \oplus t'; M)$$
$$\text{s.t.} \quad t' \in \{\Phi(T') | T' \in \mathcal{A}\} \;\; \& \;\; x = \Phi(X)$$

## 5.2 Input Perturbation Modeling

- **Modeling input feature matrix spoofing capability Φ(A)**

The location of $t_{(i)}$ can be derived as $t'(i)$ as follows:

$$(u_{(i)}, v_{(i)}, 1)^T = H \cdot (u'_{(i)}, v'_{(i)}, 1)^T,$$

$$\textbf{w.r.t.} \qquad H = \begin{bmatrix} \epsilon(\cos\theta & -\sin\theta) & \tau_x \\ \epsilon(\sin\theta & \cos\theta) & \tau_y \\ 0 & 0 & 1 \end{bmatrix}$$

**t ′(i)**: denote values of the i-th position on the spoofed input feature matrix t ′
**(u′(i),v′(i))**: denote its location
***H*** *is a* homography matrix $H(\theta, \tau, \epsilon)$

Imperial College London

## 5. Input perturbation analysis

$$\begin{aligned} \min \quad & \mathcal{L}_{\text{adv}}(x \oplus t'; M) \\ \text{s.t.} \quad & t' \in \{\Phi(T') | T' \in \mathcal{A}\} \;\; \& \;\; x = \Phi(X) \end{aligned}$$

### 5.2 Input Perturbation Modeling

- **Modeling input feature matrix spoofing capability Φ(A)**

$$t'_{(i)} = \sum_{q \in \mathcal{N}(u_{(i)}, v_{(i)})} t_{(q)}(1 - |u_{(i)} - u_{(q)}|)(1 - |v_{(i)} - v_{(q)}|),$$

**N($u_{(i)}$, $v_{(i)}$)** represents the 4-pixel neighbors (t left, top-right,bottom-left,bottom-right) at the location **( *u*(*i*), *v*(*i*) )**

Imperial College London

## 6. Generate adversarial examples

### 6.1 Design the adversarial loss

In the clustering process, each cell of the model output is filtered by its objectness value. After the clustering process, candidate object clusters are filtered by their positiveness values. Therefore, the loss function is:

$$\mathcal{L}_{adv} = \sum(1 - Q(x', \text{positiveness})Q(x', \text{objectness}))\mathcal{M}(px, py)$$

**Q(x ′, ・)** : extract the probabilities of ・ attribute from model M by feeding in adversarial example x ′.

**M** : standard Gaussian mask

**(px,py)** : attack target position chosen by the attacker.

Imperial College London

## 6. Generate adversarial examples

### 6.2 Optimization algorithm

With the Ladv design above, the optimization problem can be directly solved by using the **Adam optimizer** to obtain the transformation parameters θ, Ʈx and scalar sh by minimizing the following objective function:

$$f = \arg\min_{\theta, \tau_x, s_h} \sum (1 - Q(x', \text{positiveness}) Q(x', \text{objectness})) \mathcal{M}(px, py)$$

**Q(x ′, ・)** : extract the probabilities of ・ attribute from model M by feeding in adversarial example x ′.

**M** : standard Gaussian mask

**(px,py)** : attack target position chosen by the attacker.

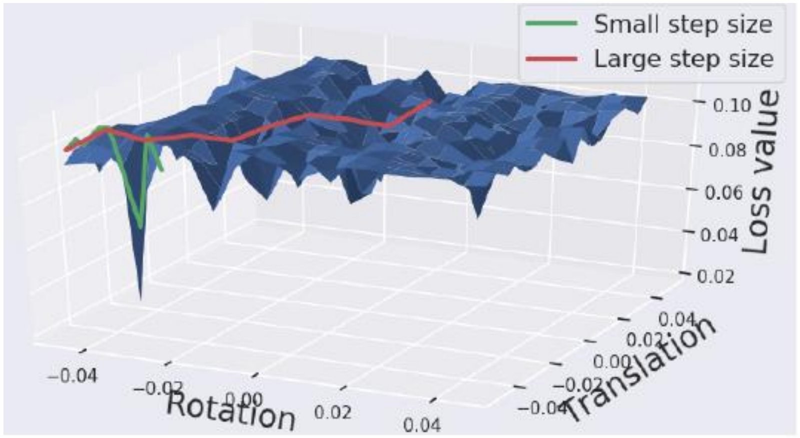
Imperial College London

# 6. Generate adversarial examples

## 6.2 Optimization algorithm

Loss surface against the transformation parameters is as below.

