2025 IEEE International Conference on Image Processing (ICIP) | 979-8-3315-2379-4/25/$31.00 ©2025 IEEE | DOI: 10.1109/ICIP55913.2025.11084759

TEXTURE- AND SHAPE-BASED ADVERSARIAL ATTACKS FOR OVERHEAD IMAGE VEHICLE DETECTION

Mikael Yeghiazaryan[1] *Sai Abhishek Siddhartha Namburu*[1] *Emily Kim*[1] *Stanislav Panev*[1]
Celso de Melo[2] *Fernando De la Torre*[1] *Jessica K. Hodgins*[1]

[1]Carnegie Mellon University, USA [2]DEVCOM Army Research Lab

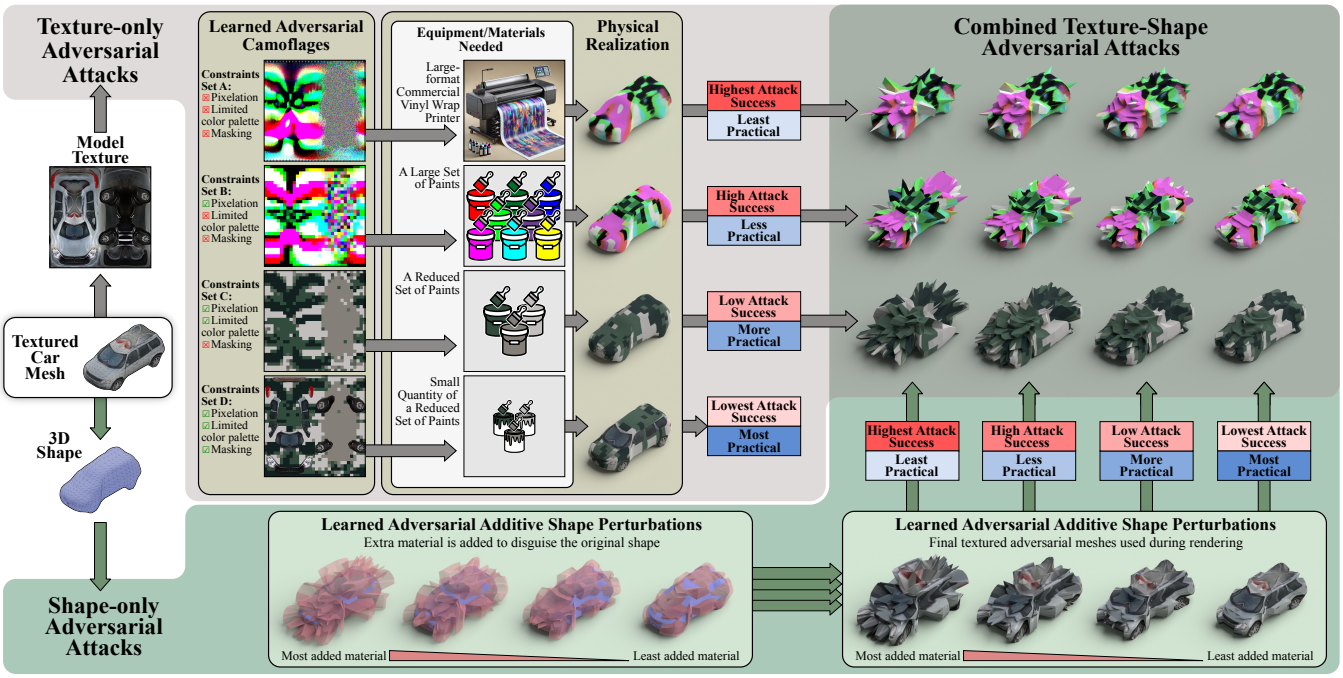

Fig. 1: We enforced various constraints on common adversarial attacks to improve their implementation in the real world. Our pipeline perturbs objects' texture, shape, or both. The latter demonstrates superior efficacy in deceiving object detectors.

ABSTRACT

Detecting vehicles in aerial images is difficult due to complex backgrounds, small object sizes, shadows, and occlusions. Although recent deep learning advancements have improved object detection, these models remain susceptible to adversarial attacks (AAs), challenging their reliability. Traditional AA strategies often ignore practical implementation constraints. Our work proposes realistic and practical constraints on texture (lowering resolution, limiting modified areas, and color ranges) and analyzes the impact of shape modifications on attack performance. We conducted extensive experiments with three object detector architectures, demonstrating the performance-practicality trade-off: more practical modifications tend to be less effective, and vice versa. We release both code and data to support reproducibility at https://github.com/humansensinglab/texture-shape-adversarial-attacks.

