TPAM: Transferable Perceptual-constrained Adversarial Meshes

Tengjia Kang[1], Yuezun Li[1], Jiaran Zhou[1,†], Shiqing Xin[2], Junyu Dong[1], Changhe Tu[2]

[1]: College of Computer Science and Technology, Ocean University of China, Qingdao, China
[2]: School of Computer Science and Technology, Shandong University, Qingdao, China

Abstract

Triangle meshes are widely used in 3D data representation due to their efficacy in capturing complex surfaces. Mesh classification, crucial in various applications, has typically been tackled by Deep Neural Networks (DNNs) with advancements in deep learning. However, these mesh networks have been proven vulnerable to adversarial attacks, where slight distortions to meshes can cause large prediction errors, posing significant security risks. Although several mesh attack methods have been proposed recently, two key aspects of Stealthiness and Transferability remain underexplored. This paper introduces a new method called Transferable Perceptual-constrained Adversarial Meshes (TPAM) to investigate these aspects in adversarial attacks further. Specifically, we present a Perceptual-constrained objective term to restrict the distortions and introduce an Adaptive Geometry-aware Attack Optimization strategy to adjust attacking strength iteratively based on local geometric frequencies, striking a good balance between stealthiness and attacking accuracy. Moreover, we propose a Bayesian Surrogate Network to enhance transferability and introduce a new metric, the Area Under Accuracy (AUACC), for comprehensive performance evaluation. Experiments on various mesh classifiers demonstrate the effectiveness of our method in both white-box and black-box settings, enhancing the attack stealthiness and transferability across multiple networks. Our research can enhance the understanding of DNNs, thus improving the robustness of mesh classifiers. The code is available at https://github.com/Tengjia-Kang/TPAM.

CCS Concepts
• ***Computing methodologies*** → ***Mesh geometry models; Shape analysis;***

1. Introduction

Triangle meshes are the most predominant and widely adopted form of 3D data representation in computer graphics. Compared to other 3D representations such as voxels and point clouds, meshes carry more geometric and topological information, performing expressiveness and efficiency in representing non-uniform and irregular surfaces. Among mesh processing tasks, mesh classification is fundamental, underpinning applications in autonomous driving [CMW*17; ZT18], virtual reality [KCMB18], shape modeling [SZTL19; PKGF21], and the medical field reality [Ric08]. With the prominent advancements in deep learning, mesh classification is now typically handled by Deep Neural Networks (DNNs) [FFY*19; HHF*19; LT20; SS21; KC22; SKB24]. Although DNNs have shown promising results, they are vulnerable to adversarial attacks, where slight distortions to input meshes can disrupt classification predictions, posing a significant security risk in mesh-related applications. Exploring these vulnerabilities can enhance our understanding of DNNs and improve the robustness of mesh classifiers.

Adversarial attacks, originating from 2D image classification [SZS*13], have been extensively studied in recent years [GSS14; PMJ*16; CW17]. Building upon these studies, efforts have expanded into the field of 3D geometry [LYS19; XQL19; ZGH*24; MCBR20; RPC*21; BYBT22; XHFB22; ZCL*23; SLA23], with particular attention to adversarial attacks for mesh networks [MCBR20; RPC*21; BYBT22; XHFB22; ZCL*23]. Typically, these methods design objectives to mislead predictions and optimize these objectives iteratively (see Sec. 3). Although these methods succeed in disrupting predictions, their real-world feasibility remains underexplored, particularly in two aspects:

- **Stealthiness:** Ideally, the distortion on meshes should be imperceptible to human observers to avoid easy detection. To constrain the distortions, existing methods (*e.g.*, [XHFB22; BYBT22]) usually set an upper bound for distortion and iteratively optimize objectives. However, this upper bound only ensures the maximum distortion and is inadequate for reflecting stealthiness, as distortions in different mesh regions impact stealthiness differently. For instance, adding perturbations

† Corresponding author.

DOI: 10.2312/pg.20241285

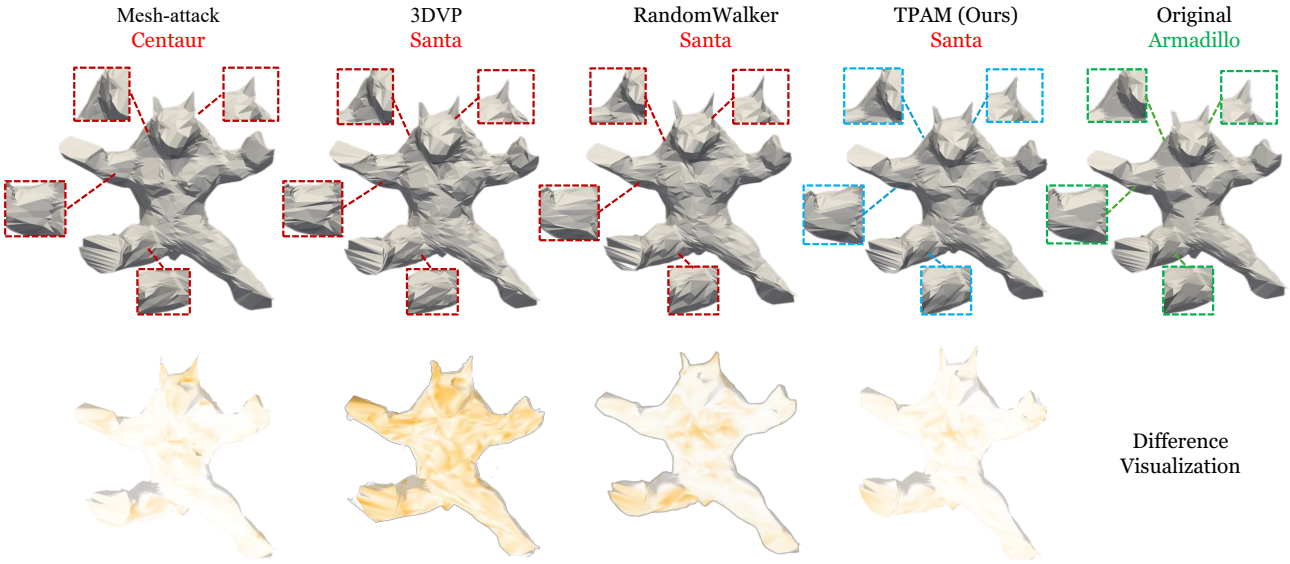

Figure 1: *Attack results of Armadillo. The adversarial mesh of ours is most similar to the original mesh. Several representative regions are highlighted by zoomed-in views in the first row. The second row visualizes the differences between the adversarial mesh of each method and the original mesh.*

in regions with high geometric frequencies (fine details, *e.g.*, hair) is less noticeable than in regions with low geometric frequencies (*e.g.*, flat board). Moreover, the optimization directions in these methods are independent of geometry, meaning the attack strength is determined solely by attacking objectives without considering any local geometric details. Other methods (*e.g.*, [MCBR20; ZCL*23]) add extra objective terms to restrict distortion, but these terms are not explicitly related to attack strength during optimization, hindering their effectiveness in maintaining stealthiness.

- **Transferability:** Exiting methods have only superficially studied transferability – the ability to disrupt unknown mesh networks using a surrogate (known) network. Most works (*e.g.*, [MCBR20; XHFB22]) focus on white-box attacks, assuming that the details (*i.e.*, the architecture and parameters) of the target network are accessible to the attackers. This assumption is impractical as the details of target networks are unlikely to be available in real-world scenarios. The study of [BYBT22] notices the importance of transferability and attempts to attack target networks using adversarial meshes generated by a surrogate network. However, this method can only transfer adversarial meshes to a single target network, limiting its applicability in real-world scenarios.

To address these limitations, we describe a new method to generate Transferable Perceptual-constrained Adversarial Meshes (TPAM). Firstly, we describe a perceptual-constrained objective term designed to properly assess the stealthiness of adversarial meshes. This term includes three soft constraints that consider the distortion and direction of each triangle, as well as the local context of the mesh. Secondly, we propose an Adaptive Geometry-aware Attack Optimization strategy that dynamically adjusts the attack strength (step size) in conjunction with optimizing the overall objectives. Specifically, we assume that the attack strength should vary across the entire mesh due to the varying geometric complexity. Thus, we propose a new Gaussian Mixture Model (GMM)-based strategy, which can partition the mesh into different regions based on local geometric complexity differences. The variance of each Gaussian component can represent the complexity and guide the adjustment of attack strength, allowing more changes in complex regions than in smooth regions. In addition, we introduce a Bayesian Surrogate Network to enhance uncertainty. Unlike conventional Deep Neural Networks (DNNs), this network samples its parameters from a learned distribution, aiming to increase the feature space overlap with target networks and improve transferability across multiple target networks. Fig. 1 shows our attack result is better than others in terms of stealthiness.

