Mesh Deformation Using Learned Priors

Master's Thesis Defense
November 26th, 2024

Committee

Min H. Kim   Sung-Eui Yoon   Minhyuk Sung

Seungwoo Yoo

School of Computing

KAIST

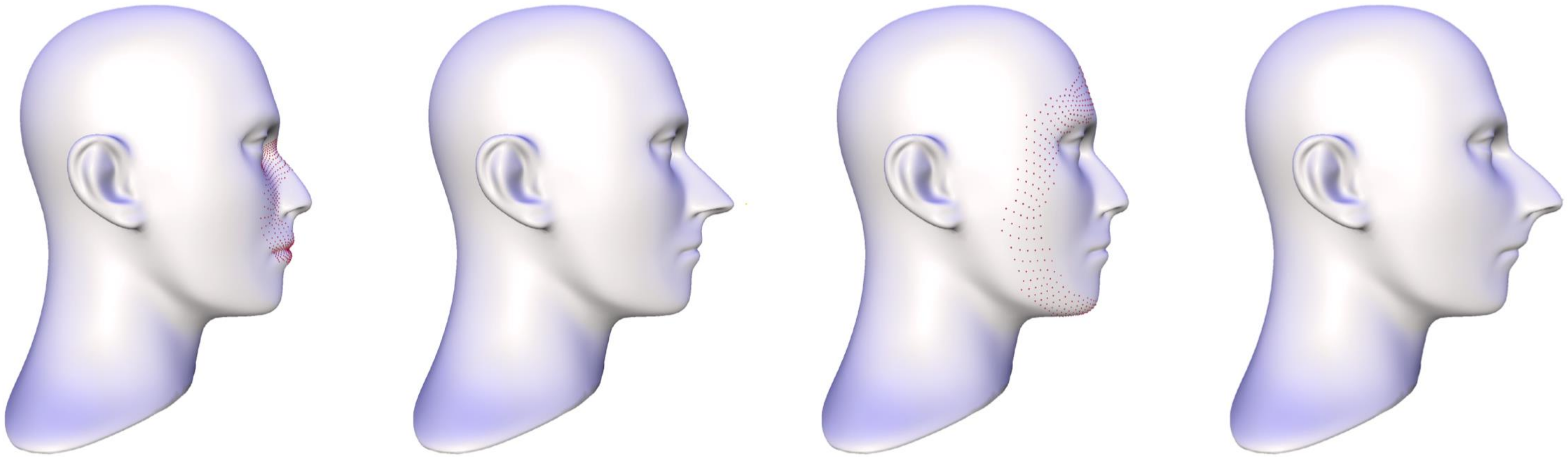
This presentation is based on the following publications:

As-Plausible-As-Possible: Plausibility-Aware Mesh Deformation Using 2D Diffusion Priors
Seungwoo Yoo*, Kunho Kim*, Vladimir G. Kim, Minhyuk Sung
CVPR 2024
* denotes equal contribution.

Neural Pose Representation Learning for Generating and Transferring Non-Rigid Object Poses
Seungwoo Yoo, Juil Koo, Kyeongmin Yeo, Minhyuk Sung
NeurIPS 2024

Mesh Deformation

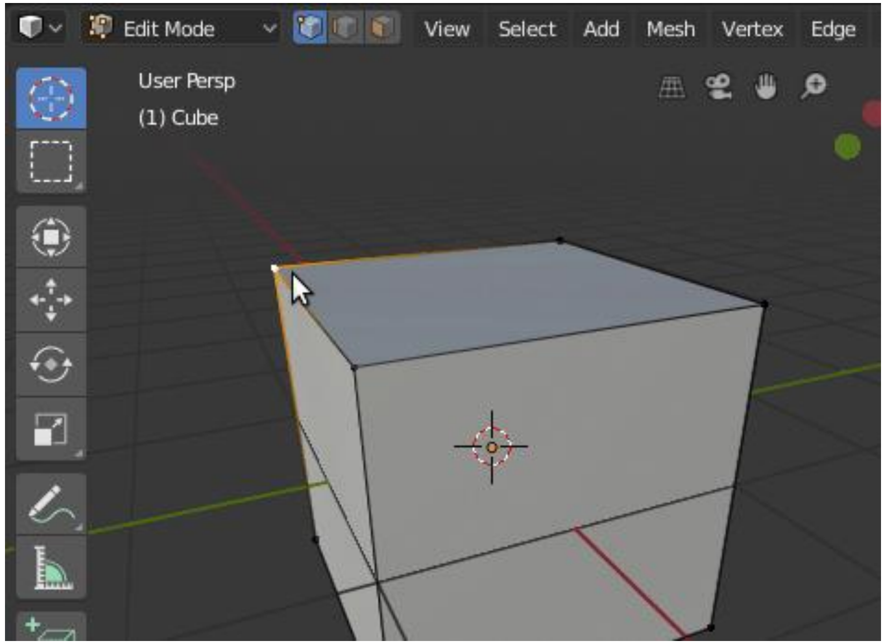
Mesh deformation has been a core research topic in geometry processing and computer graphics.



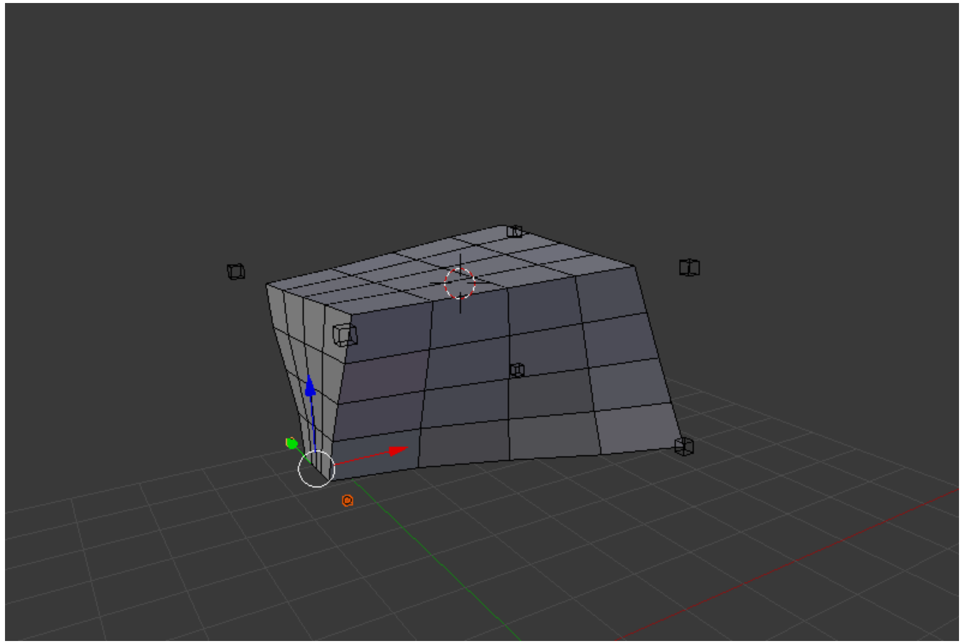
Differential Coordinates for Interactive Mesh Editing,
Lipman *et al.*, SGP 2004

Mesh Deformation

Various techniques have been proposed and integrated into graphics pipelines to offer intuitive control for human artists.



Vertex-Wise Displacement [1]

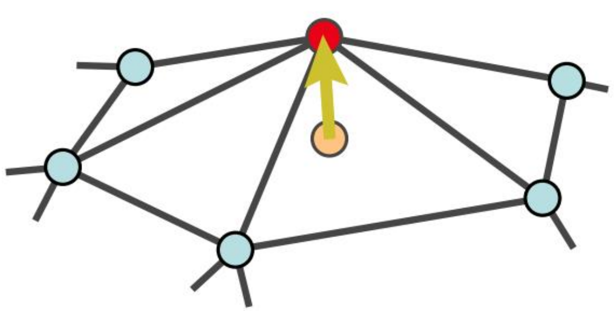


Cage-Based Deformation [2]

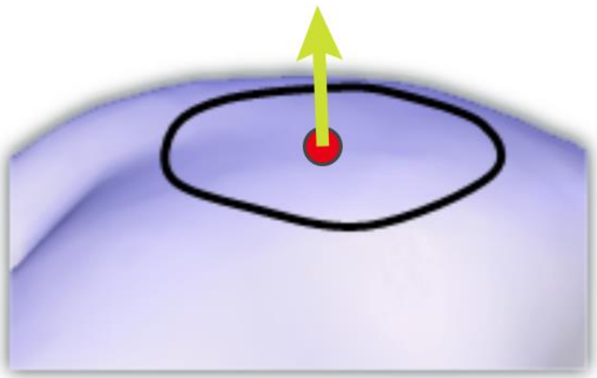
[1] https://blender.stackexchange.com/questions/120157/blender-2-8-where-is-the-axes-tool-for-moving-vertices-visually
[2] https://blendermarket.com/products/deform-pro

Mesh Deformation Using Geometric Priors

The success of existing techniques is largely attributed to leveraging **geometric priors**.

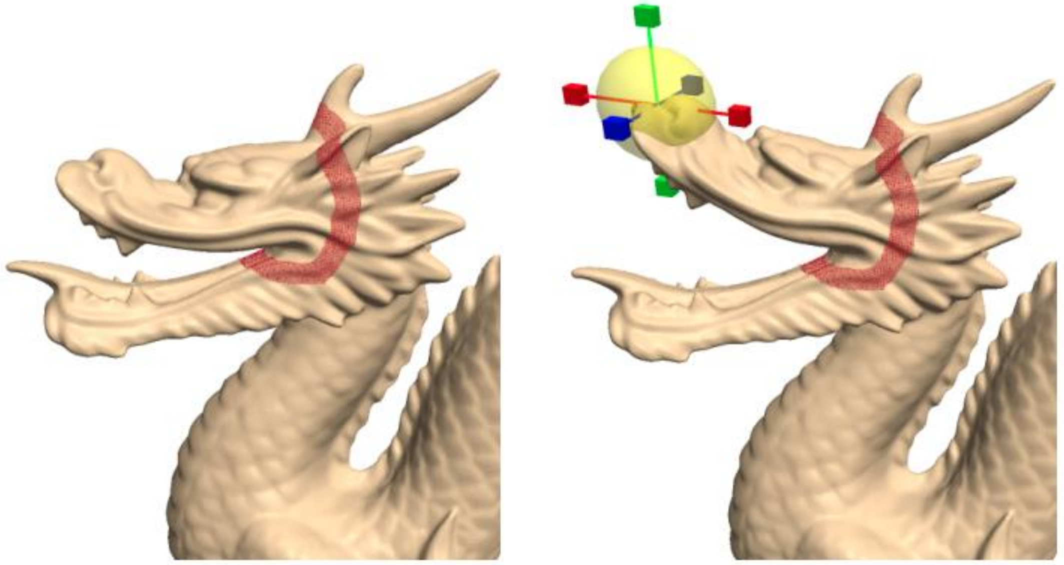


$$\delta_i = \frac{1}{d_i} \sum_{j \in N(i)} (\mathbf{v}_i - \mathbf{v}_j)$$



$$\frac{1}{|\gamma|} \int_{\mathbf{v} \in \gamma} (\mathbf{v}_i - \mathbf{v}) dl(\mathbf{v})$$

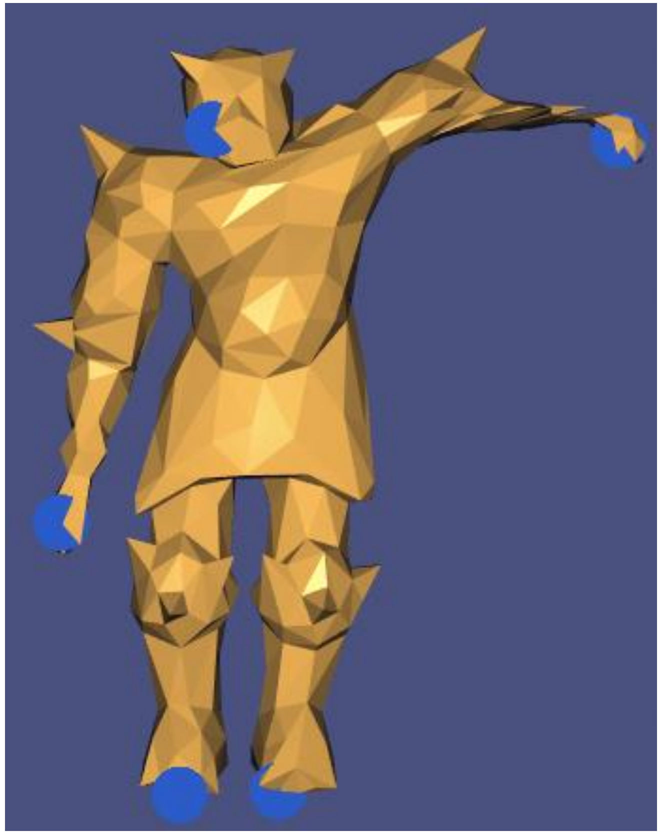
Laplacian Mesh Processing,
Sorkine, Eurographics 2005



Laplacian Surface Editing,
Sorkine *et al.*, SGP 2004

Mesh Deformation Using Geometric Priors

Handcrafted geometric priors may fail to capture behaviors of surfaces under deformation, requiring manual adjustments.

