PAPER *Special Section on Cryptography and Information Security*

Adversarial Scan Attack against Scan Matching Algorithm for Pose Estimation in LiDAR-Based SLAM

Kota YOSHIDA†a), *Student Member*, **Masaya HOJO**††, *Nonmember*, *and* **Takeshi FUJINO**††, *Member*

SUMMARY Autonomous robots are controlled using physical information acquired by various sensors. The sensors are susceptible to physical attacks, which tamper with the observed values and interfere with control of the autonomous robots. Recently, sensor spoofing attacks targeting subsequent algorithms which use sensor data have become large threats. In this paper, we introduce a new attack against the LiDAR-based simultaneous localization and mapping (SLAM) algorithm. The attack uses an adversarial LiDAR scan to fool a pose graph and a generated map. The adversary calculates a falsification amount for deceiving pose estimation and physically injects the spoofed distance against LiDAR. The falsification amount is calculated by gradient method against a cost function of the scan matching algorithm. The SLAM algorithm generates the wrong map from the deceived movement path estimated by scan matching. We evaluated our attack on two typical scan matching algorithms, iterative closest point (ICP) and normal distribution transform (NDT). Our experimental results show that SLAM can be fooled by tampering with the scan. Simple odometry sensor fusion is not a sufficient countermeasure. We argue that it is important to detect or prevent tampering with LiDAR scans and to notice inconsistencies in sensors caused by physical attacks.

key words: *instrumentation security, SLAM, LiDAR, adversarial examples*

1. Introduction

Autonomous mobile robots, such as self-driving cars, are controlled using physical information measured by various sensors. The robots measure the outside world, create a map, identify obstacles, and calculate a destination path. Camera, radar, and light detection and ranging (LiDAR) sensors are typical equipment used to measure the surrounding environment.

As autonomous mobile robots become more widespread, they are becoming increasingly targeted by attacks from adversaries. One of the ways to attack these robots is physically attacking their sensors. Adversaries have been known to irradiate a strong laser to a camera sensor to white-out the captured images and a spoofed laser to a LiDAR sensor to tamper with a measured distance [1], [2].

Sensor attacks affect subsequent systems which use acquired sensor data. Cao et al. conducted a security study on a LiDAR-based object detection system using physical sensor attacks [3]. The object detection system is important for autonomous mobile robots to determine the position of surrounding obstacles (e.g., cars, pedestrians) and avoid accidents. On the other hand, adversaries attempt to make obstacles appear or disappear to cause accidents. Cao et al. found that it was difficult to cheat the system by blindly applying existing LiDAR spoofing techniques. Thus, it is necessary to consider the object detection system when adversaries attack sensors. To cheat the object detection systems of self-driving cars, studies have investigated injecting spoofed points to LiDAR scans and creating 3D objects that can be observed as an adversarial point cloud [3]–[5].

Simultaneously localization and mapping (SLAM) is a technique to build a map from the observed surrounding environment. SLAM is necessary to robots for grasping self-location and the surrounding environment and planning a path. Similar to object detection algorithms, the SLAM algorithm may also be affected by physical attacks on sensors. In this paper, we perform a security study of a LiDAR-based SLAM algorithm. We assume that an adversary tries to fool the SLAM algorithm through a physical sensor attack. We point out a vulnerability to a scan matching algorithm on the SLAM. The scan matching algorithm is used for estimating robot pose between a series of LiDAR scans and the scans are concatenated as a map by analyzing consecutive poses. The adversary calculates perturbation (the amount of tampering with distance) of the scan by using a gradient of loss function on the scan matching algorithm. The scan matching algorithm estimates a wrong pose by tampered scan, and a concatenated scan with the wrong pose builds a wrong map. The wrong map may cause loss of self-location and wrong robot control.

Our contributions are as follows:

- We perform a security study on a SLAM algorithm. To the best of our knowledge, this work is the first to attack the SLAM algorithm by tampering with LiDAR scan data.
- We introduce an adversarial scan calculation which applies a gradient-based approach to fool a scan matching algorithm. The adversary calculates the perturbation needed to tamper with a robot pose.
- We simulate 2D-LiDAR SLAM on a virtual corridor and attack the algorithm. Our attacks are evaluated by two typical scan matching algorithms, iterative closest point (ICP) [6] and normal distribution transform (NDT) [7].

Manuscript received March 15, 2021.
Manuscript revised August 17, 2021.
Manuscript publicized October 26, 2021.

†The author is with the Graduate School of Science and Technology, Ritsumeikan University, Kusatsu-shi, 525-8577 Japan.
††The authors are with Department of Science and Engineering, Ritsumeikan University, Kusatsu-shi, 525-8577 Japan.
a) E-mail: ri0044ep@ed.ritsumei.ac.jp

DOI: 10.1587/transfun.2021CIP0017

2. Related Works

2.1 Spoofing Attacks against LiDAR

LiDAR is a method of ranging that uses light to measure distance. Most LiDARs use the time-of-flight (ToF) principle. In ToF-based LiDAR, a pulse laser is irradiated from a light source, and a light detector detects the pulse laser reflected by objects. The distance to the object L is calculated by the equation

$$L = \frac{1}{2}c\Delta t \tag{1}$$

where c is light speed and Δt is the time difference between pulse laser irradiation and detection.

Many ToF-based LiDARs adopt multi-echo technology. LiDAR can detect multiple lasers from one irradiation when a translucent object (e.g., rain, fog) reflects a part of the light on the pulse path. The handling of the multiple reflective lasers is left to subsequent systems in post-processing (e.g., selecting the strongest reflected laser to remove effects that rain and fog).

An adversary can falsify the distance to an object by exploiting additional laser for attacking [1], [2]. Just after the LiDAR irradiates the pulse laser, the adversary irradiates a pulse laser that mimics the correct reflected laser to the receiver. When a spoofed laser is detected before the correct reflected laser, Δt becomes smaller, and the measurement distance becomes shorter. When a spoofed laser is detected strongly after the correct reflected laser, Δt becomes larger, and the measurement distance becomes longer. In this way, the ToF system may misunderstand that there is an object closer or farther than the correct position.

