Adversary is on the Road: Attacks on Visual SLAM using Unnoticeable Adversarial Patch

Baodong Chen, *The Ohio State University;* Wei Wang and Pascal Sikorski, *Saint Louis University;* Ting Zhu, *The Ohio State University*

https://www.usenix.org/conference/usenixsecurity24/presentation/chen-baodong

This paper is included in the Proceedings of the 33rd USENIX Security Symposium.

August 14–16, 2024 • Philadelphia, PA, USA

978-1-939133-44-1

Open access to the Proceedings of the 33rd USENIX Security Symposium is sponsored by USENIX.

Adversary is on the Road: Attacks on Visual SLAM using Unnoticeable Adversarial Patch

Baodong Chen[1,*], Wei Wang[2,*], Pascal Sikorski[2], Ting Zhu[1,†]
[1]*The Ohio State University,* [2]*Saint Louis University*

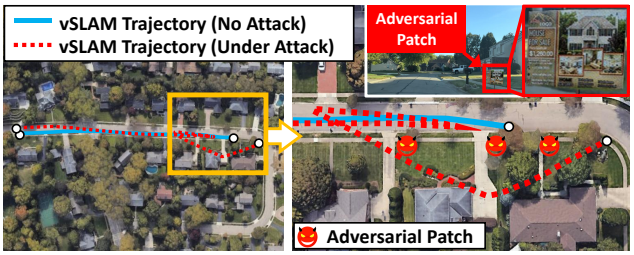
Abstract

Visual Simultaneous Localization and Mapping (vSLAM) plays a pivotal role in numerous emerging applications, including autonomous driving and robotic navigation. It mainly utilizes consecutive frames captured by image sensors to conduct localization and build high-definition maps. However, existing approaches mainly focus on building reliable and accurate vSLAM systems, while little work has been done to investigate the vulnerability of existing vSLAM systems.

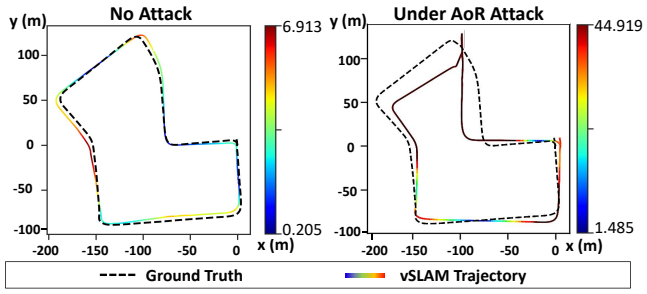
To fill the gap, we introduce an **AoR** (**A**dversary is **o**n the **R**oad) attack, which can effectively alter localization and mapping results of widely used vSLAM systems without being detected by the legitimate user. To do this, we conducted in-depth investigations on existing vSLAM systems and found that these systems are very sensitive to environmental texture changes. Building upon this insight, we design a novel adversarial patch generation mechanism that can generate unnoticeable adversarial patches to attack existing vSLAM systems. We extensively evaluate the effectiveness of the AoR attack on industry-level vehicles, robotic platforms, and four well-known open-source datasets. The evaluation results show that the AoR attack can effectively attack existing vSLAM systems and introduce extremely high localization errors (up to **713%**). To mitigate this attack, we also designed an innovative defense module to simultaneously detect abnormal environmental texture distributions and support reliable vSLAM. Our defense module is lightweight and has the potential to be applied to existing vSLAM systems.

1 Introduction

Simultaneous Localization and Mapping (SLAM) has emerged as one of the most important technologies to support real-time localization for various robotic applications, including autonomous driving [1–3], robotic navigation [4–6], and robot delivery [7–9]. To conduct SLAM, a variety of sensors, such as cameras [10, 11], LiDAR [12, 13], and radar [14, 15], are employed to sense and interpret the surroundings of the robot. Among these sensors, camera-based solutions (Visual SLAM) have distinct advantages, including low cost, high accuracy, and low system complexity. As a result, Visual SLAM (vSLAM) has been considered a practical solution and widely applied to many real-world applications [16–21].



(a) AoR Attack on a Real Vehicle in the Residential Area.



(b) Comparison of vSLAM Results on the KITTI Dataset.

Figure 1: Examples of the AoR attack on a real vehicle and KITTI Dataset: AOR attack can introduce huge errors to existing SLAM systems.

To build a vSLAM system, the key idea is to leverage the consecutively captured frames to estimate the motion and location of the camera. Then, according to the localization result, we can simultaneously construct the map using the captured camera frames. To improve the robustness and accuracy of vSLAM systems, researchers have introduced numerous novel techniques. Specifically, to design a holistic vSLAM system, one of the most well-known vSLAM algorithms, ORB-SLAM2 [22], introduced the first open-source vSLAM system for real-time and accurate localization and

*These authors contributed equally to this work.
†Corresponding author: Prof. Ting Zhu (zhu.3445@osu.edu)

mapping. Built on top of ORB-SLAM2, researchers have also introduced several novel vSLAM algorithms to improve the robustness of the vSLAM systems in different scenarios [23–26]. For example, DynaSLAM [26] can detect and filter out the moving objects in each frame to reduce the motion estimation error and improve localization accuracy in dynamic scenarios. Recently, ORB-SLAM3 [27] introduced a multi-map data association technique to further improve the accuracy and robustness of vSLAM systems.

However, existing techniques mainly focus on improving the performance of vSLAM systems, while little work has been done to study the vulnerability of these systems. Also, existing studies have only targeted specific components of vSLAM [28, 29] and often require customized hardware, performing attacks in limited scenarios [30]. Furthermore, they have not systematically analyzed the security issues in vSLAM, resulting in a significant gap in understanding potential threats and mitigation strategies.

In this work, for the first time, we comprehensively analyze the unique vulnerabilities within vSLAM and strategically exploit these weaknesses to do targeted end-to-end attacks. We discovered that the performance of the vSLAM system is intrinsically sensitive to various visual environmental factors, including texture distribution, lighting conditions, and the shapes and colors of objects. Changes in these factors can easily affect the localization and mapping accuracy. Furthermore, most of the current vSLAM systems treat all the features (e.g., key pixel points, key objects, etc.) in a frame equally without analyzing the current environment. Questions such as "Which feature is more reliable?" or "Is this feature real?" often remain unaddressed. For example, features susceptible to changing light conditions should be assigned lower importance since they may introduce high noise to vSLAM systems. However, in practice, to improve the reliability of vSLAM systems in low-texture environments, every feature in a frame is utilized for localization and mapping, which introduces inevitable errors to vSLAM systems.

Based on the above insight and analysis, we propose to attack the vSLAM systems by introducing adversarial features to existing vSLAM systems. We also aim to provide a defense strategy to mitigate this attack. To achieve this goal, we need to overcome three main challenges. **First**, existing vSLAM systems leverage various optimization techniques (e.g., Least-Square algorithm [31], RANSAC [32], etc.) to reduce errors introduced by the features in a frame. As a result, simply introducing random adversarial features may not attack vSLAM systems effectively. Therefore, we need to find the optimal adversarial features to effectively attack the vSLAM system. **Second**, most vSLAM systems are designed for autonomous vehicles and robots which operate in dynamic environments. Therefore, instead of attacking vSLAM in ***static*** scenarios, the proposed attack approach should stealthily attack existing vSLAM systems under ***dynamic*** environments. **Third**, we need to find a solution to identify the stealthy attack and mitigate its impact effectively.

To overcome the above challenges, we introduce the **AoR** (**A**dversary is **o**n the **R**oad) attack, which can effectively attack existing vSLAM systems without alerting legitimate users. The key idea of our attack is to inject unnoticeable adversarial patches into objects in the dynamic environment. These patches will introduce huge errors to vSLAM systems and can be seamlessly incorporated into common objects (i.e., advertising stickers in public spaces) without being detected by the naked eye. To generate these patches, we design a novel adversarial patch generation network to generate patches containing optimal and unnoticeable adversarial vSLAM features. Moreover, we also introduce a dynamic adjustment model to make the AoR attack robust under dynamic scenarios. Figure 1 shows examples of AoR attacks on a real vehicle platform and one of the most well-known autonomous driving datasets - KITTI [33]. As we can see from Figures 1 (a) and (b), the AoR attack introduces huge errors to the vSLAM system in both the real-world scenario and open-source dataset. In this case, robots or autonomous vehicles will get the wrong localization results and create a distorted map. Consequently, users relying on these inaccurate maps for navigation can be misled, leading to significant security risks.

To defend against the AoR attack, we propose an innovative and lightweight defense module that can identify abnormal patterns in each frame. To do this, we first convert each frame into the texture space to analyze potential adversarial patches. We then employ a variational auto-encoder model to recognize abnormal texture distributions. At last, we introduce a cross-frame optimization technique to dynamically select features in each frame for reliable localization and mapping.

Overall, our key contributions are summarized as follows:

- To the best of our knowledge, this is the ***first work*** that in-depth investigates the vulnerability of vSLAM systems. Specifically, we introduce an AoR attack, which injects unnoticeable adversarial patches into common objects. The proposed attack can effectively attack widely used vSLAM systems, introducing significant localization and mapping errors in dynamic settings without detection by legitimate users.
- We conduct extensive real-world experiments using industry-level vehicles and robot platforms under various scenarios and settings. The experiment results prove that our attack approach can attack widely used vSLAM systems and introduce harmful impacts.
- To defend against the AoR Attack, we also design a novel lightweight defense module that can detect the AoR attack and improve the reliability of vSLAM systems simultaneously. The experiment results show the effectiveness of our defense approach.

2 Vulnerability of vSLAM Systems

In this section, we first introduce and analyze the background and vulnerability of vSLAM systems. Then, we present real-world experiments to validate our findings.

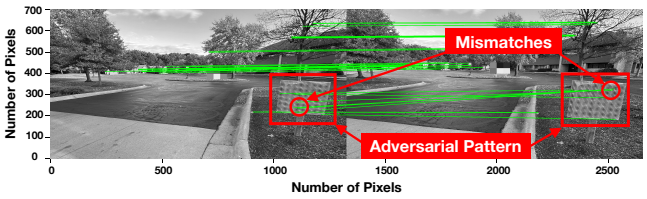


Figure 2: An example of mismatched features introduced by adversarial patterns.

2.1 Vulnerability Analysis of vSLAM Systems

The modern vSLAM system mainly consists of four core modules: (i) visual odometry (VO), (ii) backend optimization, (iii) loop closing, and (iv) map construction. The goal of VO is to estimate the motion and location of the camera between consecutive frames. The goal of backend optimization and loop closing is to reduce the noise and error introduced during the VO process. Based on the localization results, the map is then concurrently constructed using the frames captured by the camera in the map construction module.

In this paper, we found that ***the absence of a verification process during the VO process will introduce huge errors to vSLAM systems.*** Furthermore, we believe that a potential attacker may take advantage of this vulnerability to attack the vSLAM systems. Specifically, in the VO process, the first step is to extract distinctive visual features from the input images or video frames captured by the camera. These features may include corners, edges, or other salient points that can be reliably detected and tracked. Then, by mapping the same features across consecutive frames, we can find the position changes of these features and calculate the location and moving trajectory for autonomous vehicles or robots.