Problem: loss surface over the transformation parameters is noisy at a small scale (green line) and quite flat at a large scale (red line)



Solution: 1. calculate the range of the transformation parameters so that the transformed spoofed 3D point cloud is located in the target area.
2. uniformly take n samples for rotation and translation parameters and create $n^2$ samples to initiate with.

Imperial College London

## 6. Generate adversarial examples

### 6.3 Generating adversarial spoofed 3D point cloud

Using the transformation parameters θ, τ , ϵ, sh, we can express the corresponding adversarial spoofed 3D point cloud T ′ such that t ′ = Φ(T ′) with a dual transformation function GT of Gt .

We use Twx ,Twy ,Twz to denote value of coordinate (wx,wy,wz) and Ti to denote the **value of intensity** for all points in spoofed 3D point cloud T . With transformation parameters θ, τ , ϵ, sh, we can expressT′wx ,T′wy ,T′wz of the transformed adversarial spoofed 3D point cloud T ′ below:

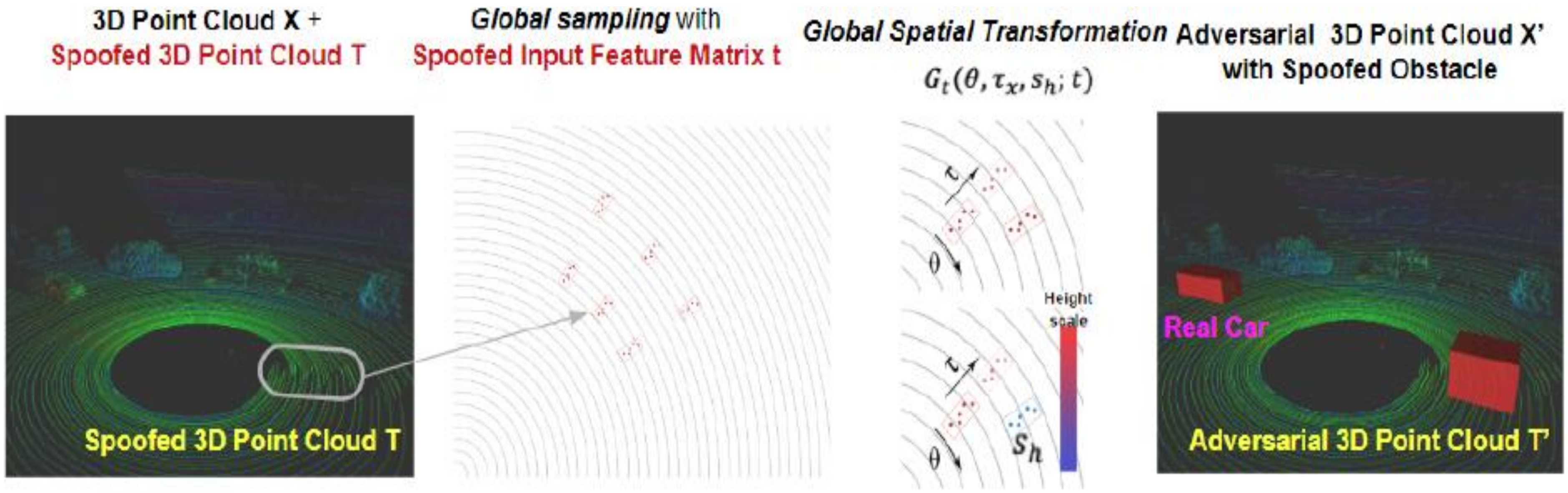
$$T_i' = T_i$$

$$\begin{bmatrix} T'_{\mathrm{w_x}} \\ T'_{\mathrm{w_y}} \\ T'_{\mathrm{w_z}} \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & \tau_x \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & s_h & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} T_{\mathrm{w_x}} \\ T_{\mathrm{w_y}} \\ T_{\mathrm{w_z}} \\ 1 \end{bmatrix}$$