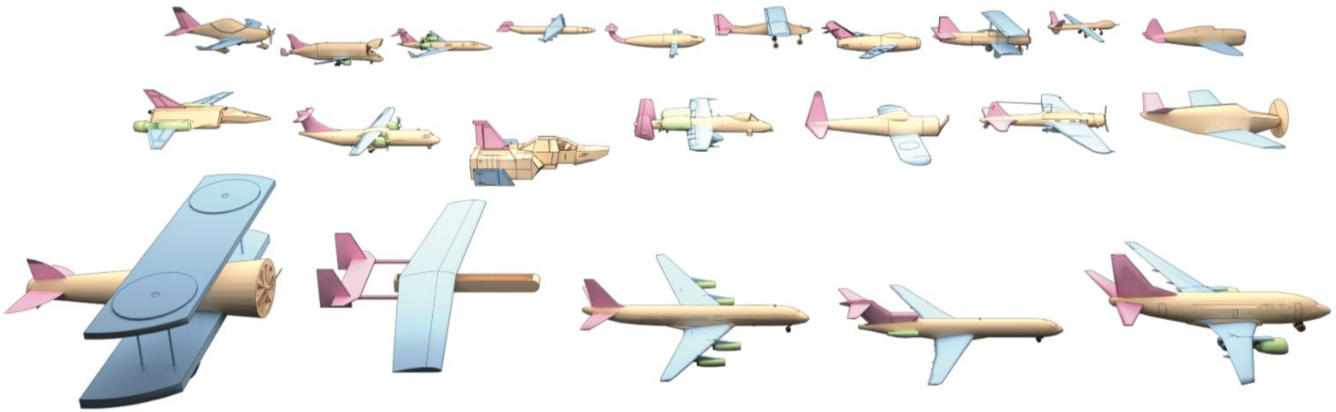


Geometry Processing – Deformation, Jacobson

Mesh Deformation Using Learned Priors

Recent approaches propose to **learn deformation priors** from **large-scale datasets** consisting of hundreds of shape exemplars.

ShapeNet Chair,
Chang *et al.*, arXiv Preprint 2015

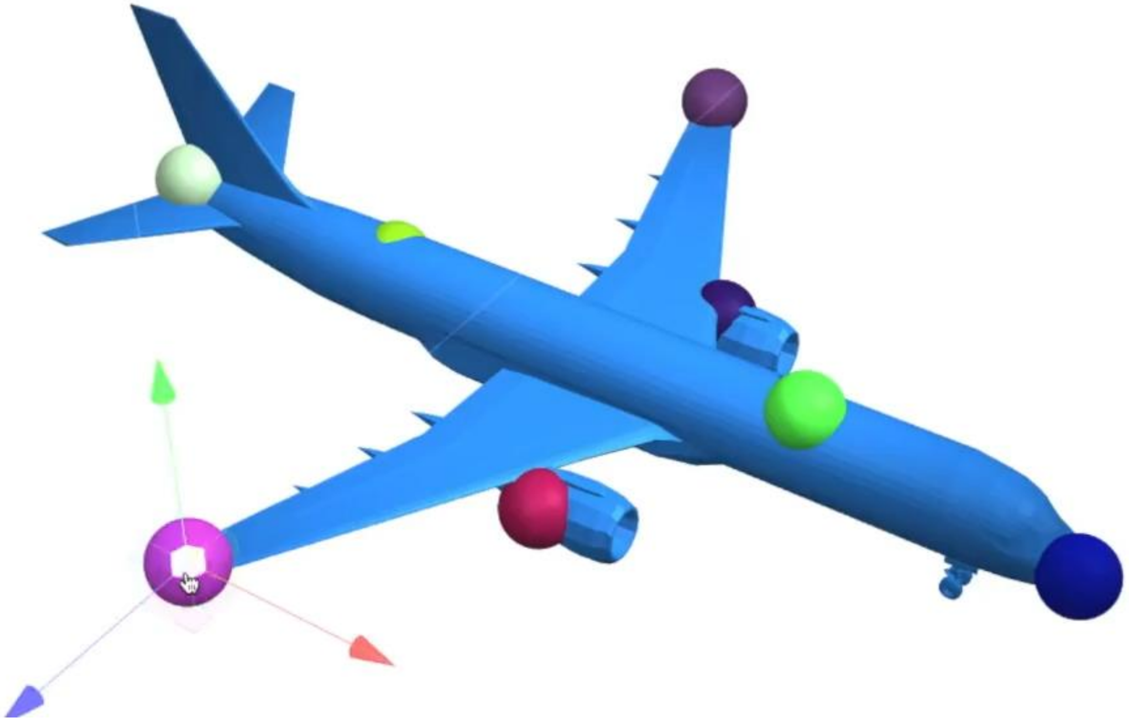


Machine Learning for 3D Data,
Stanford CS468

Mesh Deformation Using Learned Priors

Learned priors can better capture categoric-specific shape variations and reveal easy-to-use handles for making deformations.

DeepMetaHandles,
Liu *et al.*, CVPR 2021



KeypointDeformer,
Jakab *et al.*, CVPR 2021

Mesh Deformation Using Learned Priors

Collecting 3D shapes is challenging, and existing datasets are restricted to a limited number of shape categories.

ShapeNetCore Dataset
55 Categories, ~60K Examples

LAION-5B Text-to-Image Dataset
5.8B Examples

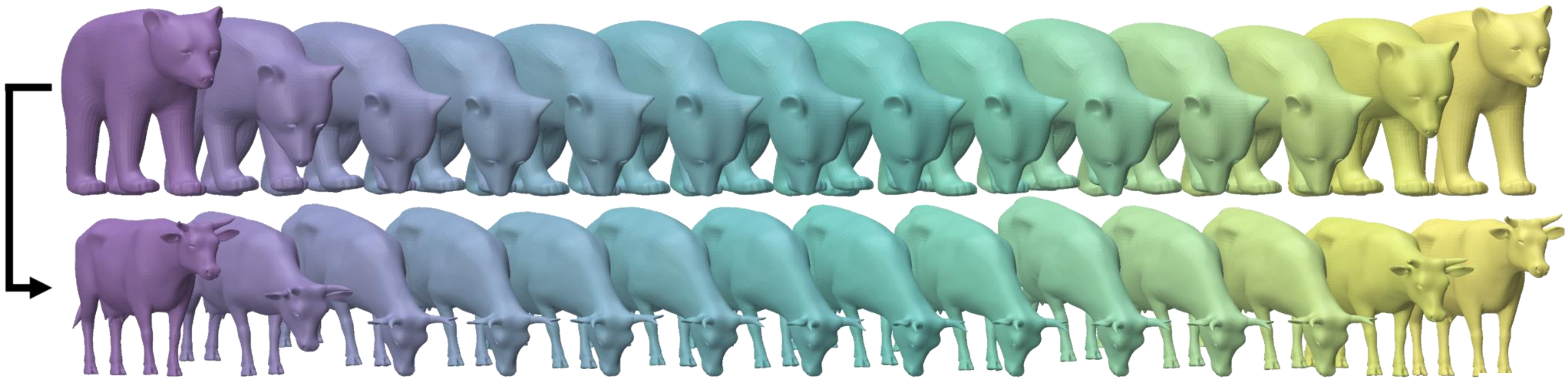
Mesh Deformation Using Learned Priors

This issue becomes more significant for shapes with minimal or no variation, making it difficult to learn such priors.

Adobe Mixamo Dataset,
Adobe

Mesh Deformation

Given a single shape and its poses, how can we transfer those poses to another shape?



Neural Pose Representation, NeurIPS 2024

Seungwoo Yoo, Juil Koo, Kyeongmin Yeo, Minhyuk Sung

Representation Learning for Non-Rigid Object Pose Transfer

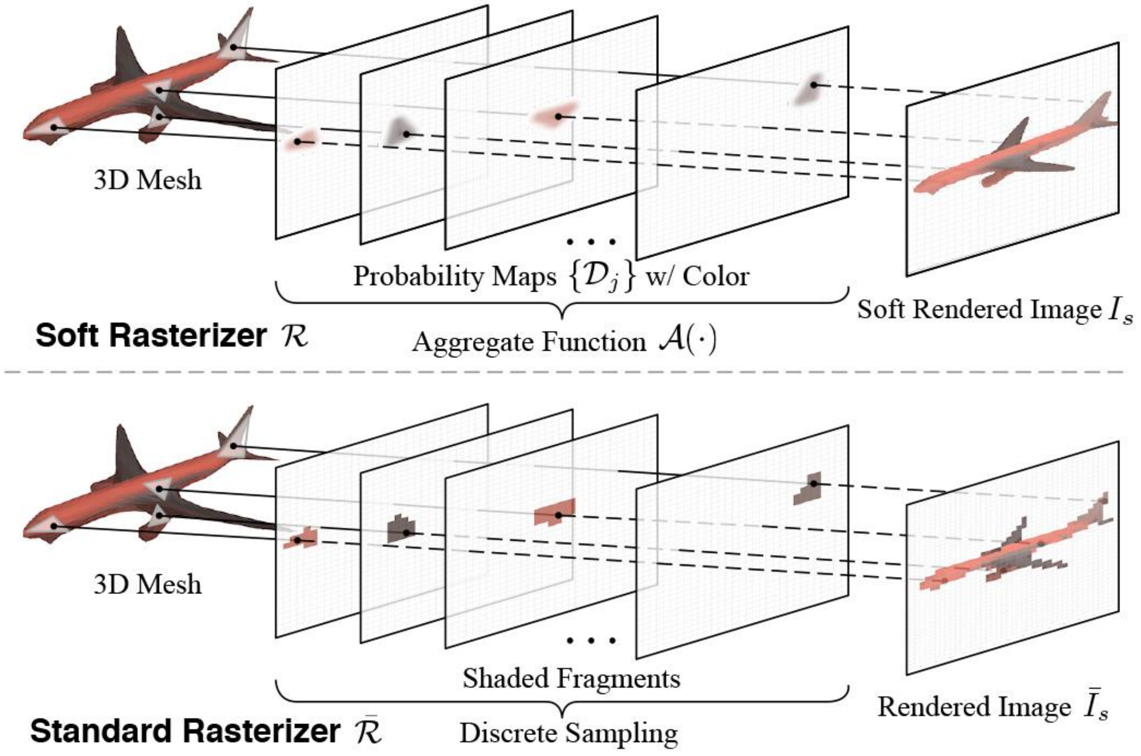
Mesh Deformation

Collecting deformation examples for arbitrary 3D shapes and categories is even infeasible.

[The BEHAVIOR Dataset of Objects](), Srivastava, CoRL 2021

Mesh Deformation

Differentiable renderers bridge images and meshes, enabling the modification of mesh properties through backpropagation.



Soft Rasterizer,
Liu *et al.*, ICCV 2019

Large Steps in Inverse Rendering of Geometry,
Nicolet *et al.*, ACM ToG 2021

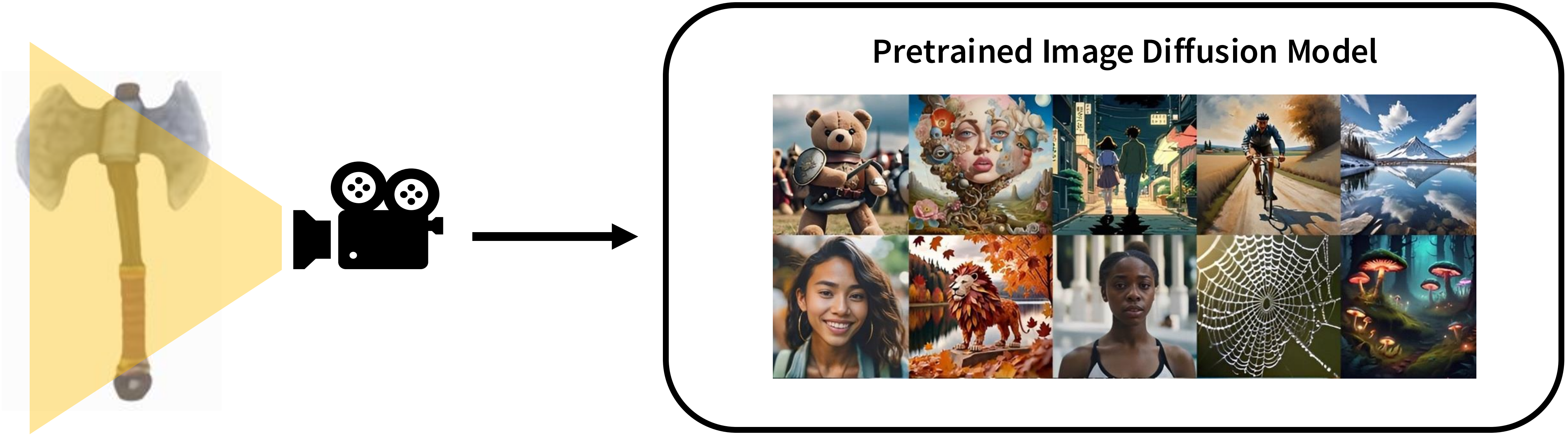
Mesh Deformation

SotA image generators are trained on **internet-scale datasets**, achieving impressive **generative capabilities**.

