NaturalADV: An Exploratory Framework to Balance Adversarial Strength and Stealth in Autonomous Driving Environments

Meriel von Stein, LESS Lab @ University of Virginia

Motivation

Deep Neural Networks (DNNs) have become integral to various real-world autonomous mobile systems, from self-driving cars to food delivery robots.

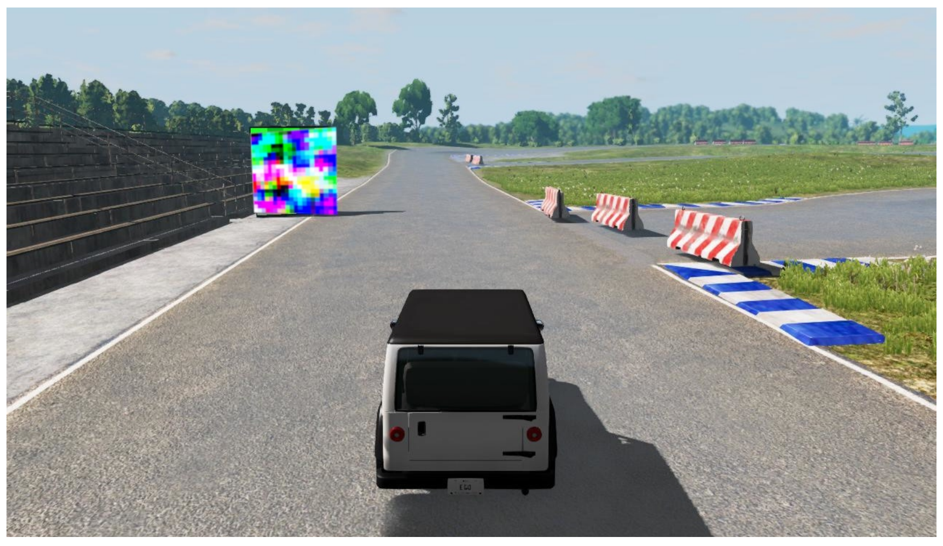


Figure 1. A DeepBillboard [3] in-situ patch attack.

However, current adversarial attack techniques often focus on maximizing the attack strength at the cost of naturalness, leading to examples that are easily detected by humans or deviate significantly from the expected input distribution. This trade-off between adversarial effectiveness and natural appearance presents a critical challenge in ensuring the robustness and reliability of DNNs in practical settings.

Perturbation Stealthiness and Naturalness

When designing adversarial attacks for deployment scenarios, it is essential to distinguish between stealthiness and naturalness. Stealthiness refers to the perceptual imperceptibility of the perturbation; a stealthy attack introduces minimal visual artifacts, making it hard for a human observer to detect any manipulation. Naturalness, on the other hand, refers to how well the adversarial example aligns with the expected distribution of inputs—whether it "looks real" or conforms to what the model would typically encounter. An attack might be stealthy but lack naturalness if, for instance, it is an abstract or unrealistic pattern that might never occur in a real-world setting.

Experiments

Perturbation Strength

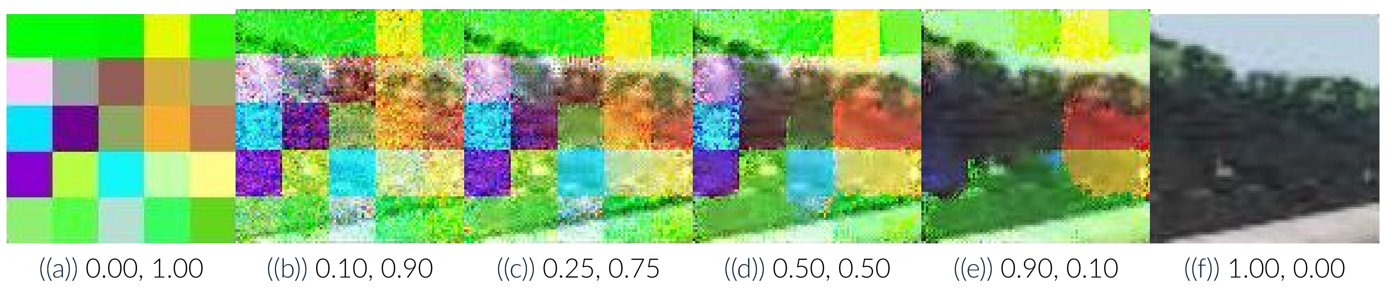
We explore a range of weights for similarity and prediction (see Equation 1) and report several performance metrics. Column 2 shows the resulting structural similarity index measure (SSIM) score when comparing the benign target patch and the generated adversarial patch. Column 3 shows the crash rate when the patch is deployed in simulation like in Figure 1. Column 4 shows the average deviation in the vehicle's trajectory from the centerline of the road.