***Index Terms*—** adversarial attacks, remote sensing, object detection

1. INTRODUCTION

Robust object detection in aerial and satellite images is vital for automating critical tasks such as traffic management, urban planning, and disaster response. State-of-the-art detectors, such as YOLO [1] and RetinaNet [2], which are based on deep neural networks (DNN), have become foundational in this domain. However, recent studies such as Szegedy *et al.* [3] have revealed that DNNs can be susceptible to adversarial examples. Given the importance of these applications, understanding this vulnerability is crucial, especially in object detection in Remote Sensing Imagery (RSI). Furthermore, there are scenarios where utilizing AAs to impede vehicle detection by computer vision systems in overhead images could

offer strategic advantages, such as military camouflage.

Our primary objective is to investigate the resilience of object detectors against adversarial vehicles in RSI scenarios under realistic constraints. Traditional AA strategies often neglect the physical implementation constraints, focusing solely on task performance. For example, adversarial texture patterns typically resemble those depicted in Figure 1 (*Texture-only attacks - Constraints Set A*). However, it is essential to consider how such complex patterns can be produced practically in the physical world. Their creation often requires specialized equipment, such as expensive vinyl wrap printers, and trained professionals to install them on vehicle surfaces.

In this study, we propose a set of constraints to ensure attacks are feasible for implementation on vehicles, effectively camouflaging them. We define a *practical adversarial mesh* as a mesh modified in texture and shape such that replicating the modifications in real life requires minimal resources or specialized equipment. We consider practicality based on installation cost, difficulty, and operation. While constrained attacks are less effective than traditional AAs [4], they offer better practicality. Shape-only attacks are less effective than unconstrained texture attacks, but combining constrained texture with shape modifications improves performance, reaching levels similar to unconstrained texture attacks (Figure 1).

Our work contributes in several ways. (1) We introduce constrained AAs for shape and texture, designed to create practical 3D camouflages capable of deceiving object detectors in RSI. These constraints facilitate a more straightforward implementation compared to unconstrained camouflages. (2) We thoroughly examine how practicality and adversarial performance relate, finding that they have an inverse relationship. (3) We developed a tool for generating synthetic overhead images, contributing to the creation of synthetic datasets.

2. RELATED WORK

Adversarial attacks (AAs) have become central to computer vision research. Szegedy *et al.* [3] introduced AAs to expose vulnerabilities in deep learning models. Research has since focused on generating adversarial examples and divided AAs into digital and physical categories [5]. Digital AAs modify image pixels imperceptibly [6], while physical AAs manipulate objects in the physical world [7]. A hybrid approach, *simulated AAs*, tests perceptible attacks in simulated environments [8]. Our work aligns with simulated AAs, using synthetic data and realistic physics-based rendering for testing.

Recent studies also explore adversarial 3D geometry, mainly in autonomous driving using point clouds and LIDARs [9, 10]. Unlike those works, our focus is solely on RGB data in remote sensing imagery (RSI), where adversarial attacks are underexplored [11]. In RSI, the demand for robust automation has risen [12], driving research on AAs. Many attacks are impractical, so we focus on realistic adversarial camouflages with real-world constraints. One of the few studies addressing physical aerial adversarial attacks is [11], which uses adversarial patches to target vehicle detectors. Our approach differs in applying stricter constraints to adversarial camouflages and vehicle modifications, while applying these modifications to the entire vehicle and at a lower geo-spatial resolution. Additionally, instead of real-world tests, we evaluated our camouflages on highly realistic synthetic data produced by a physics-based renderer.

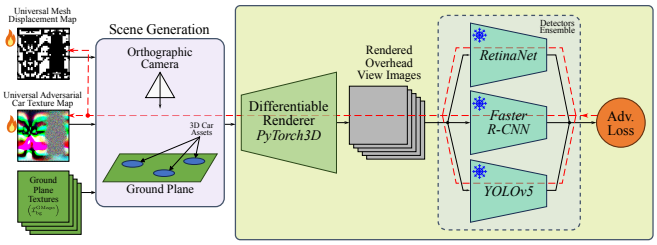

Fig. 2: Our pipeline for adversarial attacks on an ensemble of object detectors. The back-propagation path is illustrated by the red dashed line. During inference, we evaluate each model independently.