Stable Diffusion 3,
Esser *et al.*, ICML 2024

Mesh Deformation

With only 2D generative models trained on large datasets, how can we distill their prior knowledge for mesh deformation?



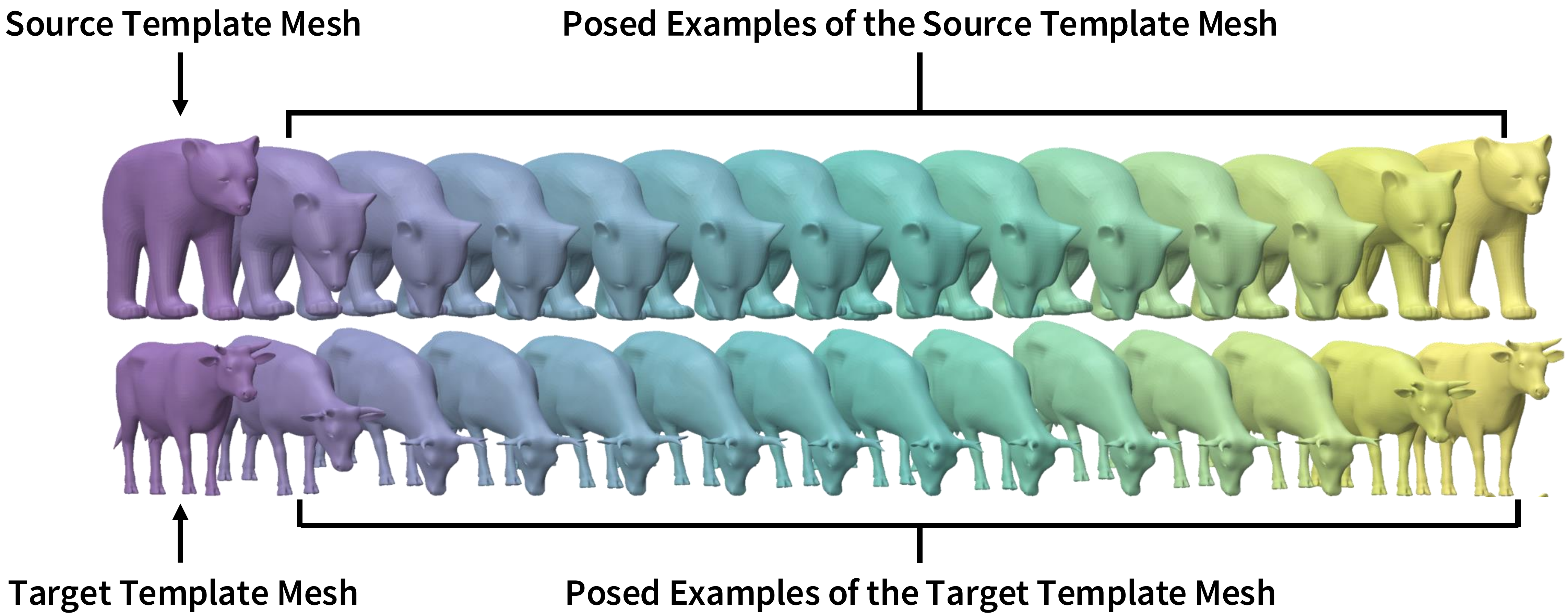
As-Plausible-As-Possible (APAP), CVPR 2024

Seungwoo Yoo*, Kunho Kim*, Vladimir G. Kim, Minhyuk Sung

Plausibility-Aware Mesh Deformation Using 2D Diffusion Priors

Representation Learning for Non-Rigid Object Pose Transfer

Problem Definition



Related Work

This problem has been extensively studied in computer graphics.
Existing works can be summarized into three categories:

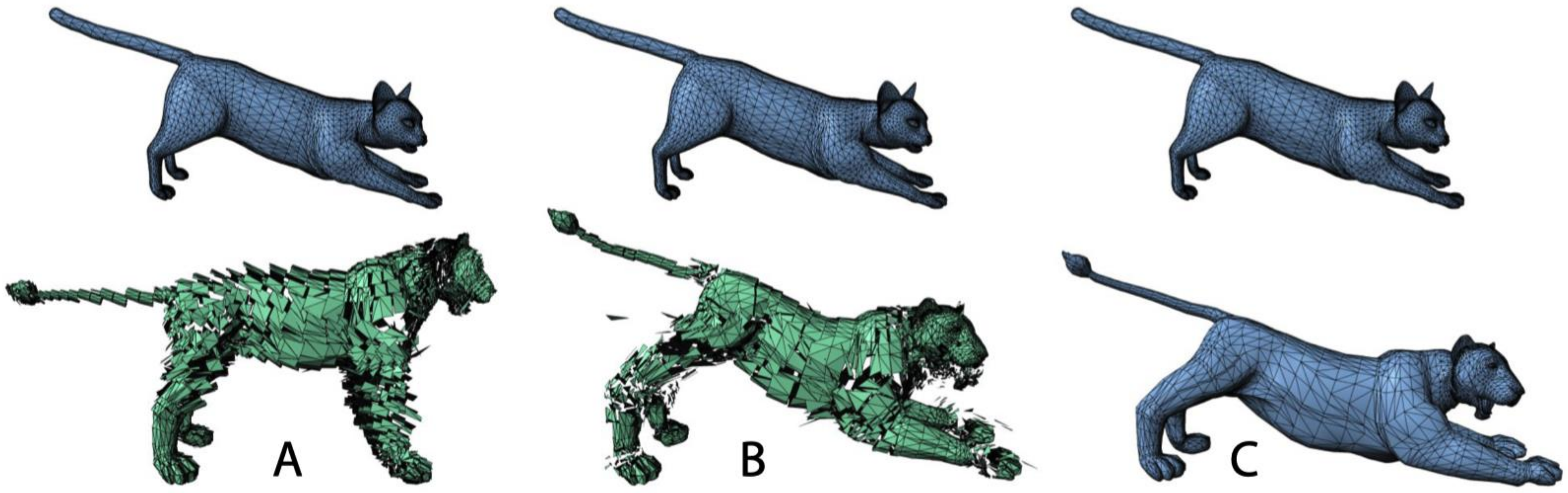
1. Pose Transfer Using Pointwise Correspondences

2. Pose Transfer Using Skeletons

3. Pose Transfer Using Learned Parameterizations

Pose Transfer Using Pointwise Correspondences

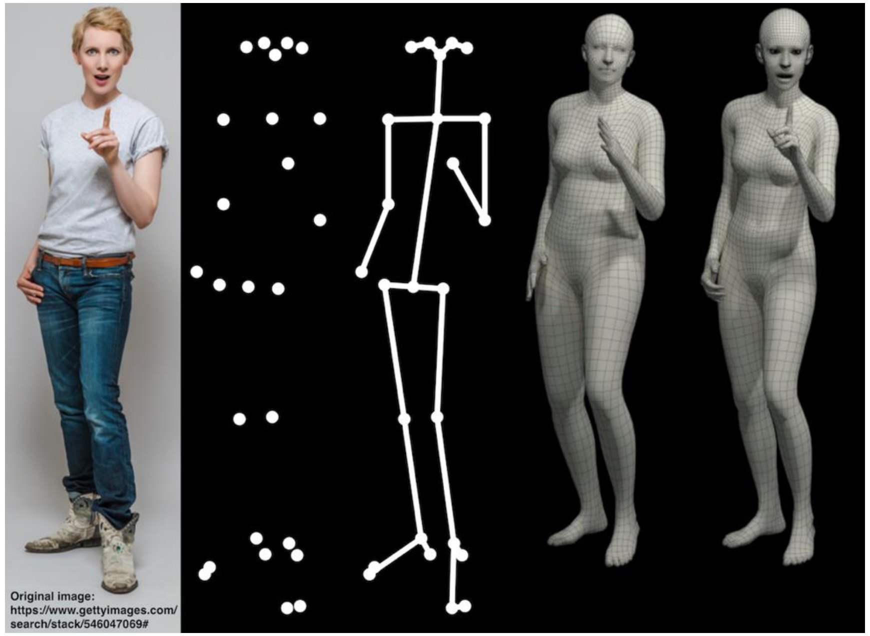
(-) Requires manual pointwise correspondence annotations.



Deformation Transfer for Triangle Meshes,
Sumner and Popovic, SIGGRAPH 2004

Pose Transfer Using Skeletons

(-) Requires a shared skeletal structure.