2.2 Adversarial Attacks against Object Detection System in Point Cloud

Szegedy et al. found adversarial examples that fool deep neural networks (DNNs) based image classification models [8]. An adversary adds a small perturbation to inputs, which causes misclassification. The fast gradient sign method (FGSM) [9] is a typical and highly effective method for calculating the perturbation of an input. The FGSM searches for η which satisfy Eq. (2) by gradually increasing ϵ in Eq. (3). Here, x is input, y is the ground truth, function f calculates the prediction of the DNN model with trained parameters, function $sign$ calculates the sign of the input, and function J calculates the distance between the ground truth y and the prediction output from the DNN model $f(x)$.

$$y \neq f(x + \eta) \tag{2}$$

$$\eta = \epsilon sign(\nabla_x J(x, y)) \tag{3}$$

Similarly, studies have calculated adversarial examples for 3D point cloud classification models [10]–[14]. An adversary can manipulate (i.e., add, remove, or move) the point cloud by using hints from gradients of the DNN models. These 3D point cloud classifiers are important parts of autonomous cars. An autonomous car detects pedestrians, cars, or other objects from the 3D point cloud acquired from the car's 3D-LiDAR. The adversary can also calculate the adversarial point cloud against the 3D point cloud classifier on the autonomous cars [3]. Furthermore, the adversary can create a 3D object which is perceived as an adversarial point cloud [4], [5].

In this paper, we do not focus on fooling machine learning algorithms such as DNN based images or point cloud classifiers. Our goal is to fool a scan matching algorithm that is used for robot pose estimation in the SLAM algorithm. Nevertheless, our attack methodology is inspired by these prior studies.

3. SLAM

3.1 System Overview

SLAM is an algorithm that performs localization and mapping simultaneously and is an important technique for autonomous mobile robots. In this paper, we focus on 2D-LiDAR SLAM. A 2D-LiDAR mounted on a robot scans horizontally and measures distances to surrounding objects. Figure 1 shows the points measured by the 2D-LiDAR on a robot placed in a corridor. A set of points measured by LiDAR scanning is called a scan. Each measurement point is represented by the robot-centric polar coordinates (l, ξ).

A typical SLAM system is shown in Fig. 2. The system consists of two stages. The frontend performs robot pose estimation, pose graph construction, and submap creation as sequential processing at each time. The robot pose is the change in robot position over time. It is estimated by a combination of scan matching, inertial or wheel sensors, or other

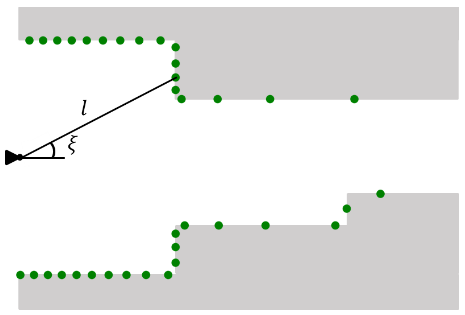


Fig. 1 Ground truth map (gray) and scan points (green) measured by 2D-LiDAR. Each scan point is represented by the robot-centric polar coordinates.

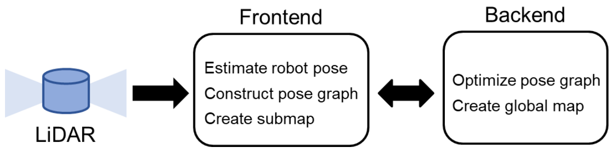


Fig. 2 Overview of typical 2D-LiDAR SLAM system.

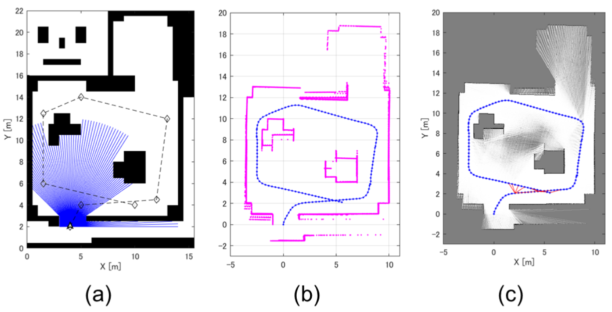


Fig. 3 (a) Ground truth map, robot path (black dashed line), robot at the initial location, and laser path of 2D-LiDAR. (b) Global map created by 2D-LiDAR SLAM [15] without odometry, implemented in Matlab. (c) Global map represented as an occupancy grid. Red lines are a pair of points of detected loop closures.

methods. The pose graph represents the robot pose from a starting location. The submap is built by overlaying a series of scans based on the estimated robot poses. The backend regularly performs pose graph optimization and global map creation as batch processing. The estimated robot poses accumulate errors; the backend optimizes the pose graph when a loop is detected and mitigates the error. The backend creates the global map by overlaying submaps with the optimized pose graph.

Figure 3 shows the result of the 2D-LiDAR SLAM [15] implemented in Matlab. A virtual robot moves around the room and performs SLAM. The robot does not have odometry sensors, so the SLAM system estimates robot poses using only a scan matching algorithm. An occupancy grid is a typical map representation method. Each cell of the occupancy grid stores a probability of the cell containing obstacles. The occupancy grid map is used for localization, path planning, and other robotics algorithms.

