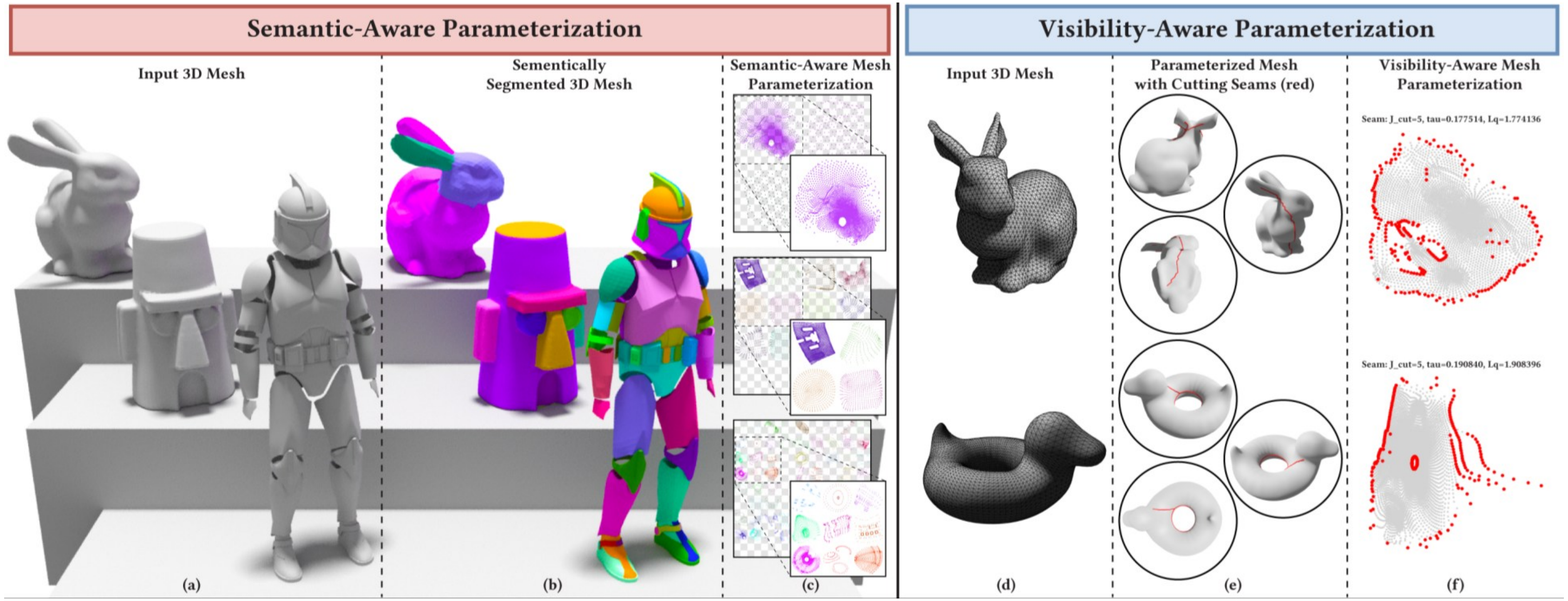
International Conference on Learning Representations (ICLR) 2026

Unsupervised Representation Learning for 3D Mesh Parameterization with Semantic and Visibility Objectives

Amirhossein Zamani [1,2,3] Bruno Roy [1] Arianna Rampini [1]

[1] Autodesk Research [2] Mila - Quebec AI Institute [3] Concordia University



Presented by: Amirhossein Zamani
2026 - March - 30

Introduction

Behind the realism in games or films like Avatar is a complex process that has long challenged 3D artists!