Fig. 3: Left: original image, right: corresponding adversarial image generated with Blender.

3. METHOD

3.1. PyTorch3D Data Generation

PyTorch3D (PT3D) data is generated using PyTorch3D [13] with 3D vehicle meshes from a GAN-based generator. We adapt and retrain the Textured 3D GAN (T3GAN) [14] to enable semantic segmentation map sampling, producing a set of meshes $\mathcal{M}$. Using GMaps backgrounds $\mathcal{I}_{\text{bg}}^{\text{GMaps}}$ and vehicle meshes $\mathcal{M}$, the differentiable renderer (DR) generates images $I = \text{DR}\left(I_{\text{bg}}^{\text{GMaps}}, M\right)$, where M includes texture T and shape S. Post-processing steps like blurring and anti-aliasing enhance realism. M can include either original or adversarial meshes, producing "original" or "adversarial" images, respectively. The pipeline supports synthetic dataset creation and ground-truth annotations, with adversarial attacks using optimized S_{adv} or T_{adv}. We use this data to train vehicle detectors and to optimize adversarial attacks.

3.2. Adversarial Optimization Pipeline

Each cycle of the attack uses PT3D to generate a batch of adversarial images $I_b = \{I_1, \ldots, I_{N_b}\}$ such that $I_k = \text{DR}\left(I_{\text{bg},k}^{\text{GMaps}}, M_k\right), \forall\ k \in [1, \ldots, N_b]$, where I_k is the k-th image from the batch, $I_{\text{bg},k}^{\text{GMaps}}$ is the k-th background image sampled from the GMaps dataset, M_k is a randomly selected mesh with shape and texture components S_k and T_k, and DR is a differentiable renderer following Section 3.1. Depending on the optimized entity, either the most recent S_{adv} or T_{adv} replaces the corresponding counterpart in M_k at the beginning of each iteration. During the attack, we ensure each image contains only one vehicle to avoid producing meshes that rely on multiple camouflages being in close proximity. The attack aims to create independently effective adversarial meshes.

Let F_i be the objective function used to train a detector model D_i. We supply I_b to D_i, producing predictions $y_{\text{pred}} = D_i(I_b)$. We then minimize a weighted loss function for an ensemble of models: $\mathcal{L} = \sum_i \lambda_i \mathbb{E}\left[F_i(D_i(I_b), y_{\text{gt}})\right], M_{\text{adv}}^{\star} = \arg\min_M \mathcal{L}(M)$, where y_{gt} are the ground-truth object locations, manually set to $\emptyset$ for adversarial attack training. We use coefficients λ_i such that the initial loss values F_i are in the same order of magnitude.

3.3. Texture-based Attacks

In texture-based attacks, we optimize a universal texture map applied to all meshes. While (**u**)nconstrained adversarial textures (abbreviated as "U") achieve excellent performance, they are impractical for real-world use. We introduce constraints to reflect practical implementation limitations: *Spatial Resolution*, *Spatial Restriction*, and *Color Restriction*.

Spatial Resolution. Applying iridescent patterns to irregular shapes such as vehicle surfaces is often challenging. We impose a texture (**pix**)elization (abbreviated as "Pix") constraint with block sizes of 16×16 px to ensure practical resolution, corresponding to approximately 15 cm on vehicle rooftops. We implement this by storing the adversarial texture as a $32 \times 32 \times 3$ tensor, then upscaling it to the original size of $512 \times 512 \times 3$ via nearest-neighbor interpolation.

Spatial Restriction. Another notable limitation is the need for vehicle camouflage to not hinder vehicle operation. We restrict the camouflage to specific areas of the vehicle using segmentation (**ma**)sks (abbreviated as "Ma"). Certain parts, such as windows, remain free from camouflage. The segmented adversarial texture map is given by $T_{\text{seg}} = T_{\text{or}} \cdot (1 - T_{\text{mask}}) + T_{\text{adv}} \cdot T_{\text{mask}}$, and T_{or} is the original texture.