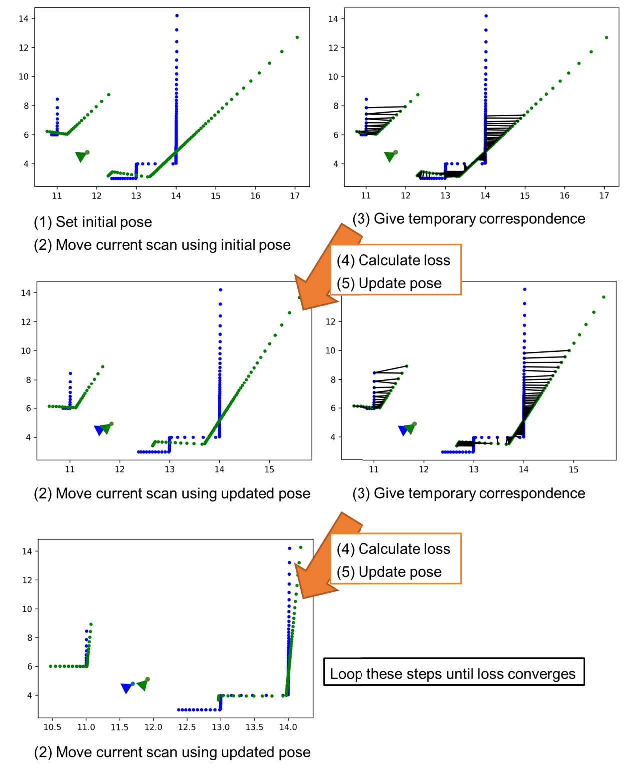
3.2 Scan Matching Algorithm

The scan matching algorithm is used for robot pose estimation in 2D-LiDAR SLAM. Iterative closest point (ICP) [6] and normal distributions transform (NDT) [7] are widely used scan matching algorithms. The scan matching algorithm aims to find a transformation in which two point clouds overlap appropriately.

In this section, we describe an ICP algorithm, specifically the point-to-point method. Our attack is based on the ICP algorithm. Figure 4 shows an overview of the scan matching. In the ICP algorithm for SLAM, one of the point clouds is a reference scan, and the other is a current scan. The reference scan is acquired at the previous time $(t-1)$ and the current scan is acquired at the present (t). The transformation such that the reference scan overlaps with the current scan is the estimated robot pose from time $(t-1)$ to (t).

The 2D-LiDAR scan which includes N sample points are represented as

$$S^p = \begin{pmatrix} l_1 & l_2 & \cdots & l_N \\ \xi_1 & \xi_2 & \cdots & \xi_N \end{pmatrix} \tag{4}$$



(1) Set initial pose
(2) Move current scan using initial pose
(3) Give temporary correspondence
(4) Calculate loss
(5) Update pose
(2) Move current scan using updated pose
(3) Give temporary correspondence
(4) Calculate loss
(5) Update pose
Loop these steps until loss converges
(2) Move current scan using updated pose

Fig. 4 Example procedure of ICP algorithm. Reference scan points are in blue and current scan points are in green. The blue triangle indicates the reference robot pose and the green triangle indicates the current robot pose during estimation.

where l is the radius and ξ is the angle of each sample point. We denote the reference scan as S^p_{ref} and the current scan as S^p_{cur}. The scan S^p is converted to Cartesian coordinates S^c by

$$S^c = \begin{pmatrix} l_1 cos\xi_1 & l_2 cos\xi_2 & \cdots & l_N cos\xi_N \\ l_1 sin\xi_1 & l_2 sin\xi_2 & \cdots & l_N sin\xi_N \end{pmatrix}. \tag{5}$$

The ICP algorithm estimates a pair consisting of rotation matrix R and translation matrix T as a robot pose. The algorithm performs the following steps.

1. Set an initial pose. Generally, the initial pose is given by known odometry if the robot has odometry sensors. If the robot does not have odometry sensors, the initial pose is set as the same as the reference scan.
2. Move the current scan using the pose. The transformed scan is calculated by the function ϕ.

$$\phi(S^c, R, T) = R \cdot S^c + T \tag{6}$$

$$R = \begin{pmatrix} cos\Delta\theta & -sin\Delta\theta \\ sin\Delta\theta & cos\Delta\theta \end{pmatrix} \tag{7}$$

$$T = \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} \tag{8}$$

where $\Delta\theta$ is the amount of rotation and $\Delta x, \Delta y$ is the amount of translation of the robot from the reference scan to the current scan.

3. Give a temporary correspondence between the transformed current scan and the reference scan. The correspondences are calculated by the k-dimensional (KD) tree constructed from the reference scan. A function $KDT(S^c_{ref}, S^c_{cur})$ returns the set of points from S^c_{cur} corresponding to each point of S^c_{ref}.
4. Calculate loss by the following cost function $\mathcal{L}$:

$$\begin{aligned}&\mathcal{L}(S^c_{ref}, S^c_{cur}, R, T) = \\ &MSE(S^c_{ref}, KDT(S^c_{ref}, \phi(S^c_{cur}, R, T)))\end{aligned} \tag{9}$$

$$MSE(a, b) = \frac{1}{I}\sum_{i=1}^{I}(a_i - b_i)^2 \tag{10}$$

5. Update the pose to minimize the loss. Gradient descent, Newton's method, or singular value decomposition (SVD) are often used.
6. Repeat the steps (2) to (5) until the loss converges.

4. Attack Methodology

4.1 Scenario

An adversary may tamper with a part of the current scan with a physical spoofing attack and fool a scan matching algorithm for pose estimation. SLAM registers an incorrect pose to a pose graph and adds a tampered current scan to a submap based on the incorrect pose. As a result, the global map generated by SLAM contains an error.

In this paper, we suppose the following conditions to evaluate our attacks. These conditions are simplified for feasibility study and we discuss realistic scenarios later in Sect. 6.

We suppose the following specifications for a robot targeted by an attack.

- 2D-LiDAR is mounted on the robot.
- Pose estimation in SLAM uses a scan matching algorithm with 2D-LiDAR scans.
- The ICP and NDT algorithms are used in the scan matching process.

We suppose the following capabilities for an adversary.

- The adversary is able to tamper with the distance of any sample point from the LiDAR scan.
- The adversary is able to calculate the robot pose at the next time step.
- The adversary is able to calculate the scan that is observed by the robot at the next time step.

4.2 Methodology

The adversary chooses an adversarial pose R_{adv}, T_{adv}. A goal of the adversary is that a scan matching algorithm estimates the target pose (R_{adv}, T_{adv}) instead of the original robot pose (R_{ori}, T_{ori}). For that purpose, the adversary tampers with a current scan S^p_{cur} using

$$S^p_{adv} = S^p_{cur} + M \circ \delta \tag{11}$$

$$\delta = \begin{pmatrix} \Delta l_1 & \Delta l_2 & \cdots & \Delta l_N \\ 0 & 0 & \cdots & 0 \end{pmatrix}. \tag{12}$$

$$M = \begin{pmatrix} m_1 & m_2 & \cdots & m_N \\ 0 & 0 & \cdots & 0 \end{pmatrix}. \quad (m \in \{0, 1\}) \tag{13}$$

where M is a mask that indicates the points the adversary can tamper with, operator "$\circ$" represents Hadamard product (product of each element of the matrix). Note that the adversary is only able to tamper with the distance of each sample point and not the angle, so the second row of δ is zero.