Imperial College London

## 6. Generate adversarial examples

### 6.4 Overall adversarial example generation process


3D Point Cloud X +
Spoofed 3D Point Cloud T
Global sampling with
Spoofed Input Feature Matrix t
Global Spatial Transformation
$G_t(\theta, \tau_x, s_h; t)$
Adversarial 3D Point Cloud X'
with Spoofed Obstacle
Spoofed 3D Point Cloud T
$\tau$
$\theta$
$s_h$
Height scale
Real Car
Adversarial 3D Point Cloud T'

Imperial College London

## 5. Evaluation and result

| Targeted position | # Spoofed points | | |
|---|---|---|---|
| | 20 | 40 | 60 |
| 2-8 meters | 87% | 82% | 90% |

Table 4: Robustness analysis results of generated adversarial spoofed 3D point cloud to variation in spoofed 3D point cloud $T \in \mathcal{S}_T$. The robustness is measured by average attack success rates.

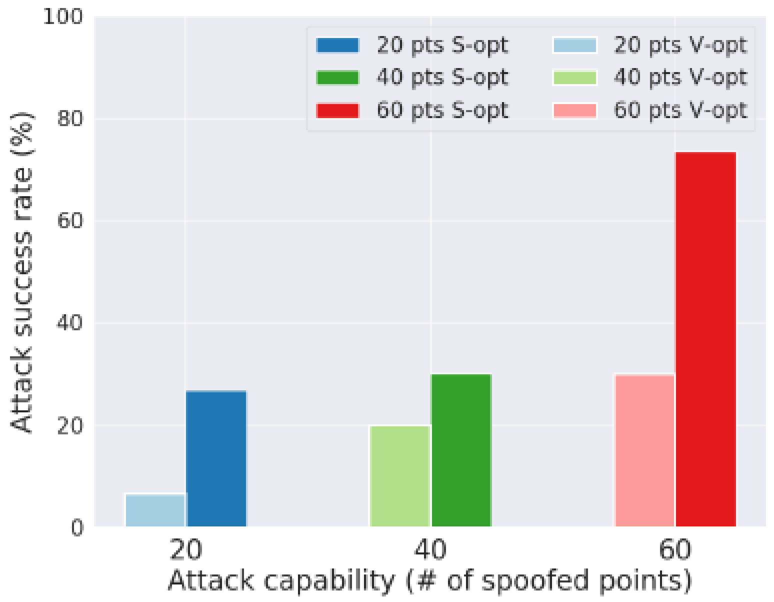


Figure 10: Attack success rate of spoofing a front-near obstacle with different number of spoofed points. V-opt refers to vanilla optimization which is directly using the optimizer and S-opt refers to sampling based optimization. We choose Adam [35] as the optimizer in both cases.

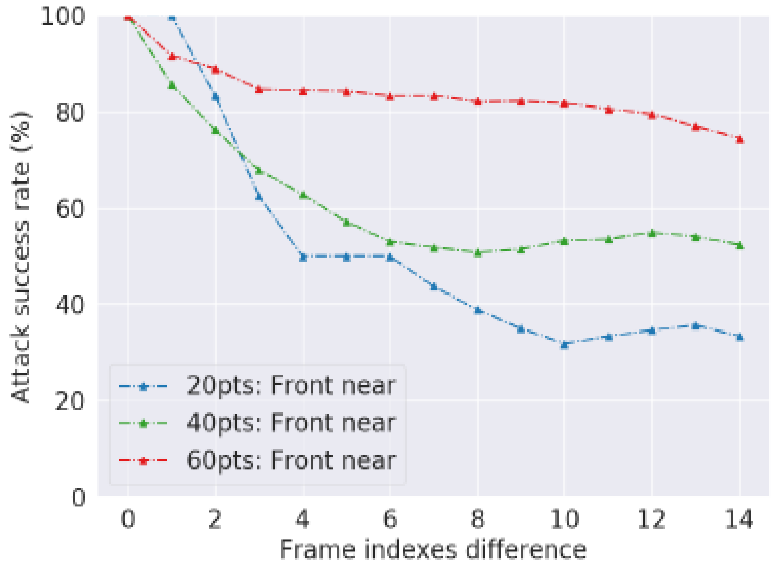


Figure 11: The robustness of the generated adversarial spoofed 3D point cloud to variations in 3D point cloud $X$. We quantify the variation in 3D point cloud $X$ as the frame indexes difference between the evaluated 3D point cloud and the 3D point cloud used for generating the adversarial spoofed 3D point cloud.

Imperial College London

# Thank you