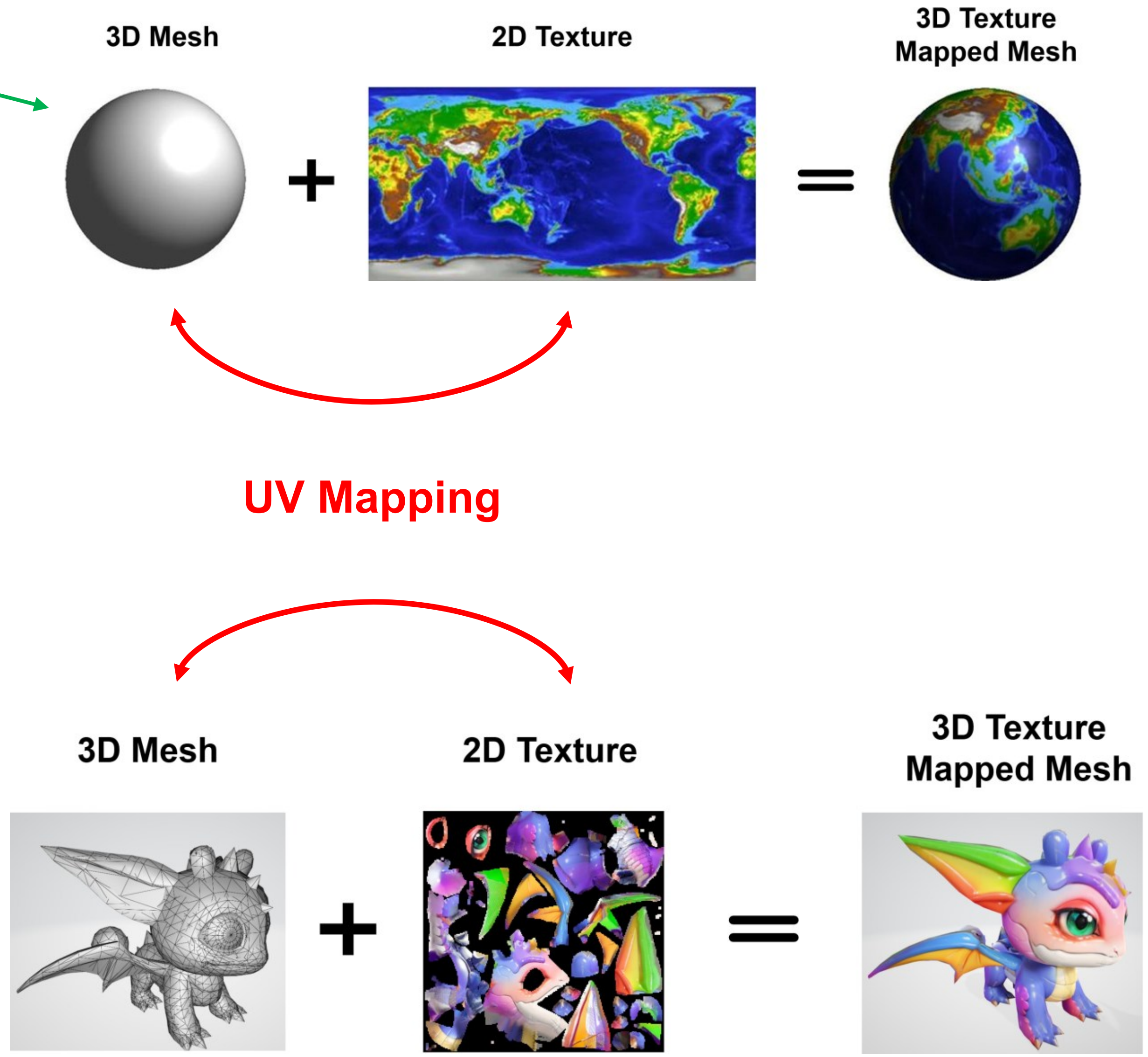
The Last of Us Part II

Red Dead Redemption 2

Avatar

Introduction

- **Preliminary**
    - Manual UV mapping by artist: mesh points ⟶ texture pixels
- **Core Challenge**
    - UV mapping = ~60% of 3D content creation time
- **Research Question**
    - Can we automate the UV mapping process using AI?

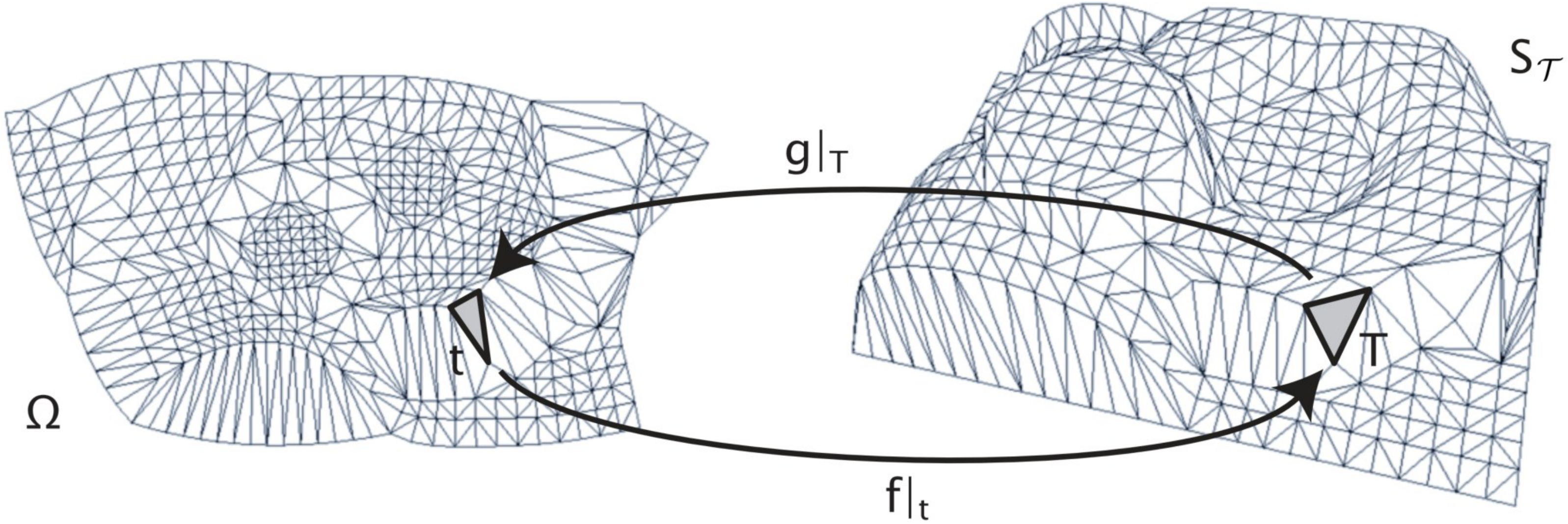


This is straightforward

But this one is complicated!

Introduction

- **Geometry-Preserving UV Parameterization:** Aim to flatten 3D surfaces to 2D while minimizing distortion and avoiding overlaps
  - Conformality (angle-preserving)
  - Equiareality (area-preserving)
  - Stretch (length-preservation)
  - Bijectivity



The Parameterization of a triangle mesh

(Image Credit: Kai Hormann, Bruno Lévy, Alla Sheffer [1])

- **Issue:** While these properties are necessary to achieve a high-quality parameterization, they are not sufficient for many downstream applications in content creation and texture synthesis