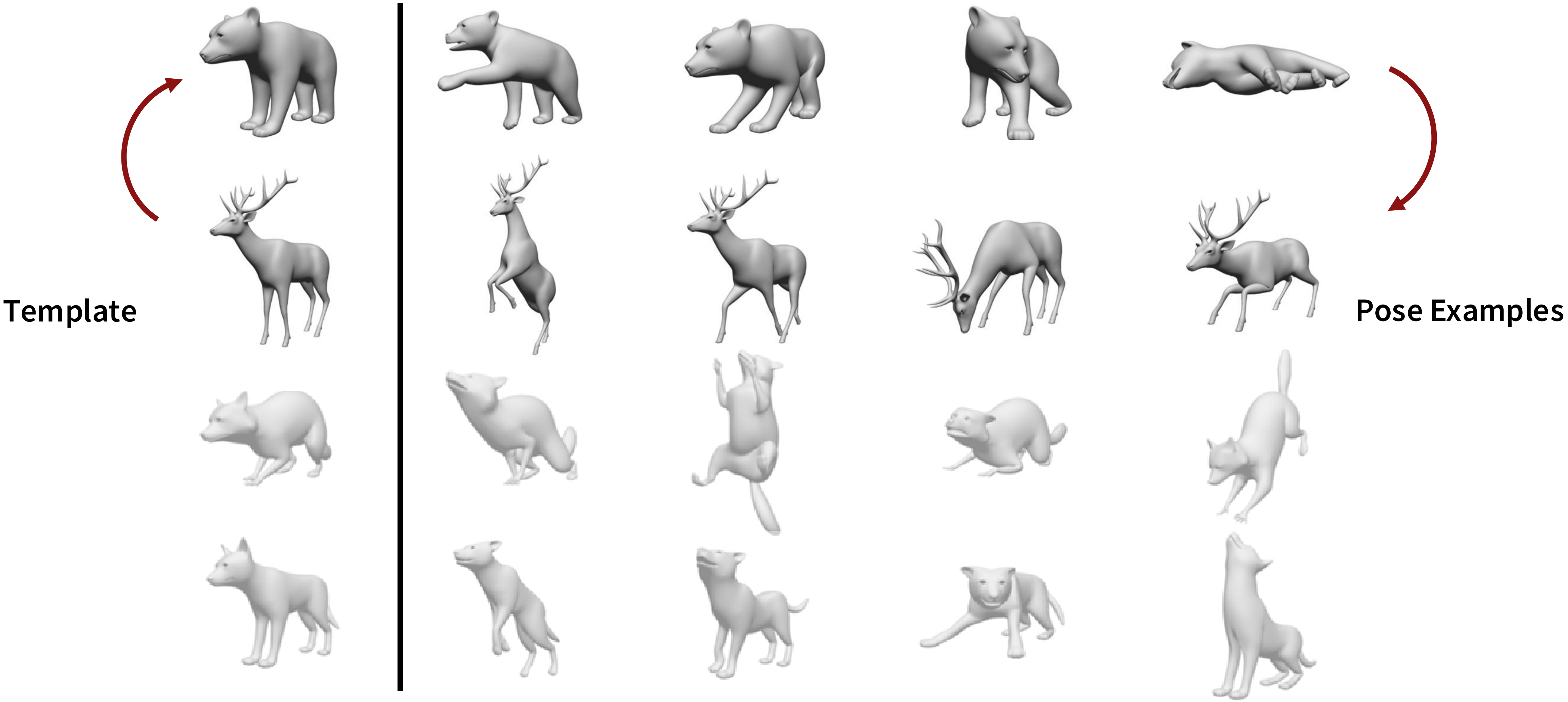


SMPL-X,
Pavlakos *et al.*, CVPR 2019

DeformingThings4D Dataset,
Li *et al.*, ICCV 2021

Pose Transfer Using Learned Parameterizations

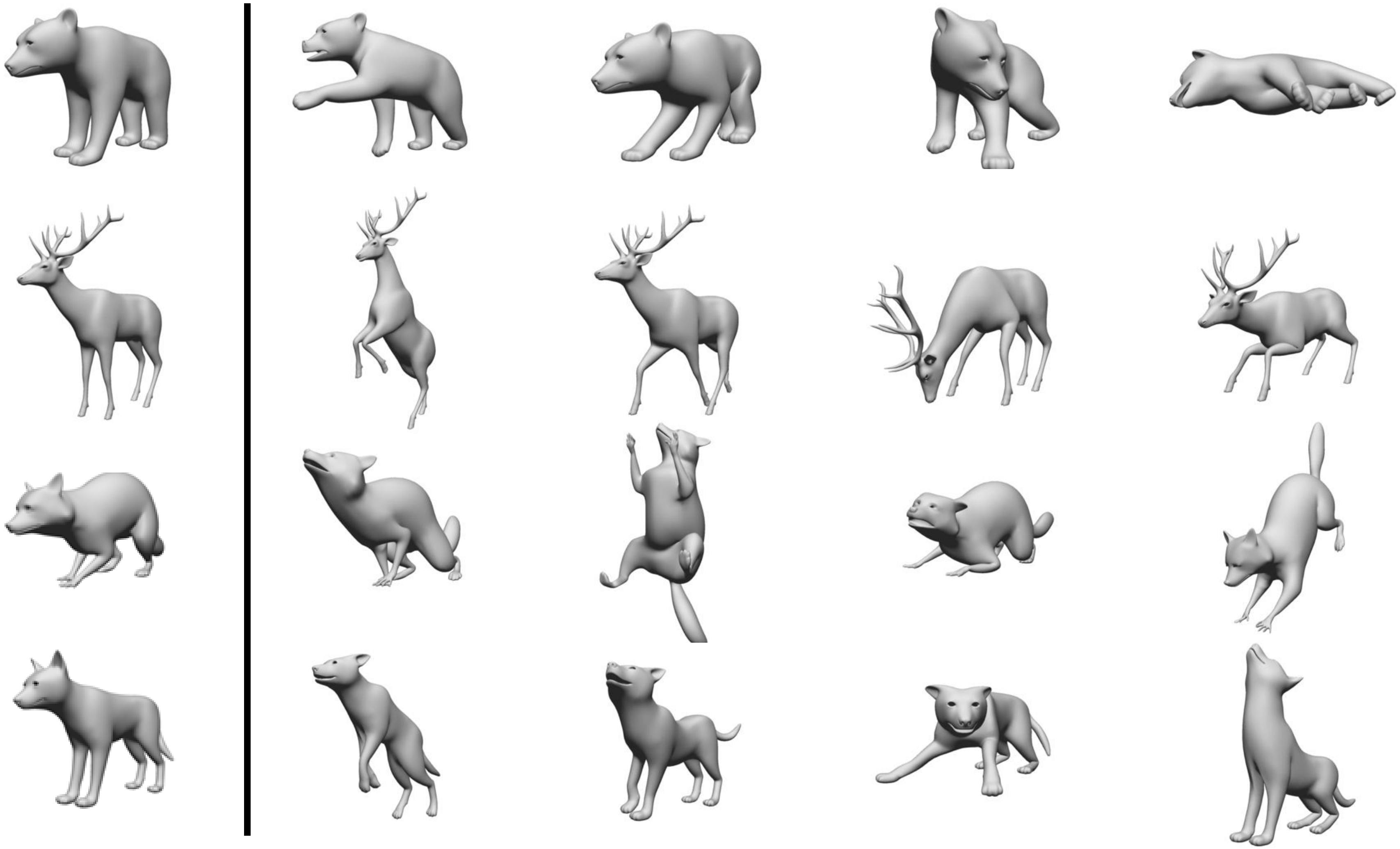
(-) Requires the target template and its posed examples.



Pose Transfer Using Learned Parameterizations

(-) Requires various target templates and examples for generalization.

Template

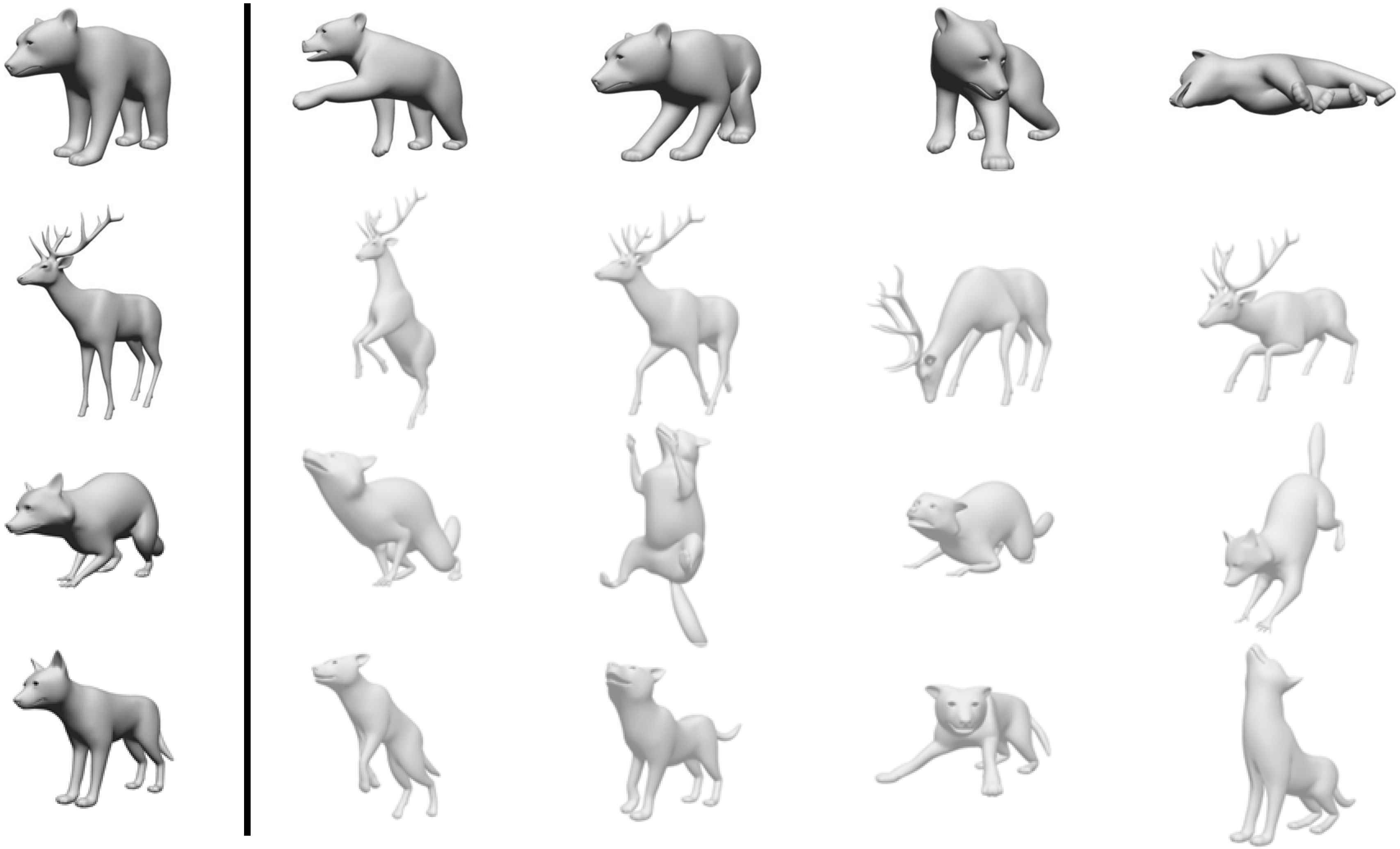


Pose Examples

Pose Transfer Using Learned Parameterizations

(-) Still requires various target templates.

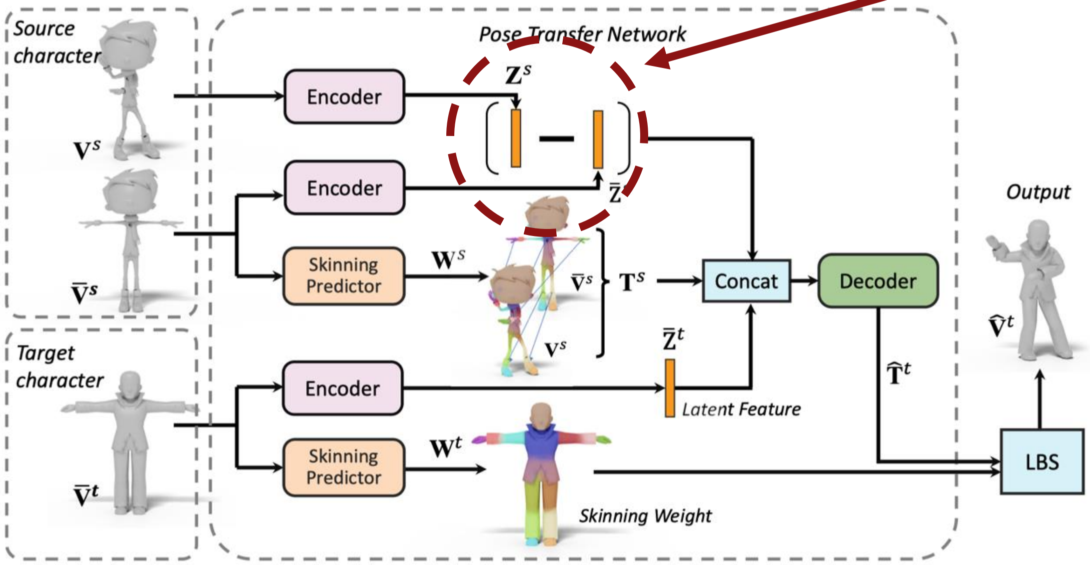
Template



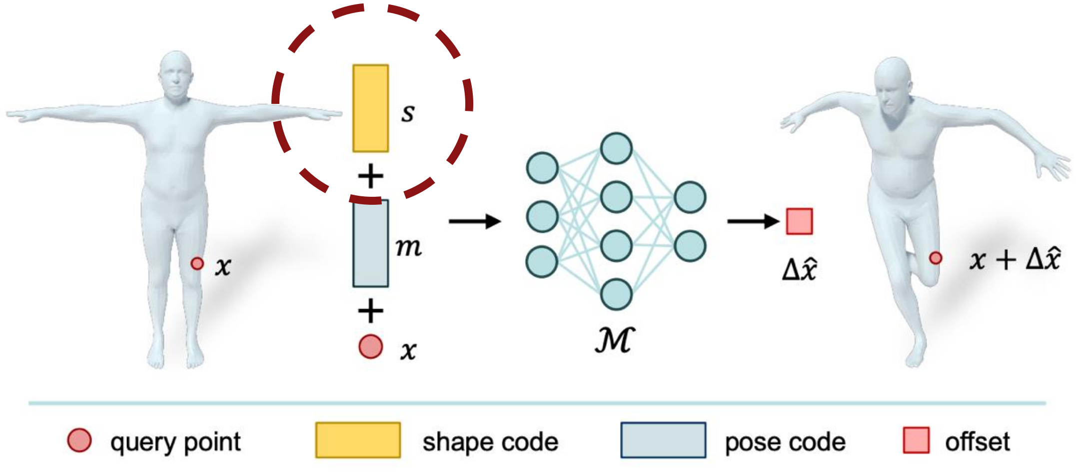
Pose Examples

Pose Transfer Using Learned Parameterizations

Encoding pose examples into abstract **global embeddings** is limiting the generalization capability.



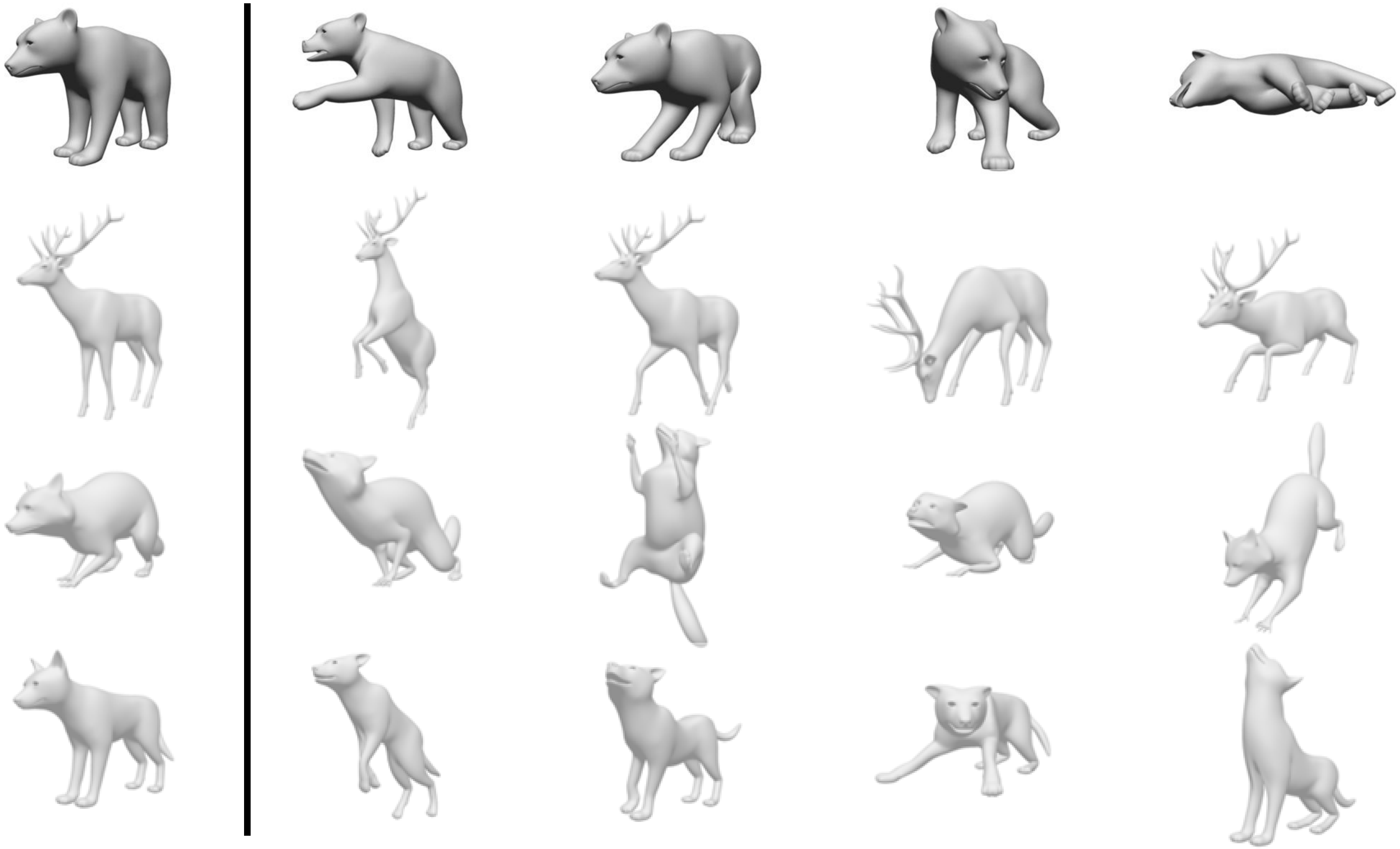
Skeleton-Free Pose Transfer for Stylized 3D Characters,
Liao *et al.*, ECCV 2022