Existing methods commonly use accuracy (ACC) as a metric, representing the percentage of test samples correctly classified. However, the conditions for calculating ACC vary among different methods. Several works do not set distortion bound [MCBR20; BYBT22], which, despite achieving high attacking performance, do not consider stealthiness. Other works [XHFB22; ZCL*23] do set a bound, but the standards for these bounds differ, making comparisons unfair. Therefore, we define a new evaluation metric, the Area Under Accuracy (AUACC), to comprehensively assess performance in terms of both attacking accuracy and stealthiness.

Extensive experiments are conducted on several recent DNN-based mesh classifiers using SHREC11 [VvJD*11] and ModelNet40 [WSK*15] datasets, comparing our method to many recent methods under both white-box and black-box attack settings. The

results demonstrate the efficacy of our method in attacking mesh classifiers.

Our contributions are four-fold:

1. We propose a new method called TPAM to generate adversarial meshes, which focuses on stealthiness and transferability that previous methods have not adequately addressed in adversarial attacks.
2. We describe a Perceptual-constrained objective term that properly assesses stealthiness by considering geometry features, including distortion and direction of each triangle, as well as local smoothness of the mesh.
3. Unlike existing methods with fixed attack steps, we propose an Adaptive Geometry-aware Attack Optimization strategy that adjusts attack steps while optimizing objectives.
4. We present a new evaluation metric (AUACC) to comprehensively evaluate attacking performance.

2. Related work

Mesh Classification. Mesh classification is a fundamental task in mesh processing, serving as a basis for many 3D applications, such as autonomous driving, virtual reality and rendering. In recent years, Deep Neural Networks (DNNs) have become the mainstream approach for mesh classification [HHF*19; FFY*19; MLR*20; LT20; KC22; SKB24]. MeshCNN [HHF*19] designs convolution and pooling operations specifically applied to the edges of meshes, in order to directly migrate CNN from image field to the field of meshes based on edge-unit feature. MeshNet [FFY*19] extracts face-unit spatial features and face-unit structural features (center, corner, normal) with a unified structure form, and feeds them into the designed combination and aggregation modules for mesh classification. PD-MeshNet [MLR*20] proposes a primal-dual graph framework to take features of edges and faces and then perform convolutions on these graphs using a dynamic attention-based network. MeshWaker [LT20] generates random walks consisting of vertices by walking along edges to extract shape features and then feeding it into a recurrent neural network to learn walking history. ExMeshCNN [KC22] introduces a descriptor to extract edge-based geodesic features and face-based geometric features and use Layer-wise Relevance Propagation (LRP) and Gradient-weighted Class Activation Mapping (Grad-CAM) to interpret its interpretability. RIMeshGNN [SKB24] leverages graph neural networks to analyze mesh-structured data, constructing a rotation-invariant network that maintains accuracy for meshes rotated in any direction.

3D Adversarial Attacks. Adversarial attacks are first studied in 2D image domain [SZS*13; GSS14; MMS*17; MFFF17] and have recently been extended to 3D geometry tasks, such as point clouds classification [LYS19; XQL19; ZGH*24], mesh classification [MCBR20; RPC*21; BYBT22; XHFB22; ZCL*23] and others [SLA23]. Note that both point clouds and meshes are 3D geometry representations. Point clouds and meshes differ significantly: point clouds are collections of discrete points without inherent connectivity, whereas meshes consist of vertices, edges, and faces with a defined topology. Consequently, methods for attacking point clouds cannot be directly adapted to mesh classification.

Several efforts have been made to attack meshes. Band-limited Perturbations (BLP) [MCBR20] proposes to perturb meshes in the spectral domain, with requirements that perturbations can occur only in the low-frequency region, inducing less noticeable perturbation. Spectral-attack [RPC*21] proposes a spectral attack across point clouds and meshes, by adding small perturbations to short eigenvalue sequences. 3DVP [XHFB22] introduces a simple vertex-based perturbation in Euclidean space, searching the optimized vertice positions based on opposite gradient fitness under a distortion bound. Mesh-attack [ZCL*23] proposes several constraints to penalize the deformation of meshes. However, this holistic optimization method lacks precise control of disturbances. Note that these above methods are all white-box attacks. Random Walker [BYBT22] explores the transferability in a black-box setting, which uses MeshWalker [LT20] as a surrogate network to simulate the features learned by a target network, and attacks the surrogate network to obtain transferable adversarial meshes. Nevertheless, this method does not develop devoted strategies for enhancing transferability and only validates it on a single target network at a time.

3. Preliminaries

Mesh Classification. Consider a triangle mesh $X = (V, F)$, where V is the set of vertices and F is the set of faces. Denote a mesh classifier as $\mathcal{H}_\theta : \mathcal{X} \to \mathcal{Y}$, where θ represents the weight parameters of this mesh classifier, $\mathcal{X}$ is the input mesh space and $\mathcal{Y}$ is the class label space. Given a mesh $X \in \mathcal{X}$, its predicted label is $y' = \mathcal{H}_\theta(X) \in \mathcal{Y}$. This prediction result is considered correct if $y' = y$, where y is the ground truth label of the mesh X.

Adversarial Meshes. The attackers aim to disrupt the prediction of mesh classifier $\mathcal{H}_\theta$ by adding slight perturbations δ on input mesh X. These perturbations δ are added on vertices in V, which in turn disturbs the faces in F as well. Let $X^* = X + \delta$ be the adversarial mesh corresponding to input mesh X. The goal of the attack is to ensure that $\mathcal{H}_\theta(X^*) \neq y$ while keeping the magnitude of the perturbations δ within a certain limit. This goal is typically formulated as a constrained optimization problem:

$$\begin{aligned} &\arg\min_{\delta} \ \mathcal{L}_{\text{atk}}(X^*, y), \\ &\text{s.t. } dis(X, X^*) \leq \epsilon, \end{aligned} \tag{1}$$

where $\mathcal{L}_{\text{atk}}$ is the attack objective, often defined as the negative of the classification objective (*e.g.*, Cross-entropy [ZCL*23]), which measures the error between the predicted label and ground truth label, and dis is a distance metric that quantifies the magnitude of perturbation (*e.g.*, ℓ_2 norm [XHFB22; ZCL*23]). The definition of $\mathcal{L}_{\text{atk}}$ can vary depending on the specific methods used [BYBT22].

The optimization process is typically iterative. Denote t as the iteration number. The adversarial mesh is generated as follows:

$$\begin{aligned} &X_0^* = X, \\ &X_{t+1}^* = X_t^* - w \cdot \nabla_{(X_t^*)} \mathcal{L}_{\text{atk}}, \end{aligned} \tag{2}$$

where w is fixed, a predefined parameter controlling the attack strength (step size), and ∇ denotes the gradients of objective $\mathcal{L}_{\text{atk}}$ with respect to the mesh X_t^*.