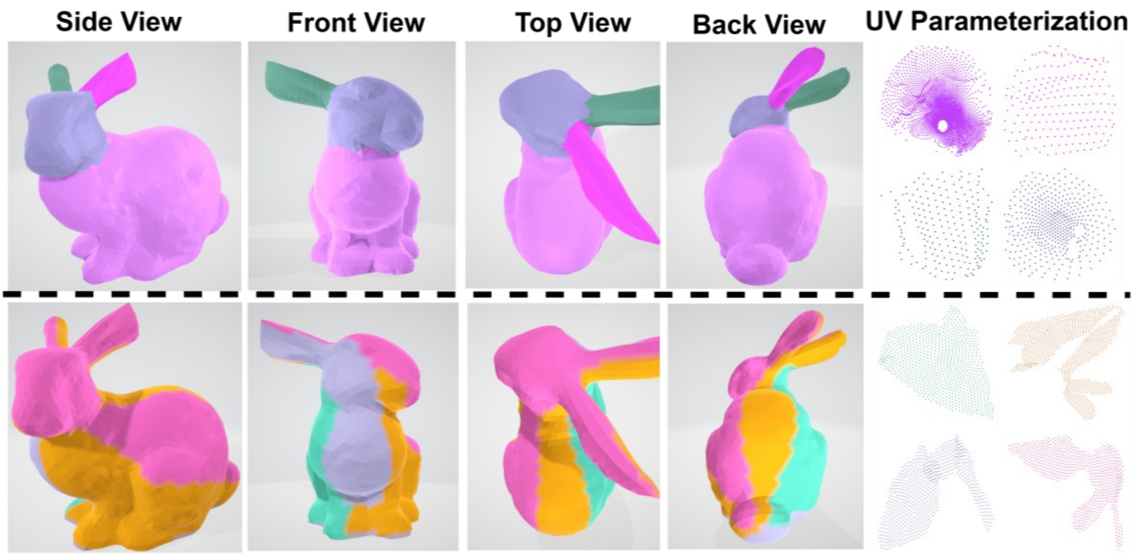
---

[1] Hormann, Kai, Bruno Lévy, and Alla Sheffer. "Mesh parameterization: Theory and practice." (2007).

Introduction

- Two important but missing properties we propose:
  - **Semantic awareness:** 2D UV charts should align with semantically meaningful surface 3D parts so that textures designed for a semantic region remain coherent and transferable across 3D shapes

This is semantic-aware

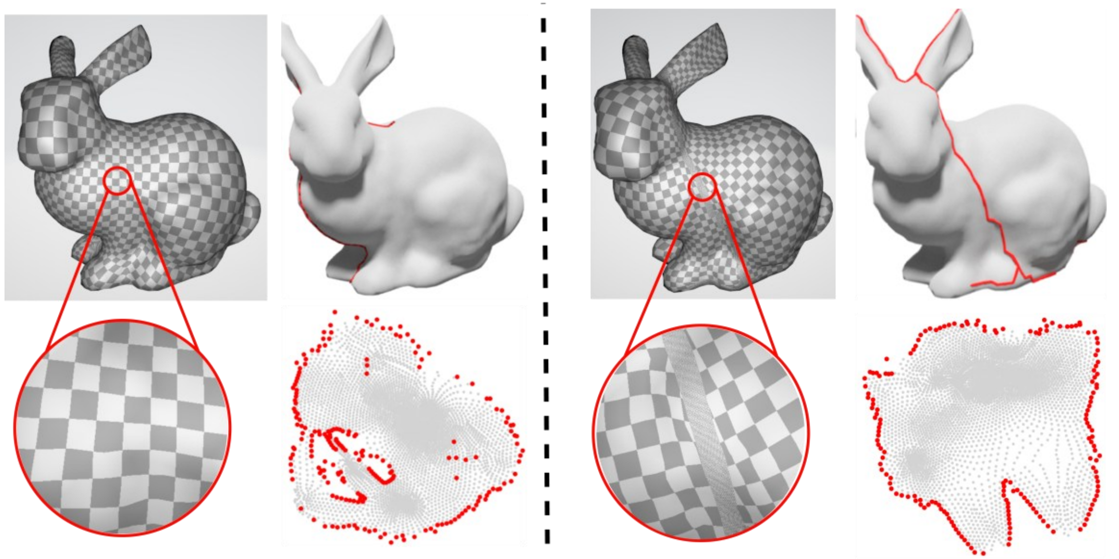


This is NOT semantic-aware

  - **Visibility awareness:** Seams should be placed where they are unlikely to be observed under typical viewpoints and lighting, so that seam artifacts are less perceptible after texturing and rendering

This is visibility-aware

This is NOT visibility-aware



Methodology – Semantic-Aware UV Learning Pipeline

Step 1: Semantic 3D Partitioning:

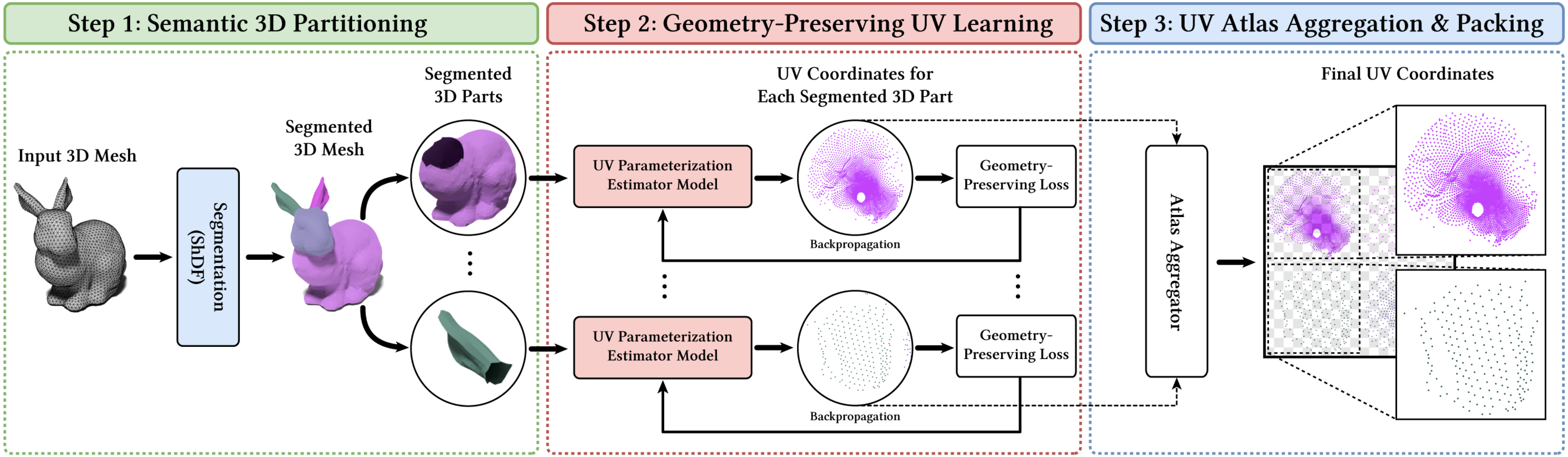
- Partition the input 3D mesh into semantic 3D parts by computing a per-vertex shape diameter function (ShDF) [2]

Step 2: Geometry-Preserving UV Learning:

- Learn a UV parameterization independently for each segmented 3D part using a multi-layer perceptron (MLP) architecture

Step 3: UV Atlas Aggregation and Packing:

- Aggregate and pack these generated UV islands into a unified UV atlas



[2] Shapira, Lior, Ariel Shamir, and Daniel Cohen-Or. "Consistent mesh partitioning and skeletonisation using the shape diameter function." *The Visual Computer* 24.4 (2008): 249-259.

Methodology – Visibility-Aware UV Learning Pipeline

① Compute Per-Vertex Ambient Occlusion on the input mesh

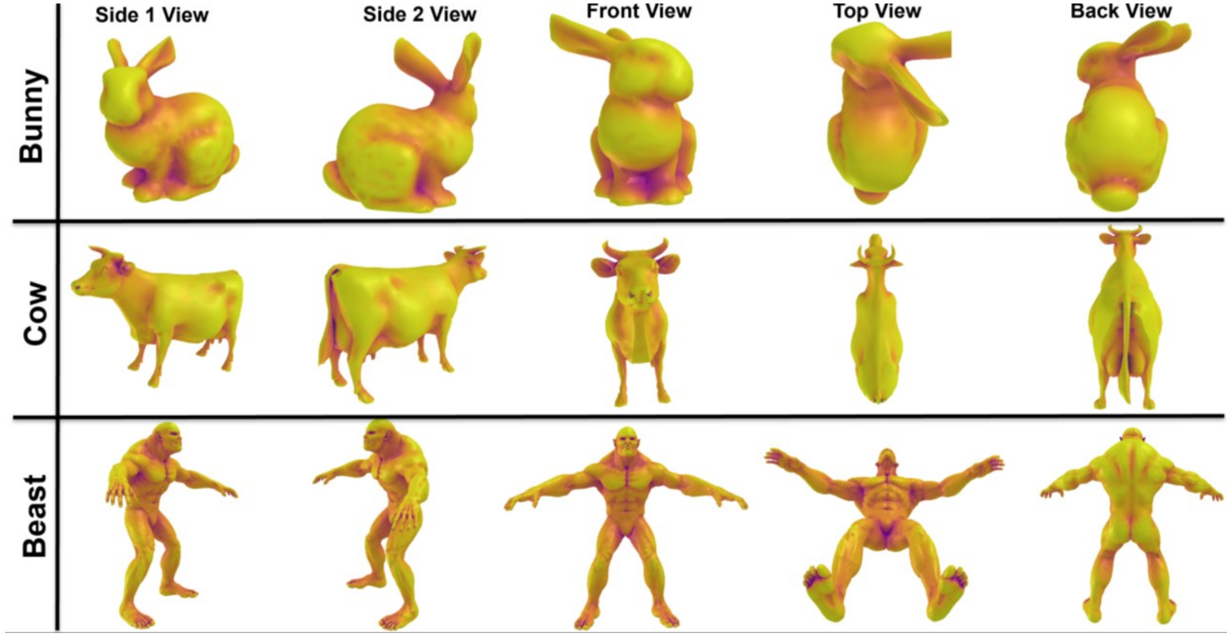
$$V(p,\omega)=\begin{cases}1, & \text{if the ray starting at } p \text{ in direction } \omega \text{ does not intersect the surface (unoccluded)}\\ 0, & \text{otherwise (occluded)}\end{cases}$$

The binary visibility function $V(p,\omega)$ measures whether direction ω is visible from a point p on the mesh surface with unit normal vector $n(p)$

$$AO(p) = \frac{1}{\pi}\int_{\Omega^{+}(p)} V(p,\omega)\,(n(p)\cdot\omega)\,\mathrm{d}\omega.$$