However, the ***first*** vulnerability issue arises as the VO process in vSLAM systems tends to select features solely based on their qualities without considering reliability, especially in environments with sparse textures. Moreover, there is no verification mechanism to validate the authenticity of these features. As a result, unreliable and untrustworthy features may be selected for localization and tracking, which introduces localization and mapping errors to vSLAM systems. The ***second*** vulnerability issue is that vSLAM systems are prone to mismatch features when pixels in a frame show similar intensity distributions. In some extreme cases, multiple selected features in one frame may be matched to a single feature in another frame. Then, during the backend optimization process, these mismatches will lead the vSLAM system in the wrong optimization direction, which introduces significant errors to vSLAM systems.

Based on the above analysis of these inherent vulnerability issues in vSLAM systems, a malicious attacker can easily attack vSLAM systems by injecting carefully designed adversarial patterns into each frame.

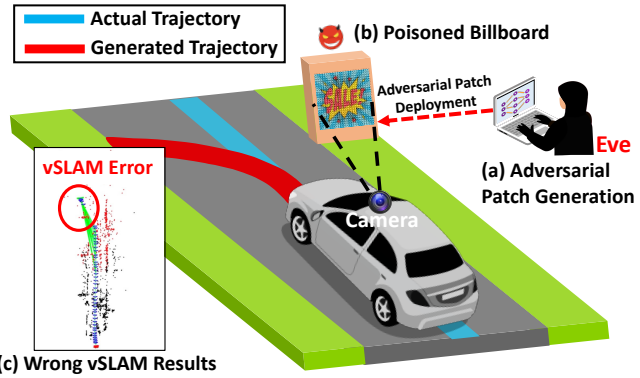


Figure 3: The Threat Model: (a) a malicious attacker (Eve) generates the unnoticeable adversarial patch and (b) deploys the patch on the billboard. In (c), the vSLAM system detects the key features in adversarial patches and outputs wrong localization and mapping results.

2.2 Preliminary Results of vSLAM Systems

To validate our analysis, we conduct a preliminary experiment in a real-world scenario in Figure 2. In this experiment, we generated an adversarial pattern containing numerous repetitive pixels. The grayscale values of these pixels are the same as each other but significantly different from the background pixels. We positioned this adversarial pattern at the roadside and utilized the implemented vSLAM system (ORB-SLAM2) [22] on our vehicle platform introduced in Section 5.1.1 for localization and mapping. During the experiment, we leveraged the widely-used ORB features [34] to extract key pixels from each frame, while leveraging the RANSAC algorithm [35] to filter out anomalies. As shown in this figure, key pixels outside the adversarial pattern are all correctly matched. However, since the features within the adversarial pattern are indistinguishable from one another, the VO process fails to accurately match these features between consecutive frames. This experiment not only proves the vulnerability of vSLAM systems but also points out a high-level strategy to attack the vSLAM system: ***the attacker can generate adversarial patterns that contain similar features to introduce errors to vSLAM systems.***

3 Models and Assumptions

In this section, we first describe the system model. Then, we introduce our threat model.

3.1 System Model

In this work, we mainly focus on investigating the vulnerability of commonly used vision-based SLAM (vSLAM) systems for autonomous vehicles and robotic applications. These vSLAM systems mainly utilize frames captured by cameras for real-time localization and mapping in both indoor and outdoor scenarios. Specifically, a vSLAM system will first select multiple key pixels in each frame as the key features during

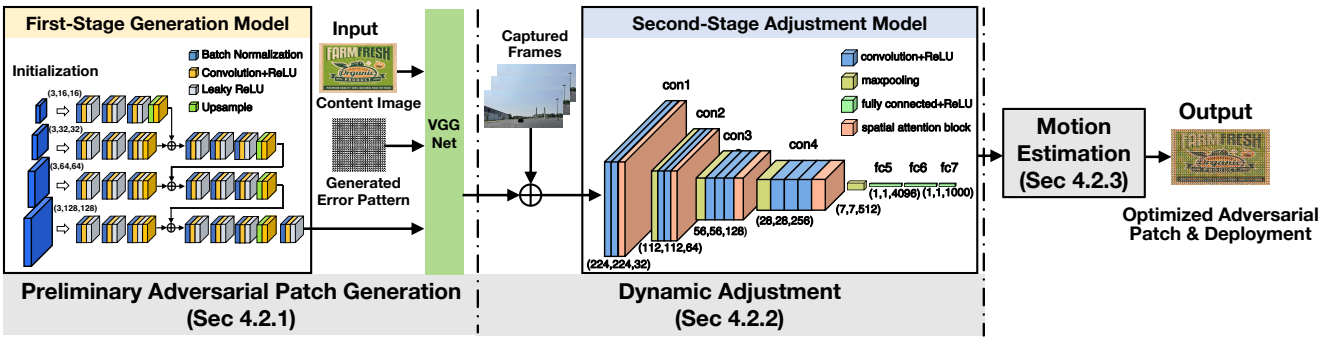


Figure 4: An overview of the AoR attack.

the visual odometry (VO) process. These selected features are then mapped and tracked across consecutive frames. Finally, based on the movement of these key features, the vSLAM system can conduct localization and mapping accurately.

3.2 Threat Model

Attack Goal. The goal of the AoR attack is to alter the localization and mapping results of vSLAM systems in autonomous vehicles or robots without being detected by the legitimate user. By doing this, a malicious attacker can potentially cause car accidents involving autonomous vehicles or divert the robots to fallacious destinations.

Assumptions. To orchestrate such an attack, we assume the attacker (Eve) can rent or buy the same hardware as the victims to know the potential attack environment and the victim's hardware platform (e.g., autonomous vehicle, robot, etc.). The attacker cannot access the victim's camera frames but can use the same hardware to collect the frames of the attack environment before launching the attack. We also assume the attacker has basic knowledge of machine learning techniques to generate optimal adversarial patches using the collected frames, but no additional computer science expertise is required to execute the attack. Then as long as the victim uses commonly-used vSLAM systems [22, 24–27], the AoR attack can effectively alter the localization and mapping results without detection by the legitimate user. The AoR attack is under a **grey-box** access assumption, where the attacker knows that the victim employs the target vSLAM system but does not have any internal technical details of the victim's system. We consider this assumption reasonable because the attacker can infer such information through preliminary experiments (Figure 2) to see if high-texture patches can affect the system's performance.

To attack the vSLAM systems in real-world scenarios, an attacker can implement unnoticeable adversarial patches on common real-world objects (e.g., billboards). While these patches appear harmless to humans, they deceive vSLAM systems used for localization in autonomous vehicles or robots. As a result, the AoR attack can stealthily introduce high localization and mapping errors to vSLAM systems. For example, as shown in Figure 3, the attacker generates and deploys adversarial patches on roadside billboards. Then, the poisoned billboard containing unnoticeable features can be easily detected by vSLAM systems in autonomous vehicles or robots and introduce errors to vSLAM results.When the victim system relies on such inaccurate vSLAM maps for decision-making, it may result in collisions with nearby objects.

4 Design of the AoR Attack

In this section, we first introduce the design overview of the AoR attack. Then, we describe the design in detail.

4.1 Design Overview

The AoR attack design, shown in Figure 4, comprises three modules: **Preliminary Adversarial Patch Generation**, **Dynamic Adjustment**, and **Motion Estimation**.

The objective of preliminary adversarial patch generation is to produce adversarial patches that contain unnoticeable adversarial features. To achieve this goal, we first select images suitable for roadside deployment as the base content image. Then we embed an ideal error pattern image that can introduce mismatches to vSLAM systems into the content image. The error pattern is designed based on the feature extraction mechanism in the VO process of vSLAM systems. To do the embedding, we design a first-stage generation model to merge the content image and the error pattern.

In the second-stage adjustment model, dynamic adjustment is designed to optimize the adversarial pattern using environmental information and maximize the attack's effectiveness. Specifically, we found that the effectiveness of the adversarial patch may vary according to different attack scenarios. Therefore, the attacker should conduct optimization based on the potential attack scenarios before attacking the target autonomous vehicle or robot. To do this, the attacker can use the same hardware to collect frame sequences of the target attack scenario. Then, the dynamic adjustment module will optimize the adversarial patterns from the preliminary adversarial patch generation module based on the captured frames. It will optimize the inner texture to make the adversarial patch targeted to the designated attack scenario while remaining benign in other situations.

We also design a motion estimation module to roughly estimate the potential movement of the target autonomous

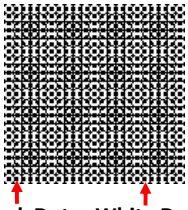


Figure 5: The ideal error pattern contains repetitive black and white dots.

vehicle or robot. Using these processes, we can find the best locations to deploy multiple adversarial patches in the target attack scenarios.

4.2 Detailed Design

In this section, we first introduce the preliminary adversarial patch generation module. Then, we show the detailed design of the dynamic adjustment module. At last, we introduce the motion estimation technique.

4.2.1 Preliminary Adversarial Patch Generation

The goal of preliminary adversarial patch generation is to create adversarial patches that can introduce vSLAM errors without being detected by legitimate users. To achieve this goal, as shown in Figure 4, we design a first-stage generation model to efficiently introduce adversarial features into the content image.

Error Pattern Design. To produce an effective error pattern, the features of the pattern must be both detectable by vSLAM systems and capable of causing mismatches over successive frames. Specifically, we generate the error pattern based on the principle of the FAST (Features from Accelerated Segment Test) [36] key feature selection algorithm which is commonly used in industry. The algorithm identifies key features by analyzing brightness differences around specific pixels, marking a pixel as a corner point if its surrounding pixels are significantly brighter or darker. As depicted in Figures 2 and 5, we created an error pattern of alternating black and white dots, distinct from the background. Because these dots are so different, the vSLAM system is likely to recognize them as key features. Moreover, due to the similarity of these dots, vSLAM systems struggle to accurately match the selected key features across consecutive frames, which introduces high errors to vSLAM systems.

Content Preservation. To attack the vSLAM system stealthily, the final adversarial patch should also look benign to legitimate users. To achieve this goal, we embed the ideal error pattern on commonly seen objects (e.g., advertisements on billboards) to hide the error pattern. Specifically, we first select multiple advertisement posters as foundational content images that are contextually suitable for roadside use. We then employ a multi-scale feed-forward neural network as the first-stage generation model to embed the ideal error pattern into the advertisement posters seamlessly. We also use a pre-trained VGG-16 model [37] as our fixed backbone to facilitate the training phase.

