Universal Adversarial Perturbations on Visual Odometry Systems

Tal Peer
Technion - Israel Institute of Technology
`tal.peer@campus.technion.ac.il`

Based on the Article "Physical Passive Patch Adversarial Attacks on Visual Odometry"
`https://arxiv.org/pdf/2207.05729.pdf`

Abstract

As we seek to deploy machine learning systems not only in virtual domains but also in real systems, it becomes critical that we examine not only whether the systems don't simply work "most of the time" but which are truly robust and reliable. Deep neural networks are known to be susceptible to adversarial perturbations, small perturbations that alter the output of the network and exist under strict norm limitations. While such perturbations are usually tailored to a specific input, it is also possible to produce universal perturbations that achieve a high rate of misclassification on a set of inputs. Universal perturbations present a more realistic use case for adversarial perturbations, as the adversary does not require to be aware of the model's input. In this project, we focus on one of the most challenging cases—monocular VO—where the only input is a monocular video stream, with the task of estimating a robot's position and orientation from visual measurements.

1 Introduction

A VO model aims to infer the motion (position and orientation) between two respective viewpoints. Such models are frequently used by visual-based autonomous navigation systems. In this project, we produced universal adversarial perturbations for a given VO model, aiming to maximize its physical 3D deviation.

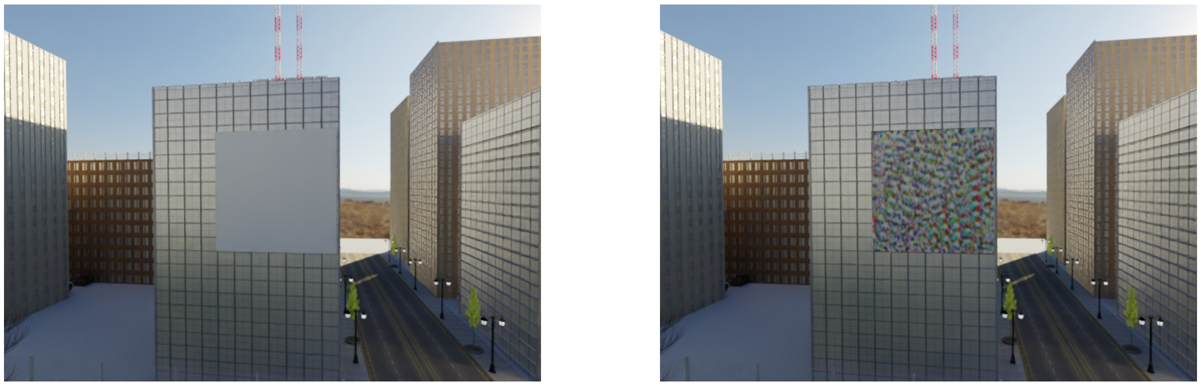
(a) Original Image (b) Perturbed Image

Figure 1: Original Image VS Perturbed Image

Papers Review:

1. **Reliable Evaluation of Adversarial Robustness with an Ensemble of Diverse Parameter-free Attacks**
   The paper discusses and proposes a number of extensions to the PGD-attack overcoming failures, such as sub-optimal step size and non-curved objective functions. Combining the proposed extensions, the authors propose the APGD method. The APGD method maintains the idea of the PGD attack optimizer, with step size halving in every few iterations called checkpoints. APGD is a parameter-free, computationally affordable, and user-independent ensemble of attacks to train or test adversarial robustness. The authors compared APGD to over 50 models from papers published at recent top machine learning and computer vision venues and achieved lower robust test accuracy in 98% of the reported results in the mentioned papers.
2. **Metrics for 3D Rotations: Comparison and Analysis**
   The paper presents a detailed analysis of six functions for measuring the distance between rotation matrices and demonstrates the importance of those distance functions to be invariant. Finally, the paper concludes that it is more efficient to use quaternions for 3D rotations. I chose the 4 criteria for the rotation loss.
3. **Guided Optical Flow Learning**
   The paper represents the most common use of the optical flow loss function, the End-Point-Error loss, also referred to as EPE. In many other papers, such as "RAFT: Recurrent All-Pairs Field Transforms for Optical Flow" and "EV-FlowNet: Self-Supervised Optical Flow Estimation for Event-based Cameras", the EPE loss is highly relevant for testing the discussed optical flow-based models. Hence, I chose to use the EPE loss for the optical flow loss.