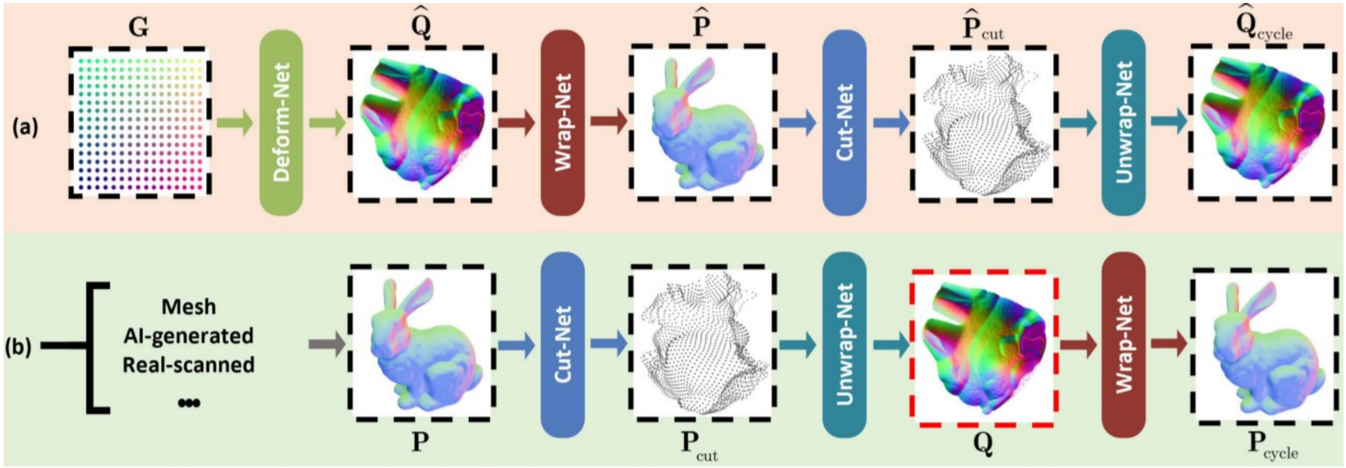
By this convention, $\mathrm{AO}(p)=1$ indicates a fully exposed point (no occlusion) and $\mathrm{AO}(p)=0$ indicates a fully occluded point.

Per-Vertex Ambient Occlusion Visualization



② Apply a base UV-parameterization backbone to obtain candidate UV islands

- Intuitively, the **bi-directional cycle mapping** [3] enforces that projecting a 3D surface patch into 2D UV coordinates and re-projecting those UVs back onto the 3D mesh yields the original data.



[3] Zhao, Yuming, et al. "FlexPara: Flexible Neural Surface Parameterization." *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2025).

AUTODESK

Methodology – Visibility-Aware UV Learning Pipeline

③ Extract UV boundary points corresponding to 3D cutting seam points

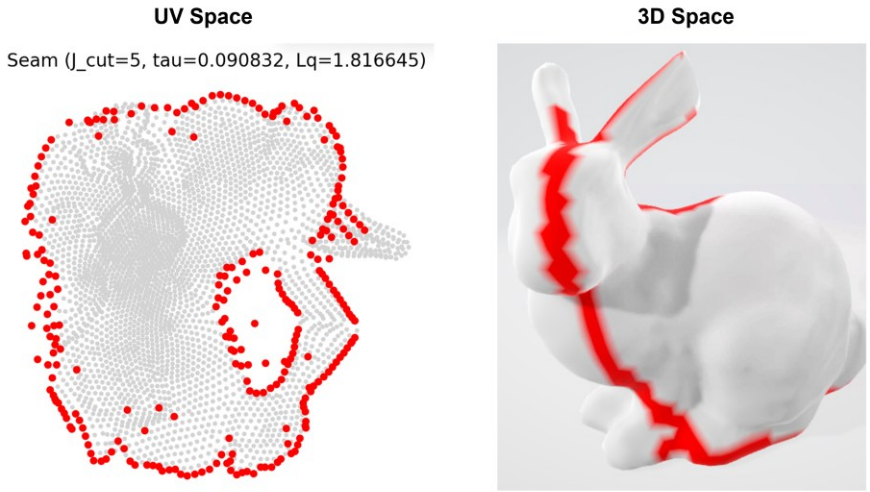
$$\eta_i = \max\left(\|q_i - q_{i,j}\|_2\right)_{j=1}^{N} \longrightarrow s_i = \sigma\big(\beta(\eta_i - \tau)\big)$$

Each point in UV space

N nearest neighbor UV coordinates

Per-vertex seam membership score

A threshold based on the UV domain size ← $\tau = \tau_{\text{scale}}\, L(Q)$ → The side length of bounding square of the UV map



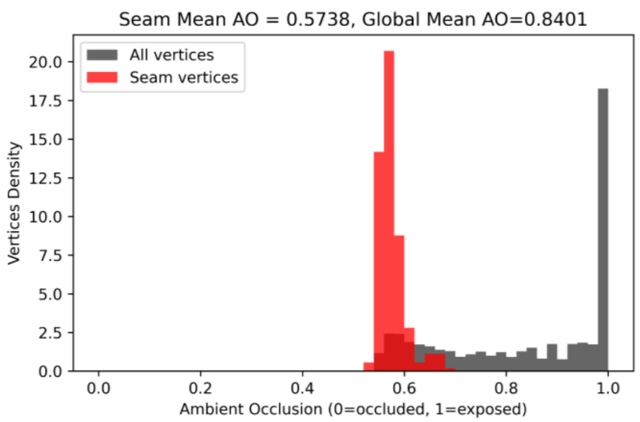
④ Minimize the AO-weighted average over the associated 3D vertices

$$\mathcal{L}_{\text{AO}} = \frac{\sum_i s_i\, AO_i}{\sum_i s_i + \varepsilon} \longrightarrow \mathcal{L}_{\text{vis}}(\theta) = \mathcal{L}_{\text{wrap}} + \mathcal{L}_{\text{cycle}} + \mathcal{L}_{\text{repel}} + \mathcal{L}_{\text{dist}} + \lambda_{\text{vis}}\, \mathcal{L}_{\text{AO}}$$