The adversary calculates a perturbation of the current scan δ in the following steps.

1. Set an initial adversarial scan as $S^p_{adv} = S^p_{cur} + M \circ \delta$ $(\delta = 0)$.
2. Calculate loss using a loss function

$$\begin{aligned}&\mathcal{L}(S^c_{ref}, S^c_{adv}, R_{adv}, T_{adv}) = \\ &MSE(S^c_{ref}, KDT(S^c_{ref}, \phi(S^c_{adv}, R_{adv}, T_{adv}))).\end{aligned} \tag{14}$$

The loss function focuses minimize the loss (Eq. (9)) on the adversarial target pose.
3. Update δ to minimize the loss.
4. Repeat steps (2) to (3) until the loss converges.

The loss function is differentiable and the adversary minimizes the loss by using the gradient method. In this paper, we use the Adam optimizer [16] to optimize the loss functions.

5. Experiment

5.1 Setup

We simulated an autonomous mobile robot and 2D-LiDAR in a virtual environment on Matlab 2020b. We prepared a virtual corridor shown in Fig. 5. The robot moved $4.61m$ in the virtual corridor from left to right along the dashed line. The 2D-LiDAR mounted on the robot acquires a scan. The LiDAR measured 120 points of distances from $-90°$ to $90°$ as the scan. The maximum measurement range of the LiDAR was $10m$. Thirteen LiDAR scans were taken while the robot was in motion. Note that the robot was only moving in the x-axis direction.

The robot performed 2D-LiDAR SLAM [15] implemented in Matlab. By default implementation, SLAM uses NDT for the scan matching algorithm. We integrated the point-to-point ICP algorithm into SLAM, and we evaluated our attack with the ICP and NDT scan matching algorithms. The ICP algorithm is the basic method to match scans, and the NDT algorithm is widely used. In the following experiment, we evaluate the effect of our attack based on how the estimated pose changed in the x-axis direction.

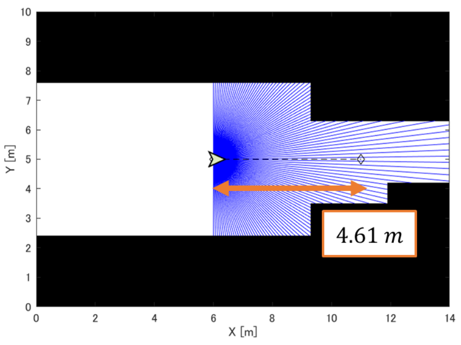


Fig. 5 Virtual corridor, robot, movement path (dashed black line), and the light path of LiDAR (blue line).

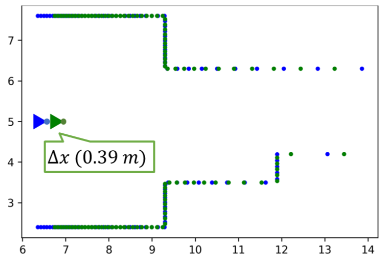


Fig. 6 Reference scan (blue) and current scan (green) are overlaid with a ground truth pose.

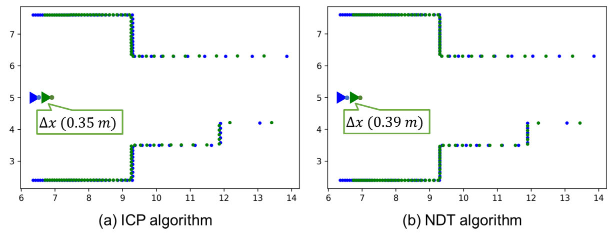


Fig. 7 Reference scan (blue) and current scan (green) overlaid with pose estimated by (a) ICP and (b) NDT without attack.

5.2 Results

5.2.1 Attack Against Pair of Scans

We selected a pair of scans that were acquired at $t = 1$ and $t = 2$. The scan at $t = 1$ is the reference scan and the scan at $t = 2$ is the current scan in this experiment. The scan matching algorithm estimates that the robot pose changed between time $t = 1$ and $t = 2$. The adversary deceives the scan matching algorithm by tampering with the current scan.

Figure 6 shows the selected pair of scans which is overlaid with the ground truth pose $\Delta x = 0.39m$. The original robot pose was $(\Delta x, \Delta y, \Delta\theta) = (0.39, 0, 0)$. We set an adversarial target pose as a half of the original pose, $(\Delta x, \Delta y, \Delta\theta) = (0.19, 0, 0)$.

Figure 7 shows the results of pose estimation by ICP and NDT. The ICP estimated $\Delta x = 0.35m$ and the NDT estimated $\Delta x = 0.39m$. We define these estimated poses as original poses to ICP and NDT.

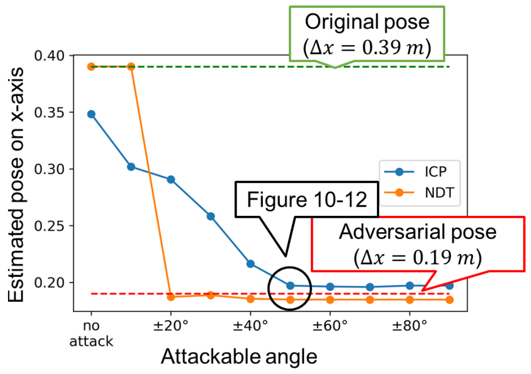


Fig. 8 The result of our attack while expanding the range of the mask from ±10° to ±90°.