2 Methods

<u>**Original Approach**</u>

- **Optimization Scheme**
  All data were used for training, without managing any partition for the evaluation set or test set.
- **Attack Optimizer**
  The pre-implemented PGD optimizer was used as an optimizer, with default hyper-parameters: Project

Step Size	0.05
Number of Epochs	100
Number of Multiple Restarts	8
Epsilon	1
Batch Size	1

  requirements were to keep other hyper-parameters in their default values such as epsilon, batch size, number of workers for data loaders, etc.
- **Loss Criterion for Training, Evaluation, and Testing**
  All of the mentioned phases were using the RMSE loss concerning the translation loss only.

<u>**New Approach**</u>

- **Optimization Scheme**
  The visual Odometry domain suffers from a lack of data. Hence, to achieve the best results on the given 50 trajectories, constructed from 5 different data-sets, I have split the data into 5 folds and iterated over them as follows: in each iteration, 4 data-sets were used for the training process, and the fifth one for evaluation. In this way, the data-set was used for the training phase 4 times in total and once for evaluation.
- **Attack Optimizer**
  To implement the APGD attack, I used the PGD framework, then added the required parameters and code implementation for the APGD. Neither the implementation nor the modification is based on the reviewed paper "Reliable Evaluation of Adversarial Robustness with an Ensemble of Diverse Parameter-free Attacks".
- **Loss Criterion for Training, Evaluation, and Testing**
  To take advantage of the rotation loss and optical flow loss, I have implemented the MSE loss for optical flow

loss and quadratic-product loss for the rotation loss. Since the pre-existing code suggested the conversion of the rotation defined by SO(3) group matrix to quaternions, I have found these objective loss functions to be the best fit for the optimization process, as they take the quaternion representation of the rotation matrix into advantage.

Algorithm 1 APGD

```
1: **Input:** $f$, $S$, $x^{(0)}$, $\eta$, $N_{\text{iter}}$, $W = \{w_0, \dots, w_n\}$
2: **Output:** $x_{\max}$, $f_{\max}$
3: $x^{(1)} \leftarrow P_{\mathcal{S}}\left(x^{(0)} + \eta \nabla f(x^{(0)})\right)$
4: $f_{\max} \leftarrow \max\{f(x^{(0)}), f(x^{(1)})\}$
5: $x_{\max} \leftarrow x^{(0)}$ **if** $f_{\max} \equiv f(x^{(0)})$ **else** $x_{\max} \leftarrow x^{(1)}$
6: **for** $k = 1$ **to** $N_{\text{iter}} - 1$ **do**
7:     $z^{(k+1)} \leftarrow P_{\mathcal{S}}\left(x^{(k)} + \eta \nabla f(x^{(k)})\right)$
8:     $x^{(k+1)} \leftarrow P_{\mathcal{S}}\Big(x^{(k)} + \alpha(z^{(k+1)} - x^{(k)})$
                   $+ (1-\alpha)(x^{(k)} - x^{(k-1)})\Big)$
9:     **if** $f(x^{(k+1)}) > f_{\max}$ **then**
10:        $x_{\max} \leftarrow x^{(k+1)}$ and $f_{\max} \leftarrow f(x^{(k+1)})$
11:    **end if**
12:    **if** $k \in W$ **then**
13:        **if** Condition 1 **or** Condition 2 **then**
14:            $\eta \leftarrow \eta/2$ and $x^{(k+1)} \leftarrow x_{\max}$
15:        **end if**
16:    **end if**
17: **end for**
```

Figure 2: Figure 1: APGD Pseudo Code.

3 Implementation and Experiments

3.1 Optimizer

As described above, my chosen optimizer is APGD. The implementation is in the "personalize attack" module. I used the pre-existing code of the PGD attack with the following modifications:

1. **Compute Checkpoint**
   The function creates the checkpoints as described before, using initial parameters for ρ-values: 0, 0.11. Note that the paper author recommends initializing the second ρ-value to 0.22, and my chosen value of 0.11 is due to the number of conducted iterations being 50, instead of 100 as the paper describes. For each checkpoint, we save the best objective function loss result and the adversarial perturbation which led to the high result loss.