To preserve the high-level information (e.g., shape, spatial relationships, etc.) of the content image in the first-stage generation model, we introduce a content loss function L_c that calculates the similarity between the final generated adversarial patch x' and the original content image x_0:

$$L_c(x', x_0) = \sum_{i=1}^{N} [F_i^l(x') - F_i^l(x_0)]^2 \quad (1)$$

In the above equation, $F_i^l(x)$ represents the $i-th$ feature map computed by the backbone network applied to image x at the $l-th$ convolutional layer and N is the number of feature maps in layer l. In practice, it has been proven that the higher layers of the network correspond to high-level information about the image, while the lower layers affect pixel-level details [38]. In this step, since our goal is to preserve the ***high-level*** information of the content image so that the final generated adversarial patch looks benign to the naked eye, we utilize the inner product of ReLU activations from the fourth layer ($l = 4$) of the VGG-16 backbone network to calculate the loss function L_c.

Adversarial Features Integration. To ensure the features in the final adversarial patch effectively introduce vSLAM errors, we introduce a feature loss function L_f to optimize the discrepancy between the generated patch and the ideal error pattern in the feature space. Here, the feature loss function L_f uses lower layers ($l = 1, 2$) of the VGG-16 model and focuses on capturing ***pixel-level*** information, such as grayscale distribution and fine textures. Specifically, the feature loss function L_f can be calculated as:

$$L_f(x', x_f) = \omega_l \sum_{l \in L_F} \sum_{i,j} \left[G_{ij}^l(x') - G_{ij}^l(x_f) \right]^2 \quad (2)$$

Here, x_f is the ideal error pattern. L_F denotes the selected layers in the backbone network, and the weight assigned to each layer is ω_l. The Gram matrix, $G_{ij}^l(x)$, calculated by multiplying the feature maps i and j in layer l, models the correlation between different feature maps in the network. Finally, we generate the optimal adversarial patches by using the high-level loss function L_c and the pixel-level loss function L_f.

4.2.2 Dynamic Adjustment

The preliminary adversarial patch generation module produces optimal adversarial patches that can work effectively in ideal attack scenarios. However, in real-world scenarios, environmental factors (e.g., the texture and color of surrounding objects, the density of objects on the road, etc.) may also affect the attack results. For example, if the texture distribution of the target environment deviates too far from the normal distribution, then the effectiveness of the adversarial patch may be diminished. To address this challenge, we introduce a dynamic adjustment module that guides the generation network in finding the optimal adversarial patch according to real-world environmental factors. The high-level idea of the dynamic adjustment module is to dynamically adapt the internal feature distribution of the generated patch in response to environmental factors.

Adjustment with Environment Loss. As depicted in Figure 4, our second-stage adjustment model employs a convolutional neural network (CNN) with attention modules. During the dynamic adjustment, this model integrates the optimal adversarial patch with frames reflecting environmental factors to create a sequence of poisoned frames. Then, the poisoned and original frame sequences are fed into the adjustment model. This adjustment model aims to maximize the difference between the poisoned and the original frames in the feature space according to the attack scenario. To achieve this, we design an environment loss function L_e that **(i)** measures the difference between the poisoned and the original frames, and **(ii)** measures the feature difference between adjacent poisoned frames simultaneously. Formally, the environment loss function L_e can be formulated as:

$$L_e = -\omega_e \sum_{e \in L_E} ([M^e(F(y)) - M^e(F(x'+y))]^2 - [F^e(x'+y) - F^e(x'+y_p)]^2) \quad (3)$$

where x' is the optimal adversarial patch, y is the captured camera frame and y_p is the previous frame of y. It's important to highlight that y and y_p are frames captured by the attacker's camera, not the victim's. They are collected before the attack to find the most effective adversarial patch for the target scenario. The combination $x'+y$ represents the poisoned frame. $F(x'+y)$ is the output from each layer in the adjustment model, while $M(F(x'+y))$ is the attention map. The set of the selected layers in the backbone network is represented by L_E, while the weight of each layer is denoted by ω_e. By using the loss function L_e, the adjustment model effectively identifies regions that have dense feature points in a frame.

Enhance Robustness. To enhance the attack's robustness to various environmental conditions in a genuine driving environment, we collect frames of the attack environment at different times under diverse weather and lighting conditions. Additionally, to address the differences in trajectory between the victim and the attacker as they navigate through the attack scenario, we employ perspective warping to adjust the perspective of the collected frames for further training.

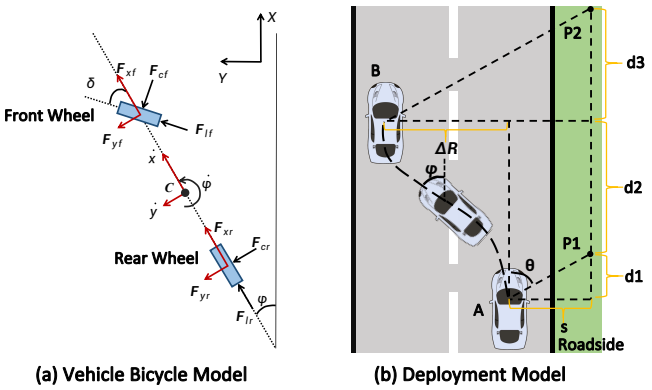
At last, a total loss function L_{total} combines the content loss, feature loss, and environment loss with weighted parameters, which can be expressed as:

$$L_{total}(x_0, x_f, x') = \alpha L_c + \beta L_f + \gamma L_e \quad (4)$$

We optimize our model using the Adam optimizer, setting an initial learning rate of 0.001 and beta coefficients at 0.9 and 0.999. To combat overfitting, we apply dropout with a rate of 0.5 and L2 regularization at a lambda value of 0.01. The training is conducted over 50 epochs with a batch size of 64, utilizing NVIDIA RTX 3080 Ti GPUs.

4.2.3 Motion Estimation

To deploy the adversarial patch and improve the effectiveness of the AoR attack in real-world scenarios, it is also important to ensure the victim's camera can see the adversarial patch.



(a) Vehicle Bicycle Model (b) Deployment Model

Figure 6: The bicycle and deployment models used for adversarial patch deployments.

We use the well-known bicycle model [39] to estimate the potential motion of the target vehicle or robot. This model, illustrated in Figure 6, defines key parameters: δ for the steering angle, C for the center of mass with front and rear distances l_f and l_r, and φ for the angle between the vehicle's longitudinal axis and the X-axis. It also includes forces such as lateral tire forces F_{cf} and F_{cr}, air drag F_a, and rolling resistance F_r. Additionally, it considers the vehicle's mass m and inertia I. The detailed bicycle model can be represented as:

$$\begin{cases} \dot{X} = v_x\cos(\varphi) - v_y\sin(\varphi) \\ \dot{Y} = v_x\sin(\varphi) + v_y\cos(\varphi) \\ \dot{\varphi} = \dfrac{v_x}{l_f + l_r}\tan(\delta) \\ m\dot{v}_x = F_x + mv_y\dot{\varphi} - 2F_{cf}\sin(\delta) - F_a - F_r \\ m\dot{v}_y = -mv_x\dot{\varphi} + 2\left(F_{cf}\cos(\delta) + F_{cr}\right) \\ I\ddot{\varphi} = 2\left(l_f F_{cf}\cos(\delta) - l_r F_{cr}\right) \end{cases} \quad (5)$$

Typically, the maximum steering angle of a vehicle is less than 40° [40]. An attacker can acquire additional vehicle or robot parameters by renting or purchasing a model identical to the target. According to this bicycle model, we can estimate the motion of the target vehicle or robot and find the optimal location to deploy the adversarial patches. As illustrated in Figure 6 (b), the vehicle transitions from $P1$ to $P2$, altering lanes from A to B. The distances involved are $d1$ (from A to $P1$), $d2$ (from $P1$ to B), and $d3$ (from B to $P2$). Analyzing these distances using the camera's angle of view, set around 62.5° [41], allows for calculating the maximum transverse (ΔR) and longitudinal (ΔR_x) displacements of the vehicle during the time window T:

$$\Delta R = \int_0^T (v_x\sin\varphi + v_y\cos\varphi)\,dt \quad (6)$$

Similarly, maximum movement ΔR_x of the victim in X direction can be estimated as:

$$\Delta R_x = d_2 + \frac{s}{\tan\theta} = \int_0^T (v_x\cos\varphi - v_y\sin\varphi)\,dt \quad (7)$$

(a) Experimental Setup (b) Campus (c) Rural Area (d) Warehouse (e) City Center (f) Parking Garage

Figure 7: The implementation of AoR attack and attack scenarios.

Based on the above equations, the distance between the adversarial patch and the victim should be larger than ΔR_x to ensure that every patch appears on the victim's camera.

5 Evaluation

To evaluate the effectiveness of the AoR attack, in this section, we conduct extensive experiments in real-world scenarios using a vehicle and a robotic platform. Then, we also study the effectiveness of the AoR attack using multiple well-known open-source datasets.

5.1 AoR Attack on the Vehicle Platform

In this subsection, we first introduce the experiment setup. Then, we show the AoR attack results on the vehicle platform.

5.1.1 Experiment Setup

vSLAM Platform. We built a standard autonomous vehicle testing platform based on a BMW X3, which is shown in Figure 7 (a). Specifically, we deployed an industry-level Orbbec Astra Pro camera [42] and implemented the target vSLAM algorithms on this platform to conduct localization and mapping. We also adapted vSLAM algorithms using BMW's parameters to handle real-world challenges, including vibration, slip, and chassis dynamics.

vSLAM Selection. We selected five popular vSLAM algorithms that are widely used in autonomous driving [16–19]. These algorithms include ORB-SLAM2 [22], ORB-SLAM3 [27], DynaSLAM [26], pl-SVO [24], and SVO Pro [25]. These algorithms deliver excellent performance with high accuracy and modest hardware resource requirements [43, 44].

Attack Execution. Under the grey-box assumption, the attacker first used the same vehicle platform as the victim to capture frames in the attack environment. These frames were then used to generate the optimal adversarial patch with our generation framework. The attacker placed the generated patches on self-made billboards and determined the optimal locations for deploying these billboards by the roadside using our proposed motion estimation approach. Once the patch was deployed at the roadside, victims using the same platform passed through the attack environment to assess the effectiveness of our attack. We conducted experiments in various real-world scenarios, including a campus, rural area, warehouse, city center, and parking garage, as depicted in Figure 7 to fully assess the AoR attack. Each scenario varied from $20m$ to $80m$. We also explored the impact of the patch configuration and various environmental conditions, with the results presented in Section 5.2. To accurately evaluate the performance of the AoR attack, we used a Real-Time Kinematics positioning device [45] to provide ground truth data and calculated the average Root Mean Square Error (RMSE) [46] within each scenario to assess the vSLAM errors.