Attack Settings. Adversarial attacks can be categorized into two types: white-box attacks and black-box attacks. Currently, most

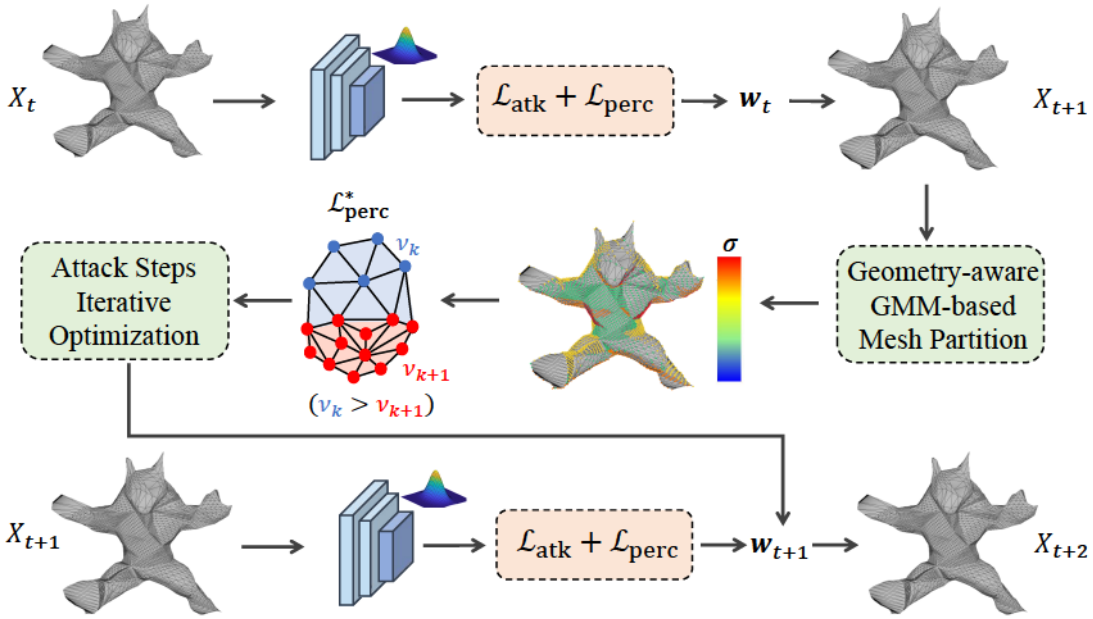

Figure 2: *Overview of TPAM. We propose a Perceptual-constrained objective $\mathcal{L}_{perc}$ and an Adaptive Geometry-aware Attack Optimization strategy to adjust attack strength $\boldsymbol{w}$ based on geometric complexity, measured by a newly proposed GMM-based strategy. We also describe a Bayesian surrogate network to further improve transferability. See text for details.*

methods are under the white-box setting, where attackers have full access to the network details, allowing them to craft specialized attack strategies. However, this setting is impractical in real-world scenarios as the details of the network are usually confidential. In black-box attacks, attackers do not have direct access to the internal structure and parameters of the target network. Consequently, a common strategy is to use adversarial meshes generated from a surrogate network (a known network that can be white-box attacked) to perform transfer attacks on the target network. The effectiveness of these attacks largely depends on the similarity between the feature spaces of the target and surrogate networks. Existing methods exhibit weak transferability as they have not fully investigated devoted strategies to enhance it. As a result, they tend to overfit the surrogate network, limiting their effectiveness when applied to the target network.

4. Method

We describe a new method, Transferable Perceptual-constrained Adversarial Meshes (TPAM), designed to enhance both the stealthiness and transferability of attacks on mesh classifiers. This method includes three major improvements. First, we propose a Perceptual-constrained Objective term to properly measure and control the stealthiness of the attack (Sec.4.1). Then we describe an Adaptive Geometry-aware Attack Optimization strategy to adjust the attack strength based on geometric complexity (Sec.4.2). Moreover, we introduce a Bayesian Surrogate Network to further enhance the Transferability (Sec.4.3). The overview of our method is shown in Fig. 2.

4.1. Perceptual-constrained Objective

Assessing the stealthiness of meshes is non-trivial as they contain much more complex geometry features than other 3D representations such as point clouds. While mesh decimation also considers preserving shapes, it typically reduces the number of polygons or vertices, making its constraints unsuitable for our purposes. Therefore, we propose a perceptual-constrained objective term dedicated to meshes. This objective term covers three perspectives: edge, normal, and context constraints.

Edge Constraint. The edge length should remain consistent after the attack. Thus the edge constraint term $\mathcal{L}_{edge}$ can be defined to penalize the variation, as

$$\mathcal{L}_{edge} = \frac{1}{d}\sum_{i=1}^{d} \left\| E(X^*)_i - E(X)_i \right\|_2^2, \tag{3}$$

where $E(X)$ is the edge set of mesh X and d is the number of edges.

Normal Constraint. To penalize changes in face orientation, the normal constraint $\mathcal{L}_{norm}$ measures the deviations of face normals, as

$$\mathcal{L}_{norm} = \frac{1}{m}\sum_{i=1}^{m} \left\| Norm(F_i^*) - Norm(F_i) \right\|_2^2, \tag{4}$$

where F^*, F are the face sets of mesh X^* and X, respectively, $Norm$ is the operation of extracting face normals, and m is the number of faces (triangles).

Context Constraint. Maintaining stealthiness requires controlling vertex position deviations within their local context. For instance, when vertices are aligned in a straight line, even minor distortions

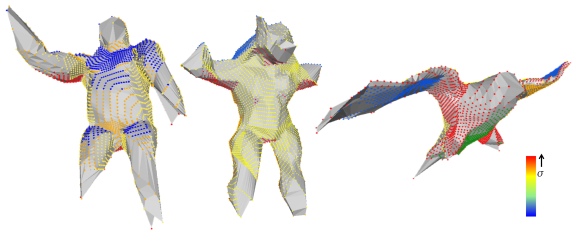

Figure 3: *GMM Partiation examples. The clusters with the same color have similar* σ *values.*

in the edges and face normals can greatly affect the geometric shape and visual appearance. Thus, we should penalize abrupt changes in vertex positions within their context.

To achieve this, we define a context constraint term $\mathcal{L}_{\text{cont}}$ that adopts Laplacian coordinates[Sor05] to represent local details. Specifically, the Laplacian coordinate of a vertex is typically derived by computing the weighted average coordinate of the vertex and those of its adjacent vertices. This method captures position information within the local neighborhood, thus assessing local geometric shape variations.

Let I be an identity matrix of size $n \times n$, where n is the number of vertices. Denote J as the mesh adjacency matrix, and $D = \text{diag}(d_1, \ldots, d_n)$as the degree matrix. The Laplacian operator is defined as $\phi = I - D^{-1}J$. The Laplacian coordinate of vertices can then be defined as $\tilde{V} = \phi V$. The context loss $\mathcal{L}_{\text{cont}}$ is given by:

$$\mathcal{L}_{\text{cont}} = \frac{1}{n}\sum_{i=1}^{n} \left\| \tilde{V}_i^* - \tilde{V}_i \right\|_2^2. \tag{5}$$

Since this term considers the relative position of a vertex to its neighbors, it promotes smooth perturbations.

Thus, the overall objective combines these terms as

$$\mathcal{L}_{\text{perc}} = \lambda_1 \mathcal{L}_{\text{edge}} + \lambda_2 \mathcal{L}_{\text{norm}} + \lambda_3 \mathcal{L}_{\text{cont}}. \tag{6}$$

4.2. Adaptive Geometry-aware Attack Optimization

Intuitively, the distortions in regions with high geometry frequencies are less notable than those in regions with low geometry frequencies. Therefore, to maintain stealthiness while increasing attacking accuracy, we can adaptively adjust the attack strength based on geometric complexity. To achieve this, we first describe a Geometry-aware GMM-based Mesh Partition strategy, which partitions the mesh into non-overlapped regions with varying degrees of geometric complexity. Then, we quantify the geometric complexity of each region as weights and describe a Geometry-aware Weighted $\mathcal{L}_{\text{perc}}$ to guide the optimization of attack steps. Finally, we outline the process of optimizing attack steps, which dynamically updates the attack steps for overall optimization.

Geometry-aware GMM-based Mesh Partition. Geometric complexity varies across different regions of the mesh, raising the question of *how to identify these regions with different degrees of geometry details.* We assume that the vertices within a region follow a specific distribution corresponding to a certain degree of geometric complexity, while different regions have different distributions. By simulating these distributions, we can partition the mesh according to the geometric complexity. To achieve this, we propose a Gaussian Mixture Model (GMM) based strategy.

Specifically, we partition the mesh into K regions, each represented by a Gaussian distribution component. Each region should meet two criteria: spatial coherence (the region should be spatially connected) and complexity representation (the region should exhibit a certain degree of complexity). For each vertex, we use its spatial coordinates and area-weighted vertex normal [Gla94] as its representation $f = (x, y, z, nx, ny, nz)$. The first three elements contribute to the spatial coherence, and the last three elements reflect geometry details. We initialize K Gaussian distributions $\{\mathcal{N}(\mu_k, \sigma_k)\}_{k=1}^K$ and weights for each distribution $\{\alpha_k\}_{k=1}^K$. Using the representations, we perform the Expectation Maximization (EM) algorithm for iterative clustering:

- E-step: Assign vertices to the corresponding distribution component by calculating the probability as

$$\arg\max_k p(k|f) = \frac{\alpha_k \cdot \mathcal{N}(f|\mu_k, \sigma_k)}{\sum_{k=1}^K \alpha_k \cdot \mathcal{N}(f|\mu_k, \sigma_k)}. \tag{7}$$

- M-step: Update $\mu_k, \sigma_k, \alpha_k$ based on just assigned vertices.