Color Restriction. Our strictest constraint limits the number of colors in the adversarial texture map, implemented in two ways: 1) fixing the color count and (**l**)earning both the (**c**)olors and their placement ("Lc"), or 2) (**f**)ixing the (**c**)olors and optimizing their placement only ("Fc"). Unlike softer constraints such as the *non-printability score*, this enforces strict color limits, a concept largely unexplored in prior works. To enforce this constraint, the color of each pixel is determined during optimization by predicting a probability distribution $p(c)$ over a fixed set of colors. This distribution is sharpened using a double softmax $s(\cdot)$ to amplify the most probable color: $p_A(c) = s(s(p(c)))$. The pixel color is initially set to $\mathbb{E}[p_A(c)]$, approximating the mode color. After optimization, each pixel is assigned $\arg\max_{c_i} p(c_i)$, producing a camouflage that satisfies the color constraint. Additional details can be found in Section S6.1 in the Supplementary Material.

Table 1: Comparison of attack practicality with prior works. Texture practicality is the first score, shape practicality is the second. Sequential and parallel combined attacks yield identical final results. Full table is in Section S6 in the Supplementary Material.

	Camouflage	PC	DI	DO	Score	Notes
Other	Du *et al.* (ON) [11]	+0	+0	+0	3	Small AA area
	Du *et al.* (OFF) [11]	00	+0	−0	0	Limited mobility
	DTA [15]	−0	−0	+0	−1	Special equipment
Our	T-U	−0	−0	−0	−3	Special equipment
	T-Ma	−0	−0	+0	−1	Special equipment
	T-PixFcMa	+0	+0	+0	+3	Lim. color stickers
	S-O	0−	0−	0−	−3	Shape modification
	C-Fc	−−	−−	−−	−6	Spec. eq./shape mod.

3.4. Shape-based Attacks

We optimize a universal perturbation in shape-based attacks using a 2D displacement map from a common UV map. Deformations extend outward from the mesh center, preserving the original shape. We enforce *Symmetry* for balanced mass and *Perturbation Magnitude* (PM) to limit deformation.

3.5. Combined Attacks

We also conduct combined attacks where both texture and shape are optimized. These can be performed sequentially or in parallel. In *sequential combined attacks*, we first optimize the texture map and then perform a shape-based attack using the adversarial texture. This allows us to evaluate the performance of both attacks sequentially. In *parallel combined attacks*, we alternate between optimizing the texture and shape. Each entity is optimized for a fixed number of steps n_{pll}, switching between them until the loss converges.

3.6. Computational Requirements

Experiments ran on a machine with an Intel Xeon Gold 6252 CPU, 755 GB RAM, and an NVIDIA Quadro RTX 6000 GPU (24 GB). Each attack took 3 hours per epoch, using up to 5 GB RAM and 12 GB GPU memory.

Table 2: Figures show mean values from synthetic models on PT3D and Blender data. "T", "R", "S", and "C" represent texture, random texture, shape, and combined attacks. Lc and Fc are mutually exclusive. PM* and Pr* denote optimal perturbation magnitude and practicality for shape attacks. See Table S3 in the Supplementary Material for the full table.

Attack	Constraints				PM*	Pr*	PT3D EASR	Blender EASR
	Pix	Lc	Fc	Ma				
T-U					—	—	95.77%	70.02%
T-Ma				✓	—	—	75.76%	44.43%
T-Pix	✓				—	—	94.75%	63.83%
T-PixLcMa	✓	✓		✓	—	—	68.39%	42.15%
T-PixFcMa	✓		✓	✓	—	—	12.70%	44.64%
R-PixFc	✓		✓		—	—	3.16%	20.24%
S-O	—	—	—	—	0.4	0.6	89.82%	78.86%
C-Fc (seq.)			✓		0.2	0.8	86.80%	70.37%
C-Fc (par.)			✓		0.2	0.8	87.11%	68.07%
C-PixFc (seq.)	✓		✓		0.2	0.8	86.83%	75.76%
C-PixFc (par.)	✓		✓		0.2	0.8	89.34%	77.86%

4. PRACTICALITY AND COMPARISONS

Our focus is not on enhancing AAs performance but exploring the impact of realistic constraints. Given the high effectiveness of unconstrained AAs, there is limited room for improvement. We evaluate our work based on three qualitative criteria: *production cost* (PC), *difficulty of installation* (DI), and *difficulty of operation* (DO), rated as good (+), insignificant (0), or bad (−). Practical camouflages score higher. See definitions in Section S6 in the Supplementary Material.