5.1.2 The AoR Attack Results

Since the evaluation results show similar trends, we mainly show the attack results on ORB-SLAM3. The full results are presented in Appendix A.2. Figure 8 shows the vSLAM errors introduced by the AoR attack in each scenario. The AoR attack significantly impacts the accuracy of vSLAM systems, as illustrated in the figures. Under normal conditions, the ORB-SLAM3 system can conduct relatively accurate localization and mapping and achieves a maximum localization error of 13.92m in rural areas. However, when the ORB-SLAM3 system is under attack, the vSLAM errors increase to $25.40m$ in rural areas, which is around **two times** as high as that of the no-attack scenario. Moreover, the increase in vSLAM errors is particularly noticeable in the warehouse scenario, where the RMSE escalates by 75%. This significant rise is attributed to the sparse textures and limited distinct features in both rural and warehouse environments. The camera has to rely more on features from the adversarial patches, thereby magnifying the vSLAM errors. It should be noted that the RMSE exceeds 13 meters across all scenarios. Given that road widths are typically less than 12 feet (3.7m) in real driving environments, such substantial errors can lead to severe outcomes, such as causing the vehicle to collide with others or veer off the road when navigating based on a distorted vSLAM map. These inaccuracies highlight the potential risks of car accidents in practical applications. **In summary,** the AoR attack can introduce high errors to vSLAM systems in various scenarios and introduce serious security risks to legitimate users.

5.2 Insight and Impact Analysis

To study the insight and impact of the AoR attack, we conduct experiments by changing different influencing factors, including the dynamic adjustment module, the number, size, orientation, and height of adversarial patches, along with different environmental conditions.

5.2.1 Effectiveness of the Dynamic Adjustment Module

To study the impact of our dynamic adjustment module, we implement a baseline attack method (named w/o DA). This method only uses the preliminary adversarial patch module

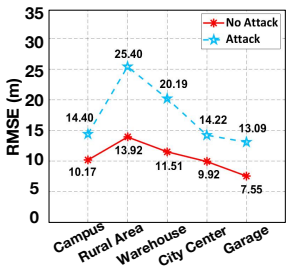


Figure 8: vSLAM errors of the AoR attack in different scenarios.

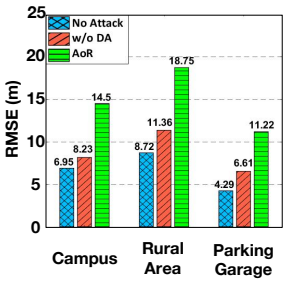


Figure 9: Effectiveness of the Dynamic Adjustment.

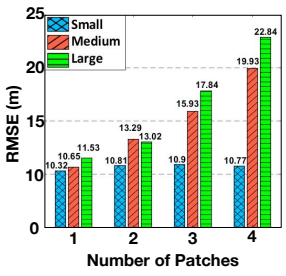


Figure 10: Impact of number and size of patches.

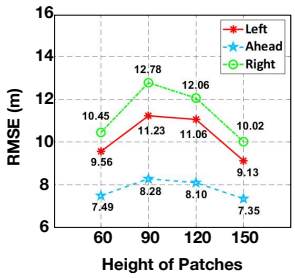


Figure 11: Impact of height and direction of patches.

Table 1: The vSLAM errors introduced by the AoR attack with various environmental conditions.

	Lighting condition			Weather			Moving objects		
	low	moderate	bright	sunny	light rain	heavy rain	< 5	5 - 10	> 10
ORB-SLAM2	9.25 / 14.31	3.22 / X	3.06 / 12.20	**2.59 / 10.27**	**14.10 / 19.82**	X / X	5.70 / 17.36	9.24 / 16.33	10.74 / 12.05
DynaSLAM	**8.16 / 15.33**	4.48 / 16.30	4.50 / 15.59	3.63 / 11.50	14.11 / X	X / X	3.14 / 15.20	**6.29 / 13.85**	**7.28 / 10.00**
ORB-SLAM3	9.34 / 15.66	**3.92 / 14.40**	**3.45 / 14.75**	2.59 / 9.40	10.82 / 14.20	9.10 / X	**3.26 / 16.06**	10.04 / 14.22	13.20 / 14.05
pl-SVO	10.63 / 15.80	13.92 / 20.10	15.49 / X	9.32 / 16.28	14.01 / X	X / X	12.84 / 19.66	12.65 / 18.53	20.16 / 23.99
SVO Pro	18.89 / 21.51	10.55 / 20.23	10.23 / 20.56	14.20 / 19.57	20.90 / 27.35	22.19 / X	8.22 / 15.29	9.10 / 13.96	18.73 / X

* For each A/B, A is the vSLAM error without the attack and B with the attack. Bold values highlight the largest change (B/A) per column.
* X means the track is lost.

to produce the adversarial patches without considering the real-world attack scenarios. We show the attack results in the rural area, warehouse, and parking garage in Figure 9.

As we can observe from the figure, the baseline attack method can only slightly affect the vSLAM results. This is because environmental factors will affect the VO process in vSLAM systems. As a result, the camera may not select the features on the adversarial pattern. On the other hand, after applying the dynamic adjustment (DA), the performance of the baseline method improved on average by 60%. This improvement is because the DA module adjusts adversarial patches dynamically based on environmental conditions. **In summary,** the dynamic adjustment module can significantly improve the effectiveness of the AoR attack.

5.2.2 Impact of the Patch Numbers and Sizes

In our experiments, we also explore the effects of varying the number and size of adversarial patches. Patch sizes are defined as Small (20×20 inches), Medium (30×30 inches), and Large (40×40 inches). We varied the number of patches placed by the roadside from 1 to 4, positioning all patches on the same side of the road.

In Figure 10, we show the vSLAM errors of ORB-SLAM3 under each setting. Notably, medium and large patches yield high errors, while the errors are significantly lower with small patches. This is because the smaller patches occupy a smaller area in the victim's camera and limited pixels in the camera can be used to render the adversarial patches. As a result, fewer features are selected from the patches, reducing the attack's effectiveness. Therefore, attackers should ensure that patch sizes are large enough for real-world effectiveness. We believe that this is a reasonable requirement since a standard advertising billboard is much larger than 20×20 inches [47]. We also notice that the vSLAM errors increase rapidly as the number of adversarial patches increases in most cases. This is because more patches will introduce severe mismatches across consecutive frames. In this case, the vehicle keeps capturing the poisoned frames, which increases the vSLAM errors. Interestingly, even if small patches collectively cover the same area as medium patches, their impact is less significant. This reduced effectiveness arises because features on a larger patch are more likely to be selected and mismatched when they are concentrated together.

5.2.3 Impact of the Patch Directions and Heights

In Figure 11, we study the effectiveness of both the patch direction and height. Throughout our experiments, adversarial patches of varying heights were positioned in distinct directions relative to the target vehicle: on its left, on its right, and directly in front of it. As shown in this figure, we can observe that the optimal heights of the adversarial patch are from 90*cm* to 150*cm*. When the height is reduced to 60*cm* or increased to 150*cm*, the camera cannot fully capture the adversarial patch. As a result, the vSLAM errors introduced by the AoR are reduced. Counterintuitively, we also notice that the AoR attack introduces higher vSLAM errors when the adversarial patches are not deployed in front of the target vehicle. This is because the features along the roadside change faster than the features in the camera's center of view as the vehicle moves on the road. This finding demonstrates

(a) Corridor (b) Office (c) Library (d) Classroom (e) Shopping Mall

Figure 12: AoR attack on the robotic platform in indoor scenarios.

the effectiveness of the AoR attack. **In summary**, the attacker can easily achieve high AoR attack effectiveness by simply deploying the adversarial patches by the roadside.

5.2.4 Robustness to Environmental Conditions

To evaluate the practicality of the AoR attack, we conduct experiments under various environmental conditions, including different lighting and weather conditions, as well as the presence of moving objects. To assess the impact under various lighting conditions, we performed the attack in low (<1000 lux), moderate (1000 to 10,000 lux), and bright light (>10,000 lux) environments. Regarding weather conditions, we evaluated the effects during sunny, light rain, and heavy rain scenarios. Additionally, we examined the influence of moving objects on the AoR attack by considering three different levels of moving object density: fewer than 5, between 5 and 10, and more than 10. In Table 1 of the evaluation results, we observe that extreme conditions such as heavy rain and a large number of moving objects may degrade the performance of the attack due to reduced visibility of the patch. However, vSLAM errors still increase significantly compared to scenarios with no attack, demonstrating the AoR attack's robustness across various environmental conditions.

5.3 AoR Attack on the Robotic Platform

To extensively study the effectiveness of the AoR attack, we also evaluate a robotic platform in indoor scenarios.

5.3.1 Experiment Setup

vSLAM Platform and vSLAM Selection. To evaluate the effectiveness of the AoR Attack on a robotic platform, we built a robotic navigation platform based on the ROSMASTER R2 [48] for indoor scenarios. We implemented vSLAM algorithms including ORB-SLAM2 [22], ORB-SLAM3 [27], DynaSLAM [26], pl-SVO [24], and SVO Pro [25] on the platform for evaluation since they are widely used in robot navigation [20, 21].

Attack Execution. To perform the AoR attack on the robotic platform, the attacker first used the same type of robotic platform as the victim to collect frames within the attack scenario under the grey-box assumption. Following this, the attacker created adversarial patches and deployed them in the environment. The victim robot then navigated through this setup to assess the effectiveness of the attack. We conducted experiments in various real-world scenarios, including a corridor, office, warehouse, library, classroom, and shopping mall, as depicted in Figure 12 to fully assess the AoR attack. Each scenario varies from $5m$ to $20m$. To quantitatively assess the attack, we utilized an Ultra-Wideband (UWB) module to provide accurate indoor positioning as ground truth to measure the Root Mean Square Error (RMSE).

Table 2: The vSLAM errors when the robotic platform is under the AoR attack in indoor scenarios.

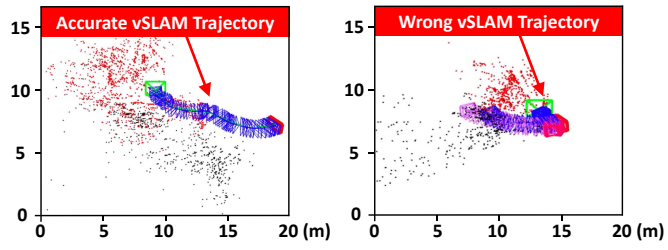
Scenarios	Corridor	Office	Library	Classroom	Mall
ORB-SLAM2	2.15 / X	**1.42 / 3.78**	2.93 / 7.20	1.90 / 4.33	2.79 / 5.20
DynaSLAM	1.98 / 5.77	1.60 / 4.08	**2.34 / 6.27**	1.87 / X	**2.53 / 5.59**
ORB-SLAM3	**1.52 / 7.02**	1.37 / 3.26	2.59 / 5.81	**1.46 / 5.96**	2.19 / 4.77
pl-SVO	5.50 / X	5.31 / 9.88	7.29 / 11.30	4.84 / 7.50	6.92 / 12.18
SVO Pro	5.39 / 9.16	4.07 / 7.59	7. 58 / 10.05	4.29 / 6. 55	7.03 / 14.11

* For each A/B, A is the vSLAM error without the attack and B with the attack. Bold values highlight the largest change (B/A) per column.
* X means the track is lost under the AoR attack.