Weights (sim, pred)	SSIM Score	Crash Rate	Avg. traj. deviation
0.00, 1.00	0.24	53%	2.75m
0.10, 0.90	0.21	22%	2.45m
0.25, 0.75	0.33	18%	2.37m
0.50, 0.50	0.46	2%	2.31m
0.75, 0.25	0.51	0%	2.21m
0.90, 0.10	0.70	0%	2.24m
1.00, 0.00	1.00	0%	2.23m

Table 1. Performance metrics for a range of weights using NaturalADV.

As Table 1 shows, perturbation strength diminishes inversely to SSIM score, where the generated patch resembles more and more closely the benign target patch. However, the generated patch still retains the ability to crash the vehicle in deployment when image similarity loss and prediction loss are equally weighted.

Perturbation Naturalness



((a)) 0.00, 1.00 ((b)) 0.10, 0.90 ((c)) 0.25, 0.75 ((d)) 0.50, 0.50 ((e)) 0.90, 0.10 ((f)) 1.00, 0.00

Figure 2. Patch appearances from original perturbation to benign target patch

Perturbation naturalness is dictated by the choice of metric, in this study SSIM. SSIM is designed to compare two images in terms of luminance, contrast, and structure, or edge detection. It was originally designed for black and white images and as a result preserves colors of the original perturbation well into the $(0.90, 0.10)$ weighting.

NaturalADV Framework Contributions

This poster presents the Natural Adversarial DNN Validation (NaturalADV) framework for balancing the trade-off between adversarial strength and naturalness of the adversarial patch's appearance. NaturalADV can incorporate a number of differentiable naturalness metrics, works with various gradient traversal algorithms, and scales to attacks represented in multiple sensor readings. Our contributions are:

- a framework, NaturalADV, to balance the trade-off between adversarial strength and naturalness for in-situ adversarial patch attacks;
- a proof of concept study showing the naturalness-strength tradeoff for the motivating example; and
- an open-source repository with tool and data for reproducibility available at `https://github.com/MissMeriel/NaturalADV`.

Perturbation Generation Loop

Figure 3 depicts a high-level overview of the generation loop for the NaturalADV framework. It takes in two images of the patch region of the deployment environment, one with the original adversarial perturbation known to have high adversarial strength (the original patch) and one that the user considers natural (the target patch), an image set `imgs` taken from an ADS navigating a driving environment without an adversarial patch, a navigation `DNN`, iterations of FGSM `iters`, a differentiable image similarity metric `similarity`, and `weights` for the two loss terms `w1` for the image similarity loss and `w2` for the perturbed prediction loss between the original perturbation and the target patch.