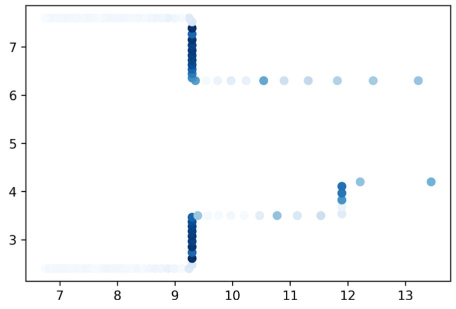


Fig. 9 Gradient of $\partial\mathcal{L}/\partial\Delta l$. The shade of color indicates the magnitude of the gradient.

We evaluated our attack against the pair of scans. We set the number of iteration of the adam optimizer as 100. Figure 8 shows the result of our attack while expanding the range of the mask M from ±10° to ±90°. As the number of attack points increases, the estimated poses were close to the adversarial pose.

Tampering points required for a successful attack depend on the terrain around the robot, but the adversary can estimate the effective points for attack by using a gradient. Figure 9 shows a gradient of $\partial\mathcal{L}/\partial\Delta l$. In this case, the wall perpendicular to the direction of the robot movement has a larger effect on scan matching.

The following describes the details of one result from Fig. 8. The mask range which is from 0° to ±50° covers the effective points estimated by the gradient. Figure 10 illustrates the mask and the difference between the current scan and the calculated adversarial scan. Figure 11 shows the transition of the loss in each iteration of the optimization process. The loss of the adversarial pose became smaller than the original pose by falsifying the measurement distance. Figure 12 shows scan matching results by ICP and NDT algorithms by using adversarial scan. Both ICP and NDT estimated the adversarial pose by the adversarial scan.

We evaluated the setting that the adversary had fewer tampering points. Figure 13 shows the result of our attack while expanding the range of the mask M from −10° to −90°. Our attack fooled the NDT even if the adversary had fewer tampering points. On the other hand, our attack fooled the

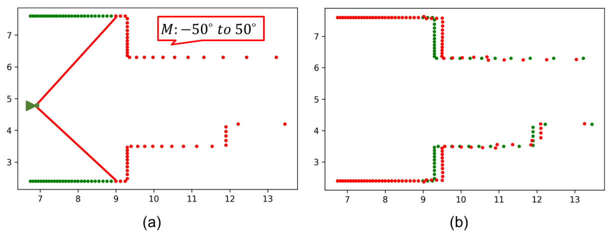


Fig. 10 (a) Mask M is set from 0° to ±50° that indicates points which the adversary can tamper with. (b) Difference between current scan (green) and adversarial scan (red).

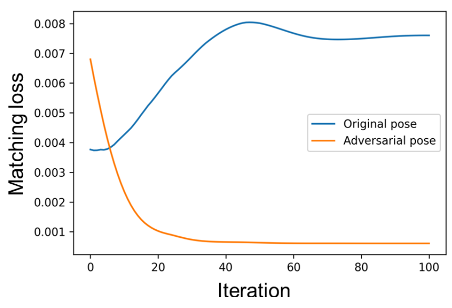


Fig. 11 Loss with original pose and adversarial pose.

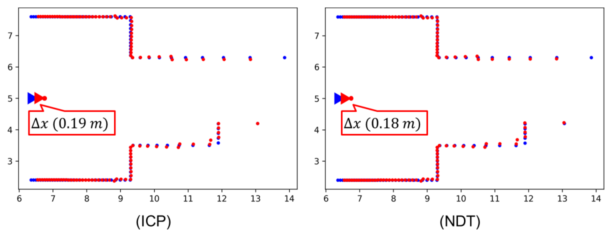


Fig. 12 Scan matching result of ICP and NDT under attack. Adversarial pose is $\Delta x = 0.19m$. The mask M is set from 0° to ±50°.

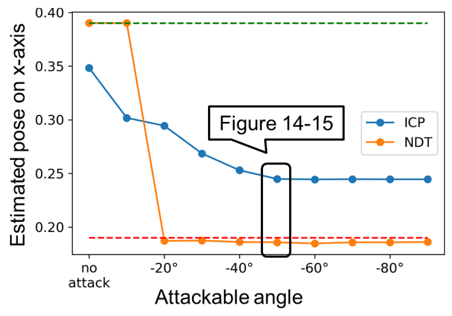


Fig. 13 The result of our attack while expanding the range of the mask from −10° to −90°.

estimated pose by ICP into shorter than the original pose, but the estimated pose did not reach the adversarial pose.

The following describes the details of one result from Fig. 13. The mask range which is from 0° to −50° covers the effective points estimated by the gradient. Figure 14 illustrates the mask and the difference between the current scan and the calculated adversarial scan. Figure 15 shows scan matching results by ICP and NDT algorithms by using

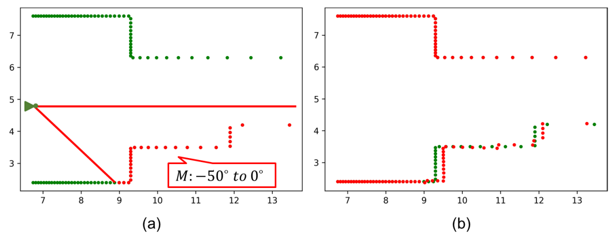


Fig. 14 (a) Mask M is set from 0° to −50°. (b) Difference between current scan (green) and adversarial scan (red).

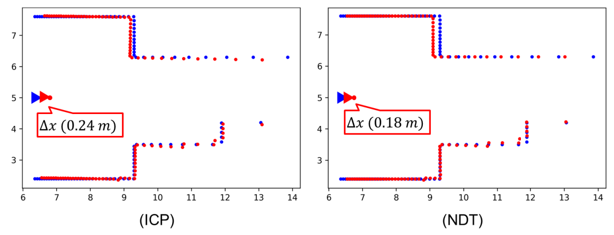


Fig. 15 Scan matching result of ICP and NDT under attack. Adversarial pose is $\Delta x = 0.19m$. The mask M is set from 0° to −50°.