2. **Update Step Size**
   The function sets the two mentioned conditions. In each checkpoint, we call the method, and the optimizer step size is halved if one of the two conditions is performed.

3. **New Gradient Ascent Step**
   The function implements the APGD optimization scheme by implementing rows 7-9 in the pseudo-code presented in Figure 1. We set $\alpha = 0.75$, which regulates the influence of the previous update on the current one, to keep a bias from the previous step. Note that in addition to the new perturbation, this function returns the accumulated loss of the training process. This is for optimization inner-checks such as early stopping.

3.2 Rotation and Optical Flow Objective Loss Function

In these experiments, I checked for different weights for the rotation and optical flow loss in the training phase to take advantage of those while optimizing the adversarial patch.

1. **Rotation Criterion**
   The chosen rotation criterion is the quat-product loss, as mentioned above.

$$\Phi_4 : S^3 \times S^3 \to \mathbb{R}^+,$$
$$\Phi_4(\mathbf{q}_1, \mathbf{q}_2) = 1 - |\mathbf{q}_1 \cdot \mathbf{q}_2|.$$

2. **Optical Flow Criterion**
   The chosen optical flow criterion is the EPE loss, which I implemented using the *torch.nn.MSE* method from the PyTorch library.

Experiments were conducted by running 50 iterations of each of the following weights:

Rotation Factor	Optical Flow Factor
0	0
0.5	0.5
0.5	1.0
1.0	0.5
1.0	1.0

Table 1: Tested Factors for Rotation and Translation

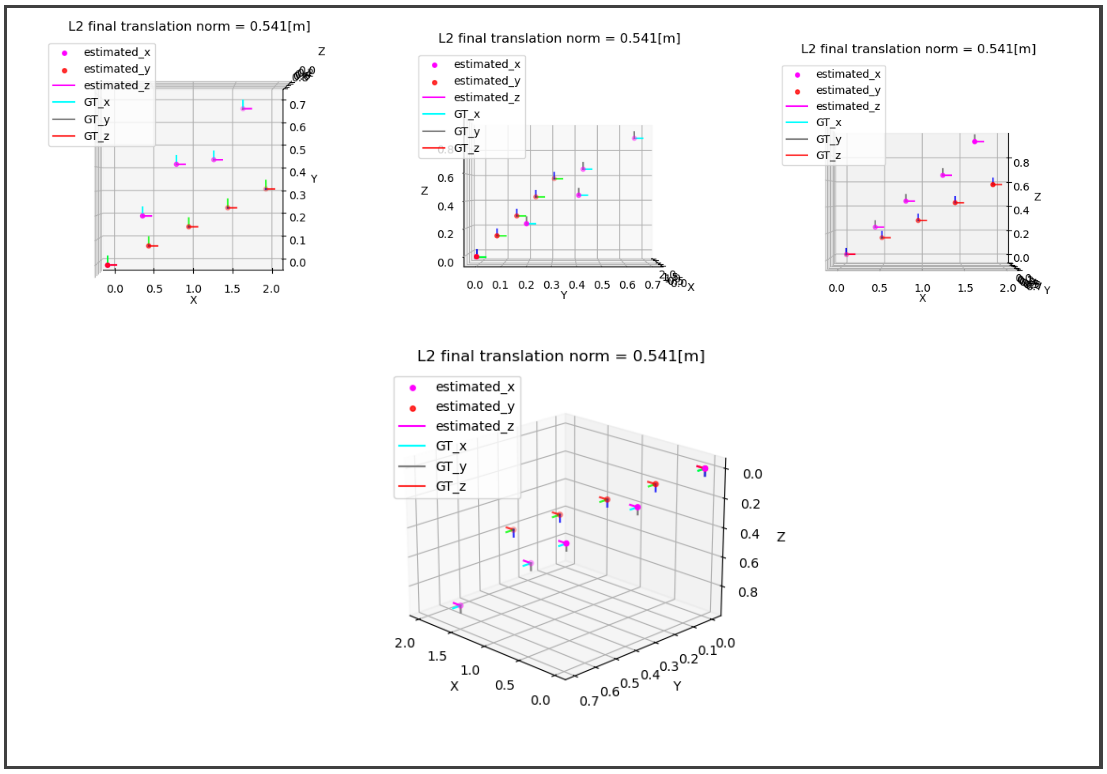

Figure 3: Translation Norm from Different Axis Views: XY, YZ, and XZ

3.3 Initializing and Updating Optimizer Step Size η

1. **Initialize** η

   In this experiment, I checked for different values to find the optimal η for the APGD optimizer. I ran 50 iterations of APGD with Momentum. In Figure 2, the evolution of the best evaluation loss is plotted for each η.

2. **Update** η

   With 50 iterations per optimizing phase, we identify 50 checkpoints at which the algorithm decides whether it is necessary to halve the current step size. We have two conditions:

   (a) **Condition 1**

   Counts in how many cases since the last checkpoint $w[j-1]$ the update step has been successful in increasing the objective function f. If this happened for at least a fraction ρ of the total update steps, then the step size is kept as the optimization is proceeding properly (I used $\rho = 0.75$).

   (b) **Condition 2**

   This holds true if the step size was not reduced at the last checkpoint and there has been no improvement in the best-found objective value since the last checkpoint. This prevents getting stuck in potential cycles.

$$1.\ \sum_{i=w_{j-1}}^{w_j-1} \mathbf{1}_{f(x^{(i+1)})>f(x^{(i)})} < \rho \cdot (w_j - w_{j-1}),$$

$$2.\ \eta^{(w_{j-1})} \equiv \eta^{(w_j)} \text{ and } f_{\max}^{(w_{j-1})} \equiv f_{\max}^{(w_j)},$$

4 Results and Discussion

To summarize, the chosen attack is APGD with a step size of 0.15, using a momentum factor of 0.75, early stopping of 20, and rotation and optical flow factors of 1.0. I use a train-evaluation split for 5 folds to adjust the initial step size since I find this hyper-parameter the most significant one.

Optimizer	Initial Step Size	Mean Loss of Adversarial RMS
APGD	0.05	0.419
APGD	0.1	3.604769
APGD	0.15	3.798201
APGD	0.2	3.157261
APGD	0.75	2.264978

As observed in Figure 4, the best start step size was 0.15. Note that Step Size 0.75 was set to ensure the step-size range.

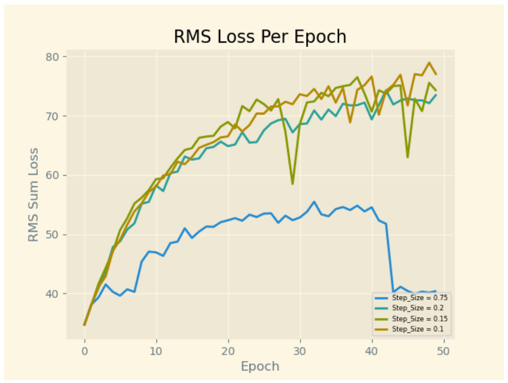

Figure 4: Initial Attack with different Step Sizes

5 Summary

This study investigates the generation of universal adversarial perturbations in Visual Odometry (VO) systems, focusing on the TartanVO model—a differentiable VO model that has achieved state-of-the-art performance in visual odometry benchmarks. The APGD attack optimizer is employed with a step size of 0.15 and a momentum factor of 0.75. By incorporating rotation and optical flow loss criteria, notable perturbations are generated, resulting in a significant increase in the physical 3D deviation of the VO model.

