PhySense: Defending Physically Realizable Attacks for Autonomous Systems via Consistency Reasoning

Zhiyuan Yu
Washington University in St. Louis
St. Louis, USA
yu.zhiyuan@wustl.edu

Ao Li
Washington University in St. Louis
St. Louis, USA
ao@wustl.edu

Ruoyao Wen
Washington University in St. Louis
St. Louis, USA
ruoyao@wustl.edu

Yijia Chen
Washington University in St. Louis
St. Louis, USA
c.amy@wustl.edu

Ning Zhang
Washington University in St. Louis
St. Louis, USA
zhang.ning@wustl.edu

Abstract

Autonomous vehicles (AVs) empowered by deep neural networks (DNNs) are bringing transformative changes to our society. However, they are generally susceptible to adversarial attacks, especially physically realizable perturbations that can mislead perception and cause catastrophic outcomes. While existing defenses have shown success, there remains a pressing need for improved robustness while maintaining efficiency to meet real-time system operations.

To tackle these challenges, we introduce PhySense, a complementary solution that leverages multi-faceted reasoning for misclassification detection and correction. This defense is built on physical characteristics, including static and dynamic object attributes and their interrelations. To effectively integrate these diverse sources, we develop a system based on the conditional random field that models objects and relationships as a spatial-temporal graph for holistic reasoning on the perceived scene. To ensure the defense does not violate the timing requirement of the real-time cyber-physical control loop, we profile the run-time characteristics of the workloads to parallelize and pipeline the execution of the defense implementation. The efficacy of PhySense is experimentally validated through simulations of datasets and real-world driving tests. It also demonstrates resiliency against adaptive attacks, and the potential of applying underlying principles to other modalities beyond vision.

CCS Concepts

• **Computing methodologies** → **Machine learning**.

Keywords

Autonomous Systems; Cyber-physical Security; Adversarial Attacks

ACM Reference Format:
Zhiyuan Yu, Ao Li, Ruoyao Wen, Yijia Chen, and Ning Zhang. 2024. PhySense: Defending Physically Realizable Attacks for Autonomous Systems via Consistency Reasoning. In *Proceedings of the 2024 ACM SIGSAC Conference on Computer and Communications Security (CCS '24), October 14–18, 2024, Salt Lake City, UT, USA*. ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3658644.3690236

CCS '24, October 14–18, 2024, Salt Lake City, UT, USA

ACM ISBN 979-8-4007-0636-3/24/10
https://doi.org/10.1145/3658644.3690236

1 Introduction

Over the past decades, autonomous vehicles emerged as a transformative technology that is seeing rapid real-world deployment. The recent advances in deep neural networks and perception systems have further fueled developments in this field. With the launch of commercial systems such as Drive Pilot [35] and Waymo [69] operating on public roads, AVs are crossing the threshold from speculation into reality. By 2032, the market size is projected to reach USD 93 billion, expanding at a CAGR of 22.8% [32].

Threats of Physically Realizable Attacks. On the other hand, these powerful autonomous systems also pose significant safety risks brought by adversarial attacks. At its core, the key functionalities of AVs are enabled by advanced DNNs and perception systems. However, they are shown to be susceptible to minor modifications on perception inputs, known as adversarial examples [27]. When such adversarial perturbations are carried out in the physical world, rather than digitally, they are often referred to as physically realizable adversarial attacks. In the context of AVs, common attack vectors include adding small patches to stop signs [22], mounting LCD screens to moving vehicles [28], and altering geometric shapes of traffic cones [9], with the goal of inducing incorrect prediction within object detection and tracking systems. Consequently, these attacks could lead to catastrophic outcomes such as traffic collisions [9, 22, 28] and braking on highways [48].

Existing Defenses. In recognition of such threats, there has been increasing research interest in developing countermeasures, which can be broadly categorized into three types based on the processing stages that they operate on. The first aims to enhance perception models through robust learning approaches [1, 50, 58, 77, 78, 89]; however, they are generally tailored to specific attacks [50] and often harm clean accuracy [71] as well as fairness [3]. The second direction aims to disrupt adversarial perturbations on inputs using various transformations [17, 34, 40, 54, 55, 63, 79, 80, 82, 93], but the overhead remains a major concern for deployment on real-time autonomous systems. The last category [47] focuses on the output of the learning component, and attempts to detect misclassified objects. However, the working principles of existing defenses mostly

focus on data without taking advantage of the underlying physical meanings. In this work, we take a step forward by incorporating physical principles in the defense.

PhySense Motivation and Overview. In contrast to DNN models, humans are generally much more resilient against adversarial examples and capable of discerning the true object "label" within a short time [92]. As revealed in recent cognitive psychological research, this can be attributed to humans' ability to associate object-specific attributes [72] and recognize inter-object relations [67]. In the context of AVs, this principle translates to discerning both the static, inherent attributes (such as the shape and texture) and dynamic, behavioral features (such as vehicles decelerating at stop signs). Unlike pixel-level features perceived by DNNs, such higher-level features often remain resilient to perturbations - since altering them requires extensive modifications in the 3D space over time, which would in turn compromise the stealthiness of attacks.

However, contemporary recognition models in AV systems do not incorporate this level of reasoning, since they are supervised by object labels only [20]. As a result, they heavily rely on non-interpretable features for recognition and are susceptible to adversarial perturbations [25, 31]. Inspired by this insight, PhySense is designed to build and integrate higher-level reasoning grounded in the physical world invariant as an additional defense layer. This reasoning produces object labels, which are then cross-checked with the perception predictions to identify discrepancies; moreover, the inference results from reasoning also offer potential clues for correcting misclassifications. At the design level, the key functionalities of PhySense are enabled by three layers, (1) the physical world modeling and object characterization, (2) the knowledge integration module, and (3) task parallelization and pipelining, each addressing a unique challenge.

Challenges. There are three main technical challenges.

C1. How to characterize objects to reflect robust physical features? The first step of PhySense is to characterize objects using physical features. While objects could be misclassified by the perception model, they still manifest physical properties (e.g., physical space occupation) that can be observed and used for defense. Guided by this principle, we first adapt a state-of-the-art 3D recognition model [30] that lifts 2D detection to 3D spaces, and then build a kinematic model adhering to physical laws (Section 5.3). This model serves as a proxy of the physical world, from which three types of features are extracted for characterization: (1) inherent attributes focusing on visual and material properties (Section 5.4); (2) dynamic object behavioral patterns (Section 5.5); and (3) inter-object relations and interactions (Section 5.6). They are selected to retain resilient spatial-temporal features while enabling efficient computation, balancing the need for performance and efficiency.

C2. How to integrate different dimensions of physical characteristics for reasoning? Reliable defense requires integrating various physical invariants as multiple layers of defense. However, this fusion poses several challenges due to the complex, multi-dimensional nature of the features. First, real-world objects in the context of transportation exhibit dynamic interconnections, thus necessitating holistic reasoning that leverages correlations. Second, these features exist in different hyper-dimensional spaces and unequally contribute to predictions, since they originate from diverse techniques and some of them uniquely characterize certain classes. Lastly, performing joint probabilistic inference on densely interconnected objects limits the scalability of defense. To address these challenges, we propose a novel framework based on conditional random field (CRF), modeling instances as graph nodes and interactions as edges. To adaptively integrate our features, we propose a new energy function that models inherent attributes and behaviors as unary terms while interactions as binary terms, with learnable weight matrices capturing the importance of features for each object class. To improve efficiency, our insight is that real-world objects also exhibit temporal continuity across consecutive frames. Therefore, we assign consistent matches between defense and perception models from prior frames as node labels in the current frame, thus accelerating the belief propagation process (Seciton 5.7).

C3. How to ensure the timeliness of PhySense? Real-time responsiveness of our defense is another key requirement for its deployment in safety-critical systems. In our preliminary exploration, we found the naive implementation of PhySense can lead to prohibitive end-to-end latency. To ensure the defense can be completed in real-time, we carefully studied the performance bottleneck of the system, and identified several opportunities to take full advantage of modern multi-core processors to parallelize and pipeline the processing. Specifically, parallelization opportunities are identified through an analysis of algorithmic dependencies, which detects workloads that can be executed concurrently due to the absence of dependencies. PhySense leverages this by decomposing these workloads into finer-grained sub-tasks, dynamically dispatched by a thread pool. Building on this parallelization, PhySense further segments the entire pipeline into multiple stages, ensuring that each stage has balanced execution times. Each stage is then executed as an independent task, optimizing resource utilization.

Evaluation. PhySense was evaluated in terms of its efficacy in detecting abnormal predictions and providing true object labels, as well as run-time efficiency. For a comprehensive evaluation, we tested on two existing datasets (nuScenes [8] and KITTI [24]), one customized dataset collected from the Carla simulator, and real-world driving tests. To better emulate real-world attacks, the adversarial examples were crafted to induce end-to-end effects on AV behaviors in the simulator. The results showed that PhySense can effectively recognize and rectify over 99% misclassified objects, without impacting the control performance of the AV. To further investigate the practicality of our approach, PhySense was also validated through real-world experiments with a Tesla Model 3 across multiple scenarios (e.g., parking lots, residential areas, main roads), different target objects (SUV and small vehicles), diverse attack types (printed patch [22], LCD displayed patterns [28], and projected perturbations [44]), and patterns with varying sizes. Lastly, to test resiliency, PhySense was tested against adaptive attackers with knowledge of the defense. While not bulletproof, its multi-faceted reasoning is shown to significantly raise the bar for attackers.

Contributions. Our contributions are outlined as follows.

- We propose PhySense[1], an integrative reasoning approach to defend against physical adversarial examples in autonomous systems. Our defense leverages robust physical attributes and correlations for multi-faceted understanding.

[1]Project Website: https://sites.google.com/view/physense

- To improve accuracy and efficiency, we propose a novel CRF-based framework that reasons object characteristics and interactions as structured spatial-temporal graphs.
- We evaluate PhySense on datasets and real-world driving tests. It achieves over 99% detection and correction accuracy against unseen attacks while maintaining run-time efficiency. It also demonstrates relative resiliency against adaptive attackers and a broader potential to apply the principles to other modalities beyond vision.

2 Background

2.1 Perception in Autonomous Vehicles

Perception is a critical component in autonomous vehicles, since an AV requires knowledge of its surroundings for decision making and safe actuation. To bridge the physical world and cyber components, perception relies on a variety of sensors like cameras, LiDARs, and radars [74, 84]. Different sensor modalities provide complementary information - for instance, cameras capture visual details like color and texture, while LiDARs and radars estimate depth and proximity by emitting laser and radio waves respectively. Through these sensors, physical elements are transformed into analog/digital signals for subsequent recognition tasks [81].

Despite the diverse raw data formats (e.g., images and point clouds) induced by different sensor modalities, they are all sent for object detection as part of the perception. Modern object detection techniques leverage DNNs to locate and classify objects from the sensor data. In the context of AVs, these networks are trained on large annotated datasets like KITTI [24] and nuScenes [8] to detect common classes such as vehicles, pedestrians, traffic signs, etc. The typical pipeline involves pre-processing raw sensor streams, extracting object-specific features using convolutional layers, classifying the features, and locating objects via bounding box regression [59, 61]. This workflow can be achieved via either a two-stage process, which uses a region proposal network to identify potential object regions and classify them using a detection network; or via a one-stage detector that performs localization and classification concurrently using a single network [43, 59].

In the same vein, object tracking builds on detection to estimate trajectories by associating detection results across consecutive frames [53]. For instance, Kalman Filters [6] model temporal evolution of object states for tracking, while data association techniques like Hungarian algorithm [37] establish frame-to-frame correspondences based on appearance and motion consistency. Recent work explored separate DNN-based detection and data association modules [23], or a single model that performs both [75]. Our work treats vision-based tracking algorithms as the target, which are most commonly studied in adversarial contexts (more details in Section 4). Moreover, we also experimentally explored the potential of PhySense to secure other modalities (Section 6.4), and showed that the key principle can be extended to other domains.