Zero-shot Pose Transfer, Wang *et al.*, CVPR 2023

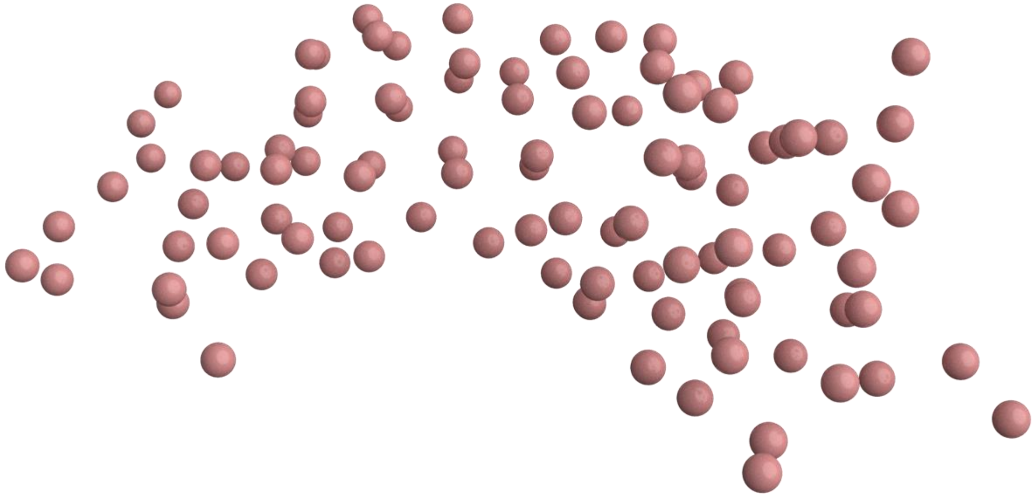
Pose Transfer Using Learned Parameterizations

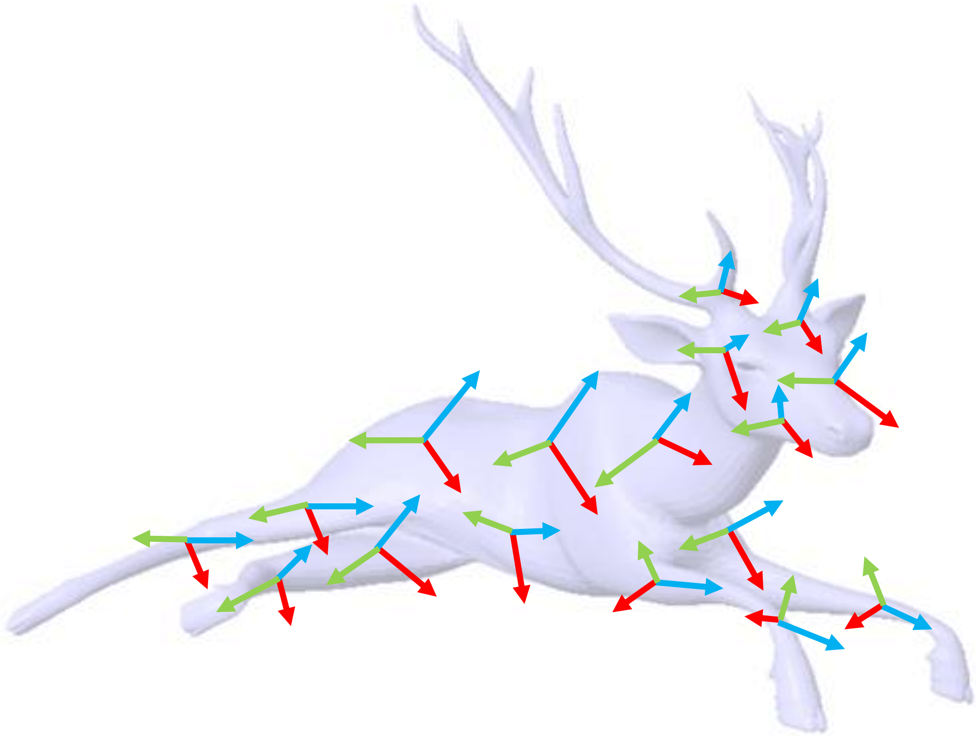
(+) Only requires one template and its posed examples.



Key Idea

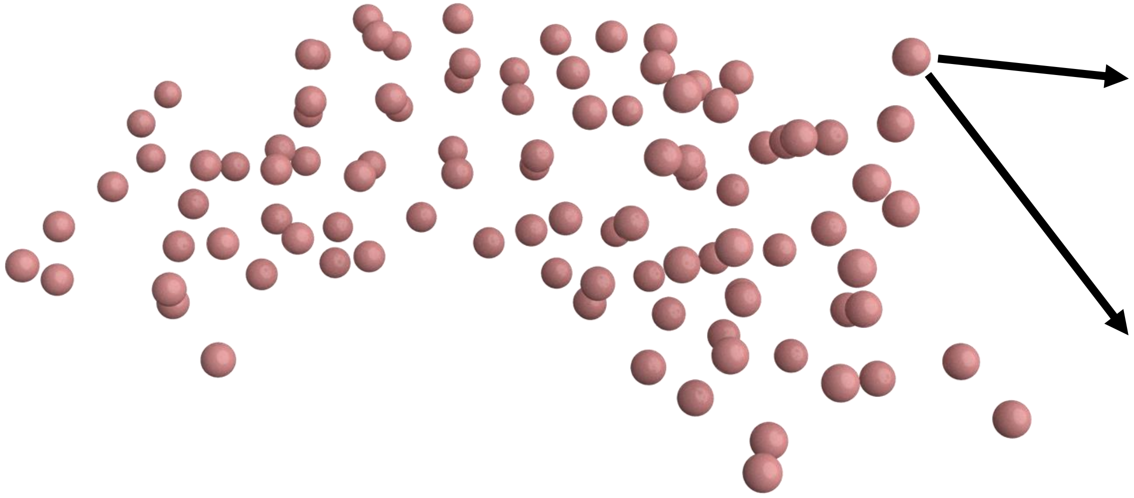
Extract poses as a keypoint-based neural representation and transfer them by predicting local surface transformations.





Keypoint-based Neural Pose Representation

Our pose representation combines **keypoints in 3D space** and **per-point neural features** to capture both shape extrinsic and intrinsic.

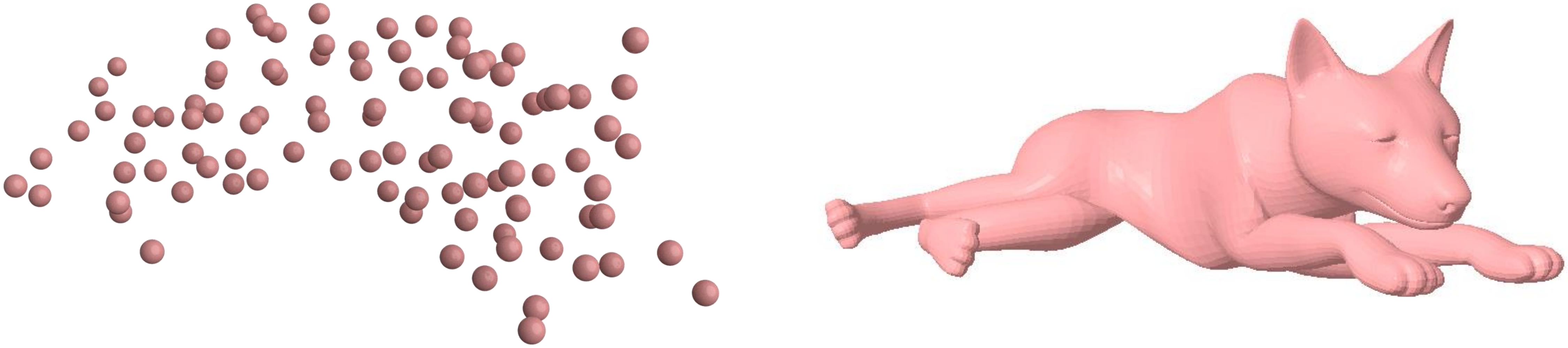


3D keypoint obtained via farthest point sampling.

Neural feature encoded from the mesh vertex coordinates.

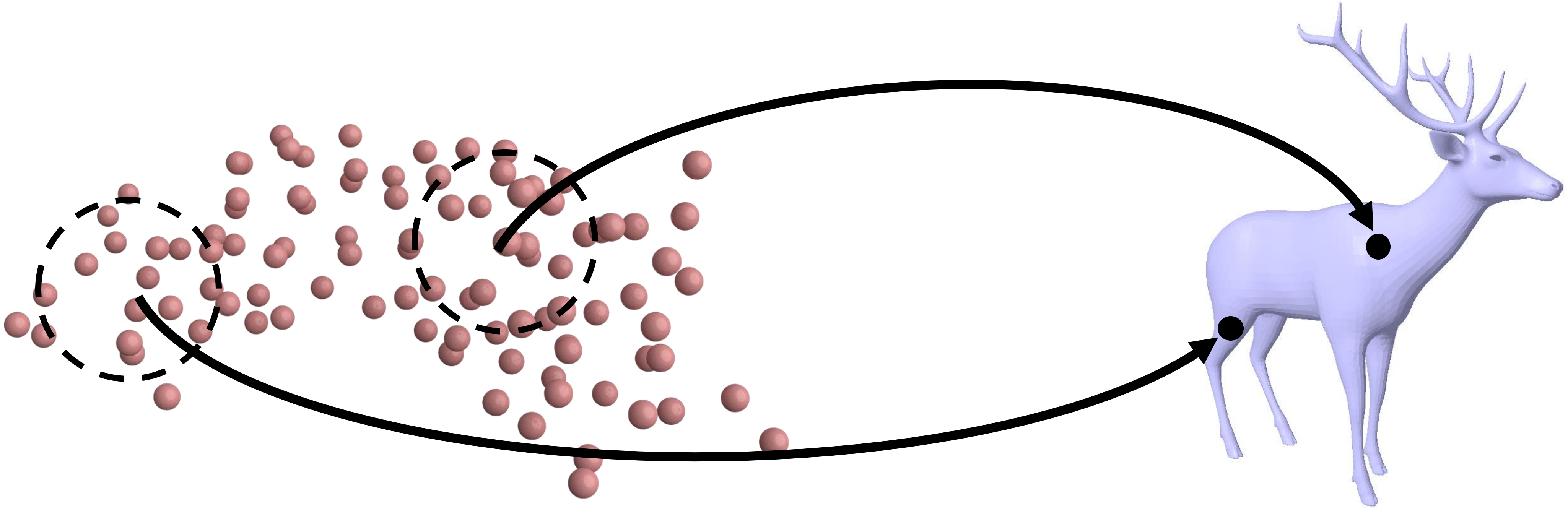
Keypoint-based Neural Pose Representation

The 3D keypoints in our representation explicitly represent **a rough silhouette** of the given pose example.