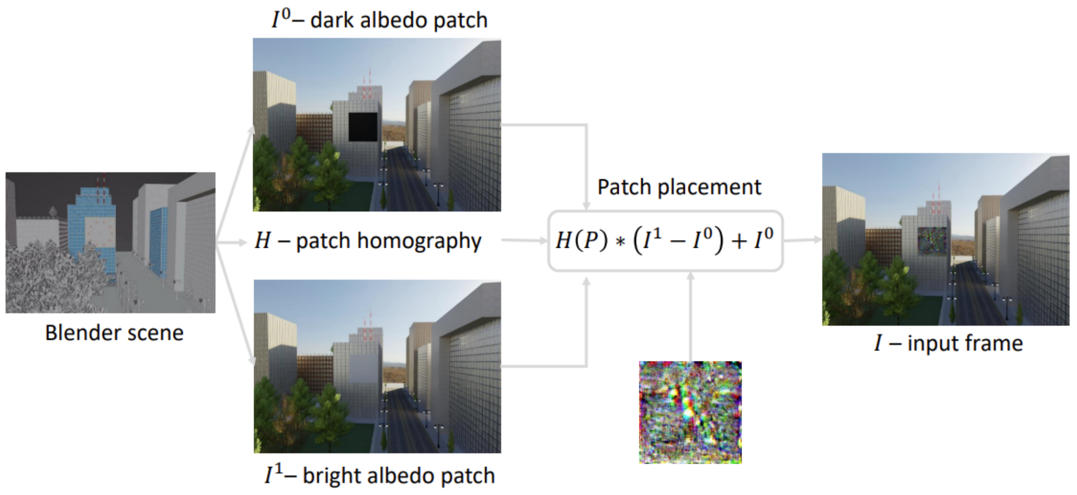

Figure 5: Process Summary

The following table presents the mean and standard deviation of the RMSE values for both the translation and rotation deviations of each approach:

Approach	Translation RMSE	Rotation RMSE
Original	0.852 (0.139)	0.013 (0.002)
New	0.873 (0.126)	0.015 (0.003)

Comprehensive experiments and evaluations are conducted to assess the impact of various hyperparameters, including the initial step size and the weights assigned to rotation and optical flow factors. The findings reveal that a rotation factor and optical flow factor of 1.0 yield the most favorable outcomes, showcasing the vulnerability of Visual Odometry systems to adversarial attacks. This underscores the critical need for robustness and security measures in such systems.

The experiments utilize the TartanVO model trained on scale-normalized trajectories from diverse synthetic datasets, enhancing generalizability to real-world scenarios. The model's robustness improves on scale-normalized trajectories, and the ground truth motions' scale is provided to the model to enhance performance. Plausible trajectory estimates are obtained over clean trajectories, both for synthetic and real data, further validating the model's effectiveness.

The evaluation of the VO model includes analysis of translation and rotation deviations, measured by Root Mean Square Error (RMSE). The results, summarized in the presented table, highlight the performance of the original and new approaches, providing insights into the model's accuracy and stability.

Overall, this project contributes to a modest yet significant understanding of passive patch adversarial attacks on vision-based navigation systems. The findings shed light on the potential risks and vulnerabilities associated with these attacks, driving the need for robust defense mechanisms and proactive strategies to ensure the integrity and reliability of autonomous systems in various operational environments.

Acknowledgments

This project is part of the final assignment for the course "Deep Learning on Computational Accelerators - 236781". Most of the code is provided by the course staff, and all rights are reserved.

Original Paper: "Physical Passive Patch Adversarial Attacks on Visual Odometry" - `https://arxiv.org/pdf/2207.05729.pdf`

The proposed method and the reported experiments is provided at my personal GitHub repository - `https://github.com/TalPeer-github/DeepLearningOnComputationalAccelerators.git`

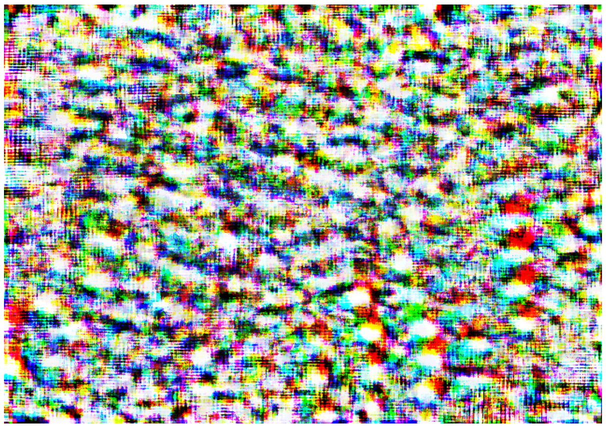
Figure 6: The Final Universal Perturbation Patch