2.2 Physical Adversarial Attacks

Since the seminal work by Goodfellow et al. [27], adversarial examples have emerged as a significant threat to broad machine learning systems. The concept was initially introduced in the image domain, revealing the vulnerability of neural network models to subtle perturbations that lead to misclassification. Over the past decade, such attacks have since evolved and expanded to other domains such as point clouds [9, 10] and audio signals [13, 83, 85]. In general, these attacks can be formulated as an optimization problem:

$$\arg\max_{\boldsymbol{\delta}} L(f(\boldsymbol{x}+\boldsymbol{\delta}), y_{\text{true}}) \qquad \text{s.t. } \|\boldsymbol{\delta}\|_p \leq \epsilon, \tag{1}$$

where $\boldsymbol{x}$ is the original input, $\boldsymbol{\delta}$ is the adversarial perturbation, $L(\cdot)$ is the distance function reflecting the attacker's goal (such as misclassification), and $\|\cdot\|_p$ represents an L_p norm that restricts the perturbation magnitude to ensure stealthiness.

Among this evolving threat landscape, physically realizable attacks present a unique line of threats due to their practical feasibility in the real world. Unlike digital attacks that manipulate data within a computational environment, physical attacks restrict the perturbations on real-world objects or environmental elements. This necessitates additional physical constraints on the manipulation space. As a general formula, the loss function is constructed as:

$$\mathbb{E}_{(\boldsymbol{T},\boldsymbol{C})\sim\mathcal{P}}\left[L\left(f\left(\boldsymbol{T}(\boldsymbol{x}+\boldsymbol{\delta});\boldsymbol{C}\right), y_{\text{true}}\right)\right] + \lambda\cdot\Omega(\boldsymbol{\delta}), \tag{2}$$

where $\mathbb{E}_{(\boldsymbol{T},\boldsymbol{C})\sim\mathcal{P}}$ represents the expectation over a distribution $\mathcal{P}$ of physical transformations $\boldsymbol{T}$ and real-world conditions $\boldsymbol{C}$ like ambient lighting [2, 22, 66]. Besides, $\lambda\cdot\Omega(\boldsymbol{\delta})$ is a weighted regularization term that enforces the perturbation to be physically plausible.

3 Existing Defenses

Existing defenses fall into three categories based on the processing stages that they operate on. The first category enhances the robustness of perception models through robust learning approaches. The key idea is to incorporate known adversarial examples (e.g., PGD [45]) during model training. For instance, Wu et al. [78] found that traditional defenses were ineffective against physical adversarial attacks and proposed adversarial training on images with rectangular occlusions. Rao et al. [58] additionally optimized patch locations for broader attacks with varying patch locations. Besides, Meta-adversarial training [50] and fast adversarial training methods [1, 77, 89] aimed to improve generalizability and reduce computational overhead. However, these defenses are limited as they are tailored to specific attacks (e.g., a specific L_p norm) and often come at the expense of clean accuracy [71] and fairness [3].

Inspired by the insight that adversarial attacks rely on manipulated inputs, a stream of defense aims to disrupt perturbations via input sanitization, employing techniques such as image compression [34], randomized smoothing [17, 40, 54], diffusion algorithms [55, 80], and broader generative models [16, 63, 82]. These defenses have shown superior performance yet have significant overhead, making them computationally prohibitive for real-time autonomous systems. Lastly, a less-explored direction focuses on perception outputs. In the context of AVs, PercepGuard [47] was recently proposed to detect object misclassification attacks. It employs an LSTM model to analyze bounding box sequences, and an alarm is raised if its output mismatches with the perception module.

As a preliminary step to address the remaining challenges, we propose PhySense as a novel approach empowered by statistical modeling, robust physical rules, and pipelining techniques. Cooperating with the AV perception module, PhySense is designed to

verify and correct adversarial predictions while preserving model utility and real-time efficiency.

4 System and Threat Model

4.1 System Model

In this study, we focus on vision-based object detection and tracking. This is because vision is de facto the most popular and well-established perception modality in contemporary AVs, and most existing physical adversarial attacks target visual inputs [84]. To enable such perception modules, the target AV is equipped with cameras that record videos in real time. The dissected image frames are sent for object detection, and the results among consecutive frames are analyzed for object tracking. The exact algorithm or model could vary, but they are all based on DNN architectures.

Additionally, to understand the potential of generalizing to broader modalities, we also conducted a small set of experiments on other sensing modalities (e.g., LiDAR) beyond vision (Section 6.4).

4.2 Threat Model

Attack Goals. The adversary's primary goal is to deceive the AV's perception system into misclassifying objects through physical perturbations. These attacks can significantly affect AV operations. For instance, an attack might cause the AV to misclassify stop signs as speed limit signs, leading to potential collisions at crossroads [22]; besides, misidentifying a car ahead as a person could trigger braking on high-speed roads [48]. On the other hand, the attacker also seeks to restrict perturbation magnitudes to avoid suspicion.

Our Scope of Attacks. This study focuses on physically realizable adversarial attacks, where the attacker can only introduce perturbations in the physical world. This differs from digital (or cyber-domain) attacks, where the attacker can arbitrarily manipulate DNN inputs. The mechanisms for realizing physical manipulation are diverse. In this study, we focus on two categories of attacks that are representative in the field, patch-based [22, 78, 91, 94] and projection-based [44, 86] attacks.

Patch-based Attacks. Adversarial patch is a well-established approach that realizes perturbations within a small area and can be physically attached to the target object. In prior work, most patches are realized on printed papers [22, 91, 94], while some recent studies have explored using monitors mounted on the vehicle to display these patches [11, 28]. This study evaluates both printed and displayed patch attacks to understand the defense effectiveness.

Projection-based Attacks. We also consider a novel approach that uses light projectors to cast perturbations onto the target [29, 44, 76]. Unlike patches that fully cover a region on the object, projected patterns are semi-transparent and overlay the object's surface, thus necessitating modeling of color blending and light diffusion effects. A unique advantage is that such attacks can be launched remotely, thus eliminating the need to physically access the target object.

Attacker Assumptions. To execute these attacks, the adversary is assumed to possess knowledge of adversarial machine learning and the high-level architecture of the target AV. Therefore, they can generate adversarial perturbations using grey-box or black-box attack methods. Furthermore, the attacker is also assumed to have access to equipment such as projectors or screens for realizing perturbations in the physical world. While our defense does not have specific assumptions on the degree of manipulation, most of the tested perturbations were constrained to certain L_p norms or restricted within small physical regions. In our evaluation, we consider both naive attackers (without the knowledge of defense) and adaptive attackers (with the knowledge of defense).

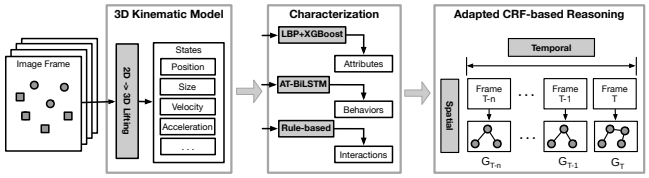

Figure 1: PhySense overview.

5 PhySense Design

5.1 Overview

The overall workflow of PhySense is depicted in Figure 1. The high-level idea is to construct a three-dimensional representation of the perceived scene and infer object labels from extracted physical features and correlations. The goal is to assign labels to individual objects such that the overall reasoning gain is maximized. Within the vision domain, we first adapt state-of-the-art 3D object detection model [30] to lift 2D detection to 3D spaces, which is used as a proxy of the physical world (Section 5.3). For effective defense, the next step is to extract features that are resilient, distinctive, and readily extractable from visual information. Guided by cognitive theory [5, 49], we employ three types of features: (1) inherent attributes that focus on visual and material properties (Section 5.4); (2) object behaviors that characterize dynamic patterns (Section 5.5); and (3) inter-object relational features capturing interactions and arrangements between objects (Section 5.6). These statistics represent robust invariant features, which are then used for reasoning through the adapted conditional random field (CRF) framework and an energy function tailored for quantifying reasoning gain (Section 5.7). At last, the defense framework is deployed with task parallelization and pipelining to improve efficiency (Section 5.8).

5.2 Problem Formulation

Let $M : \mathbb{R}^{H\times W\times C} \rightarrow \mathbb{R}^{N\times K}$ be the target vision-based perception model that takes an input image $x \in \mathbb{R}^{H\times W\times C}$ and outputs a set of N detected object instances $\mathcal{I} = \{I_1, I_2, ..., I_N\}$, where each instance I_i is a K-dimensional vector representing the probabilities over K classes. The attacker conducts attacks by applying adversarial perturbations, resulting in manipulated perception inputs $x' = x+\delta$ and the corresponding set of detected instances $\mathcal{I}' = M(x')$. Let $\mathcal{S} \subseteq \mathcal{I}$ be the subset of instances that are misclassified, such that:

$$I_i \in \mathcal{S} \Leftrightarrow \arg\max_k I_i[k] \neq \arg\max_k I_i'[k], \quad \forall I_i \in \mathcal{I} \tag{3}$$

As such, PhySense aims to identify misclassified objects and provide their true labels $\{y_i = \arg\max_k I_i[k]\}_{I_i\in\mathcal{S}}$ for defense.

5.3 Three-dimensional Kinematic Model

Traditional object detection operates solely on 2D images, which discards spatial information (e.g., depth, shape) needed to represent

physical world rules in 3D spaces. To address this, PhySense first elevates 2D detection from the perception model into a unified 3D space. This transformation is necessary as it places all perceived objects within the same world coordinate system, thus enabling consistent kinematic analysis regardless of the relative position between various objects and the camera. To achieve this, we employ a 3D recognition model adapted from a state-of-the-art approach [30]. In PhySense, this model processes image frames captured by cameras, and for each detected object, it produces eight 3D coordinates corresponding to the vertices of the 3D bounding box. These coordinates are designed to align with the world coordinate system to approximate the object's position and dimensions in physical space. Such an approach also aligns with the real-world practice that increasingly develops vision-based 3D perception [14, 18, 33].

In the 3D space, each instance I_i is associated with a set of coordinates $\mathcal{P}_i$ corresponding to the eight corners of its bounding box, where $\mathbf{p}_i^j = (x_i^j, y_i^j, z_i^j) \in \mathcal{P}_i$ represents the coordinates of the j^{th} corner of instance I_i. The centroid $\mathbf{c}_i$ is computed as the arithmetic mean of $\mathbf{P}_i$ and is used as the 3D location of instance I_i. For kinematic analysis, we calculate the velocity and acceleration along the x and y axes. In this study, the z-axis kinematic information is excluded based on a key observation: real-world AV motion along the z-axis is often much smaller than the x and y axes, making it particularly susceptible to measurement noise. Therefore, while our method can derive 3D-space representation, we choose to focus on the x-y plane to improve performance and efficiency by using more reliable information. However, PhySense is not inherently restricted to the x-y plane; instead, it is designed with the flexibility to incorporate z-axis motion information in future studies.

For instance, given the camera frame rate f_r, the acceleration along the α axis is calculated as:

$$a_i^\alpha(t) = {f_r}^2[c_i^\alpha(t) - 2c_i^\alpha(t - 1/f_r) + c_i^\alpha(t - 2/f_r)] \tag{4}$$

These statistics serve as descriptors of the current state $S_i(t)$ of instance I_i at time t. They are used for further analyses.

5.4 Characterization via Inherent Attributes

Inherent physical attributes are the features intrinsic to an object, and they are less affected by external impacts or contexts. Typical examples include sizes, color composition, texture, and geometry. However, an appropriate set of features needs to be distinctive across different classes while remaining relatively invariant among heterogeneous instances within a class; additionally, computational efficiency must also be considered. As such, we focus on 3D sizes and surface texture in this study. In contrast, other features such as color constitution could vary significantly for diverse instances, while reliable geometry extraction from solely 2D vision inputs (i.e., images) could be computationally intensive [68].

Specifically, the three-dimensional size represents the object's physical space occupation, which remains intrinsic and unchanged in the physical world. We approximate the values along three axes using 3D bounding box dimensions. Since each dimension is independent of the others, we map them separately to estimate the probability of an object belonging to a certain class. The high-level idea is to extract statistical knowledge from existing datasets, which provide rich information about the distribution of size values associated with each object class.