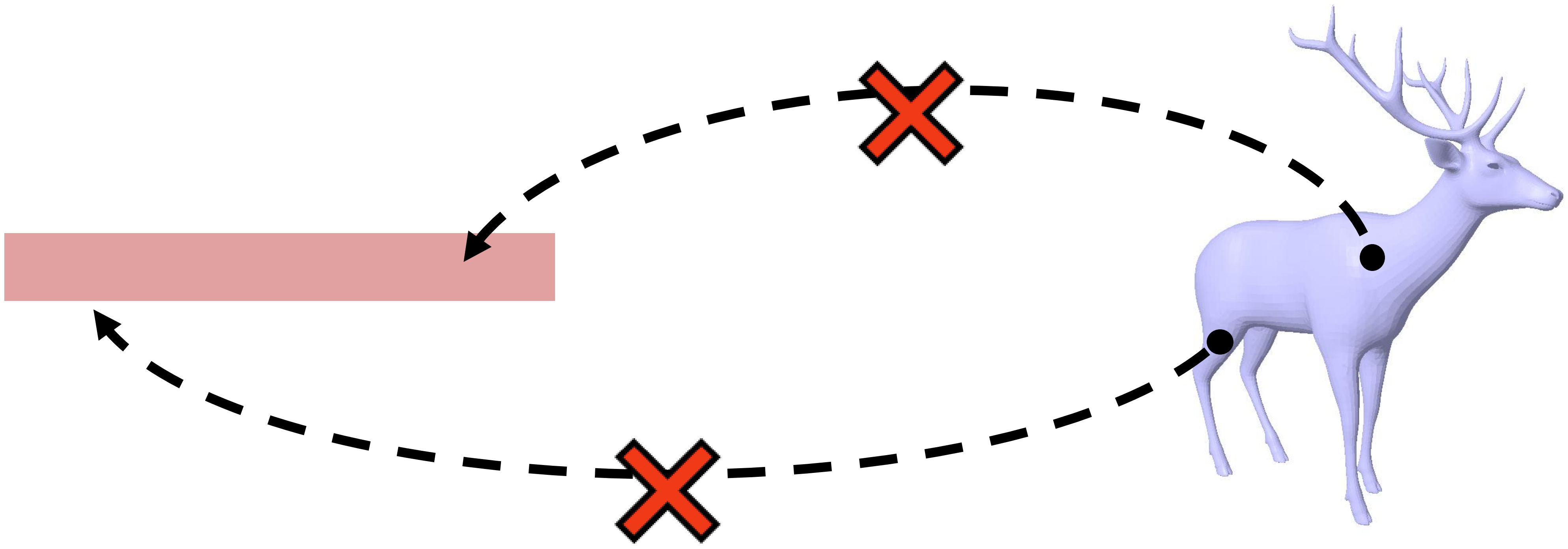
Keypoint-based Neural Pose Representation

They enable **distance-based queries**, facilitating neural feature aggregation at adjacent and relevant regions.



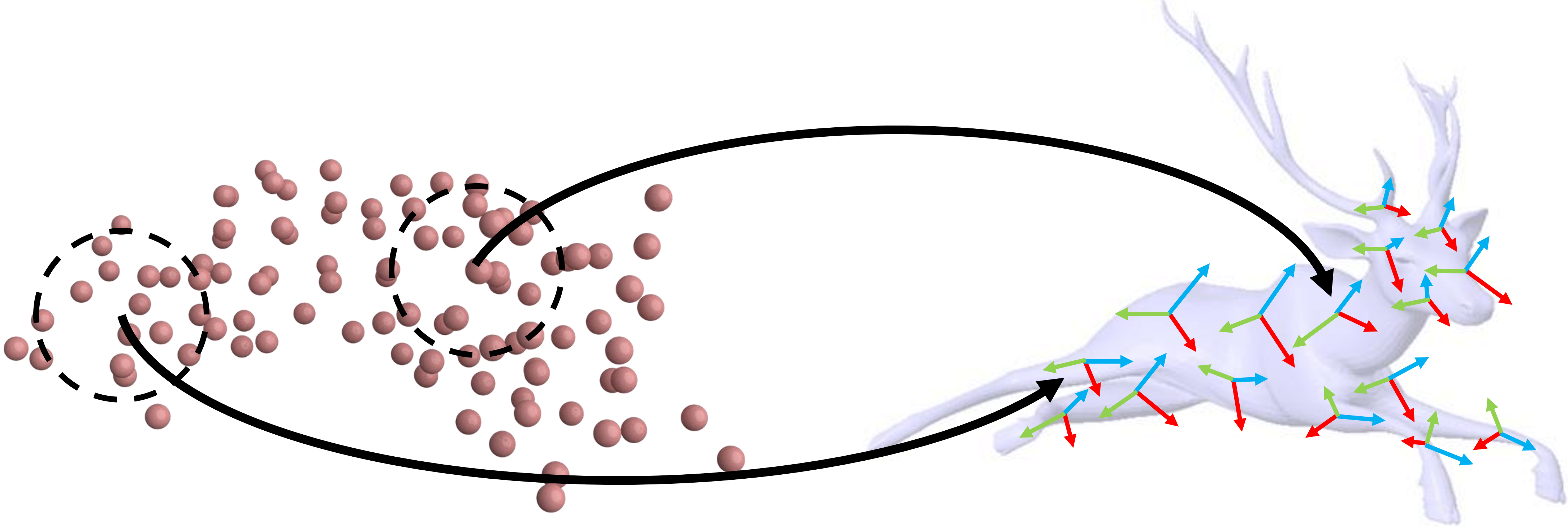
Keypoint-based Neural Pose Representation

Meanwhile, previous works encode input shapes as **abstract global embeddings**, limiting their **generalizability**.



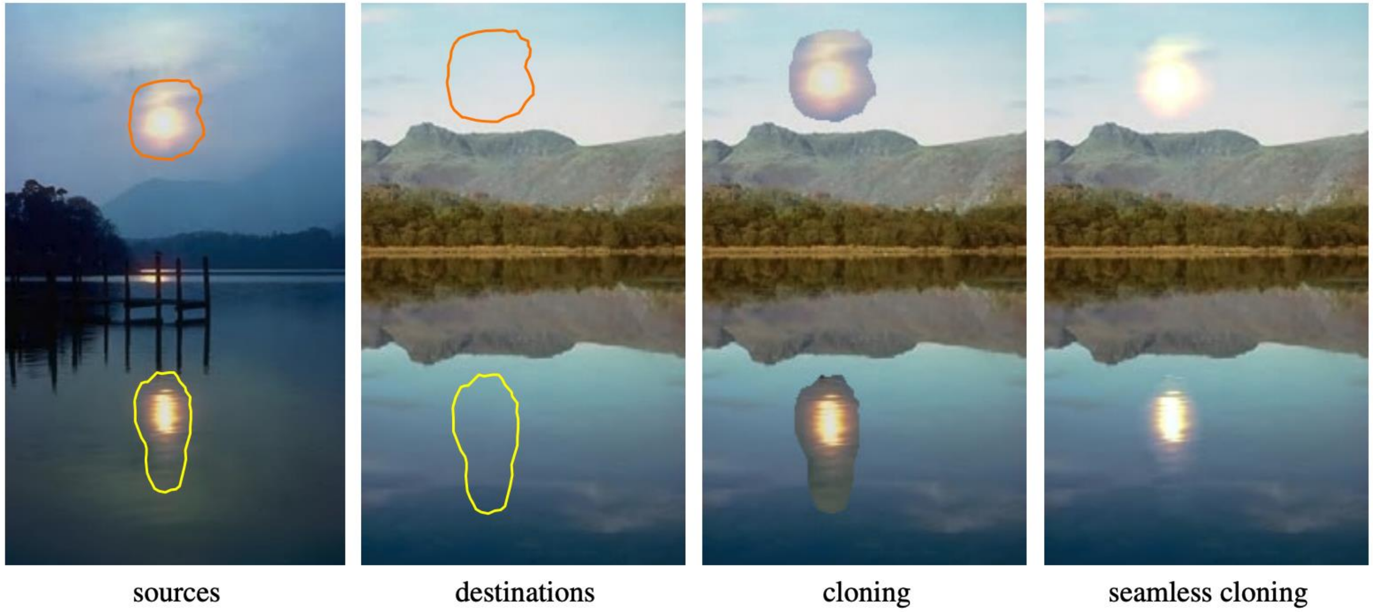
Decoding Features to Mesh Jacobians

The aggregated features are decoded to **per-triangle Jacobians** representing **local transforms**, instead of vertex coordinates.

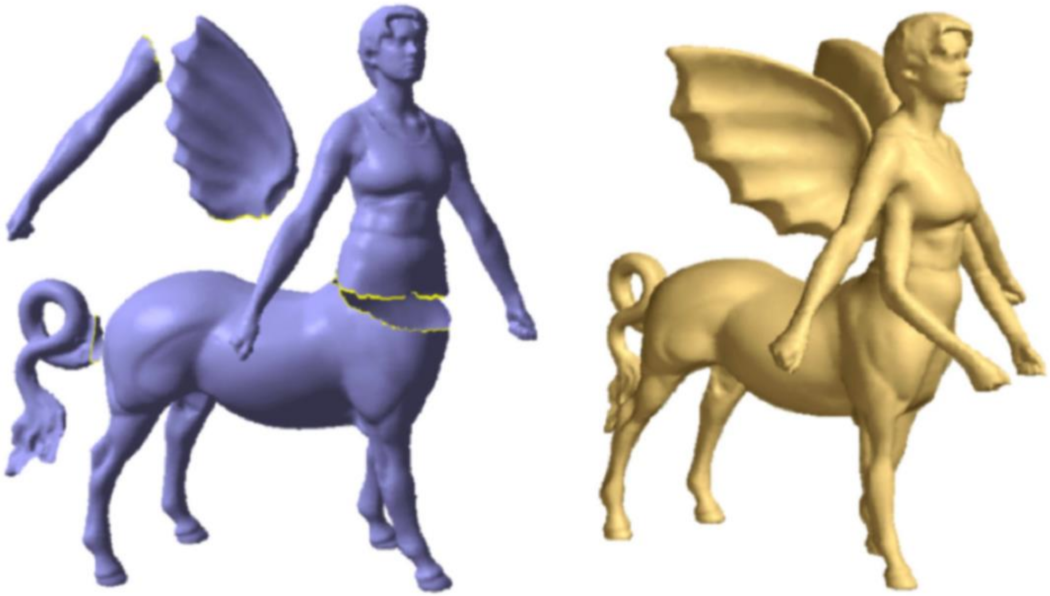


Decoding Features to Mesh Jacobians

Jacobian matrix is a gradient-domain representation, highly effective in **preserving local details**.



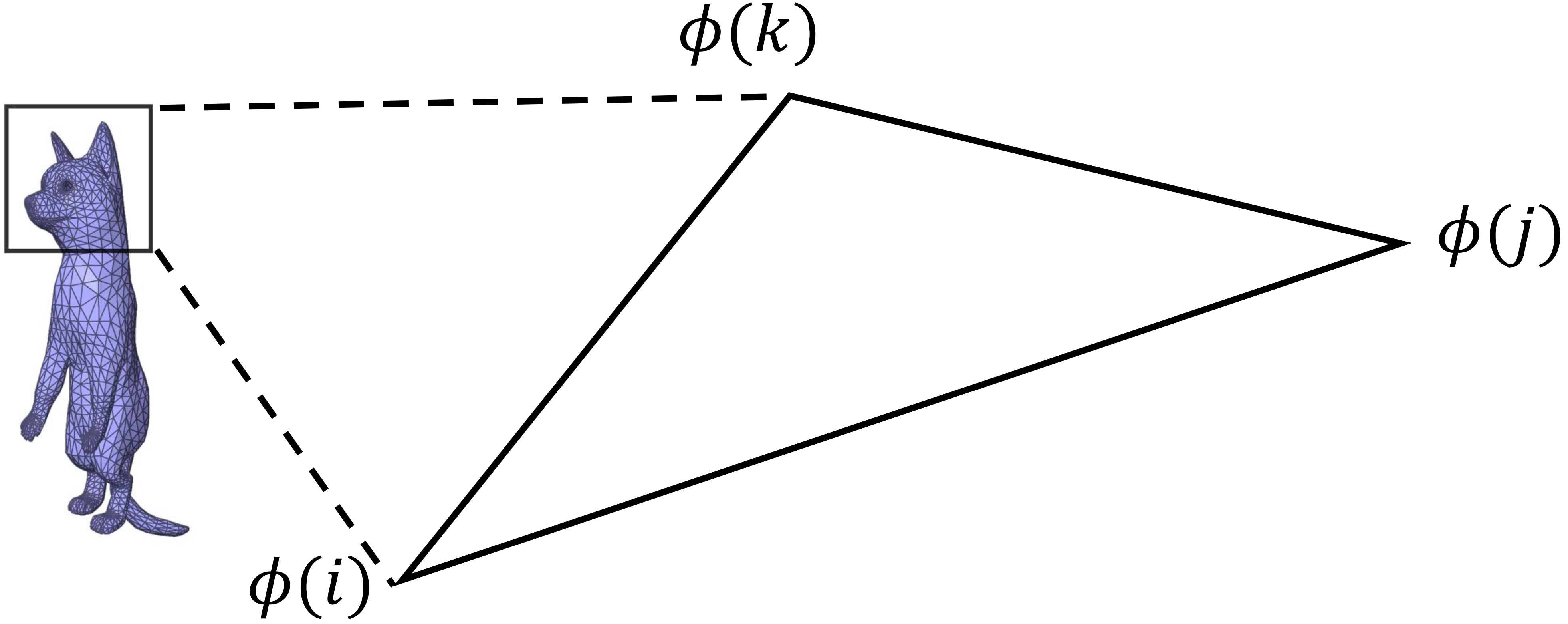
Poisson Image Editing,
Perez *et al.*, ACM ToG 2003