Safety-Critical Scenario. For safety reasons, we demonstrate the specific safety-critical consequences of the AoR attack using a robotic platform in a $2m$ wide corridor, as depicted in Figure 14. Specifically, the attacker placed a 12×12 inch adversarial patch on a trash bin to attack the robot. We set a destination for the robotic platform $8.5m$ away from the starting location. The victim utilized the ORB-SLAM3 algorithm for indoor localization and mapping and navigated based on the vSLAM results to see the impact of the attack. To establish a baseline for the robot's performance without attack, we also use the robotic platform for navigation in the same environment without attack.

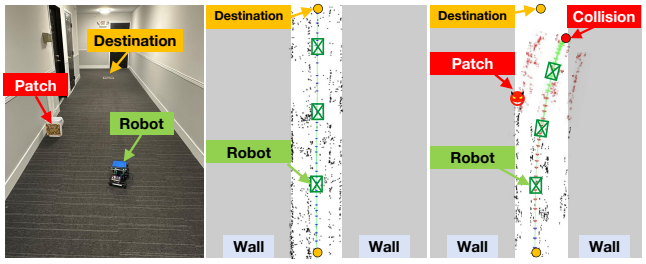
5.3.2 The AoR Attack Results

Table 2 shows the significant increase in vSLAM errors when the robotic platform is under the AoR attack in indoor scenarios. This is because the adversarial patch can continuously introduce mismatches and errors to the vSLAM systems. For example, in the corridor scenario, the RMSE of ORB-SLAM3 reaches 7.02 under the AoR attack, which is more than **4.62 times** higher than that of the no-attack scenario (1.52). Figure 13 illustrates the corresponding localization and mapping results in this scenario. As we can see from this figure, the vSLAM system accurately locates and maps the environment without attack. When the vSLAM system is under attack, it suffers from high localization errors. As a result, the vSLAM system believes that it is at the same location in the corridor due to the mismatches introduced by the adversarial patches.



(a) vSLAM results without attack. (b) vSLAM results under attack.

Figure 13: The AoR attack on the robotic platform in the corridor scenario. The green box is the current frame, and the blue boxes are past frames. Other colored boxes indicate frames relocalized after losing track.



(a) Attack scenario (b) Trajectory (no attack) (c) Trajectory (attack)

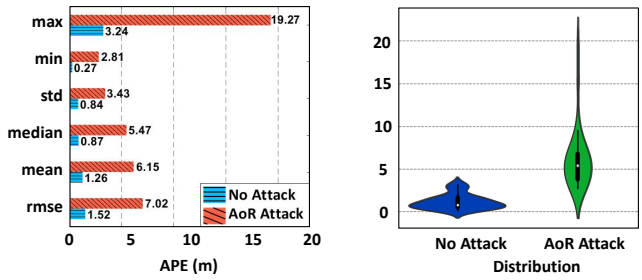
Figure 14: The safety-critical scenario experiments.

Figure 15 shows the corresponding translation errors when ORB-SLAM3 is under the AoR attack. This error defines the distances between the generated vSLAM trajectory and the ground truth driving trajectory. As seen in Figure 15 (a), the maximum translation error is as high as 19.27m. Given that the width of the corridor is less than 3m, the vSLAM system believes that the robot is actually outside the building. Figure 15 (b) illustrates the variation in translation errors, showing that while the Absolute Pose Error (APE) normally stays below 5m without an attack, it can exceed 20m in severe cases under attack.

Figure 14 illustrates the severe safety consequences of an AoR attack on a victim robot. In Figure 14 (b), the robot navigates safely to its destination in a straight line. However, as shown in Figure 14 (c), the robot believes the corridor ahead is curved after it encounters the adversarial patch on its road to the destination. Consequently, it started to turn to the wall according to this altered vSLAM map instead of moving straight. The vSLAM trajectory (green line) intersects with the corridor wall (grey), indicating a collision. In real-world applications, a delivery robot relying on such a distorted vSLAM map could be diverted to incorrect destinations and become trapped. **In summary,** the AoR attack can effectively attack vSLAM systems on robotic platforms in various indoor scenarios and introduce serious security risks.

5.4 AoR Attack on Public Datasets

In this section, we conduct experiments to study the performance of the AoR attack using well-known public datasets,



(a) Translation errors. (b) Distribution of translation errors.

Figure 15: Translation errors introduced by AoR attack in the corridor scenario.

including KITTI [33], RobotCar [49], 4Seasons [50], and Complex Urban [51].

5.4.1 Experiment Setup

The target public datasets, collected in real-world scenarios, feature diverse environmental conditions including lighting, weather, and moving objects, which reflect the complexity and reality of actual driving environments. The attacker carefully selected frames containing the target scenario to generate optimal adversarial patches. These patches were then embedded into the original benign frames at suitable locations, such as the roadside, to evaluate the effectiveness of the real-world attack. Specifically, we also performed an affine transformation on the patches during embedding to ensure the results accurately reflect genuine driving environments. The output size of each patch is set to 80×80 pixels. The average length of the selected frames ranges from 20 to 40, depending on the attack scenarios.

5.4.2 The AoR Attack Results

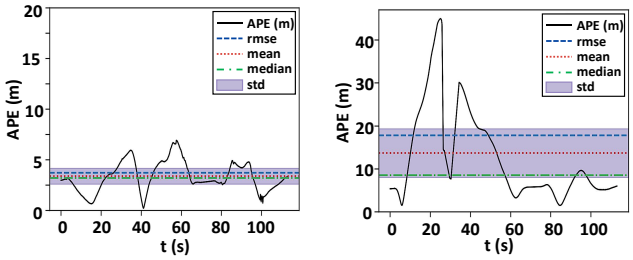
Table 3 shows the vSLAM errors introduced by the AoR Attack in city, rural areas, and parking garage scenarios. As we can see, the errors of all five vSLAM systems become much larger under the AoR attack. Moreover, all these vSLAM systems suffer track loss during the experiments, which means the autonomous vehicle or the robot cannot conduct localization and mapping at all in these scenarios. Figure 16 shows that the absolute pose error (APE) for DynaSLAM on the KITTI dataset Seq 07 is nearly **10 times** higher than that of the no-attack scenario. This experiment result demonstrates the effectiveness of the AoR attack in city scenarios.

We also notice the surprisingly high vSLAM errors in rural areas and parking garages. For example, the vSLAM error for ORB-SLAM2 is 26.03m (4Seasons-Countryside), which is around **5 times** higher than that of the no-attack scenario. Figure 17 shows the corresponding calculated vSLAM trajectory. As we can see from Figure 17 (b), the vSLAM system believes that the vehicle stops at a distance of 35m. This is because the rural area has sparse textures and fewer good features. Therefore, the vSLAM system has to select the features from the adversarial patch, which introduces significant

Table 3: The vSLAM errors introduced by the AoR attack on public datasets.

Scenarios	City		Rural Area		Parking Garage
Datasets	**KITTI**	**RobotCar**	**4Seasons**	**Complex Urban**	**4Seasons**
Sequences	Seq 07	2014-05-19	Countryside	Urban28	parking_garage_1_train
ORB-SLAM2	3.31 / 20.39	12.77 / X	**5.27 / 26.03**	10.73 / 34.94	3.78 / X
DynaSLAM	**3.25 / 23.20**	**9.04 / 19.73**	5.63 / 17.29	9.28 / 28.35	**1.58 / 9.20**
ORB-SLAM3	3.37 / 14.50	8.11 / 15.24	3.22 / 11.59	**9.30 / 31.05**	1.60 / 8.37
pl-SVO	17.62 / 23.21	15.60 / 22.29	10.17 / X	16.47 / 29.30	14.75 / X
SVO Pro	14.77 / 21.82	18.73 / X	12.49 / X	15.90 / 25.15	10.03 / X

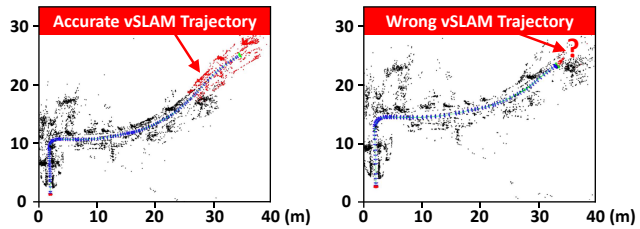
* For each A/B, A is the vSLAM error without the attack and B with the attack. Bold values highlight the largest change (B/A) per column.
* X means the track is lost under the AoR attack.



(a) Absolute pose error without the AoR attack.

(b) Absolute pose error under the AoR attack.

Figure 16: AoR attack results on the KITTI dataset (Seq 07). The attack starts at 10s and ends at 50s.



(a) vSLAM results without the AoR attack.

(b) vSLAM results under the AoR attack.

Figure 17: The AoR attack on the KITTI dataset (Seq 03).

SLAM errors. **In summary,** the AoR attack can effectively attack the vSLAM systems in various complex scenarios.

6 Defending Against the AoR Attack

Due to the severe security issues introduced by the AoR attack, we introduce a novel defense module to secure the vSLAM systems. This section first discusses the potential challenges in designing the defense module. Then, we introduce the detailed design.

6.1 Challenges in Defense

Since the vSLAM systems operate in dynamic scenarios, traditional defense strategies that are designed for static situations, such as model and input analysis approaches [52, 53], cannot work effectively to secure the vSLAM systems. Moreover, since the AoR attack uses commonly seen objects to attack the vSLAM systems, it is very difficult for common anomaly detection [54, 55] or contextual analysis [56] approaches to detect adversarial patches. In addition, although recent advances in deep learning have shown inspiring results in most vision-based tasks, the substantial computational resources and extensive training time required make them impractical for vSLAM systems. Furthermore, since the backend optimization methods in vSLAM systems barely consider the reliability of the features extracted during the VO process, the optimization direction can be easily misled by the AoR attack. Therefore, a suitable defense module should satisfy the following requirements: **(i)** high attack detection effectiveness, **(ii)** low computational overhead, and **(iii)** improving the robustness of the backend in vSLAM systems.

6.2 Design of the Defense Module

As shown in Figure 18, to defend against the AoR attack, we introduce a lightweight defense module that can effectively detect the adversarial features and improve the robustness of the vSLAM systems. This module mainly consists of three key components: **Texture Extraction**, **Anomaly Detection**, and **Cross-Frame Optimization**.

The texture extraction aims to extract the texture of captured frames using the Weber Local Descriptor (WLD) [57]. By doing this, we can make the texture of the adversarial patch noticeable in the whole frame. Then, to detect abnormal textures, the anomaly detection component uses a variational auto-encoder to identify the position of the abnormal texture in a frame sequence. To enhance the robustness of vSLAM systems against potential attacks, we've developed a cross-frame optimization component that utilizes the reliability factors of features across multiple frames. Finally, if the AoR attack is detected, the defense module will send alerts to legitimate users. In the following sections, we will introduce the details of each component in our defense module.