Consider a test instance I_{test} with size s_{test}^α along dimension α. To retrieve size knowledge associated with classes, a dataset $\mathcal{D} = (I_k, y_k)_{k=1}^N$ containing N instances is used, where I_k is the k^{th} instance and $y_k \in \mathcal{Y}$ is its ground truth class. As such, the size s_k^α along dimension α for instance I_k can be calculated using the 3D coordinates of its vertices. To approximate the distribution, values are discretized into B bins $\mathcal{B} = \{b_1, b_2, ..., b_B\}$. When s_{test}^α falls into bin b_i, the prior $P(y)$ and likelihood $P(b_i|y)$ are calculated for each class $y \in \mathcal{Y}$ based on the extracted statistics:

$$P(y) = \frac{|\{k : y_k = y\}|}{N};\ P(b_i|y) = \frac{\left|\{k : y_k = y \text{ and } s_k^\alpha \in b_i\}\right|}{|\{k : y_k = y\}|} \tag{5}$$

As such, when considering s_{test}^α, the posterior probability of I_{test} belonging to class y is calculated using Bayes' rule:

$$P(y|b_i) = \frac{\left|\{k : y_k = y \text{ and } s_k^\alpha \in b_i\}\right|}{N \cdot P(b_i)}, \tag{6}$$

where $P(b_i) = \sum_{y' \in \mathcal{Y}} P(b_i|y')P(y')$ can be precomputed as the evidence, or treated as a normalization factor.

On the other hand, surface texture reflects material properties (e.g., metal for vehicles, cloth for pedestrians) that cannot be entirely altered through restrictive physical manipulations. To use such features, we map an object's texture to the probability of belonging to a specific class. This involves randomly selecting N small regions within the object's bounding box and processing each using Local Binary Patterns (LBP) [46]. As such, each region produces a histogram with 256 bins that capture diverse texture patterns. To characterize the entire object, we normalize these N histograms to create a unified descriptor, forming a standardized LBP vector of dimensions 256×1. This latent representation is then input into an Extreme Gradient Boosting model [12], which calculates the probability that the object fits into each category.

As such, these inherent attributes are individually mapped to a probability distribution over object labels.

5.5 Characterization via Object Behaviors

Besides static attributes, dynamic behaviors also exhibit unique patterns that can characterize objects. For example, vehicles generally move in structured ways adhering to roads and speed limits, while pedestrians exhibit more irregular motion as they walk in various directions. These behavioral differences manifest in the trajectories and state changes over time. However, effectively capturing and using these complex spatio-temporal patterns poses challenges. Specifically, annotated behavioral data in the context of transportation remains scarce in existing datasets; besides, accurately modeling and categorizing the temporal dynamics is difficult.

Annotating Behaviors Using Thematic Coding Techniques. To address the first challenge, we combined existing datasets and augmented them through a structured annotation process based on thematic coding techniques [7]. As a starting point, we leveraged the LOKI dataset [26] which contains 14 types of labeled behaviors with associated object sequences. However, the given behaviors are limited to vehicles and pedestrians only. To expand the behavior catalog across more classes, three coders built upon this initial

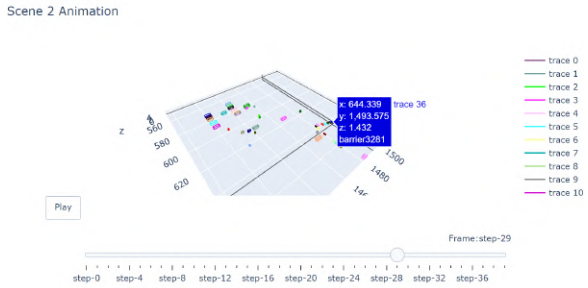

Figure 2: Example animation developed for behavior coding.

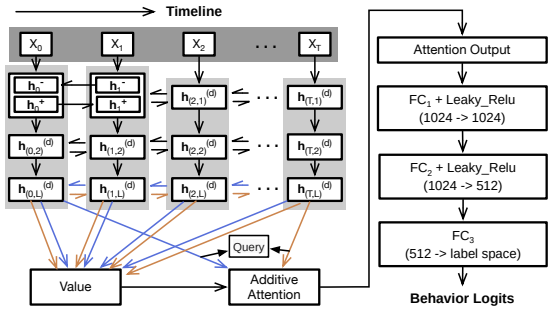

Figure 3: AT-BiLSTM model for behavior identification.

codebook and iteratively refined it through team discussion and cross-checking until a consensus was reached on the final taxonomy. The additional unlabeled sequences were sourced from nuScenes [8] and KITTI [24] datasets, with their details explained in Section 6.1.

To facilitate the annotation process, we implemented an interactive HTML-based visualization showing 3D bounding boxes with object labels and tracking IDs across frames. This enabled coders to inspect object behaviors and coordinate labels conveniently. We also included a sliding toolbar to allow repeated observation within arbitrary frame ranges. An example of our visualization is depicted in Figure 2. As a result, our annotation led to 13,857 data points containing object IDs, behavior frame ranges, and descriptive labels denoting both class and behavior (e.g. vehicle turning left). These multi-class annotations were then used to train behavioral models.

Extracting Behaviors via Sequence Models. To address the second challenge, we built on recent research of behavior recognition [90] and constructed an Attention-BiLSTM model (Figure 3) adapted to time-series object states. Specifically, Bidirectional LSTMs (BiLSTMs) [64] are employed to capture temporal contexts at each time step. By traversing sequence inputs twice from both forward and backward directions, they can encode more temporal dependencies compared to conventional LSTMs, thus improving performance for behavioral learning [65]. Their outputs are fed into an additive attention module [73] that allows for selective focus on the most salient parts of inputs. As such, the BiLSTMs and the focused feature extraction from additive attention complement each other to achieve robust behavior identification from object state sequences.

At the time t, the input is a state vector $\mathbf{x}_t \in \mathbb{R}^D$ containing the object's location, velocity, acceleration, and size. This vector is processed through L stacked BiLSTM layers as follows:

$$\mathbf{h}_{t,l}^{(d)} = \text{BiLSTM}(\mathbf{h}_{t+d,l}^{(d)}, \mathbf{x}_t, \mathbf{h}_{t,l-1}^{(d)}), \tag{7}$$

where $\mathbf{h}_{t,l}^{(d)}$ represents the hidden state at time t and layer l for direction d (either forward + or backward −). The forward and backward hidden states at time t for layer l are denoted as $\overrightarrow{\mathbf{h}}_{t,l}$ and $\overleftarrow{\mathbf{h}}_{t,l}$, each belonging to $\mathbb{R}^{d_h}$. The outputs from the last layer are concatenated and used as the value of the attention: $\mathbf{V} = [\overrightarrow{\mathbf{h}}_{t-T,L}; \ldots; \overrightarrow{\mathbf{h}}_{t,L}; \overleftarrow{\mathbf{h}}_{t-T,L}; \ldots; \overleftarrow{\mathbf{h}}_{t,L}] \in \mathbb{R}^{2\cdot d_h \times T}$; while the forward and backward outputs are concatenated as the query $\mathbf{Q} = [\overrightarrow{\mathbf{h}}_{t,L}; \overleftarrow{\mathbf{h}}_{t-T,L}] \in \mathbb{R}^{2\cdot d_h}$. Subsequently, the additive attention computes the relevance score $\mathbf{e}$ between the query $\mathbf{Q}$ and value $\mathbf{V}$ as $\mathbf{e} = \mathbf{w}^T \tanh(\mathbf{W}^Q\mathbf{Q} + \mathbf{W}^V\mathbf{V} + \mathbf{b})$, where $\mathbf{W}^Q, \mathbf{W}^V$, and $\mathbf{b}$ are learnable parameters, and $\mathbf{w}^T \in \mathbb{R}^T$ is used to compute the score. The attention weights are obtained using a softmax function over the score $\mathbf{e}$, and the output context vector is computed as the weighted sum of the value vectors. It is then processed by fully connected layers and Leaky-ReLU activations for prediction.

Physics-constrained Normalization. As the foundation of behavior identification relies on DNNs, the output distribution can diverge from physical conditions since the model is learned from data. To improve quality, we impose additional physical constraints to normalize the output space. Specifically, we follow an iterative refinement process to determine appropriate constraints for each behavior type. These constraints are designed to meet several requirements: (1) they encode fundamental physical rules and common sense priors; (2) they are readily extractable from the captured visual information, allowing efficient filtering; and (3) they contain minimal parameters and remain stable across scenarios.

With these goals, we developed a set of rules for identifying interactions. Example rules include using directional velocity to judge turns and spatial locations to identify stationary states. These constraints are applied efficiently in a post-processing stage. Let the raw DNN output be a probability distribution over behaviors $\mathbf{p}^{\text{beh}} = [p_1^{\text{beh}}, ..., p_Z^{\text{beh}}]$ where $\sum_{i=1}^{Z} p_i^{\text{beh}} = 1$. Each behavior is associated with a set of binary constraint functions $\{c_{i1}, c_{i2}, \ldots, c_{iz_i}\}$ that evaluates to 1 if satisfied. The constrained probability is:

$$\hat{p}_i^{\text{beh}} = \frac{p_i^{\text{beh}} \prod_{j=1}^{z_i} c_{ij}}{\sum_{k=1}^{Z} p_k^{\text{beh}} \prod_{j=1}^{z_k} c_{kj}} \tag{8}$$

As such, this step filters out the potential of behaviors violating constraints and renormalizes the distribution, resulting in more consistent outputs aligned with real-world observations.

5.6 Characterization via Interactions

While inherent attributes and behaviors provide self-contained characterizations of individual objects, interactions between objects also offer valuable contextual information. The key intuition is that even if an object is misclassified, it still exists physically in the world and will therefore affect other objects in the scene. For example, even if a vehicle is wrongly labeled, its follower will still react by matching velocities and maintaining distance. This reveals clues about the true class, as interactions expose physical relationships beyond an object's own properties. Motivated by this insight, we

incorporate inter-object interactions as key evidence for holistic reasoning. However, characterizing interactions poses two main challenges. First, it is difficult to detect if an interaction exists, unlike attributes or behaviors that are restricted to single objects. Second, accurately modeling the diverse, multi-agent interactions requires significant computation. This dual dilemma between efficiency and accuracy is a major obstacle in this analysis.

The key to our approach lies in the observation that, precisely inferring interaction types is not our ultimate objective. Instead, we aim to extract information to aid reasoning within reasonable costs. Inspired by prior work on interactive driving [87], a rule-based interaction identification is employed. The goal is not to classify the exact interaction, but to derive potential relationships between possibly interacting objects. Due to this reason, we do not focus on exhaustively annotating interactions for training classifiers. Instead, we source data and rules from existing datasets (i.e., INTERACTION [87, 88] and Waymo Motion [21]), and use a similar thematic coding process (Section 5.5) to refine interaction types and their rules. Specifically, we focus on three types of safety-critical objects, pedestrians, cyclists, and vehicles, resulting in 8 categories of directed object pairs (e.g., vehicle-cyclist) and 25 types of interaction characteristics (e.g., a vehicle yield to a cyclist).

During inference, the characterization consists of two phases. The first phase determines if interactions exist between a pair of objects, based on the time-to-conflict-point (TTCP) [87] and collision risk area [56] metrics. The principle is that objects expected to converge imminently are likely to interact to avoid collisions. To calculate TTCP, we first predict the conflict point based on the current trajectories of two objects, and then estimate their respective arrival times at this intersection. The objects are considered to be interacting if both the difference between these TTCPs and their maximum value fall below predefined thresholds. Complementing this, we also assess the collision risk area, which additionally covers the interactive scenarios where objects may not have predicted intersection points but are in close proximity. If the collision risk area of two objects overlaps, we consider them to be interacting. In the second phase, we apply the aforementioned rules to identify the specific types of interactions between the paired objects. These rules consider various factors such as relative distances, angles, velocities, acceleration, and the size of the overlapped collision risk area. A comprehensive list of the interaction types and their rules is provided in the extended report available on our project website.

5.7 Reasoning via Conditional Random Field

The three types of characteristics outlined above - inherent attributes, object behaviors, and inter-object interactions - capture both static and dynamic features in the physical world. While they all provide important evidence that can aid reasoning, relying on a single cue for decision-making could be unreliable. Therefore, effectively aggregating knowledge is another key problem.

Our modeling of the reasoning framework is driven by three technical challenges. First, real-world objects in transportation contexts exhibit dynamic correlations. Unlike traditional perception methods that generally isolate elements for recognition, PhySense leverages interconnections and thus requires a holistic approach. Second, the diverse knowledge sources do not contribute equally to decisions. This can be attributed to two reasons: (1) certain physical features are uniquely characteristic of specific classes, in which case they provide more information to discriminate object types; and (2) the reliability of these features varies, as they are derived from different computational techniques. For effective knowledge aggregation, a novel mechanism needs to be tailored to object classes. Third, performing joint probabilistic inference on densely interconnected instances poses scalability challenges. Therefore, improving reasoning efficiency is needed for real-time application.