After several iterations, we can obtain the final distributions $\{\mathcal{N}(\mu_k, \sigma_k)\}_{k=1}^K$ with weights $\{\alpha_k\}_{k=1}^K$. Then we can assign each vertex to its corresponding region, resulting in K regions $\{\mathcal{R}_k\}_{k=1}^K$.

Geometry-aware Weighted $\mathcal{L}_{\text{perc}}$. The above strategy allows the mesh to be partitioned into distinct regions, where the vertices within each region follow a normal distribution. The mean of this distribution represents the central location and principal orientation of the local region, while the variance indicates the dispersion of vertices within the region. We then use the variance σ_k to depict the geometric complexity of each region, *i.e.*, "larger" σ_k denotes higher complexity for the k-th region.

Since the variance σ_k is a matrix, a direct comparison of its values is not feasible. Therefore, we adopt the spherical simplification, which assumes that the covariance matrix of each Gaussian component is a diagonal matrix with equal elements on the diagonal. This implies that the variances of the data in each dimension are independent, which reduces the complexity of the problem and computational overhead, while also aligning with the actual characteristics of mesh data. This yields a scalar variance value to represent the covariance matrix.

We denote the spherical simplification as $\mathcal{S}$ and normalize these scalar values to formulate the weights $\nu_k = 1 - \frac{\mathcal{S}(\sigma_k)}{\max \mathcal{S}(\sigma_k)}$. Thus, the perceptual-constrained objective can be reformulated as

$$\mathcal{L}_{\text{perc}}^* = \sum_{k=1}^{K} \nu_k \cdot \mathcal{R}_k(\mathcal{L}_{\text{perc}}), \tag{8}$$

where $\mathcal{R}_k(\mathcal{L}_{\text{perc}})$ represents the objective calculated from the k-th region $\mathcal{R}_k$. Fig. 3 shows several visual examples. It can be seen that the clusters with low variance are concentrated in relatively flat and smooth areas, while those with high variance are concentrated in areas with more complex geometric structures.

Attack Steps Iterative Optimization. As introduced in Sec. 3, an iterative optimization is widely utilized in existing methods (see

Algorithm 1: Procedure of the proposed method TPAM.

Input: An input mesh X with a ground truth label y, a (Bayesian) surrogate network $\mathcal{H}$, maximum iterations T, distortion bound ϵ, region number K, attack objective $\mathcal{L}_{\text{atk}}$, perceptual-constrained objective $\mathcal{L}_{\text{perc}}$, initial attack step $\boldsymbol{w}$, distance metric D.

Output: Adversarial mesh X^*

1 **Initialization:** $X_0^* = X, \boldsymbol{w}_0 = \boldsymbol{w}$;
2 **for** $t \leftarrow 0$ **to** $T-1$ **do**
3 $\quad X_{t+1}^* = X_t^* - \boldsymbol{w}_t \cdot \nabla_{(X_t^*)} \mathcal{L}_{\text{atk}} + \mathcal{L}_{\text{perc}}$;
4 $\quad \boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \gamma \cdot \nabla_{(\boldsymbol{w}_t)} \mathcal{L}_{\text{perc}}^*$;
5 $\quad$ **if** $D(X_{t+1}^*, X_0) > \epsilon$ **then**
6 $\quad\quad$ Return X_t^*
7 $\quad$ **end**
8 $\quad$ **if** $\mathcal{H}(X_{t+1}^*) \neq y$ **then**
9 $\quad\quad$ Return X_{t+1}^*
10 $\quad$ **end**
11 **end**

Eq. (2)). However, the attack step w is usually fixed during optimization, overlooking the relationship between geometric complexity and attack strength. Therefore, we create a set of attack steps $\boldsymbol{w} = \{w_k\}_{k=1}^K$ for each region and describe an Attack Steps Iterative Optimization strategy that jointly optimizes the objectives $\mathcal{L}_{\text{atk}} + \mathcal{L}_{\text{perc}}$ and the attack steps $\boldsymbol{w}$. It is important to note that simultaneously optimizing them is difficult. As such, we introduce an alternative optimization strategy: During the optimization process, the attack steps for each region are preset. Then we perform an adversarial attack by optimizing the overall objectives. After obtaining the perturbations, we fix the shape of mesh and alternatively update the attack step by optimizing the perceptual-constrained objective term. The optimization process is outlined as follows:

$$\begin{aligned} &X_0^* = X, \boldsymbol{w}_0 = \boldsymbol{w}, \\ &X_{t+1}^* = X_t^* - \boldsymbol{w}_t \cdot \nabla_{(X_t^*)} \mathcal{L}_{\text{atk}} + \mathcal{L}_{\text{perc}}, \\ &\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \gamma \cdot \nabla_{(\boldsymbol{w}_t)} \mathcal{L}_{\text{perc}}^*, \end{aligned} \tag{9}$$

where γ is the step factor for optimizing $\boldsymbol{w}$. This process iterates until the maximum iteration number T is reached. The major procedure of our method is illustrated in Alg.1: Line 1 represents the initialization of adversarial mesh and attack step. Line 2 corresponds to the first stage of an iteration, where the adversarial mesh X_t^* is updated with an attack step $\boldsymbol{w}_t$. Line 3 indicates updating the adaptive attack step $\boldsymbol{w}_{t+1}$ with a step γ.

4.3. Bayesian Surrogate Network

The effectiveness of transfer attacks largely depends on the overlap between the feature spaces of the target and surrogate networks. Existing methods often use surrogate networks with fixed parameters, resulting in a stationary feature space. When there is less overlap with target networks, transferability is greatly hindered. While ensembling multiple surrogate networks can enhance feature space overlap, this approach is resource-intensive and thus impractical.

To overcome this limitation, we describe a Bayesian surrogate network to increase feature space diversity without the need to ensemble multiple networks. Specifically, we replace the conventional fully connected layers with Bayesian layers, where layer parameters are sampled from a distribution trained using maximum likelihood estimation (see Fig.4). We employ the variational Bayesian approach to train this network [KW13]. In Bayesian layers, parameters are sampled during forward propagation through a reparameterization trick. This strategy increases the uncertainty of the surrogate network, leading to a more diverse feature space and subsequently enhancing the probability of overlapping with target networks.

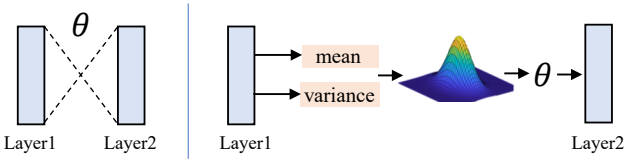

Figure 4: *Illustration of Bayesian layers, where the parameters are sampled from a learned distribution.*

4.4. Area Under Accuracy

To complement Accuracy (ACC), we describe a unified and comprehensive evaluation protocol to fairly assess the attacking performance and stealthiness. Different from existing methods that manually select a single distortion bound, we evaluate performance across various bounds and introduce the metric of Area Under Accuracy (AUACC). This metric is defined as

$$\text{AUACC} = \int_{\epsilon} \text{ACC}(\epsilon)\, d\epsilon, \tag{10}$$

where $\text{ACC}(\epsilon)$ denotes the ACC score under distortion bound ϵ.

5. Experiments

5.1. Experimental Settings

Datsets. Following existing works [MCBR20; RPC*21; BYBT22], our method is validated on two widely used datasets SHREC11 [VvJD*11] and Manifold40 [WSK*15]. SHREC11 dataset contains 600 samples with 30 categories, and each category has 20 samples. For each category, we use 16 samples for training the surrogate model and use left 4 samples for adversarial attack evaluation following [ESKB17; KC22]. The Manifold40 dataset contains 12,311 common objects categorized into 40 different classes. For this dataset, we use its official split, where 9,843 samples are for training the surrogate model and 2,488 samples are for adversarial attack evaluation.

Mesh Classifiers. Our method is performed on several recent well-known mesh classification networks, including MeshNet [FFY*19], MeshCNN [HHF*19], PD-MeshNet [MLR*20], MeshWalker [LT20], ExMeshCNN [KC22], RIMeshGNN [SKB24]. Following their default configuration, we study MeshWalker, ExMeshCNN, and RIMeshGNN on both SHREC11 and Manifold40 datasets. We explore MeshCNN, PD-MeshNet and HodgeNet on the SHREC11 dataset and investigate MeshNet on the Manifold40 dataset.