As shown in Table 1, T-PixFcMa is the most practical camouflage, despite lower effectiveness (Table 2). Du *et al.* [11] (ON) and EVD4UAV [16] achieve similar practicality scores, but their patches are too small for effective use in aerial imagery at our resolution. More details on score assignment are in Section S6 in the Supplementary Material. Results from Tables 1 to 2 highlight the trade-off between practicality and performance. While some studies excel in AA performance, they lack practicality. We conclude that optimizing for performance reduces practicality. For example, randomly generated camouflages (Table 2) show performance reduction that may not be justified without optimization. While a more rigorous analysis incorporating real data could be done (e.g., user studies for DO), this is beyond the scope due to limited resources.

5. EXPERIMENTS AND RESULTS

While Section 4 qualitatively compares practicality — considering production cost, installation, and operational complexity — this section provides a quantitative evaluation of effectiveness under different adversarial attack scenarios.

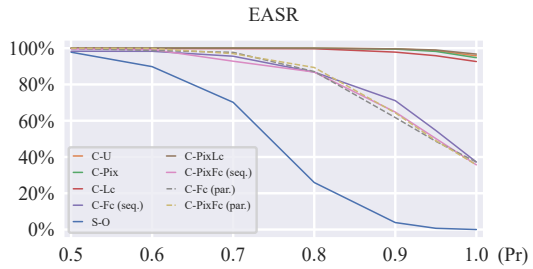

Fig. 4: EASR vs Practicality (Pr) curves for the shape-based and combined attacks on PT3D.

5.1. Evaluation metrics

We use effective attack success rate (EASR) to evaluate attack performance, defined as EASR $= \frac{|V_{d,m}|-|V_{m,d}|}{|V_{d,d}\cup V_{d,m}|}$, where the first subscript in $V_{i,j}$ indicates whether a vehicle was detected (d) or missed (m) in the original dataset, and the second subscript indicates whether it was detected (d) or missed (m) in the adversarial dataset after applying adversarial modifications. EASR is more conservative than the traditional ASR. More details can be found in Section S5.1 in the Supplementary Material.

5.2. Test Data

We describe below the datasets used to test the trained models and the camouflages generated from the attacks.

LINZ Dataset. We used the labeled LINZ dataset using LINZ Data Service aerial orthoimages[1]. The dataset was sampled into 384x384px images, resulting in 172 595 images with a 12.5 cm/px resolution. We utilized only the "Small Vehicles" class for experiments, removing other labels but preserving all images. This dataset is denoted as $\mathcal{I}^{\text{LINZ}}$. A second dataset, $\mathcal{I}^{\text{LINZ}}_{\text{bg}}$, was created by removing vehicles from $\mathcal{I}^{\text{LINZ}}$ using "Inpaint Anything" [17].

GMaps Dataset The GMaps dataset provides background images $\mathcal{I}^{\text{GMaps}}_{\text{bg}}$ for the PyTorch3D [13] data generation pipeline (Section 3.1). We extracted Google Maps (GMaps) satellite images $\mathcal{I}^{\text{GMaps}}$ with matching coordinates to the LINZ aerial dataset using QGIS[2], ensuring a direct LINZ-GMaps correspondence. Real vehicles were manually removed from these images using an image editor, resulting in the background set $\mathcal{I}^{\text{GMaps}}_{\text{bg}}$, later used in the synthetic data generation process.

Blender Data Generation Blender data tests adversarial meshes in realistic settings using the Cycles physics-based renderer [18]. Unlike PT3D, Blender generates highly realistic but non-differentiable images, excluding it from adversarial optimization. Synthetic overhead images are produced by

[1] https://data.linz.govt.nz/layer/51926-selwyn-0125m-urban-aerial-photos-2012-2013/

[2] https://www.qgis.org

Blender using LINZ backgrounds $\mathcal{I}_{\text{bg}}^{\text{LINZ}}$ and meshes M, incorporating either original or adversarial textures and shapes.

5.3. Detection Models

We used RetinaNet [2], Faster R-CNN [19], and YOLOv5 [1] for vehicle center detection in RSI, representing one-stage, two-stage, and YOLO-family detectors. Each was trained on real and synthetic data, labeled "real" and "synt" models. On real test data, synthetic models achieved 50–63% AP, while real models exceeded 80%. More details can be found in Section S1 in the Supplementary Material. We attacked synthetic model ensembles, using one for inference.