Spatial Graph. For a given scene, each object instance I_k is represented as a node v_k in the CRF graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The edges $\mathcal{E}$ encode interactions between instances. The energy function is defined over variables $\mathbf{V} = \{v_1, v_2, ..., v_K\}$ as $E(\mathbf{V}) = \sum_{k=1}^{K} \psi_u(v_k) + \sum_{(v_k,v_j)\in\mathcal{E}} \psi_b(v_k, v_j)$, where $\psi_u(v_k)$ are unary potentials capturing object attributes and behaviors, and $\psi_b(v_k, v_j)$ are binary potentials for interactions. Therefore, the overall energy is the integration of energy for all nodes. Let $y(v_k)$ be the label assigned to node v_k:

$$E(\mathbf{V}) = \sum_{k=1}^{K} \Bigg[\sum_{f\in F_a} p(y(v_k)|f) + \sum_{b\in B_{y(v_k)}} \hat{p}_b^{beh}(v_k) w_1(y(v_k)) + \sum_{(v_k,v_j)\in\mathcal{E}} f_{int}(v_k, v_j) w_2(y(v_k), y(v_j)) \Bigg], \tag{9}$$

where F_a is the set of inherent attributes, $p(y|f)$ is the posterior probability of label y given f, $B_{y(v_k)} \subseteq B$ is the subset of behaviors for label $y(v_k)$, $\hat{p}_b^{beh}(v_k)$ is the probability that v_k exhibits behavior b, $f_{int}(v_k, v_j)$ is 1 if interaction exists between v_k and v_j (0 otherwise). $w_1(y(v_k))$ and $w_2(y(v_k), y(v_j))$ are weight matrices associated with object labels, optimized with:

$$L = \frac{1}{|\mathcal{S}|} \sum_{s\in\mathcal{S}} \frac{1}{|\mathcal{V}_s|} \Big[E_{\text{infer}}(\mathbf{V}_s) - E_{gt}(\mathbf{V}_s) + C \Big], \tag{10}$$

where $\mathcal{S}$ is a batch of graphs, $E_{\text{infer}}(\mathbf{V}_s)$ is the energy calculated using inferred labels $y(v_k)$ for nodes in scene s, while $E_{gt}(\mathbf{V}_s)$ is the energy calculated using true labels $y_{gt}(v_k)$, and C is a constant.

Temporal Graph. While spatial graphs capture intra-frame consistency, another valuable insight is that real-world objects also exhibit temporal continuity across consecutive frames. A similar principle is incorporated in Kalman filtering used in control systems [36]. Inspired by this, we introduce a temporal graph that leverages cross-frame reasoning to improve robustness and efficiency.

At a high level, this temporal consistency is constructed in two main steps. First, we match graphs from previous frames to the current frame by associating similar nodes based on the distances and Intersection over Union (IoU) of their 3D bounding boxes. The second step is to verify that these labels remain consistent between matched nodes across frames and with the perception results. Nodes with consistent matches will have their labels directly assigned in the current frame, which are then utilized for more efficient belief propagation that focuses more on inferring new objects.

This temporal consistency provides two advantages. First, it enhances resiliency against corner cases where certain objects might be occasionally misidentified within spatial graphs in a few frames. Second, it significantly improves efficiency by focusing costly inference only on new objects and attacked instances, while leveraging cross-frame consistency to assign other labels.

Table 1: Run time breakdown of initial implementation.

Phase	Inherent Attributes		Behaviors	Interactions	Reasoning	
Processing	3D Size	LBP	AT-BiLSTM	Rule-based	Build Graph	CRF Inference
Time (s)	0.0058	0.0365	0.0018	0.0001	0.0118	0.0142

5.8 Optimizing Run Time Efficiency

A naive implementation of PhySense can lead to high end-to-end latency due to its long processing pipeline when designed in a sequential manner. However, ensuring the timely execution of tasks (availability) in cyber-physical systems is often equally important. Parallelization and pipelining are two of the most effective techniques for achieving this. Parallelization [19] divides a task into smaller sub-tasks that can be executed simultaneously across multiple processors or cores, while pipelining [38] breaks a process into stages that work concurrently on different parts of the task, similar to an assembly line. While these two techniques have been extensively studied over the past decades, their application to ML workloads remains less explored. To understand the runtime characteristics of the workload and to identify optimization opportunities, we begin with profiling. The results on the execution time for main components are shown in Table 1. From the result, we can observe that the majority of computational delays are CPU-bound [41], this presents a unique opportunity to leverage parallelization and pipelining at the component level to reduce the latency.

Parallelization. Intuitively, any workloads that can be executed simultaneously without dependencies are potential candidates for parallelization. To do so, PhySense decomposes the workloads into finer-grained tasks that are dynamically dispatched by a thread pool. In the thread pool, pre-allocated memory buffers are created to minimize the latency caused by runtime memory management. Additionally, we employ lock-free ring buffer [39] to enable concurrent access without blocking.

Furthermore, consistency of objects in the physical world along the time dimension also opens up an opportunity for additional optimization. In our case, the computational complexity of the LBP algorithm correlates with the number of objects detected in the image. This property can be leveraged to determine the optimal number of threads for parallelization. Insufficient thread allocation can result in suboptimal parallelization, while excessive thread allocation introduces unnecessary overhead due to context switching. Given that the number of objects typically remains stable between consecutive images, PhySense uses the object count from the previous frame to pre-allocate an appropriate number of threads.

Pipeline. Building on the results of parallelization, we further conduct a detailed analysis of the execution time for each component. The components are then grouped into stages, ensuring that each stage has a balanced execution time. Components within a stage are consolidated into a single task, with each stage executed independently by separate tasks. Upon completion of a stage, the data required for the subsequent stage is updated via a buffer. This buffer, which enables data sharing between stages, is implemented using a FIFO (first-in, first-out) queue to ensure ordered data transfer.

6 Experiments and Evaluation

Our evaluation of PhySense comprises both simulated experiments on datasets and real-world driving tests, measuring effectiveness and run-time efficiency as key criteria. While physical experiments provide the most realistic assessment, they often lack the scalability to comprehensively cover the input space. To bridge the sim-to-real gap in dataset-based evaluation, we employ transformations to adversarial examples that emulate physical conditions such as angular positioning and occlusions. For more detailed analysis, we conducted comparative experiments with state-of-the-art defenses (Section 6.2) and validated PhySense components through ablation studies (Section 6.3). To test resiliency, we also evaluated against adaptive attackers with knowledge of our defense (Section 6.6).

Figure 4: Examples of simulated data collected from Carla.

6.1 Implementation and Evaluation Setup

Datasets. Our experiments involved two of the largest multi-modal autonomous driving datasets, nuScenes [8] and KITTI [24], provided by sensor suite including cameras, radars, and LiDARs. We also collected a customized dataset using Carla simulator [52] containing 2000 videos, with 100 seconds each at 5 FPS rate (Figure 4). We focus on five representative classes shared by these datasets, "bicycle", "bus", "pedestrian", "car", and "truck". The evaluation was conducted on individual videos, each consisting of several to tens of image frames focusing on one object. The total counts of these object-specific videos were 21763 for nuScenes, 5212 for KITTI, and 26854 for our custom dataset, totaling millions of image frames. Finally, to evaluate real-world practicality, we conducted 8.6 hours of driving tests with a Tesla Model 3 across various regions including parking lots, residential areas, highways, main roads, etc.

Implementation Details. The perception and defense were implemented in PyTorch 1.11.0 as the backend. For the target perception model, we followed existing work [47] and used YOLOv3 [60] for object detection, combined with SORT [4] for tracking based on IoU between bounding boxes. Within PhySense, we implemented XGBoost with default parameters including 100 trees with a maximum depth of 6. The proposed AT-BiLSTM for behavior recognition was trained to minimize cross-entropy loss using the Adam optimizer. Key hyperparameters included a learning rate of 1e-3, batch size of 32, and 300 training epochs. More details can be found in our released implementation available on the project website.

Simulator and Hardware. We used Carla 0.9.15 for data collection. The main experiments and model training were conducted on a server with RTX 4090 GPU (24GB VRAM) and Intel i9-13900K.

6.2 Evaluation on Datasets

Evaluated Attacks. While all physical attacks aim to add perturbations in some manner, the key difference lies in the subject they are applied to and what they affect. Within this scope, some attacks manipulate the perceived objects to cause misclassification [22, 78, 91],

Figure 5: An example of a patch applied on a moving vehicle.

while others directly affect perception sensors (e.g. attaching camera patches to lenses) [42, 70, 95]. We focused on the former type of attacks in this study, since real-world attackers often lack access or physical proximity to the AV's onboard sensors.

Three main factors were considered when implementing these attacks. The first is the location of perturbations, which is constrained by the object's physical size to keep the patch within the geometry boundary. The second factor is the size of the manipulated region. Attackers can set this parameter based on the desired attack strength and imperceptibility. The last one is the exact perturbation pattern, which is optimized via L_p-bounded manipulations with constraints on realizability (e.g., printability of colors). To conduct a more comprehensive evaluation, we optimized patterns based on three existing attacks [22, 44, 91]. As CAPatch [91] was originally designed to attack image captioning models, its objective term was adjusted as object misclassification in our context. For the projection-based SLAP attack [44], we reused its provided projector-specific parameters (e.g., the achievable color spectrum of the projector) for scalable simulated evaluation. In the physical world experiments detailed in Section 6.5, we obtained these values using our own projector hardware. We also investigated the impacts of diverse source/target labels and patch sizes. In our implementation of [22, 44], the larger patch is of 400×400 in a 1024×1024 mask, while a smaller patch is of 200×200 within the mask. The patch sizes of the CAPatch attack [91] followed their original settings, averaging 14.3% of the entire image.

Considering realizability and attacker motivations, we selected vehicles, cyclists, and pedestrians as the original objects for attaching adversarial patches. This is because they possess relatively large surfaces for adhering patches and also constitute safety-critical entities in transportation. This rationale also guided our selection of target misclassification labels, focusing on changes that could induce safety risks. On the other hand, applying patches to moving entities poses unique challenges overlooked by prior work. Particularly, vehicles often move at relatively high speeds from the perception view, so their angle and position can dramatically change even over short time periods. To address this and better simulate physical patches attached to moving objects, we used the groundtruth angle matrices to spatially transform the corresponding patches. Additionally, we used box vertex distances to the ego vehicle to identify surfaces facing the perception camera. In this way, we ensured that the patches adhered to angle changes and disappeared when no longer facing the AV. An example of such transformation is depicted in Figure 5.

Experiment Methodology. Before evaluating the defense, it is critical to first ensure it operates on valid attack outcomes where instances are misclassified by the perception model. However, since perception models are not perfect, they can also make false predictions even without adversarial perturbations. As our focus is on defense performance rather than issues with the perception model or attacks, we selectively chose instance sequences where the object was initially correctly identified by the perception model but was misclassified under adversarial perturbations.

We then applied our defense and evaluated it in terms of detection accuracy, label correction efficacy, and run-time efficiency. First, the inference results from PhySense were cross-checked with those from the perception module, and any mismatch was regarded as a detected attack. As such, the detection accuracy was calculated as the number of correctly detected instances divided by the total number of misclassifications made by the AV perception model. Beyond detection, we also explored the potential that PhySense could suggest correct labels. This correction accuracy was quantified as the ratio of correctly identified labels from PhySense to the total misclassification. In addition to the two accuracy metrics, the false positive rate (FPR) and false negative rate (FNR) were measured. Specifically, false positives indicate objects that were correctly classified by perception but erroneously identified as other objects by PhySense; conversely, false negatives denote objects that were misclassified by both the perception and PhySense as wrong labels. Lastly, the efficiency of PhySense was evaluated as its average end-to-end runtime in seconds.

Table 2: Evaluation results on datasets.

Patch Size	Datasets	# Obj		PhySense				Avg. Time
		Total	Attack	Detect Acc.	Correct Acc.	FPR	FNR	
Large	nuScenes	21763	15810	0.9965	0.9834	0.0514	0.0543	0.041
	KITTI	5212	3737	1.0000	0.9999	0.0000	0.0002	0.028
	Carla	26854	13089	0.9988	0.9886	0.0505	0.0024	0.037
Small	nuScenes	21763	16833	1.0000	0.9815	0.0476	0.0557	0.044
	KITTI	5212	4263	1.0000	0.9999	0.0000	0.0002	0.026
	Carla	26854	13252	0.9985	0.9894	0.0523	0.0039	0.032

Detect Acc. = Detection Accuracy; Avg. Time denotes the average run-time per input.