6.2.1 Texture Extraction

Since the adversarial patches generated by the AoR attack are natural and unnoticeable in real-world scenarios, it is hard to detect such patches directly from the captured frame sequence using traditional vision-based detection approaches. To address this challenge, the **key insight** we found is that although adversarial patches on common objects (e.g., billboards and walls) are hard to detect, they show notably denser internal

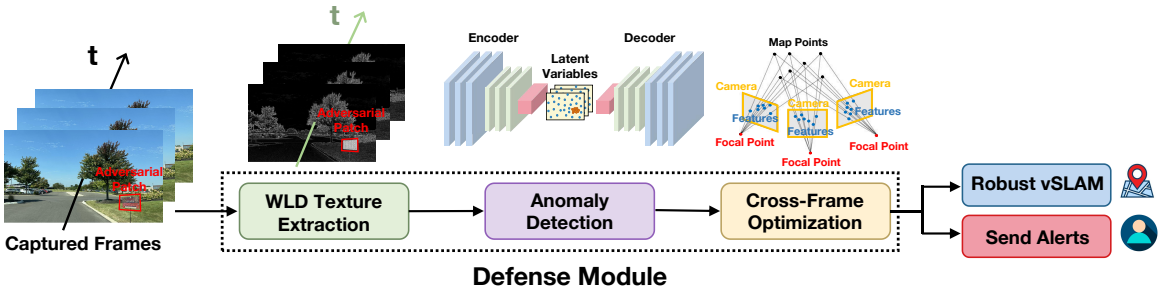


Figure 18: The overall architecture of our defense module.

textures than their surrounding common objects. To exploit this characteristic, we use Weber's Local Descriptor (WLD), which is designed to enhance texture analysis. WLD works by transforming the captured frames into a texture space, highlighting the denser textures of adversarial patches. This transformation is achieved by calculating the relative intensity difference between a pixel and its neighbors in horizontal and vertical directions, which helps to emphasize textural features. The texture distribution can be formulated as:

$$\xi(x_c) = \arctan\left(\sum_{i=0}^{P-1} \frac{x_i - x_c}{x_c}\right) \tag{8}$$

$$\Phi(x_c) = \arctan\left(\frac{D_V}{D_H}\right) \tag{9}$$

Here, x_c represents the intensity of the central pixel, while x_i refers to the intensity of a neighboring pixel, with P indicating the total number of neighbors. The variables D_V and D_H are defined as the vertical and horizontal intensity differences between the central pixel and its neighbors, respectively. To enhance efficiency and reduce complexity, the intensity difference is calculated using a sliding window with adaptive step sizes, focusing on probable regions like roadsides and buildings. This selective approach streamlines the texture extraction process, making it more manageable and effective.

6.2.2 Anomaly Detection

Since the adversarial patch becomes noticeable in texture space compared to its neighboring regions, the question now is: how can the abnormal region in the texture space be detected? To answer this question, we utilize a lightweight variational auto-encoder (VAE) [58] model to identify anomalies. The goal is to enable the model to find the abnormal regions that significantly deviate from the normal distribution. During detection, the VAE encoder first transforms the input texture images into latent variables. Then, based on the differences between the input and the output, the model can predict whether the current scenario contains the adversarial patch. Specifically, the loss function L_{oss} of the VAE model consists of the reconstruction error and a regularization parameter:

$$L_{oss} = \frac{\alpha}{m}\sum_{i=1}^{m} \|x - \hat{x}\|^2 + \frac{\beta}{2} \cdot Reg(\sigma, \mu) \tag{10}$$

In this equation, x and $\hat{x}$ are the ground truth and the prediction value, respectively. m is the number of frames. The first term is the reconstruction error, while the second term, Reg, is the regularization parameter. α and β are the weight factors for each term. Formally, Reg can be calculated as:

$$Reg(\sigma, \mu) = -\sum_{i=1}^{n} \left(1 + \log\left(\sigma_i^2\right) - \mu_i^2 - \sigma_i^2\right) \tag{11}$$

Here μ_i is the mean of the reconstructed output for feature i, and σ_i is the standard deviation of the reconstructed output.

6.2.3 Robust Optimization

An important reason why vSLAM systems do not work under the AoR attack is that adversarial features introduce huge errors during the pose optimization in the backend. Specifically, the features in the adversarial patches are carefully designed to follow the feature extraction algorithms in the VO process of vSLAM systems. As the camera captures frames when the vehicle or robot moves on the road, the adversarial features are likely to be selected as key points for pose optimization. To improve the robustness of the vSLAM system, we introduce a robust cross-frame error function E' to reduce the effects of adversarial features during optimization in the backend. Formally, we define a reliability factor r_j to measure the reliability of the feature points in each frame and its value is positively correlated with the density of surrounding features. Then, the cross-frame error function E' can be formulated as:

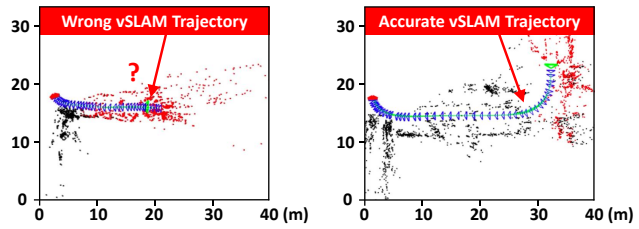
$$E' = \min_{R_i, t_i, X_j} \sum_{i=1}^{N}\sum_{j=1}^{M} \left(\frac{r_j}{\mu}\right)^2 \log\left(1 + \left(\frac{\mu}{r_j}\right)^2 E(x, p)\right) \tag{12}$$

Here μ is the weighted coefficient determined by the defender. N is the set of captured frames and M is the set of map points. $E(x, p)$ is the re-projection error that can be calculated as:

$$E(x, p) = \| x_{ji} - p(X_j, R_i, t_i, K_i)\|^2 \tag{13}$$

Here X_j denotes the jth map point, and x_{ji} represents its position in camera coordinates on the ith frame. The function p projects the map point from world coordinates to camera coordinates. The rotation matrix R and translation vector t define the pose for each frame, while K is the camera's intrinsic matrix, typically available from the manufacturer's website.

In our approach, the weight of feature points with low reliability is reduced during optimization, enhancing the influence of feature points from sparser regions (clean features). This adjustment makes the vSLAM system depend more on clean features, increasing its robustness against the AoR attack.



(a) vSLAM results under the AoR attack. (b) vSLAM results using our defense module.

Figure 19: An example of the effectiveness of our defense module on autonomous vehicle testing platform in the city center scenario.

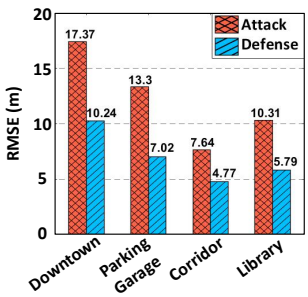


Figure 20: vSLAM errors in real-world scenarios.

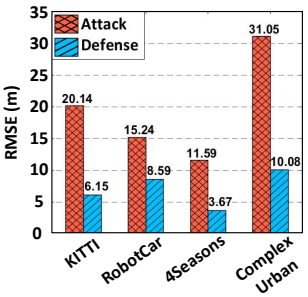


Figure 21: vSLAM errors on public datasets.

6.3 Experiment Setup

To extensively evaluate the effectiveness of our defense module, we conduct experiments in the same real-world scenarios depicted in Figure 7 and on famous public datasets including KITTI [33], RobotCar [49], 4Seasons [50], and Complex Urban [51]. During the experiments, we integrated our defense module into five commonly used vSLAM systems: ORB-SLAM2 [22], ORB-SLAM3 [27], DynaSLAM [26], pl-SVO [24], and SVO Pro [25]. In addition, we also assessed its performance in benign cases to ensure that the module does not degrade system performance. Specifically, we evaluated the precision of attack detection across 100 scenarios, comprising 38 with attacks and 62 without attacks. These scenarios are all real-world driving environments containing various environmental conditions, legitimate billboards, and moving pedestrians. Tables 4 and 5 summarize the defense results in real-world scenarios. The results on public datasets are presented in Appendix A.3.

6.4 Experiment Results

Figure 19 demonstrates the performance of our defense module in a city center scenario. In Figure 19 (a), when the autonomous vehicle makes a left turn, the vSLAM trajectory is missing due to the AoR attack. After applying our defense module, as shown in Figure 19 (b), the vSLAM system accurately conducts localization and mapping.

Figures 20 and 21 show the effectiveness of the defense module by comparing the RMSE on ORB-SLAM3. We can see that our defense module significantly reduces the vSLAM errors when vSLAM systems are under attack. For example, as shown in Figure 21, when we apply our defense module to the Complex Urban dataset, the vSLAM errors are reduced by more than **three times** compared to the attack scenario.

Table 4 demonstrates that our defense module significantly reduces vSLAM errors in real-world scenarios. More importantly, by using our defense module, the vSLAM systems never face the problem of lost tracking, demonstrating our defense module's effectiveness. Table 5 shows the detection results of our defense module in real-world scenarios. Our system correctly raises alarms for all 36 potential threats, resulting in no false alarms. This indicates that our defense module does not impact vSLAM performance when there is no attack, as it only switches to the robust mode when attacks are detected. **In summary**, our defense module can **(i)** effectively detect the AoR attack and reduce the vSLAM errors, **(ii)** enhance the security of the vSLAM system, and **(iii)** enable reliable localization and mapping under the AoR attack.

Table 4: The effectiveness of the defense module in real-world scenarios.

Scenarios	City Center	Parking Garage	Corridor	Library
ORB-SLAM2	X / 5.21	7.23 / 4.24	X / 3.10	5.26 / 3.52
DynaSLAM	**10.95 / 5.31**	7.45 / 4.21	5.77 / 3.95	**6.27 / 2.59**
ORB-SLAM3	17.37 / 10.24	13.30 / 7.02	7.64 / 4.77	10.31 / 5.79
pl-SVO	8.22 / 5.20	4.09 / 3.13	**7.02 / 4.18**	5.81 / 2.56
SVO Pro	X / 12.14	**19.70 / 10.58**	9.16 / 8.60	10.05 / 9.32

* For each C/D, C is the vSLAM error under attack, and D is with our defense module. Bold values highlight the largest change (C/D) per column.
* X means the track is lost under the AoR attack.

Table 5: Robustness of the defense strategy.

	Detection Results	
	Alarm	**No Alarm**
Attack	TP = 36	FN = 2
No Attack	FP = 0	TN = 62

7 Discussion

7.1 Ethics and Safety Considerations

Our research does not target specific commercial products; the BMW X3 is used solely as an example within our vehicle-based vSLAM system testing setup. Furthermore, all safety-critical experiments are conducted in controlled environments, enabling human operators to intervene at any time to prevent harm to users or public systems. For instance, a human driver can immediately take over the vehicle, and we can remotely take control of the wheeled robot, equipped with a remote controller, to mitigate potential risks. We found that the AoR attack could introduce significant errors into vSLAM systems, misleading those who rely on vSLAM maps for navigation. Given the crucial role of vSLAM algorithms in autonomous driving [16–19] and robot navigation [20, 21], their vulnerabilities pose significant threats. We hope our findings will highlight these security issues and enhance the robustness of

vSLAM systems.