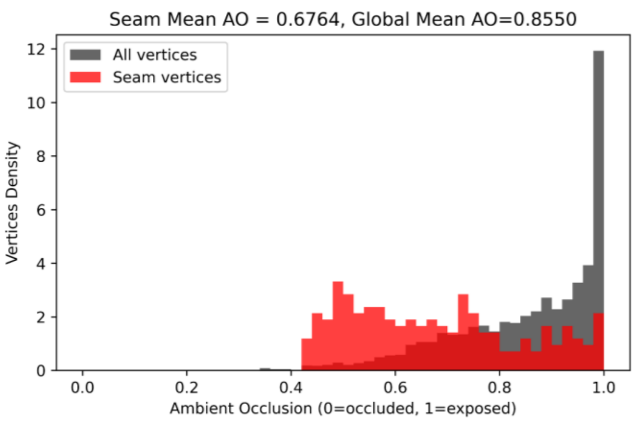
Per-vertex ambient occlusion

A small constant to avoid division by zero

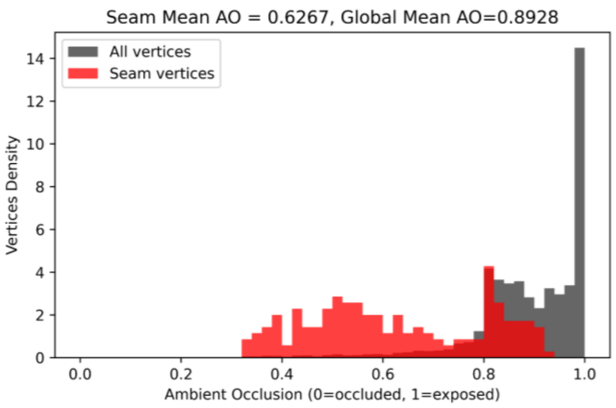
Duckie



Cow



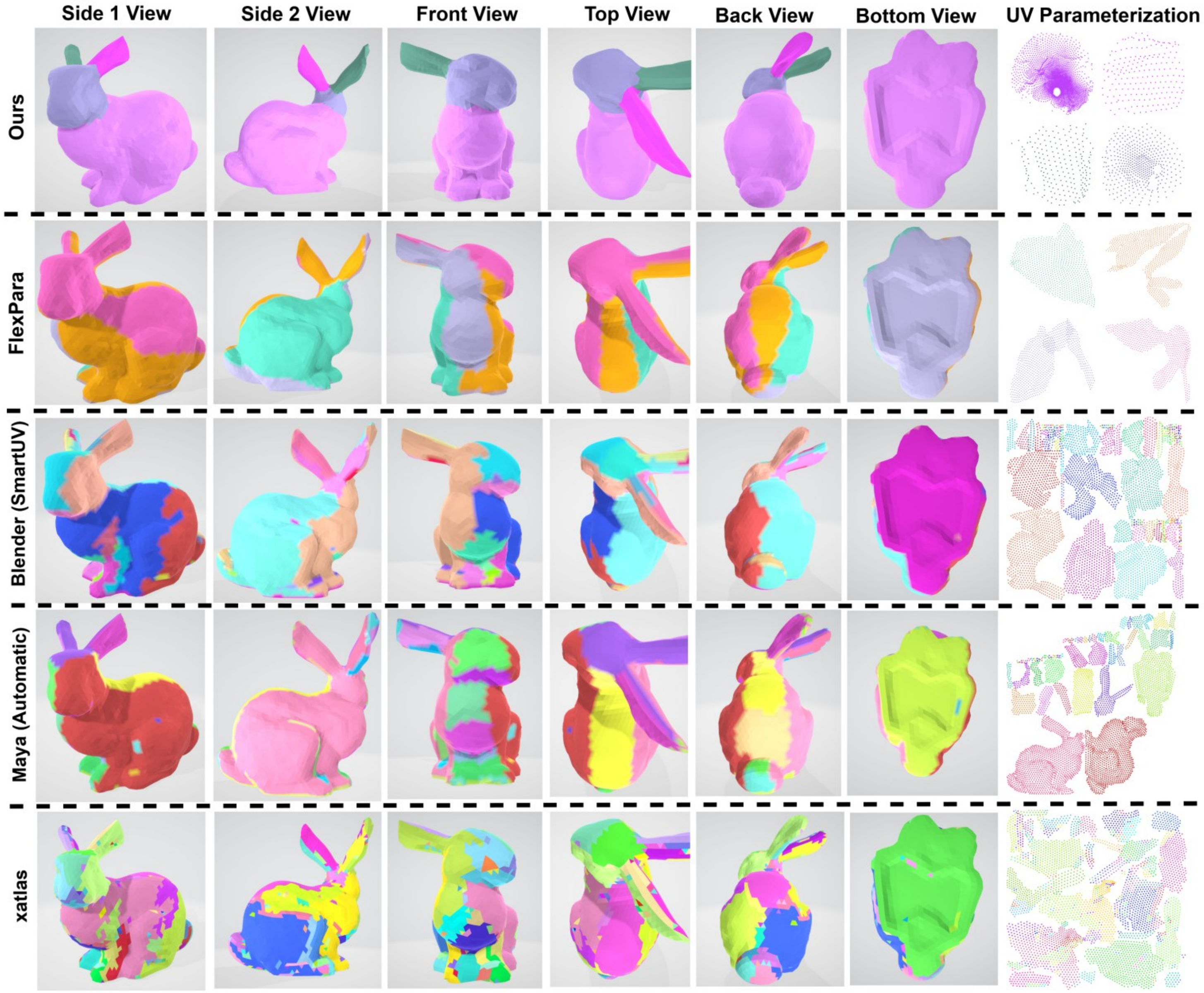
Fish



Comparative Results

Comparison of Semantic-Aware UV Parameterization Results