Results. The main results are summarized in Table 2. The number of objects represents the counts of sequences, wherein each contains several to tens of frames. Overall, PhySense achieves over 99% detection rate and correctly recovers 98% misclassified labels across datasets, showing the effectiveness of our approach. The correction accuracy is slightly lower than detection, because this metric additionally requires PhySense to precisely identify true labels. Besides, FPR and FNR are both relatively low. Upon investigation, we found that these missed cases mainly focused on confusing "Truck" and "Car" classes. This is because objects within the two classes exhibit similar characteristics, including 3D sizes (many trucks have similar sizes to vehicles [62]), behaviors (e.g., changing lanes and making turns), and interactions with other objects (e.g., following front cars). However, in practical scenarios, AVs are likely to take similar actions when encountering either category of objects, therefore such confusion may not lead to serious safety risks.

Moreover, we observe that the specific attack method and patch size do not significantly affect the performance. Intuitively, a larger patch could potentially degrade protection, since a larger surface area is covered by adversarial patterns that could affect texture analysis. However, the performance of PhySense remains stable even when facing large patches. Similarly, we found that PhySense

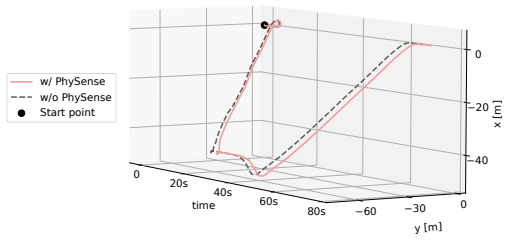

Figure 6: AV trajectories with and without PhySense.

achieved similar correction accuracy across the three implemented attacks, comparable to the overall performance. This can be attributed to our relatively robust feature extraction mechanisms. For instance, the texture analysis is designed to sample multiple small regions within the object ROI, each going through LBP and subsequently aggregated. This process can be less affected by varying patch sizes and diverse adversarial patterns generated by different attacks. Additionally, PhySense does not rely solely on visual features; they are weighted against other factors such as sizes and behaviors during reasoning. This multi-faceted approach further improves resiliency across attack methods and strengths.

Run-time Efficiency Results. The measured run time of PhySense is 0.037 seconds on average. Even though it increases the end-to-end delay, we found the delay did not introduce new control problems, owing to the pipeline design of framework processing in modern AV software. Based on the run-time breakdown of the key processing modules, we found the main bottleneck lies in the LBP texture extraction, which operates on $N = 8$ sampled regions per object to ensure robustness. To improve efficiency, reducing N is an option, however, it risks degrading performance if texture information becomes unreliable. We further investigate the impact of this parameter in ablation studies (Section 6.3).

To further understand how such delays impact AV behaviors, we deployed PhySense in the simulator, recorded the traveled path, and then used it to measure the control deviations introduced by the overhead of PhySense. Specifically, we set the vehicle to track a reference path in the simulated world and then measured the actual traveled trajectory with and without PhySense. The difference between the two trajectories was then calculated to quantify the control deviation. Figure 6 shows a sample pair of such trajectories. We can observe that the vehicle behaves almost identically under both settings. The average deviation between these two trajectories was $0.0266m$ for the x-axis and $0.0251m$ for the y-axis.

Comparison with Other Defenses. To understand the improvements PhySense could offer over prior defenses, a comparative study was conducted with three state-of-the-art defenses, PercepGuard [47], Jujutsu [16], and DiffPure [55]. PercepGuard employs an LSTM-based model to predict object labels from 2D bounding box trajectories, while Jujutsu and DiffPure are input purification approaches. The difference is that DiffPure processes the entire image to remove perturbations, while Jujutsu first localizes potential regions of adversarial perturbations and replaces them with reconstructed contents. We used their official implementations on GitHub with original hyperparameters for consistency.

The input purification defenses, Jujutsu and DiffPure, achieved correction accuracy of 11.4% and 25.3% respectively. Jujutsu was mainly hindered by its high detection FNR of 79.8% and imprecise localization of adversarial regions, while DiffPure was affected by its inaccurate content reconstruction. Specifically, Jujutsu's detection mechanism works by transplanting the region with the most salient features to images of other classes; if the classification on both images matches, the transplanted area is marked as adversarial. This mechanism was originally designed for universal adversarial patches that are effective on any object and can consistently induce the same misclassified label. However, in the context of physical adversarial attacks, we observed difficulties in simultaneously ensuring real-world robustness and consistent misclassification results across all object classes over a time period. In the dynamic physical world, adversarial patterns may not cause identical misclassification results when applied to other objects, thus they are wrongly regarded as benign by Jujutsu. Furthermore, such saliency-based localization of adversarial regions may not always be accurate. To quantify this, we calculated the IoU between the actual adversarial region and the one identified by Jujutsu. The mean IoU across tested images was 0.51, indicating that many patches were only partially sanitized and could still induce misclassification. DiffPure, on the other hand, applies transformations to all image inputs. However, we found the results often contain abnormal, repeated patterns, such as multiple wheel-shaped patterns on a vehicle. The root cause is that the model was pre-trained on images of dimensions 256×256, while the AV's perception model (YOLO in our setup) processes images of 416×416. Therefore, the model's receptive field and learned representations may not scale coherently on larger images. While DiffPure altered the classification results for 98.2% of attack images after processing, only 25.8% of them aligned with the true labels. Moreover, we found the modifications introduced by diffusion processing could sometimes cause misclassifications of the originally benign images as well.

In contrast, PercepGuard achieved a detection rate of 95.9% across all adversarial examples, indicating its reliable performance in attack detection. On the other hand, it only achieved 72.7% correction accuracy with an FPR of 0.233. The results across different categories revealed that PercepGuard achieved a 100% correction rate for the vehicle class but was less effective in identifying pedestrians and cyclists, often misclassifying them as vehicles. This discrepancy could be attributed to two reasons. First, the perturbations not only changed the perception results (i.e., labels) but also caused non-negligible variations in the bounding boxes within the AV perception module. During the training process of the defense, PercepGuard was enforced to associate these altered bounding boxes with the true label (i.e., vehicles), thus inadvertently biasing the model towards vehicles even when these features did not belong to this class. This also explained its relatively high FPR, where benign objects were sometimes identified as vehicles too. Second, we observed that some misclassified object sequences had missing frames. This was because adversarial examples, especially those realized in the physical world, often fail to consistently attack every frame under constraints. This resulted in objects appearing to "jump" between locations and were interpreted by PercepGuard as moving significantly faster than they were. As a result, some cyclists with perturbations were misclassified as vehicles in the defense.

Table 3: Correction accuracy in ablation studies.

Char. Features	Size	Texture	Behavior	Interaction
	0.702	0.870	0.833	0.886
# Texture Region	1	3	6	8
	0.952	0.958	0.970	0.983
Texture Descriptor	LPQ		GLCM	LBP
	0.960		0.971	0.983
Behavior Model	Bev-1		Bev-2	Bev-3
	0.977		0.973	0.983

These findings reveal the limitation of relying solely on perception bounding boxes, as they could be prone to bias and manipulations. To mitigate this issue, PhySense integrates multi-faceted characterization and employs holistic reasoning, thus enhancing resiliency and reliability. As such, while PhySense is not bulletproof of misclassification, it is designed to raise the bar for attackers, as altering multiple types of features consistently over time would require extensive changes in the physical environment. To further investigate how well PhySense works under such manipulations, it was evaluated against adaptive attackers in Section 6.6.

6.3 Ablation Studies

In addition to evaluating the overall performance of PhySense, we further conducted ablation studies to dissect the effectiveness of its individual components. These studies were structured around two lines of ablation groups: (1) the characterization features in PhySense were individually decoupled, and (2) the techniques for feature extraction and analysis were varied.

Decoupling Characterization Features. This line of studies involves four ablation groups, each having one characterization feature removed from PhySense. Specifically, we individually removed 3D sizes, textures, behaviors, and interactions from reasoning and re-train the defense model for each test. The results measured by correction accuracy are summarized in the first row of Table 3. It can be observed that the performance degraded significantly with the removal of each feature, showing that these characteristics all contribute to PhySense to non-negligible extents. Following the validation of the importance of these features, we proceeded to investigate the techniques for extracting and analyzing them.

Comparing Design Alternatives. We considered alternatives for three major components: the number of sampled regions for texture analysis, texture descriptor, and behavior recognition model. The first factor, as discussed in Section 6.2, is a hyperparameter that balances robustness and efficiency for texture analysis. We examined the impact of randomly sampled 1, 3, 6, and 8 regions for aggregated texture analysis. Additionally, we individually employed LBP, GLCM [51], and LPQ [57] as texture descriptors. As for the behavior model, we designed three sets of frameworks: (1) inspired by existing studies on modeling consumer journey, we adaptively designed a transformer with multi-head attention combined with three-layered MLP, denoted as *Bev-1*; (2) a three-layered LSTM model combined with additive attention and MLP with Leaky-ReLU activation, denoted as *Bev-2*; and (3) the current behavior model with AT-BiLSTM structure (Section 5.5), denoted as *Bev-3*.

Figure 7: Our real-world driving tests involved different target vehicles, patch sizes, environments, and attack vectors.

The results are summarized in Table 3, suggesting that our design achieves optimal in the current form. However, we only examined several alternative designs as tuning defense performance is not our primary objective. This is further discussed in Section 7.

6.4 PhySense Applied to Other Modalities

The underlying principle of PhySense is built on a 3D kinematic model for high-level reasoning. While our main focus lies within the vision domain, however, the same principle could potentially apply to other sensing modalities that support 3D reconstruction. Here we consider LiDAR as the representative module that approximates the physical world via light-projected 3D point clouds. For the 3D detection model serving as the lifting tool (Section 5.3), we employed Apollo v2.5 which operates on LiDAR raw data and outputs 3D bounding boxes of detected objects. This 3D representation of the perceived scene, together with the visual information is then used in PhySense framework for evaluation. The tested accuracy achieved 85.6%, which is lower than the main vision-based results in Section 6.2. Upon inspection, we found this is because the LiDAR model sometimes missed detection of objects in the point cloud (even though no adversarial manipulations on point clouds were conducted). To mitigate this issue and improve the LiDAR-based approach, a potential direction is to associate its results with vision outputs, forming the de facto sensor fusion solution. However, we leave it to future work as it is not the focus of this study, and our threat model focuses on single-modality attacks and defenses.

6.5 Evaluation via Real-world Driving Tests

Realizing Attacks in the Physical World. To test the real-world practicality of PhySense, we implemented attacks with three types of realization vectors, building upon existing approaches [22, 28, 44]. Specifically, we realized the created adversarial examples in the physical world by (1) printing them on paper and attaching them to target vehicles, (2) displaying them on screens mounted in the trunk, and (3) projecting them at a distance with a projector. In implementing these attacks, we considered four main factors, including scenarios (e.g., parking lots, residential areas, main roads), target vehicles (a small two-door vehicle and a larger SUV), attack vectors (printed patches, displayed patterns on screens, and projected perturbations), and patterns sizes (realized with different sizes of printing papers, monitors, and projector distances). Some examples are depicted in Figure 7, from left to right are (a) the smaller target vehicle carrying a patch displayed on the larger monitor in a parking lot, (b) the pattern projected on a larger target vehicle via a projector placed at a distance, and (c) a printed patch attached to the smaller vehicle on a main road. These illustrative

figures only represent a subset of our tests that fully crossed the aforementioned factors; for example, we also tested the larger target vehicle carrying the larger display.

We employed a Tesla Model 3 (2023 made) as the ego vehicle that the attacker aims to mislead. This vehicle is equipped with on-board cameras that record videos automatically. The recordings were extracted via a formatted USB drive inserted in the vehicle's USB port. For light projection attack [44], we used a ViewSonic PS502X projector with 4000 Lumens and a resolution of 1024×768, which are the same as the one used in the original study. Moreover, its optimization scheme incorporates the modeling of the projector's achievable color spectrum and reflectivity of projection surfaces, which are specific to the projector hardware and target objects. To adapt the attack in our physical experiments, we followed the same process that iteratively shined a set of colors on the surface of the target vehicle and collected the outputs captured by the camera. The data was then fit into the model to approximate physical transformations. However, we found that attacks realized via monitor display and light projection often failed in outdoor scenarios during daytime, affected by intense ambient illumination. As such, our evaluation against these attacks focused on dark environments such as evening roads and indoor settings like garages, consistent with existing evaluation strategies [28, 44]. While exploring effective attacks with bright ambient light is an important future direction, such scenarios still represent realistic threats to perception systems.