7.2 Limitations and Future Work

One limitation of our attack is that position changes caused by the attack may diminish its effectiveness. During the attack, its position changes as the vehicle or robot navigates using the distorted vSLAM map. These changes make the attack-influenced frames somewhat different from those used for patch generation, potentially reducing the attack's effectiveness. However, our design ensures that the adversarial patch remains effective as long as it is visible to the camera, even if the patch's position in the frames shifts slightly. When the position changes are significant, the victim has likely already been diverted from the driving area, indicating a successful attack. In future work, we will explore the impact of such position changes during the attack more deeply and try to incorporate them during patch generation to realize more accurate attacks.

Additionally, our defense strategy could potentially be bypassed by attackers. Attackers might develop new adversarial patches specifically designed to evade the detection techniques currently used by our defense module. For instance, they could use machine learning models to generate patches that more closely mimic legitimate environmental textures. However, our defense strategy can also continually learn from new types of attacks and automatically adjust the defense mechanisms. This adaptive security approach is essential for recognizing and addressing changing threats over time.

8 Related Work

Simultaneous Localization and Mapping (SLAM) plays a pivotal role in enabling robots and autonomous vehicles to understand their current locations and make navigation decisions. Among existing solutions [10–15, 59–62], camera-based solutions (Visual SLAM) have been adopted for many real-world applications [16–21] due to their lightweight nature, cost-effectiveness, and ability to provide richer environmental representations [60].

One of the most widely recognized Visual SLAM(vSLAM) systems is ORB-SLAM2 [22], which provides a robust and efficient method with various camera setups. To enhance the ability to handle dynamic elements, DynaSLAM [26] effectively detects moving objects and filters them out to build a reliable map. Building upon its predecessors, the ORB-SLAM3 [27] system utilizes a multi-map data association technique to achieve highly accurate and robust real-time localization and mapping. To integrate the benefits of feature-based and direct method-based vSLAM systems, SVO [23] introduced a novel VO method enhancing vSLAM performance. Building on this, pl-SVO [24] utilizes point and line features for a more robust vSLAM system. SVO Pro [25] upgrades SVO by incorporating a sliding window backend and loop closure technique, further enhancing mapping accuracy.

However, existing works mainly focus on improving the performance of the vSLAM system, while little work has focused on the vulnerability of the vSLAM systems. Several studies [28, 29] attack certain components of the vSLAM system using highly noticeable images for human eyes. In addition, the ICSL attack [30] introduces an IR light-based attack approach to attack the SLAM system. However, their work requires customized hardware and can only attack ORB-SLAM2 in an indoor parking lot at night. Importantly, these studies often do not investigate the underlying reasons and conditions for their success, leading to uncertainty about the effectiveness of these attacks. Different from their works, in this paper, we are the first to conduct an in-depth analysis of the unique vulnerabilities within vSLAM systems and to strategically exploit these vulnerabilities to implement targeted end-to-end attacks. Specifically, we propose an AoR attack, which can effectively attack multiple commonly used vSLAM systems in various real-world scenarios. Furthermore, we also designed a defense module to protect vSLAM systems from the harmful impact of the AoR attack.

Attacks on Image Recognition Systems. There are extensive works that explored the security issues of camera-based systems on autonomous vehicles [63–67]. Using adversarial patches can harm the image recognition system [68–70] on various tasks. For example, GhostImage [68] exploits lens flare effects and auto-exposure control to project adversarial patterns into camera-based image classification systems and causes misclassification. TPatch [69] introduces a physical adversarial patch that uses specific signals to execute hiding, creating, and altering attacks on the vision-based perception module of a targeted autonomous vehicle. In contrast to previous work, our goal is to disrupt the crucial processes of real-time localization and mapping in vSLAM rather than simply causing image misclassification. Additionally, our attack is effective across sequential frames within dynamic environments and leads to serious safety issues. Thus, these previous works are not applicable to attack vSLAM.

9 Conclusion

In this paper, we propose the first work that in-depth investigates security issues in vSLAM systems. Specifically, we introduce the AoR attack, which can effectively attack commonly used vSLAM systems in autonomous vehicles and robots. Our attack has the following advantages: **(i)** it can easily be implemented in various real-world scenarios; **(ii)** no need for expensive hardware, and **(iii)** it can introduce significant errors to vSLAM systems without being detected by legitimate users. Moreover, given the severe harmful impacts of the attack, we also present a lightweight defense module to counter the AoR attack. We believe that the attack method and defense strategy outlined in this paper will contribute to the development of more secure vSLAM systems.

10 Acknowledgments

This work is partially supported by NSF grants CNS-1652669, CNS-2305246 and CCF-2347888. We thank the anonymous

shepherd and reviewers for their valuable input in improving this manuscript.

References

[1] T. Qin, Y. Zheng, T. Chen, Y. Chen, and Q. Su, "A light-weight semantic map for visual localization towards autonomous driving," in 2021 IEEE International Conference on Robotics and Automation (ICRA), 2021.

[2] Y. Pan, P. Xiao, Y. He, Z. Shao, and Z. Li, "Mulls: Versatile lidar slam via multi-metric linear least square," in 2021 IEEE International Conference on Robotics and Automation (ICRA), 2021.

[3] H. Zhang, H. Uchiyama, S. Ono, and H. Kawasaki, "Motslam: Mot-assisted monocular dynamic slam using single-view depth estimation," in 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2022.

[4] D. Fourie, N. R. Rypkema, P. V. Teixeira, S. Claassens, E. Fischell, and J. Leonard, "Towards real-time non-gaussian slam for underdetermined navigation," in 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2020.

[5] H. Huang, W.-Y. Lin, S. Liu, D. Zhang, and S.-K. Yeung, "Dual-slam: A framework for robust single camera navigation," in 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2020.

[6] P. Karkus, S. Cai, and D. Hsu, "Differentiable slam-net: Learning particle slam for visual navigation," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021.

[7] H. Wang, C. Wang, and L. Xie, "Intensity-slam: Intensity assisted localization and mapping for large scale environment," IEEE Robotics and Automation Letters, 2021.

[8] K. J. Singh, D. S. Kapoor, K. Thakur, A. Sharma, A. Nayyar, S. Mahajan, and M. A. Shah, "Map making in social indoor environment through robot navigation using active slam," IEEE Access, 2022.

[9] Z. Meng, C. Wang, Z. Han, and Z. Ma, "Research on slam navigation of wheeled mobile robot based on ros," in 2020 5th International Conference on Automation, Control and Robotics Engineering (CACRE), 2020.

[10] A. Pumarola, A. Vakhitov, A. Agudo, A. Sanfeliu, and F. Moreno-Noguer, "Pl-slam: Real-time monocular visual slam with points and lines," in 2017 IEEE international conference on robotics and automation (ICRA), 2017.

[11] W. Wang, D. Zhu, X. Wang, Y. Hu, Y. Qiu, C. Wang, Y. Hu, A. Kapoor, and S. Scherer, "Tartanair: A dataset to push the limits of visual slam," in 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2020.

[12] K. Liu, A. Xiao, J. Huang, K. Cui, Y. Xing, and S. Lu, "D-lc-nets: Robust denoising and loop closing networks for lidar slam in complicated circumstances with noisy point clouds," in 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2022.

[13] X. Zhao, S. Yang, T. Huang, J. Chen, T. Ma, M. Li, and Y. Liu, "Superline3d: Self-supervised line segmentation and description for lidar point cloud," in Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part IX, 2022.

[14] G. Kim, Y. S. Park, Y. Cho, J. Jeong, and A. Kim, "Mulran: Multimodal range dataset for urban place recognition," in 2020 IEEE International Conference on Robotics and Automation (ICRA), 2020.

[15] P. Gao, S. Zhang, W. Wang, and C. X. Lu, "Dc-loc: Accurate automotive radar based metric localization with explicit doppler compensation," in 2022 International Conference on Robotics and Automation (ICRA), 2022.

[16] S. Zheng, J. Wang, C. Rizos, W. Ding, and A. El-Mowafy, "Simultaneous localization and mapping (slam) for autonomous driving: Concept and analysis," Remote Sensing, 2023.

[17] J. Cheng, L. Zhang, Q. Chen, X. Hu, and J. Cai, "A review of visual slam methods for autonomous driving vehicles," Engineering Applications of Artificial Intelligence, 2022.

[18] K. L. Lim and T. Bräunl, "A review of visual odometry methods and its applications for autonomous driving," CoRR, 2020.

[19] A. Tourani, H. Bavle, J. L. Sanchez-Lopez, and H. Voos, "Visual slam: What are the current trends and what to expect?" Sensors, 2022.

[20] Y. Ock, H. Kang, and J. Lee, "Modified orb-slam algorithm for precise indoor navigation of a mobile robot," The Journal of Korea Robotics Society, 2020.

[21] M. Filipenko and I. Afanasyev, "Comparison of various slam systems for mobile robot in an indoor environment," in 2018 International Conference on Intelligent Systems (IS), 2018.

[22] R. Mur-Artal and J. D. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," IEEE transactions on robotics, 2017.

[23] C. Forster, M. Pizzoli, and D. Scaramuzza, "Svo: Fast semi-direct monocular visual odometry," in 2014 IEEE international conference on robotics and automation (ICRA), 2014.

[24] R. Gomez-Ojeda, J. Briales, and J. Gonzalez-Jimenez, "Pl-svo: Semi-direct monocular visual odometry by combining points and line segments," in 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2016.

[25] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza, "Svo: Semidirect visual odometry for monocular and multicamera systems," IEEE Transactions on Robotics, 2016.

[26] B. Bescos, J. M. Fácil, J. Civera, and J. Neira, "Dynaslam: Tracking, mapping, and inpainting in dynamic scenes," IEEE Robotics and Automation Letters, 2018.

[27] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam," IEEE Transactions on Robotics, 2021.

[28] M. H. Ikram, S. Khaliq, M. L. Anjum, and W. Hussain, "Perceptual aliasing++: Adversarial attack for visual slam front-end and back-end," IEEE Robotics and Automation Letters, 2022.

[29] Y. Nemcovsky, M. Jacoby, A. M. Bronstein, and C. Baskin, "Physical passive patch adversarial attacks on visual odometry systems," in Proceedings of the Asian Conference on Computer Vision, 2022.

[30] W. Wang, Y. Yao, X. Liu, X. Li, P. Hao, and T. Zhu, "I can see the light: Attacks on autonomous vehicles using invisible lights," in Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security, 2021.

[31] Y. Yue, X. Wang, and Z. Zhan, "Single-point least square matching embedded method for improving visual slam," IEEE Sensors Journal, 2023.

[32] H. Seok and J. Lim, "Rovo: Robust omnidirectional visual odometry for wide-baseline wide-fov camera systems," in 2019 International Conference on Robotics and Automation (ICRA), 2019.

[33] "Kitti dataset," https://www.cvlibs.net/datasets/kitti/, 2023.

[34] "Orb feature," https://docs.opencv.org/3.4/d1/d89/tutorial_py_orb.html, 2023.

[35] "Random sample consensus," https://en.wikipedia.org/wiki/Random_sample_consensus, 2023.