5.4. Texture-based Attacks

We modify a vehicle's texture in *texture-based attacks* to conceal it from detectors, starting with random adversarial textures. There are twelve distinct texture settings, each evaluated on an ensemble of three synthetic models (RetinaNet, Faster R-CNN, YOLOv5). We compare these adversarial textures with four random texture maps (R-*). Results on PT3D test data are shown in Table 2. In Fc experiments, five colors are determined via K-means clustering of background pixels, while Lc experiments involve model-learned color placement.

To account for the distribution gap between real and synthetic datasets, we repeat the experiments using Blender. The results, presented in Table 2, show that constraints reduce performance but increase practicality. Example images are in Figure 3, with additional examples in the Supplementary Material. The performance-practicality trade-off remains even when testing on a different domain, like Blender. We also observe that unrestrained color distribution results in saturated colors, a common but underexplored issue in prior works. Further details are in Section S1 the Supplementary Material.

5.5. Shape-based Attacks

In *shape-based attacks*, we alter the geometry of the vehicle sacrificing practicality for improved adversarial performance. We link practicality, denoted as Pr, to the perturbation magnitude PM as $\text{Pr} = 1 - \text{PM}$, where $\text{PM} \in [0, 1]$. $\text{Pr} = 0$ indicates extreme mesh perturbation and $\text{Pr} = 1$ indicates no perturbation for a more realistic scenario.

Our goal is to minimize PM (maximize Pr) in shape-based attacks while maximizing EASR performance. We assess multiple attacks on synthetic models across a range of Pr values. Refer to the curve in Figure 4 under "Original" for details (utilizing original vehicle textures). Given an EASR-vs-Pr curve, we compute a practicality metric P1 as the harmonic mean of the EASR and Pr values for each point on the curve, *i.e.* $\text{P1} = 2\frac{\text{EASR}\cdot\text{Pr}}{\text{EASR}+\text{Pr}}$. We then pick the attack with the highest P1 as the optimal one. See the results, denoted as S-O in Table 2. The results suggest that when no adversarial texture is utilized along with a deformed vehicle geometry, the deformation must be large to achieve good performance, which is expensive to produce and difficult to install and operate, rendering it impractical.

5.6. Combined Attacks

In the *combined attacks*, we aim to boost the adversarial performance by attacking both mesh entitites: texture and shape. We discard the adversarial textures that use masking because a modified mesh is hard to segment into semantically meaningful parts. Thus, this leaves us with six adversarial texture maps for mesh-based attacks, each with its texture setting.

Sequential Attacks. We conduct six sequential attacks, each using one of the six adversarial textures from a non-masked setup. We follow the methodology in Section 5.5 to determine the optimal PM. The results, labeled C-* in Table 2, reveal significant insights. While the improvement over texture attacks without the fixed colors constraint (T-U, T-Pix, T-Lc, T-PixLc) is unjustified due to practicality loss, the fixed colors constraint (T-Fc and T-PixFc) justifies sacrificing some practicality for better performance. Compared to the huge practicality loss in shape-based attacks, the sequential blend of adversarial texture and shape is more efficient than shape-only attacks. We evaluate the resulting adversarial meshes on Blender data where the $\text{PM}^\star \neq 0$.

Parallel Attacks. We conduct parallel attacks using only two adversarial texture maps, Fc and PixFc, to reduce the PM even further than the sequential attacks. The results in Table 2 suggest no significant gain in switching to parallel attacks.

6. CONCLUSION

This study outlines a methodology for developing effective camouflage strategies to conceal vehicles in RSI. We also study the performance-practicality trade-off when implementing adversarial camouflages. While our findings could be misused, it is vital for the research community to be aware of the vulnerabilities in current models that we highlight. We show an inverse relationship between practicality and performance: unconstrained adversarial textures are highly effective against vehicle detection systems, while practical constrained textures are easier to implement but less effective. Shape-only attacks are also less impactful than texture attacks, but combining both can achieve results similar to unconstrained textures. Notably, sequential and parallel executions of shape and texture attacks demonstrate similar adversarial performance. Additionally, we present two pipelines for generating synthetic aerial images: using a differentiable renderer and a physics-based renderer.

Acknowledgements

We thank Brent Lance for valuable feedback and support. This work was supported in part by the Army Research Lab.