Compared Methods. Our method is compared with recent

Table 1: *ACC↓ (%) | AUACC↓ (%) of one-to-one transfer attack on SHREC11 datasets. The surrogate network is MeshWalker. The* ***best*** *scores are highlighted by bold and underlining, whereas the* ***second-best*** *scores are highlighted in bold.*

Methods	MeshWalker	MeshCNN	PD-MeshNet	ExMeshCNN	RIMeshGNN	Average
Original accuracy	99.17	99.17	99.17	96.67	100	98.84
BLP [MCBR20]	75.83 | 83.04	84.17 | 94.67	64.17 | 87.13	93.33 | 95.42	75.00 | 91.50	78.50 | 90.35
Spectral-attack [RPC*21]	47.50 | 65.17	45.83 | 83.80	46.67 | 81.37	89.17 | 94.54	48.33 | 77.98	55.50 | 80.57
RandomWalker [BYBT22]	**25.00 | 53.67**	**26.67 | 60.33**	**29.17 | 65.13**	**78.33 | 91.04**	**33.33 | 68.58**	**38.50 | 67.75**
3DVP [XHFB22]	29.17 | 58.46	35.83 | 72.67	35.83 | 69.71	84.17 | 93.54	40.00 | 76.08	43.67 | 74.29
Mesh-attack [ZCL*23]	25.83 | 60.58	41.67 | 69.92	39.17 | 71.33	90.00 | 94.08	50.00 | 81.28	49.33 | 75.44
TPAM	**16.67 | 48.63**	**21.67 | 62.83**	**26.67 | 65.21**	**77.50 | 90.46**	**28.33 | 64.96**	**34.17 | 66.42**

attack methods, 3DVP [XHFB22], Mesh-attack [ZCL*23], BLP [MCBR20], Spectral-attack [RPC*21] and RandomWalker [BYBT22]. Since the methods of 3DVP, Mesh-attack, BLP, and Spectral-attack have not released their codes, we reproduce these methods rigorously following their paper. For RandomWalker, we utilize its official codes. Note that the mesh shapes in the Manifold40 dataset are regular. Spectral-attack and Mesh-attack are hardly adapted to it, as they easily distort the meshes due to their inner pipeline. For a fair comparison, all the methods share the same attacking configuration such as the same termination requirements, the same distortion bound, etc.

Implementation Details. Our method is implemented using Tensorflow 2.4.1 with a Nvidia 3060 GPU. We employ Meshwalker as our surrogate network. Following previous methods [XQL19; BYBT22], we employ ℓ_2 norm as a distance metric to measure the bound of distortion ϵ. The parameters in perceptual-constrained objective is set to $\lambda_1 = 1, \lambda_2 = 0.1, \lambda_3 = 10$. The region number partitioned from meshes is set to $K = 8$. The maximum iteration is set to $T = 10000$. The step in optimizing $\boldsymbol{w}$ is set to $\gamma = 3000$.

5.2. Results

Following [BYBT22], we evaluate our method under a setting of *one-to-one transfer attack*, training the surrogate network using the output of a target network, and then attacking this target network with adversarial meshes generated from the surrogate network. We also explore a more challenging setting, the *one-to-many transfer attack*, where we attempt to attack multiple target networks using a parameter-fixed surrogate network trained on one of the target networks.

One-to-one Transfer Attack. We validate the performance using two metrics: the ACC metric, commonly used in existing methods, and the AUACC metric, proposed in this paper. For the calculation of ACC, we empirically set the upper bound as $\epsilon = 0.02$, as this distortion level is visually acceptable based on a user study. Table 1 shows the ACC performance of different methods on the SHREC11 dataset. Note that the surrogate network is MeshWalker. As such, the first column of MeshWalker represents the *white-box attack*, as the surrogate network and target network are the same. The results show that our method outperforms others by a large margin, reducing the ACC by approximately 9.33% in white-box attack and by $5\%, 2.5\%, 0.8\%, 5\%$ on MeshCNN, PD-MeshNet, and RIMeshGNN, respectively. For AUACC, our method achieves the highest score, surpassing RandomWalker by 1.33%. It can be seen that all methods show compromised performance on ExMeshCNN, likely due to its use of enriched features such as edge-based geodesic and face-based geometric features, which increases the robustness. Nonetheless, our method still outperforms the second-best method, RandomWalker.

To validate these methods under the AUACC metric, we increase the bound ϵ from 0 to 0.02 with an interval of 0.001 and calculate the ACC for each bound. Then we calculate the area under the ACC curve and normalize it to the range of $[0, 1]$ by dividing by the maximum area, *i.e.*, a rectangular area denoted by ACC = 0, ACC = 1, $\epsilon = 0$, and $\epsilon = 0.02$. The results, shown in Table 1, indicate that our method comprehensively surpasses others on average across various bounds, demonstrating its effectiveness in maintaining both stealthiness and attack accuracy.

One-to-many Transfer Attack. In this setting, we utilize the output of MeshCNN to train the surrogate network MeshWalker and then perform the transfer attack on all target networks except MeshWalker. Table 2 shows the ACC and AUACC scores of different methods. It can be seen that our method achieves superiority compared to other methods on all target networks, exceeding the second-best method RandomWalker by 4.79% at ACC on average, and 1.07% at AUACC on average. These results highlight that our method has better transferability than others, enhancing the application in real-world scenarios.

Moreover, we validate these attacking methods on the Manifold40 dataset. The results are shown in Table 3. We can observe that our method can averagely outperform the second-best method by 2.77% at the ACC score and 0.5% at the AUACC score.

6. Ablation Study

Effect of Bayesian Layers. This part studies the effect of Bayesian layers in surrogate networks. Denote "without" as "w/o" and "with" as " w/". Experiments are conducted on the SHREC11 and Manifold40 datasets under one-to-many transfer attack settings on PD-MeshNet, ExMeshCNN, and RIMeshGNN classifiers. The results are shown in Table 4 and Table 5, which reveal that using Bayesian layers can notably reduce both ACC and AUACC scores by $0.84\%/4.42\%, 4.17\%/1.04\%, 5\%/3.84\%$ on SHREC11 dataset and $4.17\%/1.04\%, 5\%/3.84\%$ on Manifold40 dataset. This demonstrates the effectiveness of Bayesian layers in enhancing transferability.

Various Combinations of $\mathcal{L}_{\text{perc}}$. We experiment on the SHREC11 dataset to verify the effectiveness of each term in the proposed perceptual-constrained objective $\mathcal{L}_{\text{perc}}$. The results are shown in

Table 2: *ACC↓ (%) | AUACC↓ (%) of one-to-many transfer attack on SHREC11 dataset. The surrogate network is MeshWalker.*

Methods	MeshCNN(♠)	PD-MeshNet	ExMeshCNN	RIMeshGNN	Average
Original accuracy	99.17	99.17	96.67	100	98.75
BLP [MCBR20]	84.17 | 94.67	62.50 | 87.79	93.33 | 95.25	71.67 | 90.88	77.42 | 92.65
Spectral-attack [RPC*21]	45.83 | 83.38	48.33 | 81.96	90.83 | 94.79	50.00 | 79.02	58.75 | 84.79
RandomWalker [BYBT22]	**26.67** | **60.33**	**22.50** | **63.33**	**79.17** | **90.83**	**28.33** | **62.88**	**39.17** | **69.84**
3DVP [XHFB22]	35.83 | 72.67	23.33 | 68.37	83.33 | 92.71	38.33 | 70.31	45.71 | 76.52
Mesh-attack [ZCL*23]	41.67 | 69.92	40.00 | 71.92	89.17 | 94.25	40.83 | 75.48	52.92 | 77.89
TPAM	**21.67** | **62.83**	**18.33** | **58.71**	**77.50** | **90.46**	**20.00** | **61.08**	**34.38** | **68.77**

Table 3: *ACC↓ (%) | AUACC↓ (%) of one-to-many transfer attack on Manifold40 dataset. The surrogate network is MeshWalker.*

Methods	MeshNet(♠)	ExMeshCNN	RIMeshGNN	Average
Original accuracy	92.02	88.90	89.99	90.30
BLP [MCBR20]	86.88 | 89.89	82.38 | 85.87	82.73 | 87.75	84.16 | 87.84
RandomWalker [BYBT22]	**77.25** | **86.00**	78.17 | **84.27**	**76.60** | **84.68**	**77.34** | **84.98**
3DVP [XHFB22]	81.81 | 88.59	**77.47** | 85.12	78.20 | 84.87	79.16 | 86.19
TPAM	**78.00** | **87.48**	**73.26** | **83.84**	**72.45** | **82.12**	**74.57** | **84.48**

Table 4: *Effect of Bayesian layers on SHREC11 dataset.*

Methods	PD-MeshNet		ExMeshCNN		RIMeshGNN	
	ACC	AUACC	ACC	AUACC	ACC	AUACC
w/o Bayesian	19.17	63.13	81.67	91.50	25.00	64.92
w/ Bayesian	18.33	58.71	77.50	90.46	20.00	61.08

Table 5: *Effect of Bayesian layers on Manifold40 dataset.*

Methods	ExMeshCNN		RIMeshGNN	
	ACC	AUACC	ACC	AUACC
w/o Bayesian	77.23	84.27	76.74	84.70
w/ Bayesian	73.26	83.84	72.45	82.12

Table 6: *Various combinations of* $\mathcal{L}_{\text{perc}}$.