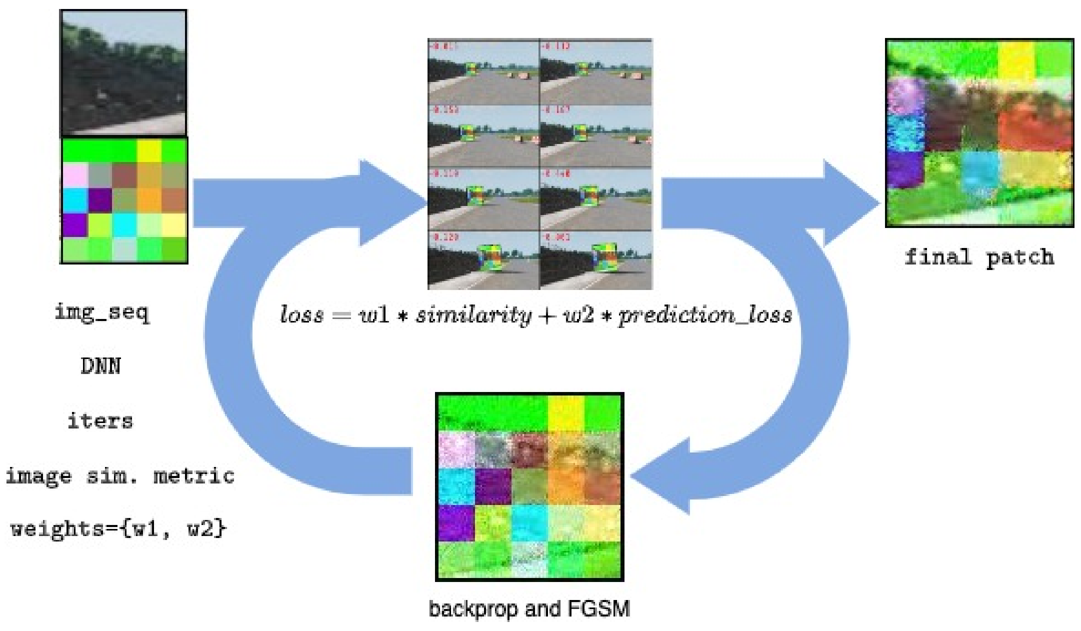


Figure 3. NaturalADV perturbation generation loop.

The framework alternates between calculating the loss function (see Equation 1) and backpropagation combined with Fast Sign Gradient Method (FGSM) to adjust the similarity of the patch to match the target image, while still retaining the adversarial strength of the original high-strength patch. Similarity and strength are prioritized according to parameterized `weights`. After `iters` loops, the generation loop exits and returns the final patch for injection into a driving environment.

NaturalADV Loss Function

The perturbation loop relies on a loss function for which preserving image similarity versus perturbation strength has been parameterized:

$$\begin{aligned} loss = w1 \times similarity(img_{natural}, img_{originial\_patch}) \\ +w2 \times L1(DNN(imgs + img_{originial\_patch}), DNN(imgs + img_{natural\_patch})) \end{aligned} \tag{1}$$

where $similarity$ is any differentiable image similarity metric (e.g. SSIM, German-McClure, Welsch, etc.), $img_{natural}$ is the natural image is that supposed to be an image or a patch? WE need to be super careful here differentiating patch and the image whole image, seems like we are mixing them up? we want the adversarial patch to look like, $img_{originial\_peturbation}$ is the high-strength but unnatural-looking (or un-stealthy) original adversarial patch, $DNN(imgs + img_{originial\_peturbation}$ is the DNN prediction output for the original high strength perturbation, $DNN(imgs + img_{natural\_peturbation}$ is the DNN prediction output for the current version of the natural perturbation, and w_x are weights to prioritize image similarity loss or DNN prediction loss.

References

[1] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song.
Natural adversarial examples.
In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15262–15271, 2021.

[2] Meriel von Stein, David Shriver, and Sebastian Elbaum.
Deepmaneuver: Adversarial test generation for trajectory manipulation of autonomous vehicles.
IEEE Transactions on Software Engineering, 2023.

[3] Husheng Zhou, Wei Li, Zelun Kong, Junfeng Guo, Yuqun Zhang, Bei Yu, Lingming Zhang, and Cong Liu.
Deepbillboard: Systematic physical-world testing of autonomous driving systems.
In *2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE)*, pages 347–358, 2020.