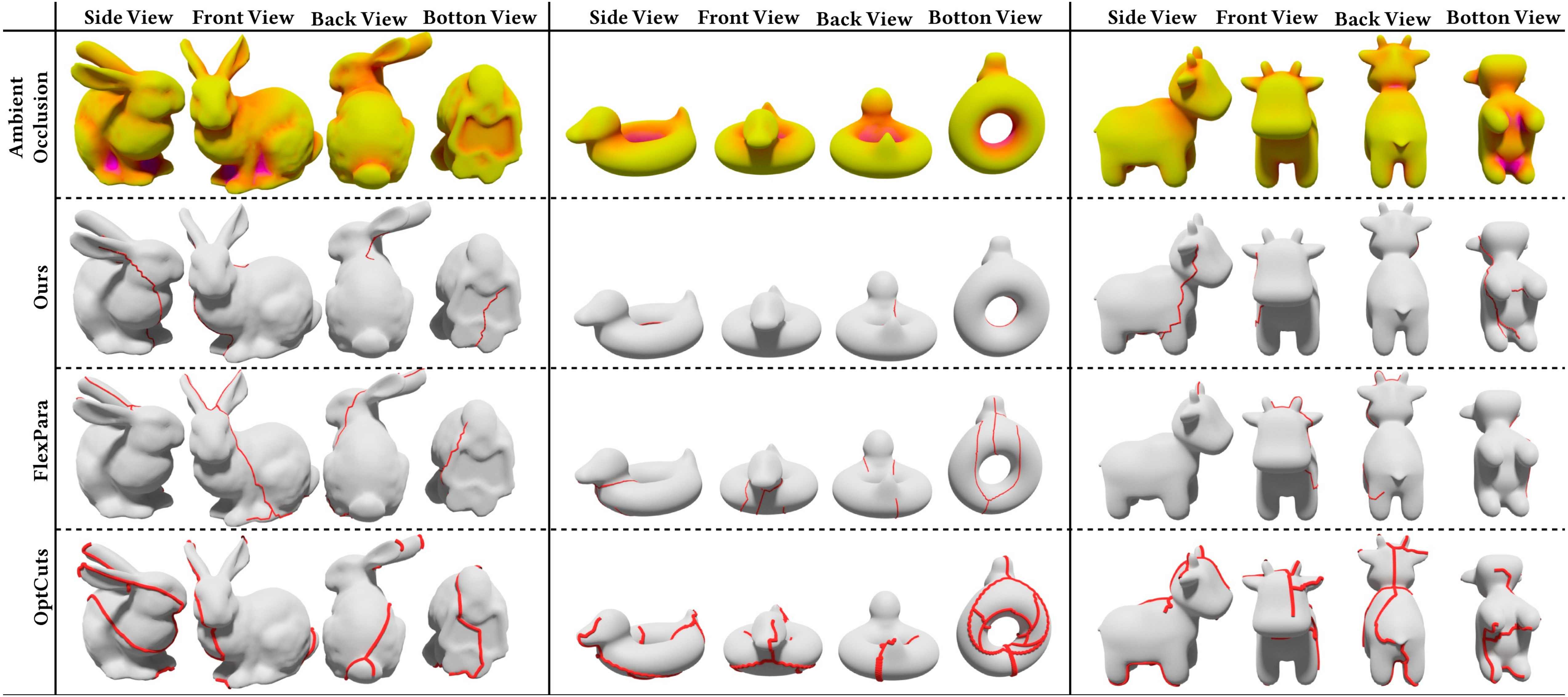
- Produces UV charts that align closely with the mesh's semantic 3D parts (e.g., clearly identifiable rabbit head, ears, body), unlike baseline methods.
- The main advantage: Simplifies texture painting and improves workflow efficiency for UV/layout artists.