$\mathcal{L}_{\text{edge}}$	$\mathcal{L}_{\text{norm}}$	$\mathcal{L}_{\text{cont}}$	ACC	AUACC
✓			20.83	63.42
	✓		21.67	62.96
		✓	21.67	62.67
✓	✓		22.50	62.46
✓		✓	19.17	61.50
	✓	✓	20.83	62.01
✓	✓	✓	19.17	59.72

Table 6. It can be seen that using all three objective terms yields the best performance, demonstrating the effectiveness of each objective term.

Effect of various γ. This part studies the effect of various γ shown in Eq.(9). To do so, we conduct an experiment to attack PD-MeshNet in one-to-one transferable attacks on the SHREC11 dataset with various γ in $[500, 3000, 5000, 10000, 20000]$. It can be seen that our method is not sensitive to the value of γ. To strike a good balance between ACC and AUACC, we set $\gamma = 3000$ in our main experiment.

Effect of the Number K of Components in GMM. This part studies the effect of the region number K in GMM-based strategy.

Table 7: *Effect of various* γ.

γ	ACC	AUACC
500	29.17	66.45
3000	26.67	65.21
5000	26.67	64.38
10000	28.33	64.73
20000	30.00	65.46

The experiment is conducted in one-to-one transferable attacks on SHREC11 dataset with various k in range of $\{1, ..., 10\}$. The target network is MeshCNN. Note that $k = 1$ corresponds to treating the whole mesh as a region. Moreover, we study an extreme case, where K is set as the number of vertices in mesh. This setting treats each vertex as a region. Fig. 5 shows the ACC histogram for various k values. It can be observed that using GMM ($K > 1$) reduces the ACC score, demonstrating the effectiveness of the Geometry-aware GMM-based strategy. It also has a certain effect when $K = N$ compared to $K = 1$. However, this setting corresponds to adjusting attack steps vertex-by-vertex. Since a single vertex cannot express geometry details, it only performs slightly better than $K = 1$. Meanwhile, since this setting performs on each vertex, it can introduce large computational overhead, limiting its practical use. In the main experiments, we use $K = 8$, as it achieves the lowest ACC score.

7. Further Analysis

ACC Curves. Fig. 6 shows the ACC curves of one-to-one transfer attack of different methods on SHREC11 dataset. It can be seen that our method does not show superiority when the bound is small. It is because our method is designed to maintain stealthiness, thus it likely disperses the distortion into regions with high geometric complexity. Nevertheless, given the small bound, the distortions in these regions may not have an adequate effect. The performance of our method becomes better with the bound increasing, demonstrating the good balance between attacking performance and stealthi-

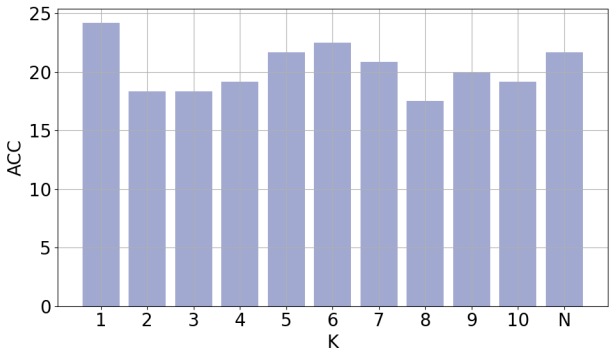

Figure 5: *Effect of the number K of components in GMM.*

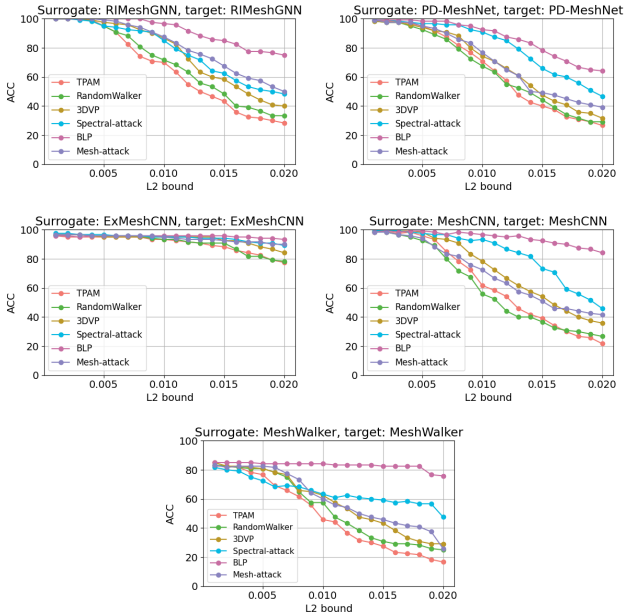

Figure 6: *ACC curves of one-to-one transfer attack on SHREC11 dataset.*

ness. A similar trend is also observed in Fig. 7, which shows the ACC curves of one-to-many transfer attacks of different methods.

More Examples of Attack Results. Fig 8 shows two meshes with fine details using the original resolution. Fig. 9 shows more examples of attack results of our method. These examples are all from the SHREC11 dataset with a simplified 500 faces. It can be seen that these adversarial meshes have less noticeable distortions but successfully disrupt the correct predictions.

Gray-box Attack. This part discusses a new setting, the Gray-box attack, where the architecture of the surrogate network is the same as the target network but the parameters are different. To investigate this setting, we set the surrogate network as MeshWalker. On the SHREC11 dataset, the surrogate network is trained using the output of MeshCNN while the target network is trained using the ground truth. On the Manifold40 dataset, the surrogate network is trained using the output of MeshNet while the target network is

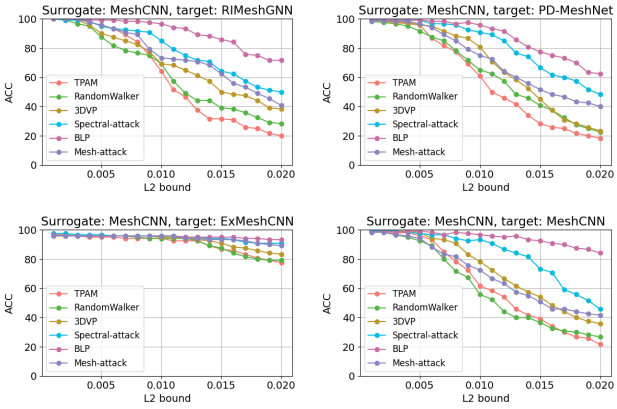

Figure 7: *ACC curves of one-to-many transfer attack on SHREC11 dataset.*

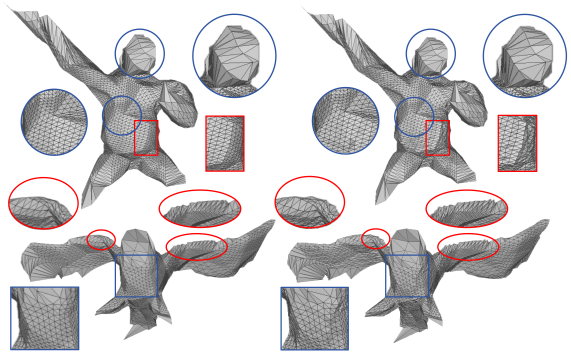

Figure 8: *Meshes with fine details represented by triangles. More perturbations are added to complex regions (red) and less to smooth regions (blue).*

trained using the ground truth. The results are shown in Table 8. It can be seen that our method still performs favorably under this setting. Our method without Bayesian layers achieves the best performance, which outperforms the second best method RandomWalker by 7.5%, 2.96% at ACC and AUACC on the SHREC11 dataset, and by 2.22%, 0.37% at ACC and AUACC on Manifold40 dataset. Nevertheless, by using Bayesian, the performance slightly drops. It aligns with our knowledge, as Bayesian increases the uncertainty, thus reducing overfitting to the same architecture.