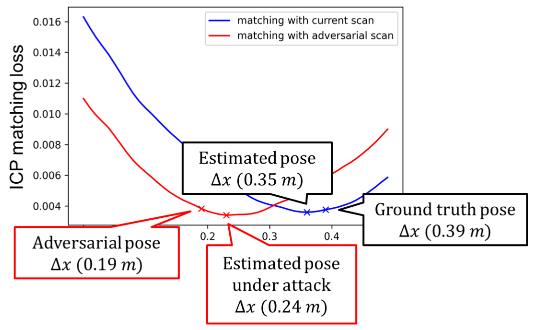


Fig. 16 ICP scan matching loss around ground truth pose by using the current scan (without attack) and the adversarial scan (with attack).

adversarial scan.

The reason why the estimated pose did not reach the adversarial pose is that the ICP scan matching loss at the adversarial pose could not satisfy the global minimum by limitation of the number of tampering points. Figure 16 shows the ICP scan matching loss around ground truth pose by using the current scan (without attack) and the adversarial scan (with attack). The loss curve was shifted to the left but the global minimum was $\Delta x = 0.24m$. The adversary tries to minimize the scan matching loss at the adversarial pose using Eq. (14) but it is not able to guaranteed the loss is whether the global minimum.

5.2.2 Attack against a Series of Scans

We evaluated our attack against a series of scans. The 2D-LiDAR SLAM [15] creates an occupancy grid map from the scans. We prepared two SLAMs that using the ICP and NDT scan matching respectively. The total ground truth robot transition was $4.61m$. Note that the ground truth robot pose was moving only in the x-axis direction. We set the

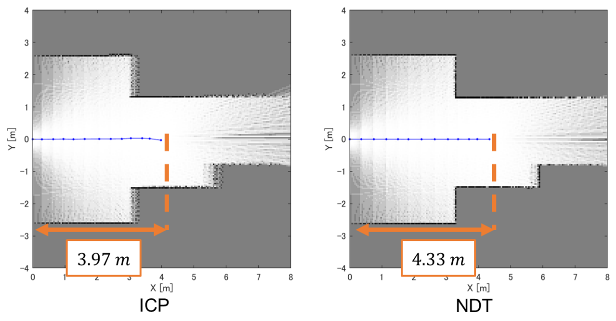


Fig. 17 Occupancy grids generated by SLAM with ICP and NDT scan matching. The total ground truth robot transition is $4.61m$.

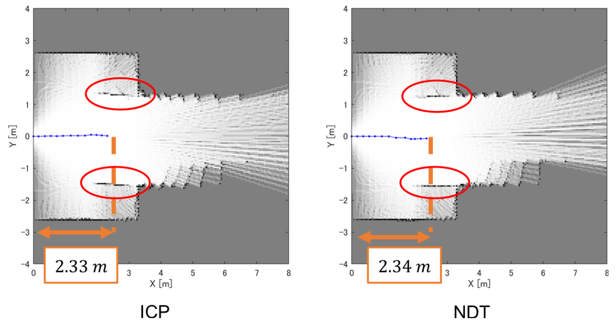


Fig. 18 Occupancy grids generated by SLAM with ICP and NDT scan matching under the attack. Occupancy grids generated by SLAM with ICP and NDT scan matching. The total adversarial robot transition is $2.3m$.

adversarial poses to 1/2 of the ground truth poses, a total transition was $2.3m$. We set the mask from 0° to ±50°. It is the same as Fig. 10(a).

Figure 17 shows the result of each SLAM with ICP and NDT. Figure 18 shows the result of each SLAM under the attack. Both of the SLAM were affected by the attack and the total travel distances were shorten. The estimated total transition of SLAM with ICP was $2.33m$ and with NDT is $2.34m$. They are close to adversarial pose. The shape of the corridor has significantly changed. Due to the robot travel distance was shorten, a virtual sidewall appeared at the red circle region on Fig. 18.

5.2.3 Odometry Sensor Fusion as Countermeasure

Odometry sensor fusion is one possible countermeasure against sensor attacks. Odometry can be acquired by a rotary encoder or inertial measurement units (IMUs). The Odometry sensor fusion improves the robustness of scan matching. Even if the scan matching algorithms estimate a wrong robot pose under our attack, the pose may be corrected by using odometry sensor fusion.

One of the simplest ways to fuse the odometry to the scan matching algorithm is to set the odometry as an initial pose. The scan matching algorithm can start the optimization step near the ground truth by using odometry.

We evaluated the scan matching-based pose estimation with odometry by using scan pairs from $t = 1$ and 2 similar to the experiment in Sect. 5.2.1. Figure 19 shows the result of

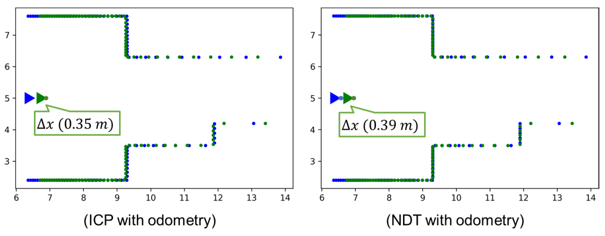


Fig. 19 Reference scan (blue) and current scan (green) overlaid with estimated pose by the ICP (left) and NDT (right) algorithms with odometry.

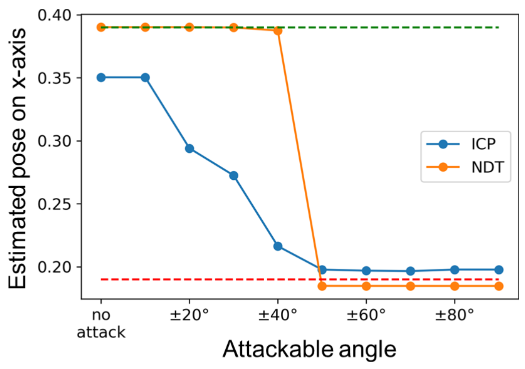


Fig. 20 The result of our attack while expanding the range of the mask M from 0° to ±90°. These scan matching algorithms were used odometry to initial pose.

ICP and NDT algorithms with odometry. The ICP algorithm estimated a pose $\Delta x = 0.35$ with or without odometry, but the ground truth pose is $\Delta x = 0.39$. When the ICP uses odometry sensor fusion, the ICP calculates the pose which satisfies the minimum scan matching loss from the given odometry. According to Fig. 16, the blue line is the scan matching loss around ground truth pose by using the current scan (without attack), and the scan matching loss at the estimated pose $\Delta x = 0.35m$ is global minimum. Therefore, the ICP reaches the estimated pose even if the ICP starts the search from the ground truth pose.