Comparative Results

Comparison of Visibility-Aware UV Parameterization Results

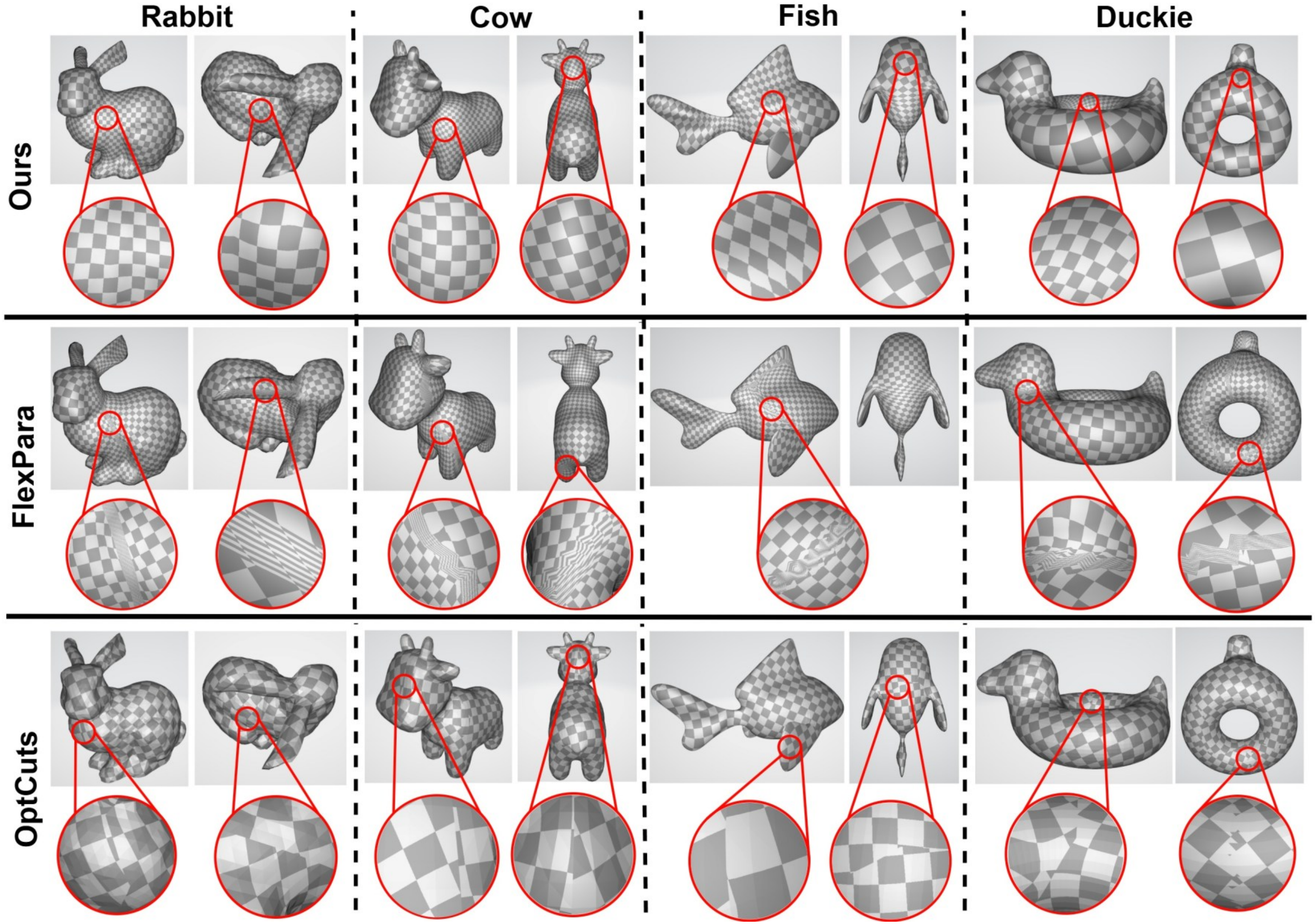
Steer cutting seams toward occluded regions!



Comparative Results

Comparison of Checkerboard Texturing using Different UV Parameterization Methods

- Produces improving texture continuity and consistency from typical viewpoints.



Comparative Results

User Preference Study

- 115 Participants = 70 general + 45 expert participants (UV/layout and modeling artists)

Table 1. User-study with 70 general participants

Method	General User Preference Percentage
Visibility-Awareness Evaluation	
OptCuts [1]	1.43 %
FlexPara (single-Chart) [2]	7.14 %
Ours - Visibility-Aware Param	**91.42 %**
Semantic-Awareness Evaluation	
xatlas [3]	4.28 %
Blender [4]	4.28 %
Autodesk Maya [5]	10 %
FlexPara (multi-Chart) [2]	7.14 %
Ours - Semantic-Aware Param	**74.29** %

Table 2. User-study with 45 expert participants

Method	Expert User Preference Percentage
Visibility-Awareness Evaluation	
OptCuts [1]	1.48 %
FlexPara (single-Chart) [2]	4.81 %
Ours - Visibility-Aware Param	**93.70 %**
Semantic-Awareness Evaluation	
xatlas [3]	2.66 %
Blender [4]	5.33 %
Autodesk Maya [5]	8.44 %
FlexPara (multi-Chart) [2]	3.33 %
Ours - Semantic-Aware Param	**80.22** %

Takeaway: The trade-off between semantic-/visibility-awareness and geometry-preserving properties can be controlled in our pipeline.

More importantly, the semantic and visibility properties are more often valuable in downstream tasks such as mass production, semantic editing, and 3D asset reuse, where consistent UV semantic and low seam visibility significantly reduce manual effort.

Quantitative Comparison

Table 3. Quantitative comparison of visibility-aware UV parameterization

Method	Visibility ↓	Conformality ↑	Equiareality ↑	Time (s) ↓
OptCuts [1]	0.7855	**0.9341**	**0.8934**	240
FlexPara (Single-Chart) [2]	0.8604	0.9097	0.6759	2
Ours - Visibility-Aware Param	**0.6065**	0.9175	0.6093	2
Ours - Semantic+Visibility-Aware Param	0.6534	0.9153	0.6369	15

Table 4. Quantitative comparison of semantic-aware UV parameterization

Method	Semantic Awareness		Conformality ↑	Equiareality ↑	Inference Time (sec) ↓
	Hamming Distance ↓	Rand Index ↑			
xatlas [3]	0.8896	0.7023	**0.9792**	**0.9341**	12
Blender [4]	0.8634	0.7173	0.9289	0.9023	< 1
Maya [5]	0.8615	0.7125	0.9272	0.8746	< 1
FlexPara (Multi-Chart) [2]	0.5980	0.6902	0.9592	0.7606	2
Ours - Semantic-Aware Param	**0.3188**	**0.8151**	0.9123	0.6707	15
Ours - Semantic+Visibility-Aware Param	0.3212	0.8087	0.9153	0.6369	15

[1] Li, Minchen, et al. "Optcuts: Joint optimization of surface cuts and parameterization." *ACM transactions on graphics (TOG)* 37.6 (2018): 1-13.
[2] Zhao, Yuming, et al. "FlexPara: Flexible Neural Surface Parameterization." *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2025).
[3] jpcy. xatlas: Mesh parameterization / uv unwrapping library. https://github.com/jpcy/xatlas, 2025.
[4] Blender Foundation. *Blender - a 3D creation suite (Version 4.5)*. Blender Foundation, 2025. URL: https://www.blender.org.
[5] Autodesk, Inc. *Autodesk Maya (Version 2025.x)*. Autodesk, Inc., 2025. URL: https://www.autodesk.com/products/maya/overview.

Thank you for your attention!