[36] M. Trajković and M. Hedley, "Fast corner detection," Image and vision computing, 1998.

[37] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in 3rd International Conference on Learning Representations, ICLR, 2015.

[38] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image style transfer using convolutional neural networks," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2016.

[39] P. Polack, F. Altché, B. d'Andréa Novel, and A. de La Fortelle, "The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles?" in 2017 IEEE intelligent vehicles symposium (IV), 2017.

[40] D. Berjoza et al., "Research in kinematics of turn for vehicles and semitrailers," in Proceedings of 7th International Scientific Conference on Engineering for Rural Development, 2008.

[41] "Mono camera," https://www.continental-automotive.com/en-gl/Passenger-Cars/Autonomous-Mobility/Enablers/Cameras/Mono-Camera, 2023.

[42] "Qrbbec astra camera," https://www.orbbec.com/products/structured-light-camera/astra-series/, 2023.

[43] C. Mannila, "Robustness of state-of-the-art visual odometry and slam systems," 2023.

[44] I. A. Kazerouni, L. Fitzgerald, G. Dooly, and D. Toal, "A survey of state-of-the-art on visual slam," Expert Systems with Applications, 2022.

[45] "Rtk gps: Understanding real-time kinematic gps technology," https://globalgpssystems.com/gnss/rtk-gps-understanding-real-time-kinematic-gps-technology/.

[46] "Root mean square deviation," https://en.wikipedia.org/wiki/Root-mean-square_deviation, 2023.

[47] "Billboard size," https://www.adquick.com/blog/what-is-the-standard-size-of-a-billboard/, 2023.

[48] "Rosmaster robot," https://category.yahboom.net/products/rosmaster-r2, 2023.

[49] "Robotcar dataset," https://robotcar-dataset.robots.ox.ac.uk/datasets/, 2023.

[50] "4seasons dataset," https://cvg.cit.tum.de/data/datasets/4seasons-dataset, 2023.

[51] "Complexurban dataset," https://sites.google.com/view/complex-urban-dataset, 2023.

[52] Y. Gao, C. Xu, D. Wang, S. Chen, D. C. Ranasinghe, and S. Nepal, "Strip: A defence against trojan attacks on deep neural networks," in Proceedings of the 35th Annual Computer Security Applications Conference, 2019.

[53] B. Wang, Y. Yao, S. Shan, H. Li, B. Viswanath, H. Zheng, and B. Y. Zhao, "Neural cleanse: Identifying and mitigating backdoor attacks in neural networks," in 2019 IEEE Symposium on Security and Privacy (SP), 2019.

[54] K. K. Santhosh, D. P. Dogra, and P. P. Roy, "Anomaly detection in road traffic using visual surveillance: A survey," ACM Computing Surveys (CSUR), 2020.

[55] C. Huang, Z. Yang, J. Wen, Y. Xu, Q. Jiang, J. Yang, and Y. Wang, "Self-supervision-augmented deep autoencoder for unsupervised visual anomaly detection," IEEE Transactions on Cybernetics, 2021.

[56] B. Nassi, Y. Mirsky, D. Nassi, R. Ben-Netanel, O. Drokin, and Y. Elovici, "Phantom of the adas: Securing advanced driver-assistance systems from split-second phantom attacks," in Proceedings of the 2020 ACM SIGSAC conference on computer and communications security, 2020.

[57] J. Chen, S. Shan, C. He, G. Zhao, M. Pietikäinen, X. Chen, and W. Gao, "Wld: A robust local image descriptor," IEEE transactions on pattern analysis and machine intelligence, 2009.

[58] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," in 2nd International Conference on Learning Representations, ICLR, 2014.

[59] C. Yu, Z. Liu, X.-J. Liu, F. Xie, Y. Yang, Q. Wei, and Q. Fei, "Ds-slam: A semantic visual slam towards dynamic environments," in 2018 IEEE/RSJ international conference on intelligent robots and systems (IROS), 2018.

[60] S. Sumikura, M. Shibuya, and K. Sakurada, "Openvslam: A versatile visual slam framework," in Proceedings of the 27th ACM International Conference on Multimedia, 2019.

[61] L. Li, X. Kong, X. Zhao, W. Li, F. Wen, H. Zhang, and Y. Liu, "Sa-loam: Semantic-aided lidar slam with loop closure," in 2021 IEEE International Conference on Robotics and Automation (ICRA), 2021.

[62] J. Jiao, Y. Zhu, H. Ye, H. Huang, P. Yun, L. Jiang, L. Wang, and M. Liu, "Greedy-based feature selection for efficient lidar slam," in 2021 IEEE International Conference on Robotics and Automation (ICRA), 2021.

[63] C. Yan, Z. Xu, Z. Yin, S. Mangard, X. Ji, W. Xu, K. Zhao, Y. Zhou, T. Wang, G. Gu et al., "Rolling colors: Adversarial laser exploits against traffic light recognition," in 31st USENIX Security Symposium (USENIX Security 22), 2022.

[64] W. Jia, Z. Lu, H. Zhang, Z. Liu, J. Wang, and G. Qu, "Fooling the eyes of autonomous vehicles: Robust physical adversarial examples against traffic sign recognition systems," in 29th Annual Network and Distributed System Security Symposium, NDSS, 2022.

[65] Y. Zhong, X. Liu, D. Zhai, J. Jiang, and X. Ji, "Shadows can be dangerous: Stealthy and effective physical-world adversarial attack by natural phenomenon," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2022.

[66] H. Wu, S. Yunas, S. Rowlands, W. Ruan, and J. Wahlström, "Adversarial driving: Attacking end-to-end autonomous driving," in 2023 IEEE Intelligent Vehicles Symposium (IV), 2023.

[67] Y. Cao, C. Xiao, A. Anandkumar, D. Xu, and M. Pavone, "Advdo: Realistic adversarial attacks for trajectory prediction," in European Conference on Computer Vision, 2022.

[68] Y. Man, M. Li, and R. Gerdes, "GhostImage: Remote perception attacks against camera-based image classification systems," in 23rd International Symposium on Research in Attacks, Intrusions and Defenses (RAID), 2020.

[69] W. Zhu, X. Ji, Y. Cheng, S. Zhang, and W. Xu, "TPatch: A triggered physical adversarial patch," in 32nd USENIX Security Symposium (USENIX Security 23), 2023.

[70] X. Liu, H. Yang, Z. Liu, L. Song, Y. Chen, and H. Li, "DPATCH: an adversarial patch attack on object detectors," in Workshop on Artificial Intelligence Safety co-located with the Thirty-Third AAAI Conference on Artificial Intelligence (AAAI), 2019.

A Appendix

A.1 Effect of Random Adversarial Features

Since existing vSLAM systems leverage various optimization techniques (e.g., Least-Square algorithm [31], RANSAC [32], etc.) to reduce potential errors introduced by the features in a frame, simply adding random features in the environment won't effectively affect the performance of the system. To show the impact of random features, we tested ORB-SLAM2

(a) Extracted features

(b) Feature matching results

Figure 22: The effect of random adversarial features.

Table 6: The vSLAM errors when the vehicle platform is under the AoR attack.

Scenarios	Campus	Rural Area	Warehouse	City Center	Garage
ORB-SLAM2	13.54 / 18.04	12.14 / 20.30	14.95 / X	8.69 / 18.93	**7.20 / 16.15**
DynaSLAM	**11.85 / 20.36**	13.03 / X	14.35 / 20.05	**8.34 / 20.17**	8.10 / 16.33
ORB-SLAM3	10.17 / 14.40	13.92 / 25.40	11.51 / 20.19	9.92 / 14.22	7.55 / 13.09
pl-SVO	16.39 / 22.25	15.36 / 22.92	**12.96 / 23.37**	15.39 / X	13.32 / 21.51
SVO Pro	15.28 / 20.30	**12.03 / 25.64**	16.84 / 27.97	15.60 / 22.18	10.33 / 19.70

* For each A/B, A is the vSLAM error without the attack and B with the attack. Bold values highlight the largest change (B/A) per column.
* X means the track is lost under the AoR attack.

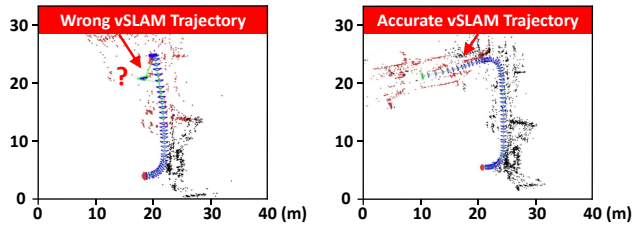
in real-world driving environments by altering grayscale values of randomly selected pixels in consecutive frames. However, as shown in Figure 22 (a), these injected features were rarely chosen as key features, and mapping results remained unaffected (Figure 22 (b)).

A.2 AoR Attack Results on Vehicle Platform

Table 6 shows the impact of the AoR attack on vSLAM systems in various environments. The RMSE (Root Mean Square Error) consistently increases under attack. For example, in the city center, DynaSLAM's RMSE rises from 8.34m to 20.17m, indicating significant navigational disruptions due to the AoR attack. The provided data in Table 6 not only quantifies the severity of AoR attacks across different scenarios but can also serve as a benchmark for evaluating the robustness of various vSLAM systems under adversarial conditions.

A.3 Defense Effectiveness on Public Datasets

Table 7 shows that our defense module significantly reduces vSLAM errors across all four datasets. Figure 23 shows an example of using our defense module on 4Seasons dataset. During an AoR attack, as the victim vehicle makes a left turn (Figure 23 (a)), the vSLAM map inaccurately shows it still facing forward with a slight position change, potentially leading to accidents. After implementing our defense, the vSLAM map is corrected and reliably accurate for navigation, which confirms the effectiveness of our defense module.



(a) vSLAM results under the AoR attack.

(b) vSLAM results using our defense module.

Figure 23: An example of the effectiveness of our defense module on the 4Seasons dataset.

Table 7: The effectiveness of our defense module on public datasets.

Scenarios	City		Rural Area		Parking Garage
Datasets	**KITTI**	**RobotCar**	**4Seasons**	**Complex Urban**	**4Seasons**
Sequences	Seq 07	2014-05-19	Countryside	Urban28	parking_garage_1_train
ORB-SLAM2	20.39 / 3.36	X / 15.00	**26.03 / 5.25**	34.94 / 12.61	X / 3.70
DynaSLAM	**23.20 / 3.07**	**19.73 / 10.94**	17.29 / 5.99	**28.35 / 8.20**	**9.20 / 1.60**
ORB-SLAM3	14.50 / 4.15	15.24 / 8.59	11.59 / 3.67	31.05 / 12.89	8.37 / 2.02
pl-SVO	23.21 / 19.15	22.29 / 13.08	X / 12.55	29.30 / 15.26	X / 10.02
SVO Pro	21.82 / 15.26	X / 16.90	X / 9.13	25.15 / 17.32	X / 8.07

* For each C/D, C is the vSLAM error under attack, and D is with our defense. Bold values indicate the largest change per column.
* X means the track is lost under the AoR attack.