8. Discussions

Tentative Study on Different Type Classifiers. This part explores the performance of our method on a different type of classifier, SubdivNet [HLG*22]. In contrast to other mesh classifiers, SubdivNet takes a mesh with subdivision sequence connectivity as input. Therefore, we first attack the meshes and then map the adversarial meshes into one with the loop subdivision sequence connectivity, the input of SubdivNet. Table 9 shows the performance comparison of our method with BLP and RandomWalker on the SHREC11

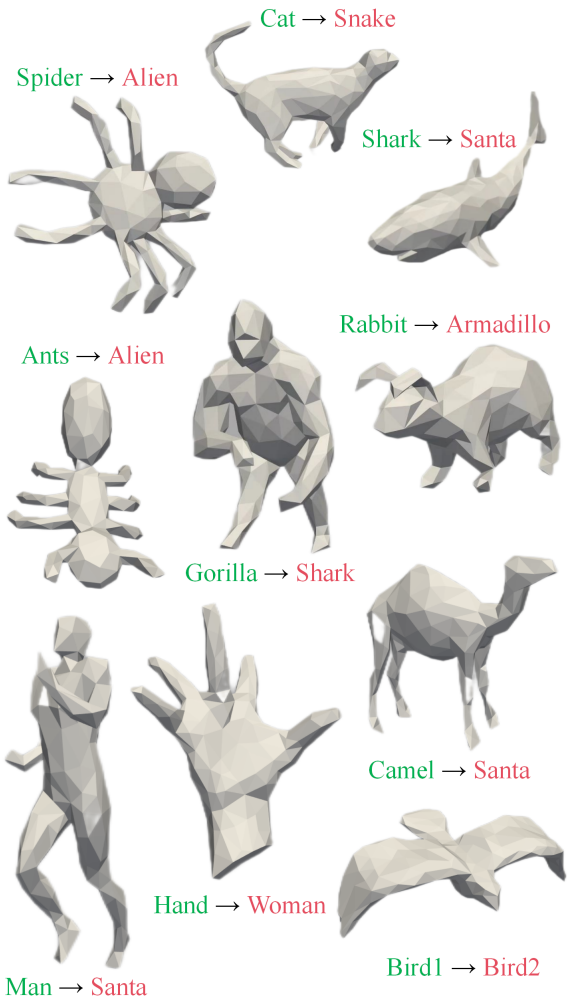

Figure 9: *More examples of attack results.*

Table 8: *Gray-box attack: The surrogate network is MeshWalker. The surrogate network is trained using MeshCNN on SHREC11 dataset and trained using MeshNet on the Manifold40 dataset. The target network is MeshWalker.*

Methods	SHREC11		Manifold40	
	ACC	AUACC	ACC	AUACC
BLP [MCBR20]	96.67	98.92	85.32	86.73
Spectral-attack [RPC*21]	88.33	97.42	-	-
RandomWalker [BYBT22]	30.83	67.79	64.40	77.74
3DVP [XHFB22]	42.50	78.88	72.74	81.21
Mesh-attack [ZCL*23]	45.83	73.96	-	-
TPAM w/o Bayesian	23.33	64.83	62.18	77.37
TPAM w/ Bayesian	25.83	67.33	67.57	79.22

dataset. It can be seen that our method can still achieve the best performance.

Feasibility on Open Meshes. Besides evaluating on closed meshes in the main experiments, we also test our method on open meshes to demonstrate its generalizability. To obtain the labeled open meshes, we remove specific vertices and faces from the SHREC11 dataset using MeshLab [CCC*08]. Our method can be applied to this scenario with minor modifications to the perceptual-constrained objectives. Since certain vertices and faces are removed, norm and context constraints no longer apply, so we rely only on the edge constraint. Fig. 10 shows an example attacked by our method, showing that it maintains high stealthiness while still misleading the classifier.

Table 9: *ACC↓ (%) | AUACC↓ (%) of one-to-many transfer attack on SHREC11 dataset, The target network is SubdivNet. The surrogate network is MeshWalker.*

Methods	SubdivNet
Original accuracy	100
BLP [MCBR20]	93.33 | 98.37
RandomWalker [BYBT22]	64.17 | 87.62
TPAM	**62.50 | 86.74**

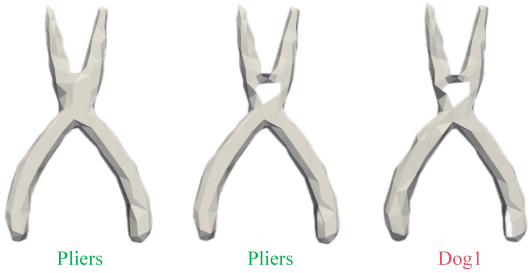

Figure 10: *An example of attacks on an open mesh. From left to right denotes original closed mesh, open mesh, and adversarial open mesh.*

Relation with Mesh Decimation. Mesh Decimation, also known as mesh simplification, aims to reduce the complexity of a 3D mesh while balancing visual fidelity and computational efficiency [PPZ22; XLW*24]. This process involves reducing the number of polygons or vertices in the mesh, making the model less resource-intensive to render or process while maintaining as much of its original appearance and details as possible. Although mesh decimation also needs to preserve shape, it significantly differs from the perceptual-constrained metric in our method.

Since meshes before and after decimation typically have different number of vertices, leading to changes in topology, non one-to-one correspondence metrics are often utilized. These include Chamfer distance [PPZ22; XLW*24; PBZ22] and Hausdorff distance [XLW*24] for measuring the deviation of point sets as well as normal consistency [XLW*24], triangle collision [PPZ22] and Edge Crossings [PPZ22] for assessing the consistency of meshes.

In contrast, our method aims to add minimal distortion on the position of vertices, ensuring that the topology of meshes remains unchanged, with the same number of vertices and edges. Therefore, the constraints used in our perceptual-constrained metric ensure a one-to-one correspondence, *i.e.*, measuring the deviation of

corresponding edges or norms before and after attack. Moreover, to maintain the stealthiness of attack, we also introduce context constraint to restrict the vertex deviation within local context.

9. Conclusion

In this paper, we introduce a new method called Transferable Perceptual-constrained Adversarial Meshes (TPAM) to investigate two key aspects in adversarial attacks: stealthiness and transferability. Concretely, we introduce a Perceptual-constrained objective term to properly measure the stealthiness of meshes, and then an Adaptive Geometry-aware Attack Optimization strategy to adjust attacking strength according to local geometric complexity. Moreover, we introduce a Bayesian Surrogate Network to improve its uncertainty, leading to the enhancement of transferability. Lastly, we describe a new metric called the Area Under Accuracy (AUACC) to comprehensively evaluate attacking performance. Extensive experiments on two datasets with various mesh classifiers demonstrate the superiority of our method in both white-box and black-box attacks.

Acknowledgements

This work was supported by NSFC under grant No.62102380, NSF of Shandong province under grant No.ZR2021QF095 and China Postdoc Science Foundation under grant No.2021M693022.

References

[BYBT22] BELDER, AMIR, YEFET, GAL, BEN-ITZHAK, RAN, and TAL, AYELLET. "Random Walks for Adversarial Meshes". *ACM SIGGRAPH 2022 Conference Proceedings*. 2022, 1–9 1–3, 6–8, 10.

[CCC*08] CIGNONI, PAOLO, CALLIERI, MARCO, CORSINI, MASSIMILIANO, et al. "MeshLab: an Open-Source Mesh Processing Tool". *Eurographics Italian Chapter Conference*. 2008 10.

[CMW*17] CHEN, XIAOZHI, MA, HUIMIN, WAN, JI, et al. "Multi-view 3d object detection network for autonomous driving". *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 2017, 1907–1915 1.

[CW17] CARLINI, NICHOLAS and WAGNER, DAVID. "Towards evaluating the robustness of neural networks". *2017 IEEE Symposium on Security and Privacy*. 2017, 39–57 1.