Poisson Mesh Editing,
Yu *et al.*, ACM ToG 2004

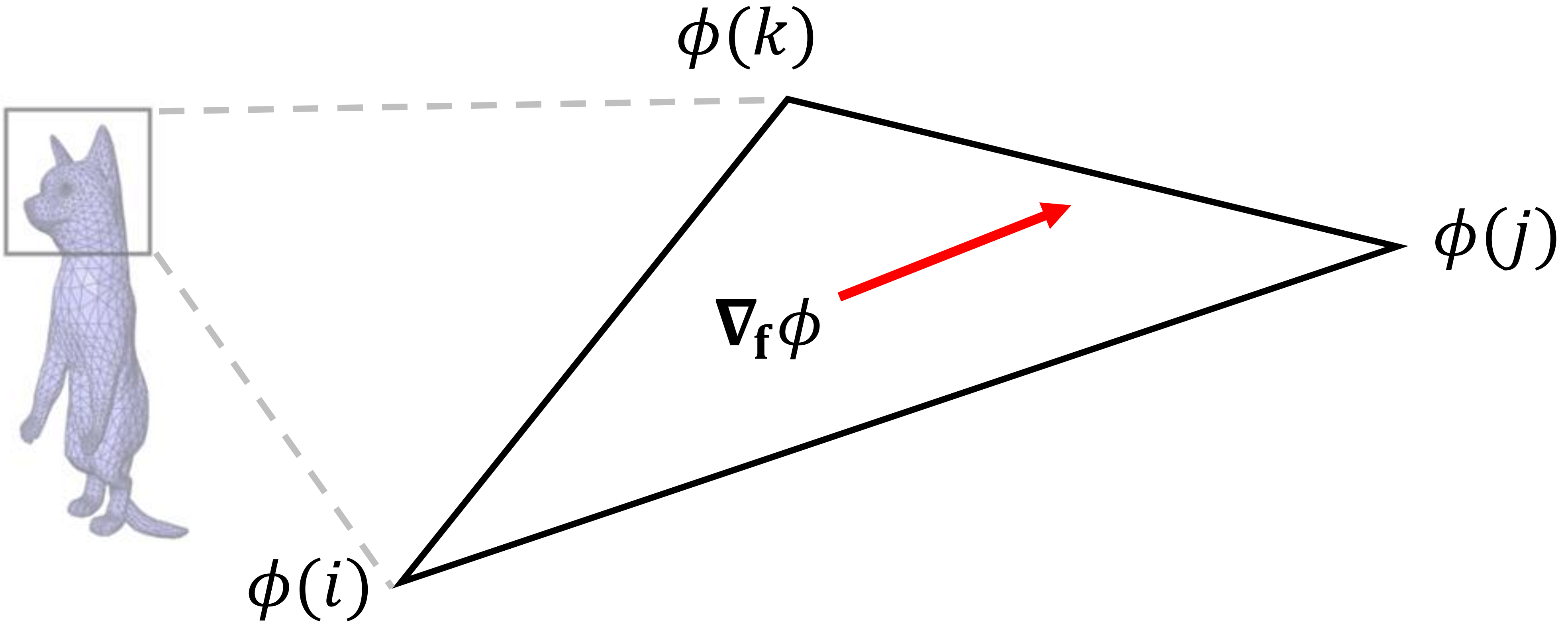
Decoding Features to Mesh Jacobians

Given a mesh $\mathcal{M} = (\mathbf{V}, \mathbf{F})$ with vertices $\mathbf{V}$ and faces $\mathbf{F}$, consider the spatial derivative $\nabla\phi$ of a scalar-valued function $\phi\colon \mathbf{V} \to \mathbb{R}$.



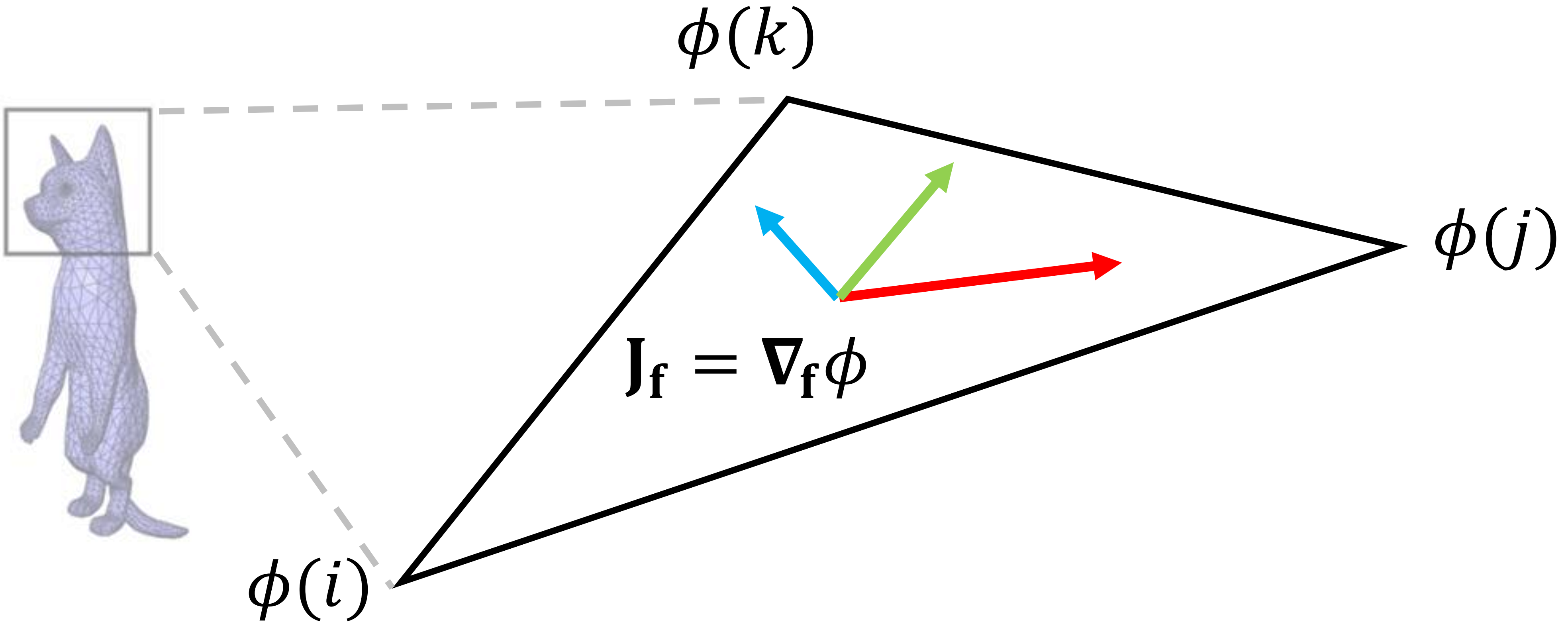
Decoding Features to Mesh Jacobians

Assuming the piece-wise linearity of ϕ, we discretize **the derivative** at each triangle $\mathbf{f} \in \mathbf{F}$ using the per-triangle gradient operator $\nabla_{\mathbf{f}}$.



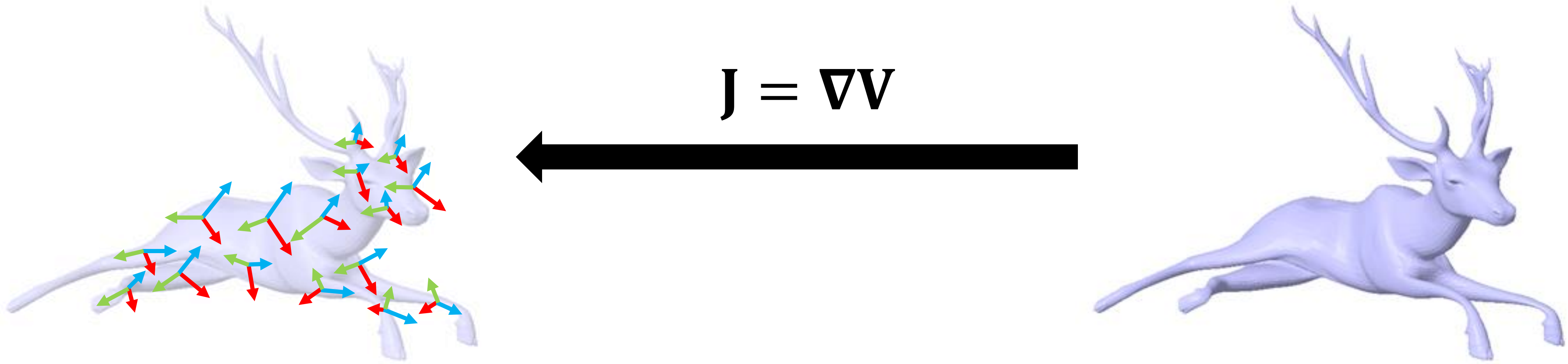
Decoding Features to Mesh Jacobians

Regarding each component of vertex coordinates as such a function, we define the per-triangle Jacobian matrix $\mathbf{J_f} = \mathbf{\nabla_f V} \in \mathbb{R}^{3\times3}$.



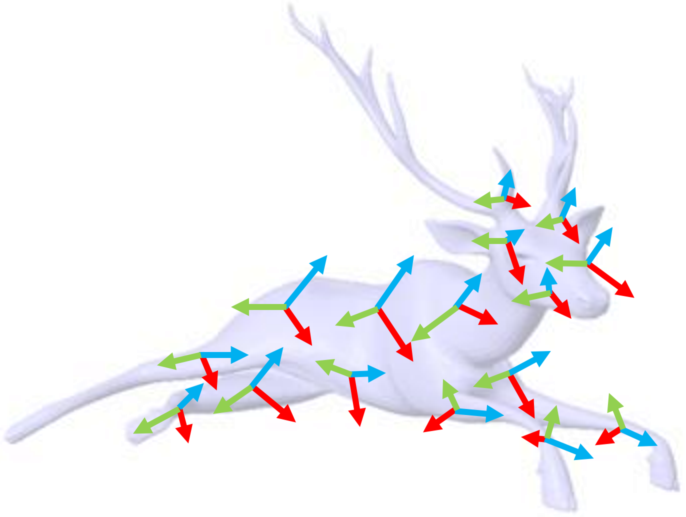
Decoding Features to Mesh Jacobians

We can represent $\mathcal{M}$ as a collection of per-triangle Jacobian matrices, denoted as the Jacobian field $\mathbf{J} \in \mathbb{R}^{3|\mathbf{F}|\times 3}$ of $\mathcal{M}$.



Decoding Features to Mesh Jacobians

Conversely, $\mathcal{M}$ can be recovered from a given Jacobian field $\mathbf{J}$ by solving the Poisson's equation.



$$\mathbf{J} = \nabla\mathbf{V}$$

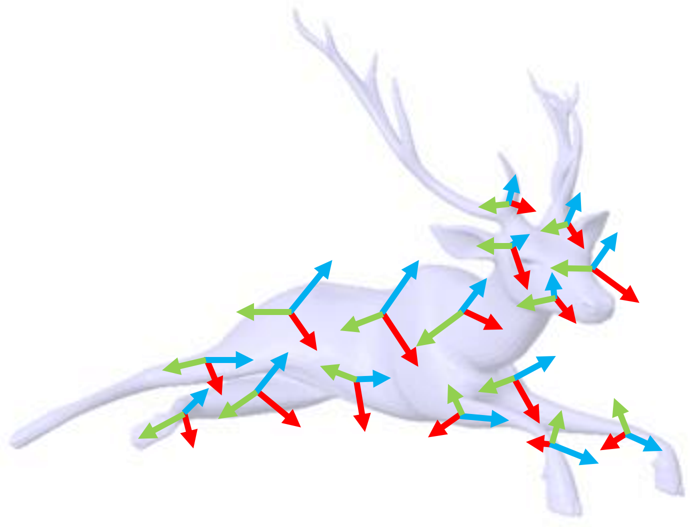
$$\mathbf{V}^* = \underset{\mathbf{V}}{\operatorname{argmin}} \parallel \mathbf{LV} - \boldsymbol{\nabla}^T \mathcal{A}\mathbf{J} \parallel^2$$

Mesh Laplacian ($\mathbb{R}^{|\mathbf{V}| \times |\mathbf{V}|}$)

Mass Matrix ($\mathbb{R}^{3|\mathbf{F}| \times 3|\mathbf{F}|}$)