We evaluated our attack against scan matching algorithms with an odometry sensor fusion. We prepared odometry data from the ground truth. It assumes that accurate odometry can be obtained. The adversary is able to tamper with the range of the mask which is visualized in Fig. 10(a), from 0° to ±50°.

Figure 20 shows the result of our attack similar to Fig. 8 with ground truth odometry. The estimated results by ICP gradually closed to the adversarial pose by increasing the attack points. It indicates that ICP can not mitigate the effects of the attack by using odometry. The NDT estimated the original pose when the angle of the attack points is smaller than ±40°. It indicates that NDT can mitigate the effects of the attack by using odometry when the attack points are less. However, it affected by our attack when the attack points increase.

We also evaluated our attack with fewer attack points. The adversary is able to tamper with the range of the mask which is visualized in Fig. 14(a), from 0° to −50°. Figure 21

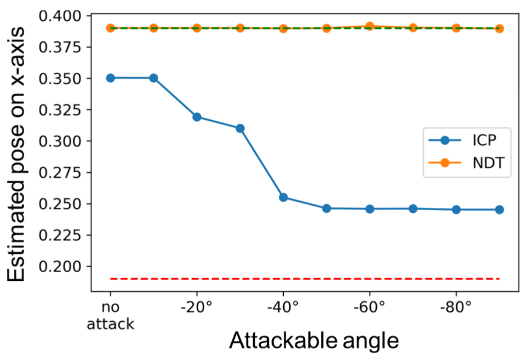


Fig. 21 The result of our attack while expanding the range of the mask M from 0° to −90°. These scan matching algorithms were used odometry to initial pose.

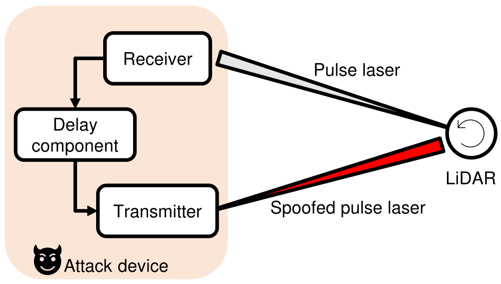


Fig. 22 Structure of typical attack device for physical spoofing attack.

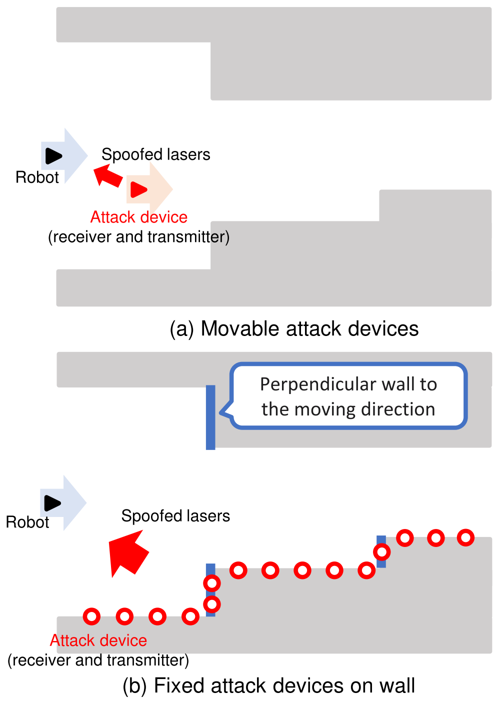

Robot
Spoofed lasers
Attack device
(receiver and transmitter)
(a) Movable attack devices
Perpendicular wall to the moving direction
Robot
Spoofed lasers
Attack device
(receiver and transmitter)
(b) Fixed attack devices on wall

Fig. 23 Expected attack device arrangements in actual attack scenario. (a) Adversary makes the attack device (red triangle) move in parallel with the target robot. (b) Adversary arranges attack devices (red) along the walls of the corridor. The adversary can also arrange attack devices with priority at the perpendicular wall to the moving direction (blue) of the target robot because the wall has a large effect to scan matching.

shows the result of our attack with ground truth odometry. The result of ICP is similar to Fig. 13 and it indicates that ICP can not mitigate the attack. On the other hand, the NDT estimated the original pose even if the adversary tampered with the scan points from 0° to −90°. These results suggest that odometry sensor fusion is a promising countermeasure for the NDT algorithm.

6. Discussion

In the above sections, we assumed that an adversary can tamper with the distance of any sample points from LiDAR scans in simulation. In this section, we discuss the basic case study for realizing an actual attack. The adversary should consider the limitations such as attack device arrangements and specifications in the actual attacks.

As shown in Fig. 22, physical spoofing attacks against LiDAR require an attack device that contains a receiver and transmitter [1], [2]. The receiver observes pulse laser from LiDAR for determining attack timing and after an appropriate delay, the transmitter irradiates a spoofed laser.

The adversary also needs to deploy multiple attack devices for satisfying all attack angles even if the target robot moves. There are two ways to deploy the attached devices in the real world as shown in Fig. 23; (a) the adversary makes a movable attack device that moves in parallel with the target robot, or (b) the adversary arranges attack devices at regular intervals along the walls of the corridor. In case(a), the attack angle is kept constant, then conditions as same as the simulation are satisfied if the movable target device can spoof the scan points around the mask M area. On the contrary, the attack angles of consecutive LiDAR scans change according to the movement of the target robot, when the position of the spoofed laser transmitter is fixed as shown in case(b). Many attack devices must be deployed on the corridor to satisfy the simulation conditions for deceiving the robot on the pathway from the start and the goal position. Placing many attack devices on the pathway get the attack cost larger. The idea for reducing the attack devices is that considering the effect of each sample point on the loss, which can be calculated using the gradient of $\partial\mathcal{L}/\partial\Delta l$ as shown in Fig. 9. The sample point on the perpendicular wall to the moving direction has the large effect on the pose estimation. Therefore, the attack device should be mainly deployed on the perpendicular walls rather than on the wall parallel to the moving direction. The future work is necessary for studying the detailed arrangement of attack devices.