7. REFERENCES

[1] Glenn Jocher, "YOLOv5 by Ultralytics," May 2020. 1, 5

[2] Lin, Tsung-Yi and Goyal, Priya and Girshick, Ross and He, Kaiming and Dollar, Piotr, "Focal Loss for Dense Object Detection," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 1, 5

[3] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus, "Intriguing properties of neural networks," *arXiv preprint arXiv:1312.6199*, 2013. 1, 2

[4] Suryanto, Naufal and Kim, Yongsu and Larasati, Harashta Tatimma and Kang, Hyoeun and Le, Thi-Thu-Huong and Hong, Yoonyoung and Yang, Hunmin and Oh, Se-Yoon and Kim, Howon, "ACTIVE: Towards Highly Transferable 3D Physical Camouflage for Universal and Robust Vehicle Evasion," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2023, pp. 4305–4314. 2

[5] Syed M. Kazam Abbas Kazmi, Nayyer Aafaq, Mansoor Ahmad Khan, Ammar Saleem, and Zahid Ali, "Adversarial Attacks on Aerial Imagery : The State-of-the-Art and Perspective," in *2023 3rd International Conference on Artificial Intelligence (ICAI)*, 2023, pp. 95–102. 2

[6] Andrew Du, Yee Wei Law, Michele Sasdelli, Bo Chen, Ken Clarke, Michael Brown, and Tat-Jun Chin, "Adversarial Attacks against a Satellite-borne Multispectral Cloud Detector," in *2022 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, 2022, pp. 1–8. 2

[7] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song, "Robust Physical-World Attacks on Deep Learning Visual Classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2

[8] Alexey Kurakin, Ian Goodfellow, Samy Bengio, et al., "Adversarial Examples in the Physical World," 2016. 2

[9] Huangxinxin Xu, Fazhi He, Linkun Fan, and Junwei Bai, "D3AdvM: A direct 3D adversarial sample attack inside mesh data," *Computer Aided Geometric Design*, vol. 97, pp. 102122, 2022. 2

[10] Kibok Lee, Zhuoyuan Chen, Xinchen Yan, Raquel Urtasun, and Ersin Yumer, "ShapeAdv: Generating Shape-Aware Adversarial 3D Point Clouds," *arXiv preprint arXiv:2005.11626*, 2020. 2

[11] Andrew Du, Bo Chen, Tat-Jun Chin, Yee Wei Law, Michele Sasdelli, Ramesh Rajasegaran, and Dillon Campbell, "Physical Adversarial Attacks on an Aerial Imagery Object Detector," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, January 2022, pp. 1796–1806. 2, 3, 4

[12] Xingkui Zhu, Shuchang Lyu, Xu Wang, and Qi Zhao, "TPH-YOLOv5: Improved YOLOv5 Based on Transformer Prediction Head for Object Detection on Drone-captured Scenarios," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 2778–2788. 2

[13] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari, "Accelerating 3D Deep Learning with PyTorch3D," *arXiv:2007.08501*, 2020. 2, 4

[14] Dario Pavllo, Jonas Kohler, Thomas Hofmann, and Aurelien Lucchi, "Learning Generative Models of Textured 3D Meshes From Real-World Images," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021, pp. 13879–13889. 2

[15] Suryanto, Naufal and Kim, Yongsu and Kang, Hyoeun and Larasati, Harashta Tatimma and Yun, Youngyeo and Le, Thi-Thu-Huong and Yang, Hunmin and Oh, Se-Yoon and Kim, Howon, "DTA: Physical Camouflage Attacks Using Differentiable Transformation Network," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022, pp. 15305–15314. 3

[16] Huiming Sun, Jiacheng Guo, Zibo Meng, Tianyun Zhang, Jianwu Fang, Yuewei Lin, and Hongkai Yu, "Evd4uav: An altitude-sensitive benchmark to evade vehicle detection in uav," *arXiv preprint arXiv:2403.05422*, 2024. 4

[17] Tao Yu, Runseng Feng, Ruoyu Feng, Jinming Liu, Xin Jin, Wenjun Zeng, and Zhibo Chen, "Inpaint Anything: Segment Anything Meets Image Inpainting," *arXiv preprint arXiv:2304.06790*, 2023. 4

[18] "Blender Cycles," https://docs.blender.org/manual/en/latest/render/cycles/index.html, Accessed: 2024-03-06. 4

[19] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in *Advances in Neural Information Processing Systems*, C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, Eds. 2015, vol. 28, Curran Associates, Inc. 5