[ESKB17] EZUZ, DANIELLE, SOLOMON, JUSTIN, KIM, VLADIMIR G, and BEN-CHEN, MIRELA. "GWCNN: A metric alignment layer for deep shape analysis". *Computer Graphics Forum*. Vol. 36. 5. 2017, 49–57 6.

[FFY*19] FENG, YUTONG, FENG, YIFAN, YOU, HAOXUAN, et al. "Meshnet: Mesh neural network for 3d shape representation". *Proceedings of the AAAI conference on artificial intelligence*. Vol. 33. 01. 2019, 8279–8286 1, 3, 6.

[Gla94] GLASSNER, ANDREW. "I.6. - Building Vertex Normals from an Unstructured Polygon List". *Graphics Gems*. 1994, 60–73 5.

[GSS14] GOODFELLOW, IAN J, SHLENS, JONATHON, and SZEGEDY, CHRISTIAN. "Explaining and harnessing adversarial examples". *arXiv preprint arXiv:1412.6572* (2014) 1, 3.

[HHF*19] HANOCKA, RANA, HERTZ, AMIR, FISH, NOA, et al. "Meshcnn: a network with an edge". *ACM Transactions on Graphics (TOG)* 38.4 (2019), 1–12 1, 3, 6.

[HLG*22] HU, SHI-MIN, LIU, ZHENG-NING, GUO, MENG-HAO, et al. "Subdivision-based mesh convolution networks". *ACM Transactions on Graphics (TOG)* 41.3 (2022), 1–16 9.

[KC22] KIM, SEONGGYEOM and CHAE, DONG-KYU. "Exmeshcnn: An explainable convolutional neural network architecture for 3d shape analysis". *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2022, 795–803 1, 3, 6.

[KCMB18] KANCHAN, BAHIRAT, CHENGYUAN, LAI, MCMAHAN, RYAN P., and BALAKRISHNAN, PRABHAKARAN. "Designing and Evaluating a Mesh Simplification Algorithm for Virtual Reality". *ACM Transactions on Multimedia Computing Communications and Applications* 14.3s (2018), 1–26 1.

[KW13] KINGMA, DIEDERIK P and WELLING, MAX. "Auto-encoding variational bayes". *arXiv preprint arXiv:1312.6114* (2013) 6.

[LT20] LAHAV, ALON and TAL, AYELLET. "Meshwalker: Deep mesh understanding by random walks". *ACM Transactions on Graphics (TOG)* 39.6 (2020), 1–13 1, 3, 6.

[LYS19] LIU, DANIEL, YU, RONALD, and SU, HAO. "Extending adversarial attacks and defenses to deep 3d point cloud classifiers". *2019 IEEE International Conference on Image Processing*. 2019, 2279–2283 1, 3.

[MCBR20] MARIANI, GIORGIO, COSMO, LUCA, BRONSTEIN, ALEXANDER M, and RODOLA, EMANUELE. "Generating Adversarial Surfaces via Band-Limited Perturbations". *Computer Graphics Forum*. Vol. 39. 5. 2020, 253–264 1–3, 6–8, 10.

[MFFF17] MOOSAVI-DEZFOOLI, SEYED-MOHSEN, FAWZI, ALHUSSEIN, FAWZI, OMAR, and FROSSARD, PASCAL. "Universal adversarial perturbations". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, 1765–1773 3.

[MLR*20] MILANO, FRANCESCO, LOQUERCIO, ANTONIO, ROSINOL, ANTONI, et al. "Primal-dual mesh convolutional neural networks". *Advances in Neural Information Processing Systems*. 33 (2020), 952–963 3, 6.

[MMS*17] MADRY, ALEKSANDER, MAKELOV, ALEKSANDAR, SCHMIDT, LUDWIG, et al. "Towards deep learning models resistant to adversarial attacks". *arXiv preprint arXiv:1706.06083* (2017) 3.

[PBZ22] POTAMIAS, ROLANDOS ALEXANDROS, BOURITSAS, GIORGOS, and ZAFEIRIOU, STEFANOS. "Revisiting point cloud simplification: A learnable feature preserving approach". *European Conference on Computer Vision*. 2022, 586–603 10.

[PKGF21] PASCHALIDOU, DESPOINA, KATHAROPOULOS, ANGELOS, GEIGER, ANDREAS, and FIDLER, SANJA. "Neural parts: Learning expressive 3d shape abstractions with invertible neural networks". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2021, 3204–3215 1.

[PMJ*16] PAPERNOT, NICOLAS, MCDANIEL, PATRICK, JHA, SOMESH, et al. "The limitations of deep learning in adversarial settings". *2016 IEEE European Symposium on Security and Privacy*. 2016, 372–387 1.

[PPZ22] POTAMIAS, ROLANDOS ALEXANDROS, PLOUMPIS, STYLIANOS, and ZAFEIRIOU, STEFANOS. "Neural mesh simplification". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2022, 18583–18592 10.

[Ric08] RICHARD A. Robb, PH. D. "Abstract 3-Dimensional Visualization in Medicine and Biology". *Handbook of Medical Imaging* 26.7 (2008), 685–712 1.

[RPC*21] RAMPINI, ARIANNA, PESTARINI, FRANCO, COSMO, LUCA, et al. "Universal spectral adversarial attacks for deformable shapes". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2021, 3216–3226 1, 3, 6–8, 10.

[SKB24] SHAKIBAJAHROMI, BAHAREH, KIM, EDWARD, and BREEN, DAVID E. "Rimeshgnn: A rotation-invariant graph neural network for mesh classification". *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2024, 3150–3160 1, 3, 6.

[SLA23] STOLIK, TOMER, LANG, ITAI, and AVIDAN, SHAI. "SAGA: Spectral Adversarial Geometric Attack on 3D Meshes". *Proceedings of the IEEE International Conference on Computer Vision*. 2023, 4284–4294 1, 3.

[Sor05] SORKINE, OLGA. "Laplacian mesh processing". *Eurographics (State of the Art Reports)* 4.4 (2005), 1 5.

[SS21] SMIRNOV, DMITRIY and SOLOMON, JUSTIN. "Hodgenet: Learning spectral geometry on triangle meshes". *ACM Transactions on Graphics (TOG)* 40.4 (2021), 1–11 1.

[SZS*13] SZEGEDY, CHRISTIAN, ZAREMBA, WOJCIECH, SUTSKEVER, ILYA, et al. "Intriguing properties of neural networks". *arXiv preprint arXiv:1312.6199* (2013) 1, 3.

[SZTL19] SUN, CHUN-YU, ZOU, QIAN-FANG, TONG, XIN, and LIU, YANG. "Learning adaptive hierarchical cuboid abstractions of 3d shape collections". *ACM Transactions on Graphics (TOG)* 38.6 (2019), 1–13 1.

[VvJD*11] VELTKAMP, REMCO C, van JOLE, STEFAN, DRIRA, HASSEN, et al. "SHREC'11 Track: 3D Face Models Retrieval." *3DOR@ Eurographics*. 2011, 89–95 2, 6.

[WSK*15] WU, ZHIRONG, SONG, SHURAN, KHOSLA, ADITYA, et al. "3d shapenets: A deep representation for volumetric shapes". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, 1912–1920 2, 6.

[XHFB22] XU, HUANGXINXIN, HE, FAZHI, FAN, LINKUN, and BAI, JUNWEI. "D3AdvM: A direct 3D adversarial sample attack inside mesh data". *Computer Aided Geometric Design*. 97 (2022), 102122 1–3, 7, 8, 10.

[XLW*24] XU, RUI, LIU, LONGDU, WANG, NINGNA, et al. "CWF: Consolidating Weak Features in High-quality Mesh Simplification". *ACM Transactions on Graphics (TOG)* 43.4 (2024), 1–14 10.

[XQL19] XIANG, CHONG, QI, CHARLES R, and LI, BO. "Generating 3d adversarial point clouds". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, 9136–9144 1, 3, 7.

[ZCL*23] ZHANG, JINLAI, CHEN, LYUJIE, LIU, BINBIN, et al. "3d adversarial attacks beyond point cloud". *Information Sciences*. 633 (2023), 491–503 1–3, 7, 8, 10.

[ZGH*24] ZHANG, JIANPING, GU, WENWEI, HUANG, YIZHAN, et al. "Curvature-Invariant Adversarial Attacks for 3D Point Clouds". *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 38. 7. 2024, 7142–7150 1, 3.

[ZT18] ZHOU, YIN and TUZEL, ONCEL. "Voxelnet: End-to-end learning for point cloud based 3d object detection". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, 4490–4499 1.