Evaluation Results. Our evaluation using this data followed a similar strategy in Section 6.2, and the results are summarized in Figure 8. We found that the overall correction accuracy is comparable to the performance on datasets, with a slight drop due to occasional occlusions from passing-by objects. Such issues hinder our ability to build accurate 3D models and the subsequent characterization, thus affecting defense accuracy. This highlights the challenge of capturing real-life driving videos. In controlled environments, the target vehicle carrying the patch is often fully visible due to simplicity of the scenario; in contrast, real-world scenarios are more dynamic and unpredictable. The perception camera's view can be occluded by various moving objects, including pedestrians, bicycles, and other vehicles. Under these conditions, PhySense can effectively mitigate impacts from small or short-duration occlusions, due to its robust 3D modeling and the use of temporal graphs to maintain consistency with previously unoccluded frames. However, addressing large and persisting occlusions remains an open question in the broad field of 3D recognition. Given the continuous advancements being made in recent years [15], we do not consider it a fundamental limitation of PhySense. To address this challenge and improve our defense, one potential direction is to augment the 3D recognition model with additional occluded data. This method involves systematically modifying training images to simulate various occlusion scenarios, which can be implemented by overlaying shapes or objects of different sizes, positions, and transparencies onto target objects. As such, this encourages the model to learn partial information and infer complete object structures. However, applying such enhancements also requires consideration of the balance between occlusion handling and overall accuracy, as well as the need to mitigate the risks of overfitting to occlusion patterns. We leave such exploration to future work.

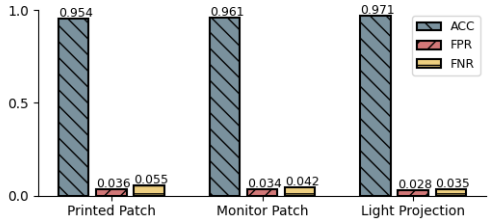

Figure 8: Evaluation results for real-world driving tests.

In addition, the defense accuracy and efficiency tested on the three types of attacks did not differ significantly. As adversarial patterns and realization vectors vary when different attacks are applied, they mainly affect visual features within localized regions of the target object. These variations have different levels of impact on textual analysis, due to their varying perturbations and altered surface reflectivity (e.g., caused by papers or projected light). However, PhySense remains relatively robust as it incorporates additional physical features for high-level reasoning, making it less affected by specific physical vectors. Moreover, the average run time for real-world driving tests is 0.039s, which is also consistent with that measured on the datasets.

6.6 PhySense against Adaptive Attackers

We further examined the resiliency against adaptive attackers with knowledge of PhySense. In this case, they have two primary adversarial goals - deceiving the perception model and bypassing PhySense. To achieve this, the attack needs to ensure the predicted label from the defense matches the output of the perception model.

Adaptive Attack Strategies. We consider three types of adaptive attack strategies in this study. First, knowing the exact physical features used in PhySense, the attacker may try to disrupt the feature extraction process that leads to the desired output. Second, the attacker could directly use the outputs of both the perception model and PhySense as feedback to guide the optimization of adversarial patterns. Lastly, the attacker can exploit the unique mechanism of temporal graphs that relies on defense results from previous frames to mislead the current frame. In these cases, the attacker's optimization still follows the form of Eq. 2, but with an additional term in the loss function measuring consistency between the perception model and defense outputs.

For implementation, we focused on interrupting texture analysis and behavior recognition phases for the first type of attack due to two reasons. First, they are critical features used in PhySense, since their weighted contributions are relatively high among others based on our inspection of the weighting matrices. Second, they both involve using machine learning algorithms, which are inherently susceptible to adversarial perturbations. As for the second type of attack, we used the sum of the probability of the target class from both the perception model and PhySense to quantify adversarial gain. Lastly to exploit temporal graphs, the attacker needs to bypass PhySense in consecutive frames, such that the misclassified label from prior frames will be directly assigned to the associated object in the current frame. However, a key challenge for these attacks lies in the lack of gradient propagation for adaptive

perturbation optimization, because PhySense also incorporates components that are not differentiable. To address this, we consider the attacker employing a search-based algorithm, specifically an adapted genetic algorithm where noise candidates are modeled as individual genomes. The fitness function is defined as the gain of objective goals and physical constraints are termed as penalties. A unique advantage of this approach is fast convergence speed within a large search space. Using these strategies, we individually adapted the optimization framework of existing attacks [22, 44, 91] to create adversarial examples.

Evaluation Results. The results revealed that the majority of adaptive perturbations often failed when applied to the test set. Only 12.0% and 7.0% of the adversarial examples generated by the first and second strategy could induce at least one successful bypass of PhySense on object sequences, while the third attack did not succeed. Upon investigation, we found that although they could induce misclassification on both the perception model and PhySense, most of their results were inconsistent between the two over time, thus failing to bypass our defense. We also tested them in the physical world following the same settings in Section 6.5, and found they rarely succeed in deceiving PhySense. This highlights an inherent difficulty for attackers, as they need to ensure consistent misclassification output between two systems, which could often be broken by dynamic physical conditions.

Among these strategies, the first attack appears to be most effective, potentially due to its fined-grained optimization on the selected important features. In contrast, the second attack is of coarse granularity, which disregards the inner workings of PhySense and directly optimizes on the fitness calculated based on final outputs. While this approach should be intuitively better since it implicitly incorporates all the feature sets, a key obstacle lies in the computation time. During the search process, each perturbation candidate has to iterate through all the training samples to quantify the overall effectiveness. While PhySense is designed to be efficient, the total processing time could be large. Given unlimited queries and time, the second strategy could achieve better adversarial perturbations, but this adds a significant burden on the attacker. Lastly for the third strategy, although it appears an obvious attack surface since PhySense directly skips inference on certain objects if they are considered consistent in previous frames, exploiting this mechanism requires the attacker to successfully compromise a set of history frames. While compromising a single frame is non-trivial as demonstrated by the first two adaptive attacks, this poses even more challenges to the attacker. In our implementation, we observed that the optimization was often trapped in local minimum and the fitness was difficult to improve. Therefore, the results revealed the feasibility for attackers to adaptively craft samples to bypass PhySense, however, they often face the challenge of dynamic physical conditions and excessive costs raised by our defense.

Summary and Discussions. Overall, the results indicated that PhySense remains relatively resilient when facing adaptive attackers. This is rooted in three reasons: first, our approach leverages diverse physical characteristics in spatial-temporal domains, which are relatively robust and hard to alter for malicious goals. Second, PhySense aggregates these features via holistic reasoning on graph-based structures, therefore compromising some of the features does not completely invalidate the protection. Lastly, the attackers aiming to evade PhySense have to incorporate multiple terms into optimization along with the adversarial goal. When bounded by physical constraints, the search space is significantly narrowed and therefore makes it much more difficult to find working perturbations satisfying multiple goals. However, with considerable computational resources and time, the attacker could still find the perturbations that compromise the majority of leveraged features while also misleading the perception model. To address this and further improve resiliency, a potential direction is to leverage more robust physical features, but the run-time efficiency could be a key obstacle in this regard. Additionally, while no defense is bulletproof, we suggest adopting a defense-in-depth approach by designing and integrating efficient input purification techniques. For more discussions on limitations and future directions please see Section 7.

7 Discussion and Limitations

Limited Set of Physical Characteristics Used for Defense. In this work, we focus on a limited set of physical features; therefore, the reasoning developed in PhySense is only limited to the scope related to those features. While involving more features could improve defense, they are also likely to induce larger overhead due to the need for additional computation and larger-scale inference. Moreover, simply adding more features does not always guarantee enhanced security, as their reliability and informativeness vary. As such, finding an optimal feature set balancing efficacy and efficiency is the key. To this end, our features were selected across both spatial and temporal domains, with joint consideration of uniqueness, physical grounding, and computational costs. Another limitation of PhySense stems from its reliance on sensor readings, making it potentially vulnerable to sensor attacks [81] that could lead to inaccurate physics-based characterization. To address these limitations and broaden the defense capabilities, future work could explore more diverse multi-modal features and combine hardware-based solutions like interference shielding and filtering. **Limitations in Feature Extraction Techniques.** For the same set of features, the techniques used for extraction are equally important to affect performance. In PhySense, these span DNN models, conventional CV algorithms like LBP, and rule-based identification. Some techniques may be coarse-grained (e.g. interaction rules), potentially degrading recovery accuracy. We currently address this by learning class-specific weights to balance contribution in the reasoning model. Besides, we experimentally validated in ablation studies that our design components were optimal among a set of alternatives. Further improvements could employ more advanced techniques for higher-fidelity extraction, such as modeling behaviors and interactions with more complex learning components.

Applications beyond Adversarial Defense. Outside the realm of defending adversarial attacks, techniques developed for PhySense can broadly enhance perception. The key insight is that reasoning higher-level characteristics provides valuable information complementary to DNN perception models. Therefore, it can serve as a general mechanism integrated into pipelines for improved scene understanding. More broadly, they also have potential applications in other domains like computer vision, enabling contextual holistic analysis to address challenges like occlusions. For general systems,

PhySense provides principles to integrate top-down contextual reasoning with bottom-up feature extraction for robustness.

8 Conclusion

In this work, we propose PhySense, an integrative reasoning-based defense for label recovery from physical adversarial examples in autonomous systems. Complementary to existing defenses, PhySense leverages robust physical world characteristics of objects and their relationships for multi-faceted understanding. For efficient and accurate reasoning, a novel CRF-based framework is proposed to model objects and correlations as structured spatial-temporal graphs. To improve efficiency for practical deployment, it is further optimized via task parallelization and pipelining based on workload profiles. The efficacy of PhySense is validated through experiments on both simulated datasets and real-world driving tests.

Acknowledgment

We thank the reviewers for their valuable feedback. This work was partially supported by the NSF (CNS-2038995, CNS-2154930, CNS-2238635, CNS-2403758), ARO (W911NF-24-1-0155), and Intel.

References

[1] Maksym Andriushchenko and Nicolas Flammarion. 2020. Understanding and Improving Fast Adversarial Training. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 16048–16059.

[2] Anish Athalye et al. 2018. Synthesizing robust adversarial examples. In *International conference on machine learning*. PMLR, 284–293.

[3] Philipp Benz, Chaoning Zhang, Adil Karjauv, and In So Kweon. 2021. Robustness may be at odds with fairness: An empirical study on class-wise accuracy. In *NeurIPS 2020 Workshop on Pre-registration in Machine Learning*. PMLR, 325–342.

[4] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. 2016. Simple online and realtime tracking. In *2016 IEEE international conference on image processing (ICIP)*. IEEE, 3464–3468.

[5] Irving Biederman. 1987. Recognition-by-components: a theory of human image understanding. *Psychological review* 94, 2 (1987), 115.

[6] Gary Bishop, Greg Welch, et al. 2001. An introduction to the kalman filter. *Proc of SIGGRAPH, Course* 8, 27599-23175 (2001), 41.

[7] Richard E Boyatzis. 1998. *Transforming qualitative information: Thematic analysis and code development*. sage.

[8] Holger Caesar, Varun Bankiti, Alex H Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. 2020. nuscenes: A multimodal dataset for autonomous driving. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 11621–11631.

[9] Yulong Cao et al. 2021. Invisible for both camera and lidar: Security of multi-sensor fusion based perception in autonomous driving under physical-world attacks. In *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE, 176–194.

[10] Yulong Cao, Chaowei Xiao, Benjamin Cyr, Yimeng Zhou, Won Park, Sara Rampazzi, Qi Alfred Chen, Kevin Fu, and Z Morley Mao. 2019. Adversarial sensor attack on lidar-based perception in autonomous driving. In *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*. 2267–2281.

[11] Amirhosein Chahe, Chenan Wang, Abhishek Jeyapratap, Kaidi Xu, and Lifeng Zhou. 2023. Dynamic adversarial attacks on autonomous driving systems. *arXiv preprint arXiv:2312.06701* (2023).