7. Conclusion

Autonomous robots are controlled using physical information acquired by various sensors. Physical attacks on these sensors tamper with the observed values, interfering with control of the robots. Recently, sensor spoofing attacks that target control algorithms have become a growing threat.

In this paper, we presented a new attack against the LiDAR-based SLAM algorithm. An adversary tries to fool the scan matching algorithm in SLAM which is used for pose estimation. The pose is registered to the pose graph and is used to grow the submap with the current scan. The adversary calculates perturbation of the scan by using a gradient of loss function in the scan matching algorithm and performs a physical spoofing attack on the basis of the perturbation. When the LiDAR acquires an adversarial scan, the scan matching algorithm estimates the wrong robot pose, and the submap created by the wrong pose and adversarial scan.

We simulated 2D-LiDAR SLAM in a virtual corridor. We attacked and evaluated two typical scan matching algorithms, ICP and NDT. First, we attacked scan matching between a pair of scans and tampered with the estimated robot pose. Our attack fooled both the ICP and NDT scan matching algorithms. Next, we attacked a series of scans acquired by a moving robot. Our attack tampered with an estimated pose graph and a submap generated by SLAM. Finally, we evaluated a simple odometry sensor fusion algorithm as a potential countermeasure. The NDT algorithm was able to mitigate our attack when the adversary's attack points are less but it was affected when the attack points increase. The ICP algorithm was not able to mitigate our attack.

Our experimental results show that SLAM can be fooled by tampering with the scan. Simple odometry sensor fusion is a promising countermeasure for NDT but is not sufficient. We argue that it is important to detect or prevent tampering with LiDAR scans and to notice inconsistencies in multiple sensors caused by physical attacks.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Number JP18K11308.

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, no.2015, p.995, 2015.

[2] H. Shin, D. Kim, Y. Kwon, and Y. Kim, "Illusion and dazzle: Adversarial optical channel exploits against lidars for automotive applications," Cryptographic Hardware and Embedded Systems – CHES 2017, Lecture Notes in Computer Science, vol.10529, pp.445–467, Springer Verlag, 2017.

[3] Y. Cao, C. Xiao, B. Cyr, Y. Zhou, W. Park, S. Rampazzi, Q. Alfred Chen, K. Fu, and Z. Morley Mao, "Adversarial sensor attack on LiDAR-based perception in autonomous driving," Proc. 2019 ACM SIGSAC Conference on Computer and Communications Security, pp.2267–2281, 2019.

[4] Y. Cao, C. Xiao, D. Yang, J. Fang, R. Yang, M. Liu, and B. Li, "Adversarial objects against LiDAR-based autonomous driving systems," arXiv:1907.05418, July 2019.

[5] 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," Proc. 29th USENIX Security Symposium, pp.877–894, 2020.

[6] P.J. Besl and N.D. McKay, "A method for registration of 3-D shapes," IEEE Trans. Pattern Anal. Mach. Intell., vol.14, no.2, pp.239–256, 1992.

[7] P. Biber and W. Strasser, "The normal distributions transform: A new approach to laser scan matching," IEEE International Conference on Intelligent Robots and Systems, pp.2743–2748, 2003.

[8] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," 2nd International Conference on Learning Representations, ICLR 2014 - Conference Track Proceedings, Dec. 2013.

[9] I.J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings, International Conference on Learning Representations, ICLR, Dec. 2015.

[10] K. Lee, Z. Chen, X. Yan, R. Urtasun, and E. Yumer, "ShapeAdv: Generating Shape-Aware Adversarial 3D Point Clouds," arXiv:2005.11626, May 2020.

[11] D. Liu, R. Yu, and H. Su, "Adversarial shape perturbations on 3D point clouds," Computer Vision – ECCV 2020 Workshops, pp.88–104, Springer, Cham, Aug. 2020.

[12] A. Hamdi, S. Rojas, A. Thabet, and B. Ghanem, "AdvPC: Transferable adversarial perturbations on 3D point clouds," Computer Vision – ECCV 2020, Lecture Notes in Computer Science, vol.12357, pp.241–257, Springer Science and Business Media Deutschland GmbH, aug 2020.

[13] D. Liu, R. Yu, and H. Su, "Extending adversarial attacks and defenses to deep 3D point cloud classifiers," Proc. International Conference on Image Processing, ICIP, pp.2279–2283, IEEE Computer Society, Sept. 2019.

[14] J. Yang, Q. Zhang, R. Fang, B. Ni, J. Liu, S. Jiao, and Q. Tian, "Adversarial attack and defense on point sets," Technical Report, 2019.

[15] W. Hess, D. Kohler, H. Rapp, and D. Andor, "Real-time loop closure in 2D LIDAR SLAM," Proc. IEEE International Conference on Robotics and Automation, pp.1271–1278, June 2016.

[16] D.P. Kingma and J. Ba, "Adam: A method for stochastic optimization," arXiv:1412.6980, 2017.

Kota Yoshida received his B.E. and M.E. in electronic engineering from Ritsumeikan University in 2017 and 2019. He is currently a doctoral student at the Graduate School of Science and Technology, Ritsumeikan University. His research interests include machine learning and hardware security. He is a member of IEICE, IEEE.

Masaya Hojo received his B.E. in electronic engineering from Ritsumeikan University in 2021. He is currently a student at the Graduate School of Science and Technology, Ritsumeikan University. His research interests include machine learning and instrumentation security.

Takeshi Fujino was born in Osaka, Japan, on March 17, 1962. He received his B.E. and M.E., and Ph.D. in electronic engineering from Kyoto University, Kyoto, Japan, in 1984, 1986, and 1994. He joined the LSI Research and Development center, Mitsubishi Electric Corp. in 1986. Since then, he had been engaged in the development of micro-fabrication processes, such as electron beam lithography, and embedded DRAM circuit design. He has been a professor at Ritsumeikan University since 2003. His research interests include hardware security such as side-channel attacks and physically unclonable functions. He is a member of IEICE, IPSJ, IEEE.