[12] Tianqi Chen, Tong He, Michael Benesty, Vadim Khotilovich, Yuan Tang, Hyunsu Cho, Kailong Chen, Rory Mitchell, Ignacio Cano, Tianyi Zhou, et al. 2015. Xgboost: extreme gradient boosting. *R package version 0.4-2* 1, 4 (2015), 1–4.

[13] Tao Chen, Longfei Shangguan, Zhenjiang Li, and Kyle Jamieson. 2020. Metamorph: Injecting inaudible commands into over-the-air voice controlled systems. In *Network and Distributed Systems Security (NDSS) Symposium*.

[14] Xiaozhi Chen, Kaustav Kundu, Ziyu Zhang, Huimin Ma, Sanja Fidler, and Raquel Urtasun. 2016. Monocular 3d object detection for autonomous driving. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2147–2156.

[15] Yongjian Chen, Lei Tai, Kai Sun, and Mingyang Li. 2020. Monopair: Monocular 3d object detection using pairwise spatial relationships. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 12093–12102.

[16] Zitao Chen, Pritam Dash, and Karthik Pattabiraman. 2023. Jujutsu: A two-stage defense against adversarial patch attacks on deep neural networks. In *Proceedings of the 2023 ACM Asia Conference on Computer and Communications Security*. 689–703.

[17] Jeremy Cohen et al. 2019. Certified adversarial robustness via randomized smoothing. In *international conference on machine learning*. PMLR, 1310–1320.

[18] Ben Dickson. 2021. Tesla AI chief explains why self-driving cars don't need lidar. https://bdtechtalks.com/2021/06/28/tesla-computer-vision-autonomous-driving/.

[19] Chen Ding, Xipeng Shen, Kirk Kelsey, Chris Tice, Ruke Huang, and Chengliang Zhang. 2007. Software behavior oriented parallelization. *ACM SIGPlan Notices* 42, 6 (2007), 223–234.

[20] Nikita Dvornik, Julien Mairal, and Cordelia Schmid. 2018. Modeling visual context is key to augmenting object detection datasets. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

[21] Scott Ettinger et al. 2021. Large Scale Interactive Motion Forecasting for Autonomous Driving: The Waymo Open Motion Dataset. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 9710–9719.

[22] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. 2018. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1625–1634.

[23] Christoph Feichtenhofer et al. 2017. Detect to track and track to detect. In *Proceedings of the IEEE international conference on computer vision*. 3038–3046.

[24] Andreas Geiger, Philip Lenz, and Raquel Urtasun. 2012. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition*. IEEE, 3354–3361.

[25] Robert Geirhos et al. 2019. ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *7th International Conference on Learning Representations, ICLR 2019*.

[26] Harshayu Girase et al. 2021. LOKI: Long Term and Key Intentions for Trajectory Prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 9803–9812.

[27] Ian J. Goodfellow et al. 2015. Explaining and Harnessing Adversarial Examples. In *3rd International Conference on Learning Representations, ICLR 2015*.

[28] Shahar Hoory et al. 2020. Dynamic adversarial patch for evading object detection models. *arXiv preprint arXiv:2010.13070* (2020).

[29] Chengyin Hu et al. 2023. Adversarial color projection: A projector-based physical-world attack to DNNs. *Image and Vision Computing* 140 (2023), 104861.

[30] Hou-Ning Hu, Yung-Hsu Yang, Tobias Fischer, Trevor Darrell, Fisher Yu, and Min Sun. 2022. Monocular quasi-dense 3d object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45, 2 (2022), 1992–2008.

[31] Andrew Ilyas et al. 2019. Adversarial examples are not bugs, they are features. *Advances in neural information processing systems* 32 (2019).

[32] Mordor Intelligence. 2023. Autonomous Vehicle Market Size & Share Analysis - Growth Trends & Forecasts (2023 - 2028). https://www.mordorintelligence.com/industry-reports/autonomous-driverless-cars-market-potential-estimation.

[33] Mohamed Isse. 2020. How Mobileye and Tesla are Tackling 3D Perception. https://www.autovision-news.com/whitepaper/multiple-computer-vision-engines-how-mobileye-and-tesla-are-tackling-3d-perception/.

[34] Xiaojun Jia, Xingxing Wei, Xiaochun Cao, and Hassan Foroosh. 2019. Comdefend: An efficient image compression model to defend adversarial examples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*.

[35] Rachyl Jones. 2024. Mercedes becomes the first automaker to sell autonomous cars in the U.S. that don't come with a requirement that drivers watch the road. https://fortune.com/2024/04/18/mercedes-self-driving-autonomous-cars-california-nevada-level-3-drive-pilot/.

[36] Rudolph Emil Kalman. 1960. A new approach to linear filtering and prediction problems. (1960).

[37] Harold W Kuhn. 1955. The Hungarian method for the assignment problem. *Naval research logistics quarterly* 2, 1-2 (1955), 83–97.

[38] Monica Lam. 1988. Software pipelining: An effective scheduling technique for VLIW machines. In *Proceedings of the ACM SIGPLAN 1988 conference on Programming Language design and Implementation*. 318–328.

[39] Leslie Lamport. 1977. Proving the correctness of multiprocess programs. *IEEE transactions on software engineering* 2 (1977), 125–143.

[40] Alexander Levine and Soheil Feizi. 2020. (De) Randomized smoothing for certifiable defense against patch attacks. *Advances in Neural Information Processing Systems* 33 (2020), 6465–6475.

[41] Ao Li, Marion Sudvarg, Han Liu, Zhiyuan Yu, Chris Gill, and Ning Zhang. 2022. Polyrhythm: Adaptive tuning of a multi-channel attack template for timing interference. In *2022 IEEE Real-Time Systems Symposium (RTSS)*. IEEE, 225–239.

[42] Juncheng Li, Frank Schmidt, and Zico Kolter. 2019. Adversarial camera stickers: A physical camera-based attack on deep learning systems. In *International Conference on Machine Learning*. PMLR, 3896–3904.

[43] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. 2016. Ssd: Single shot multibox detector. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*. Springer, 21–37.

[44] Giulio Lovisotto et al. 2021. {SLAP}: Improving physical adversarial examples with {Short-Lived} adversarial perturbations. In *30th USENIX Security Symposium (USENIX Security 21)*. 1865–1882.

[45] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. 2018. Towards Deep Learning Models Resistant to Adversarial Attacks. In *6th International Conference on Learning Representations, ICLR 2018*.

[46] Topi Mäenpää and Matti Pietikäinen. 2005. Texture analysis with local binary patterns. In *Handbook of pattern recognition and computer vision*. World Scientific.

[47] Yanmao Man et al. 2023. That person moves like a car: Misclassification attack detection for autonomous systems using spatiotemporal consistency. In *32nd USENIX Security Symposium (USENIX Security 23)*. 6929–6946.

[48] Yanmao Man, Raymond Muller, Ming Li, Z Berkay Celik, and Ryan Gerdes. 2022. Evaluating perception attacks on prediction and planning of autonomous vehicles. In *USENIX Security Symposium Poster Session*.

[49] David Marr and Herbert Keith Nishihara. 1978. Representation and recognition of the spatial organization of three-dimensional shapes. *Proceedings of the Royal Society of London. Series B. Biological Sciences* 200, 1140 (1978), 269–294.

[50] Jan Hendrik Metzen, Nicole Finnie, and Robin Hutmacher. 2021. Meta Adversarial Training against Universal Patches. In *ICML 2021 Workshop on Adversarial Machine Learning*. https://openreview.net/forum?id=sePThSlRHr

[51] P Mohanaiah, P Sathyanarayana, and L GuruKumar. 2013. Image texture feature extraction using GLCM approach. *International journal of scientific and research publications* 3, 5 (2013), 1–5.

[52] Raymond Muller. 2022. Drivetruth: Automated autonomous driving dataset generation for security applications. In *Workshop on Automotive and Autonomous Vehicle Security (AutoSec)*.

[53] Raymond Muller, Yanmao Man, Ming Li, Ryan Gerdes, Jonathan Petit, and Z. Berkay Celik. 2024. VOGUES: Validation of Object Guise using Estimated Components. In *33rd USENIX Security Symposium (USENIX Security 24)*. USENIX Association, Philadelphia, PA, 6327–6344. https://www.usenix.org/conference/usenixsecurity24/presentation/muller

[54] Muzammal Naseer, Salman Khan, and Fatih Porikli. 2019. Local gradients smoothing: Defense against localized adversarial attacks. In *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, 1300–1307.

[55] Weili Nie, Brandon Guo, Yujia Huang, Chaowei Xiao, Arash Vahdat, and Anima Anandkumar. 2022. Diffusion Models for Adversarial Purification. In *International Conference on Machine Learning (ICML)*.

[56] Byeongjoon Noh and Hwasoo Yeo. 2022. A novel method of predictive collision risk area estimation for proactive pedestrian accident prevention system in urban surveillance infrastructure. *Transportation research part C: emerging technologies*.

[57] Ville Ojansivu and Janne Heikkilä. [n. d.]. Blur insensitive texture classification using local phase quantization. In *Image and Signal Processing: 3rd International Conference, ICISP 2008. Cherbourg-Octeville, France, July 1-3*. Springer.

[58] Sukrut Rao, David Stutz, and Bernt Schiele. 2020. Adversarial training against location-optimized adversarial patches. In *European Conference on Computer Vision*. Springer, 429–448.

[59] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. 2016. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 779–788.

[60] Joseph Redmon and Ali Farhadi. 2018. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767* (2018).

[61] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems* 28 (2015).

[62] Kaustubh V Sakhare, Tanuja Tewari, and Vibha Vyas. 2020. Review of vehicle detection systems in advanced driver assistant systems. *Archives of Computational Methods in Engineering* 27, 2 (2020), 591–610.

[63] Pouya Samangouei, Maya Kabkab, and Rama Chellappa. 2018. Defense-GAN: Protecting Classifiers Against Adversarial Attacks Using Generative Models. In *International Conference on Learning Representations*.

[64] Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing* 45, 11 (1997), 2673–2681.

[65] Sima Siami-Namini, Neda Tavakoli, and Akbar Siami Namin. 2019. The performance of LSTM and BiLSTM in forecasting time series. In *2019 IEEE International conference on big data (Big Data)*. IEEE, 3285–3292.

[66] Dawn Song et al. 2018. Physical adversarial examples for object detectors. In *12th USENIX workshop on offensive technologies (WOOT 18)*.

[67] Lynn KA Sörensen et al. 2023. Mechanisms of human dynamic object recognition revealed by sequential deep neural networks. *PLOS Computational Biology* (2023).

[68] Junshu Tang, Tengfei Wang, Bo Zhang, Ting Zhang, Ran Yi, Lizhuang Ma, and Dong Chen. 2023. Make-it-3d: High-fidelity 3d creation from a single image with diffusion prior. In *Proceedings of the IEEE/CVF international conference on computer vision*. 22819–22829.

[69] Trisha Thadani. 2024. Waymo robotaxis can hit California highways after state approval. https://www.washingtonpost.com/technology/2024/03/01/waymo-expands-california-los-angeles-highways/.

[70] Kalibinuer Tiliwalidi. 2023. Adversarial Camera Patch: An Effective and Robust Physical-World Attack on Object Detectors. *arXiv preprint arXiv:2312.06163*.

[71] Dimitris Tsipras, Shibani Santurkar, Logan Engstrom, Alexander Turner, and Aleksander Madry. 2019. Robustness May Be at Odds with Accuracy. In *International Conference on Learning Representations*.

[72] Leonard Elia Van Dyck, Roland Kwitt, Sebastian Jochen Denzler, and Walter Roland Gruber. 2021. Comparing object recognition in humans and deep convolutional neural networks—an eye tracking study. *Frontiers in Neuroscience* (2021).

[73] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems* 30 (2017).

[74] Zifan Wang, Lin Zhang, Qinru Qiu, and Fanxin Kong. 2023. Catch you if pay attention: Temporal sensor attack diagnosis using attention mechanisms for cyber-physical systems. In *2023 IEEE Real-Time Systems Symposium (RTSS)*. IEEE.

[75] Zhongdao Wang, Liang Zheng, Yixuan Liu, Yali Li, and Shengjin Wang. 2020. Towards real-time multi-object tracking. In *European Conference on Computer Vision*. Springer, 107–122.

[76] Huixiang Wen, Shan Chang, and Luo Zhou. 2023. Light projection-based physical-world vanishing attack against car detection. In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE.

[77] Eric Wong, Leslie Rice, and J. Zico Kolter. 2020. Fast is better than free: Revisiting adversarial training. In *International Conference on Learning Representations*.

[78] Tong Wu, Liang Tong, and Yevgeniy Vorobeychik. 2020. Defending Against Physically Realizable Attacks on Image Classification. In *International Conference on Learning Representations*.

[79] Chong Xiang, Saeed Mahloujifar, and Prateek Mittal. 2022. {PatchCleanser}: Certifiably robust defense against adversarial patches for any image classifier. In *31st USENIX Security Symposium (USENIX Security 22)*. 2065–2082.

[80] Chaowei Xiao, Zhongzhu Chen, Kun Jin, Jiongxiao Wang, Weili Nie, Mingyan Liu, Anima Anandkumar, Bo Li, and Dawn Song. 2023. DensePure: Understanding Diffusion Models for Adversarial Robustness. In *The Eleventh International Conference on Learning Representations*.

[81] Chen Yan, Hocheol Shin, Connor Bolton, Wenyuan Xu, Yongdae Kim, and Kevin Fu. 2020. Sok: A minimalist approach to formalizing analog sensor security. In *2020 IEEE Symposium on Security and Privacy (SP)*. IEEE, 233–248.

[82] Jongmin Yoon, Sung Ju Hwang, and Juho Lee. 2021. Adversarial purification with score-based generative models. In *International Conference on Machine Learning*. PMLR, 12062–12072.

[83] Zhiyuan Yu, Yuanhaur Chang, Ning Zhang, and Chaowei Xiao. 2023. {SMACK}: Semantically Meaningful Adversarial Audio Attack. In *32nd USENIX Security Symposium (USENIX Security 23)*. 3799–3816.

[84] Zhiyuan Yu, Zack Kaplan, Qiben Yan, and Ning Zhang. 2021. Security and privacy in the emerging cyber-physical world: A survey. *IEEE Communications Surveys & Tutorials* 23, 3 (2021), 1879–1919.

[85] Zhiyuan Yu, Shixuan Zhai, and Ning Zhang. 2023. Antifake: Using adversarial audio to prevent unauthorized speech synthesis. In *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security*. 460–474.

[86] LI Yufeng, YANG Fengyu, LIU Qi, LI Jiangtao, and CAO Chenhong. 2023. Light can be Dangerous: Stealthy and Effective Physical-world Adversarial Attack by Spot Light. *Computers & Security* (2023), 103345.

[87] Wei Zhan et al. 2019. INTERACTION Dataset: An INTERnational, Adversarial and Cooperative moTION Dataset in Interactive Driving Scenarios with Semantic Maps. *arXiv:1910.03088 [cs, eess]* (Sept. 2019).

[88] Wei Zhan, Liting Sun, Di Wang, Yinghan Jin, and Masayoshi Tomizuka. 2019. Constructing a highly interactive vehicle motion dataset. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 6415–6420.

[89] Dinghuai Zhang, Tianyuan Zhang, Yiping Lu, Zhanxing Zhu, and Bin Dong. 2019. You Only Propagate Once: Accelerating Adversarial Training via Maximal Principle. In *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), Vol. 32. Curran Associates, Inc.

[90] Liang Zhang, Guangming Zhu, Lin Mei, Peiyi Shen, Syed Afaq Ali Shah, and Mohammed Bennamoun. 2018. Attention in convolutional LSTM for gesture recognition. *Advances in neural information processing systems* 31 (2018).

[91] Shibo Zhang, Yushi Cheng, Wenjun Zhu, Xiaoyu Ji, and Wenyuan Xu. 2023. {CAPatch}: Physical Adversarial Patch against Image Captioning Systems. In *32nd USENIX Security Symposium (USENIX Security 23)*. 679–696.

[92] Zhenglong Zhou and Chaz Firestone. 2019. Humans can decipher adversarial images. *Nature communications* 10, 1 (2019), 1334.

[93] Hong Zhu, Shengzhi Zhang, and Kai Chen. 2023. Ai-guardian: Defeating adversarial attacks using backdoors. In *2023 IEEE Symposium on Security and Privacy (SP)*. IEEE, 701–718.

[94] Wenjun Zhu, Xiaoyu Ji, Yushi Cheng, Shibo Zhang, and Wenyuan Xu. 2023. TPatch: A Triggered Physical Adversarial Patch. In *32nd USENIX Security Symposium (USENIX Security 23)*. USENIX Association, Anaheim, CA, 661–678.

[95] Alon Zolfi, Moshe Kravchik, Yuval Elovici, and Asaf Shabtai. 2021. The translucent patch: A physical and universal attack on object detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 15232–15241.

Table 4: Interaction Categories and Conditions

Object Pair ($o_1 - o_2$)	Interaction Categories	Rules	Identification
Pedestrian-Pedestrian	Accompany	① $v_1, v_2 \leq 2$ ② $D(o_1, o_2) \leq 2$ ③ $\lvert v_1 - v_2 \rvert \leq 1$	① ∧ ② ∧ ③
Vehicle-Pedestrian	Pre-Yield	① $D(o_1, o_2) \geq 2$ ② $a_1 \leq -0.1$ OR $v_1 \leq 0.1$ AND $v_2 \leq 2$ ③ $\exists t > 0 : d(p_1(t), p_2(t)) = 0$ AND $\Delta DTCP_{1,2} < 0$	① ∧ ② ∧ ③
	Post-Yield	① $D(o_1, o_2) \geq 2$ ② $a_1 \leq -0.1$ OR $v_1 \leq 0.1$ AND $v_2 \leq 2$ ③ $\forall t > 0 : d(p_1(t), p_2(t)) \neq 0$ AND $\Delta DTCP_{1,2} > 0$	① ∧ ② ∧ ③
	After-Yield	① $D(o_1, o_2) \geq 2$ ② $\Delta DTCP_{1,2} > 0$ AND $a_1 \geq 0.1$ AND $v_2 \leq 2$	① ∧ ②
Pedestrian-Vehicle	Pre-Yield	① $D(o_1, o_2) \geq 2$ ② $a_1 \leq -0.1$ OR $v_1 \leq 0.1$ ③ $\exists t > 0 : d(p_1(t), p_2(t)) = 0$ AND $\Delta DTCP_{1,2} < 0$	① ∧ ② ∧ ③
	Post-Yield	① $D(o_1, o_2) \geq 2$ ② $a_1 \leq -0.1$ OR $v_1 \leq 0.1$ ③ $\forall t > 0 : d(p_1(t), p_2(t)) \neq 0$ AND $\Delta DTCP_{1,2} > 0$	① ∧ ② ∧ ③
	After-Yield	① $D(o_1, o_2) \geq 2$ ② $\Delta DTCP_{1,2} > 0$ AND $a_1 \geq 0.1$	① ∧ ②
Vehicle-Cyclist/ Cyclist-Vehicle	Yielding	① $D(o_1, o_2) \geq 2$ ② $a_1 \leq 0.1$ OR $v_1 \leq 0.1$ OR $a_2 \leq 0.1$ OR $v_2 \leq 0.1$ ③ $\exists t > 0 : d(p_1(t), p_2(t)) = 0$ AND $\Delta DTCP_{1,2} < 0$	① ∧ ② ∧ ③
	Post-Yield	① $D(o_1, o_2) \geq 2$ ② $a_1 \leq 0.1$ OR $v_1 \leq 0.1$ OR $a_2 \leq 0.1$ OR $v_2 \leq 0.1$ ③ $CRA_{o_1 \cap o_2} < 0.3$	① ∧ ② ∧ ③
	After-Yield	① $D(o_1, o_2) \geq 2$ ② $\Delta DTCP_{1,2} > 0$ AND $a_1 \geq 0.1$ AND $a_2 \geq 0.1$	① ∧ ②
	Following	① $D(o_1, o_2) \geq 2$ ② $\lvert v_1 - v_2 \rvert \leq 2$ AND $CRA_{o_1 \cap o_2} > 0.3$	① ∧ ②
	Move in parallel	① $D(o_1, o_2) \geq 2$ ② $D(o_1, o_2) < 3$ AND $(\Delta\theta_{\mathbf{v_1},\mathbf{v_2}} < \pi \times \frac{15}{180}$ OR $\lvert \pi - \Delta\theta_{\mathbf{v_1},\mathbf{v_2}} \rvert < \pi \times \frac{15}{180})$	① ∧ ②
	Pre-Overtake	① $D(o_1, o_2) \geq 2$ ② $a_1 \geq 0.1$ AND $v_1 > v_2$ AND $\Delta CRA_{o_1 \cap o_2} < 0$ ③ $\Delta\theta_{\mathbf{v_1},\mathbf{v_2}} < \pi \times \frac{30}{180}$	① ∧ ② ∧ ③
	Post-Overtake	① $D(o_1, o_2) \geq 2$ ② $\Delta CRA_{o_1 \cap o_2} > 0$ AND $\Delta\theta_{\mathbf{v_1},\mathbf{v_2}} < \pi \times \frac{30}{180}$	① ∧ ②
Cyclist-Pedestrian	Pre-Yield	① $a_1 \leq -0.1$ OR $v_1 \leq 0.1$ AND $v_2 < 2$ ② $\exists t > 0 : d(p_1(t), p_2(t)) = 0$ AND $\Delta DTCP_{1,2} < 0$	① ∧ ②
	Post-Yield	① $a_1 \leq -0.1$ OR $v_1 \leq 0.1$ AND $v_2 < 2$ ② $\forall t > 0 : d(p_1(t), p_2(t)) \neq 0$ AND $\Delta DTCP_{1,2} > 0$	① ∧ ②
	After-Yield	① $\Delta D(o_1, o_2) > 0$ AND $a_1 \geq -0.1$ AND $v_2 < 2$	①
Pedestrian-Cyclist	Pre-Yield	① $a_1 \leq 0.1$ OR $v_1 \leq 0.1$ AND $v_2 < 7$ ② $\exists t > 0 : d(p_1(t), p_2(t)) = 0$ AND $\Delta DTCP_{1,2} < 0$	① ∧ ②
	Post-Yield	① $a_1 \leq 0.1$ OR $v_1 \leq 0.1$ AND $v_2 < 7$ ② $\forall t > 0 : d(p_1(t), p_2(t)) \neq 0$ AND $\Delta DTCP_{1,2} > 0$	① ∧ ②
	After-Yield	① $\Delta D(o_1, o_2) > 0$ AND $a_1 \geq 0.1$ AND $v_2 < 7$	①
Cyclist-Cyclist	Potential Turning	① $v_1, v_2 < 7$ ② $a_1 \geq 0.1$ OR $a_2 \geq 0.1$ AND $\Delta\theta_{\mathbf{v_1},\mathbf{v_2}} < 0$	① ∧ ②
	Following	① $v_1, v_2 < 7$ ② $\lvert v_1 - v_2 \rvert < 2$ AND $CRA_{o_1 \cap o_2} \geq 0.3$	① ∧ ②
	Move in parallel	① $v_1, v_2 < 7$ ② $D(o_1, o_2) < 3$ AND $(\Delta\theta_{\mathbf{v_1},\mathbf{v_2}} < \pi \times \frac{15}{180}$ OR $\lvert \pi - \Delta\theta_{\mathbf{v_1},\mathbf{v_2}} \rvert < \pi \times \frac{15}{180})$	① ∧ ②
	Pre-Overtake	① $v_1, v_2 < 7$ AND $1 - \cos\theta_{\mathbf{v_1},\mathbf{a_1}} < 0.1$ ② $a_1 \geq -0.1$ OR $v_1 > v_2$ ③ $\Delta CRA_{o_1 \cap o_2} < 0$ OR $CRA_{o_1 \cap o_2} < 0.1$	① ∧ ② ∧ ③
	Post-Overtake	① $v_1, v_2 < 7$ ② $\Delta CRA_{o_1 \cap o_2} > 0$ AND $\Delta\theta_{\mathbf{v_1},\mathbf{v_2}} < \pi \times \frac{30}{180}$	① ∧ ②

[1] Prerequisite: (a) two nodes have an intersection along the velocity direction AND $\lvert TTCP_1 - TTCP_2 \rvert \leq 2$ AND $\max(TTCP_1, TTCP_2) \leq 5$; (b) $CRA_{o_1 \cap o_2} > 0$
[2] CRA = Collision Risk Area; TTCP = time-to-conflict-point; DTCP = distance-to-conflict-point;