Decoding Features to Mesh Jacobians

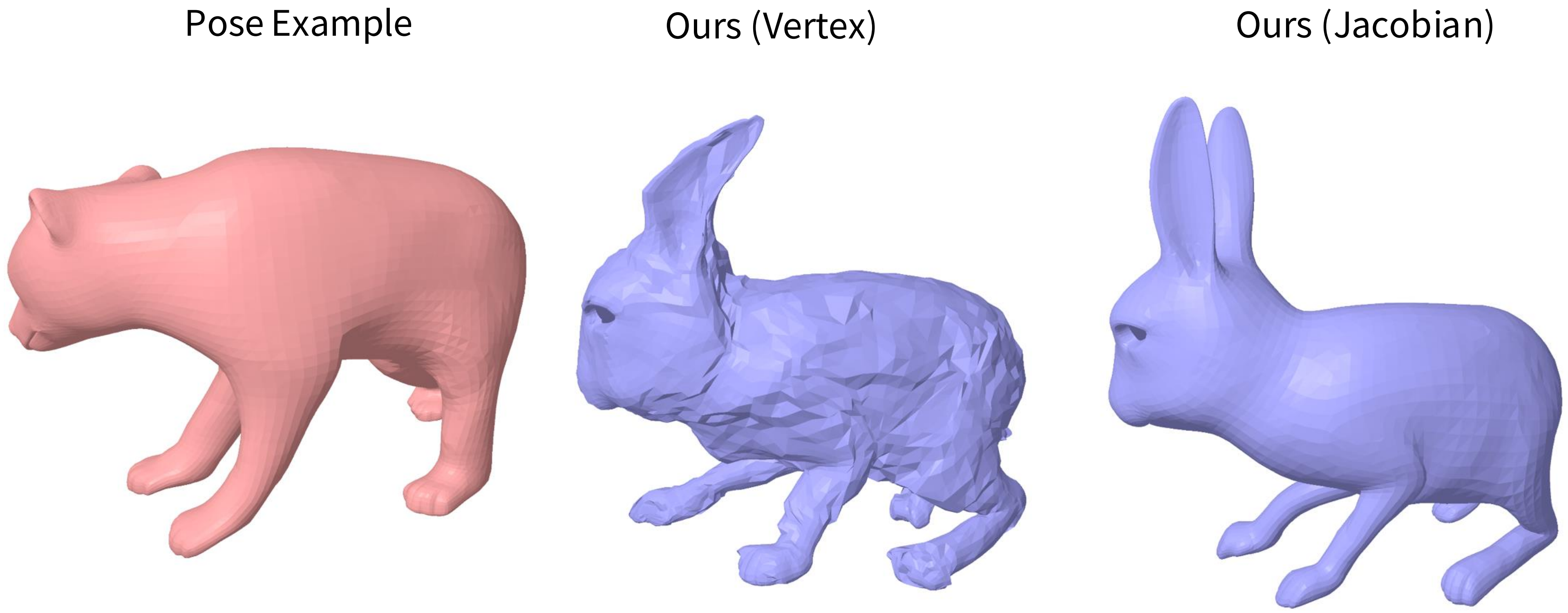
Being a dual representation of $\mathcal{M}$, representing a shape as a Jacobian field offers several advantages:



- **Local Detail Preservation**
- **Differentiability**
- **Prefactorization for Fast Forward Passes**

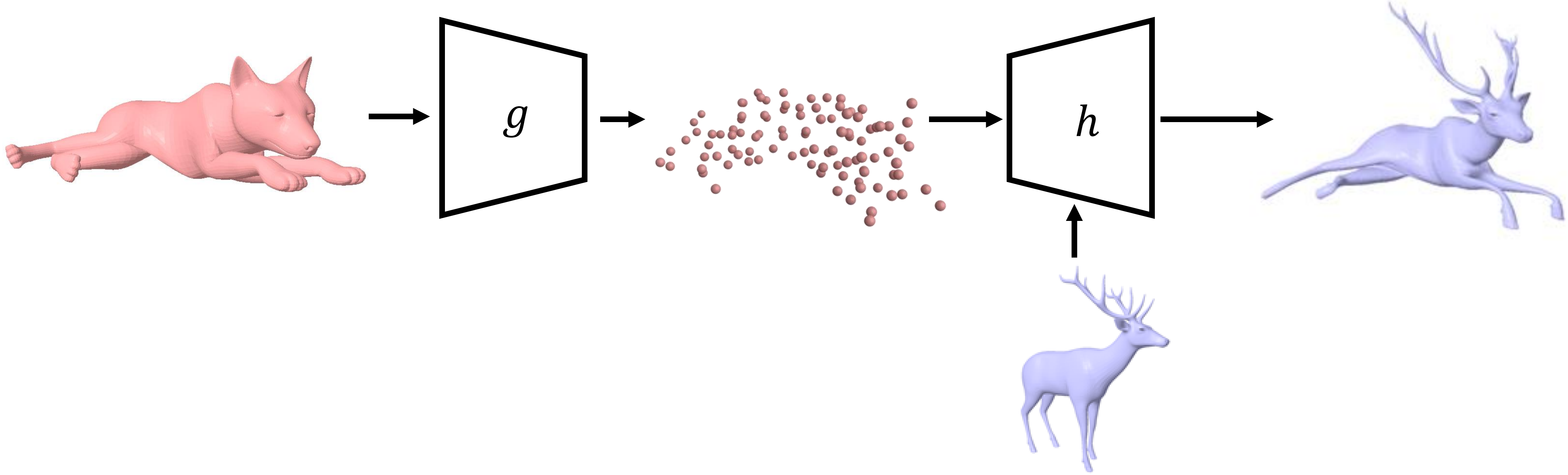
Decoding Features to Mesh Jacobians

Predicting Jacobian fields instead of vertex coordinates plays a crucial role in producing smooth surfaces after pose transfer.



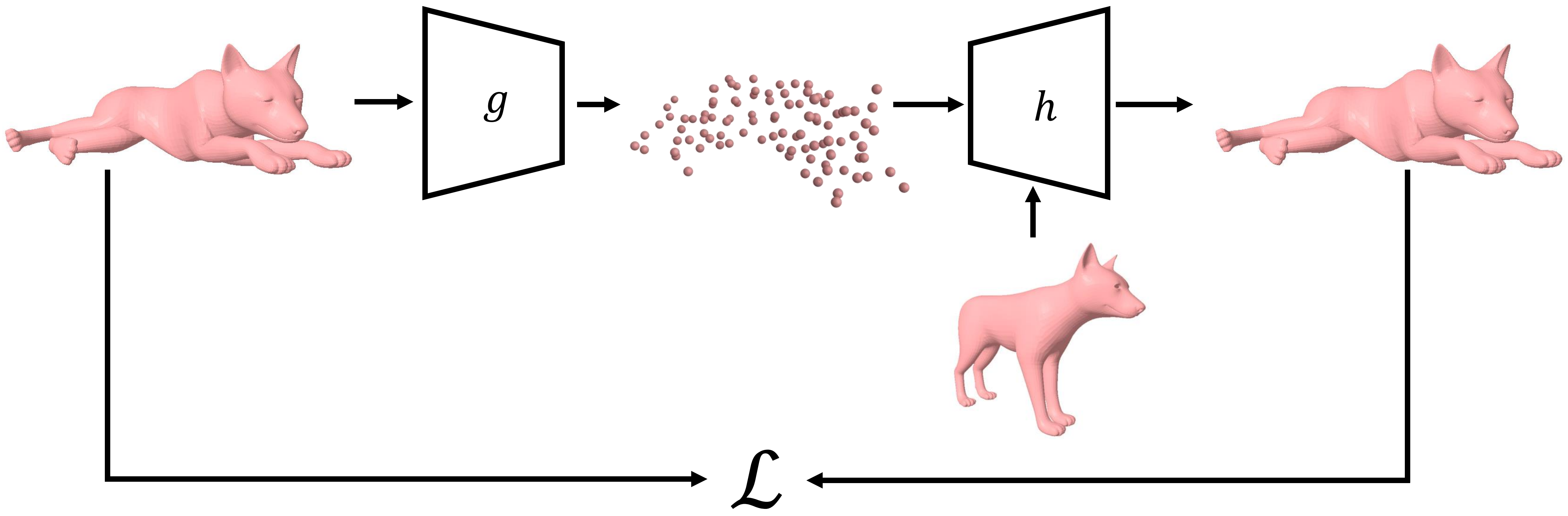
Neural Pose Representation Learning

The **pose extractor** g and **pose applier** h are designed to extract our pose representation and transfer the pose to different shapes.



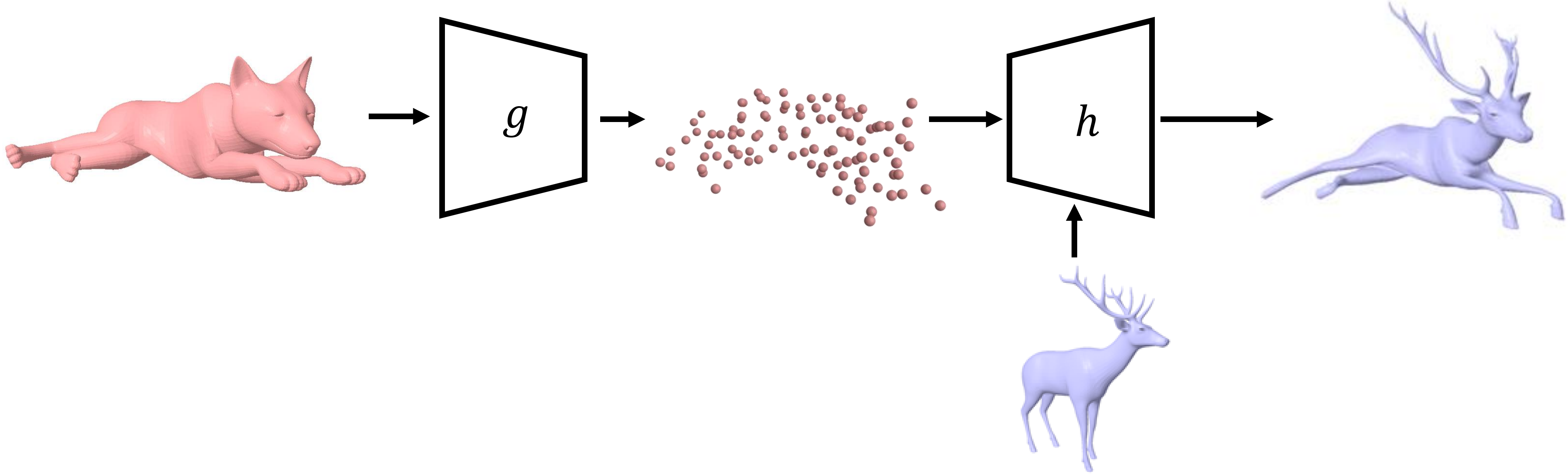
Neural Pose Representation Learning

During training, we update the network parameters by minimizing the **reconstruction loss** using **the source template and its pose examples**.



Neural Pose Representation Learning

At inference time, we only **replace the template mesh** given to the pose applier to transfer poses.



Experiment Setup

We consider the current state-of-the-art methods as our baselines:

- **Neural Jacobian Fields (NJF)** [1]
- **Skeleton-Free Pose Transfer (SPT)** [2]
- **Zero-shot Pose Transfer (ZPT)** [3]

[1] Neural Jacobian Fields: Learning Intrinsic Mappings of Arbitrary Meshes, Aigerman *et al.*, ACM ToG 2022
[2] Skeleton-Free Pose Transfer for Stylized 3D Characters, Liao *et al.*, ECCV 2022
[3] Zero-shot Pose Transfer for Unrigged Stylized 3D Characters, Wang *et al.*, CVPR 2023

Experiment Setup

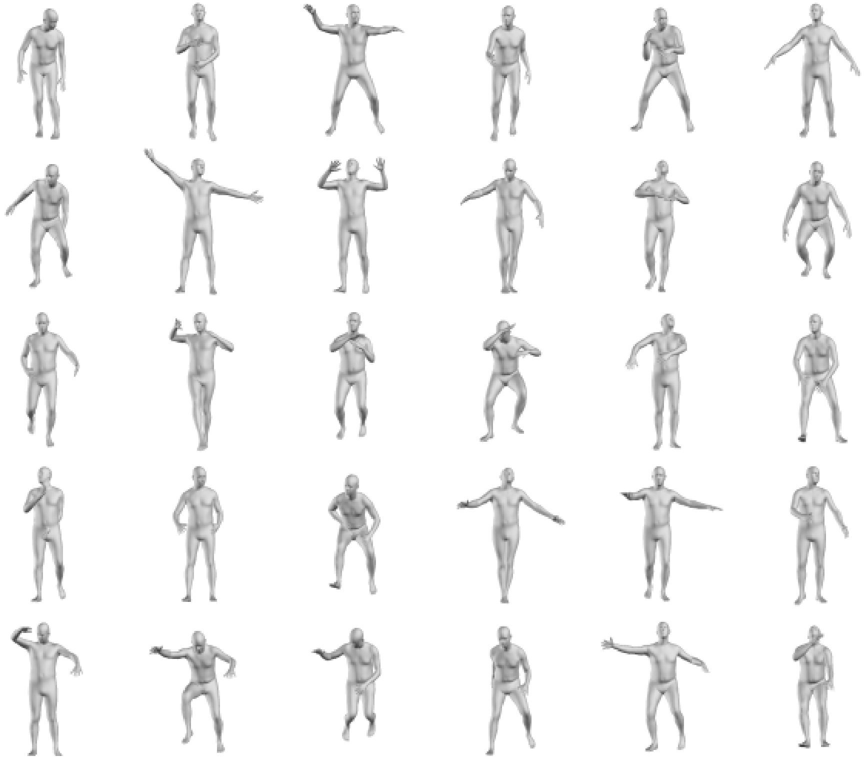
We use **DeformingThings4D-Animals** to test pose transfer among shapes with **no shared skeletal structure**.

DeformingThings4D Dataset,
Li *et al.*, ICCV 2021

- 9 shapes with 300 pose examples;
- Transfer poses to the other 8 shapes.

Experiment Setup

We populate **SMPL human body** shapes with known **vertex-wise correspondences** to measure pose transfer accuracy.



SMPL-X,
Pavlakos *et al.*, CVPR 2019

- 1 human shape with 300 pose variations;
- 40 different human shapes for testing;
- Correspondences for quantitative evaluation.

Experiment Setup

We additionally collect **9 stylized characters** from the Adobe Mixamo dataset to assess **generalization** in **real-world scenarios**.

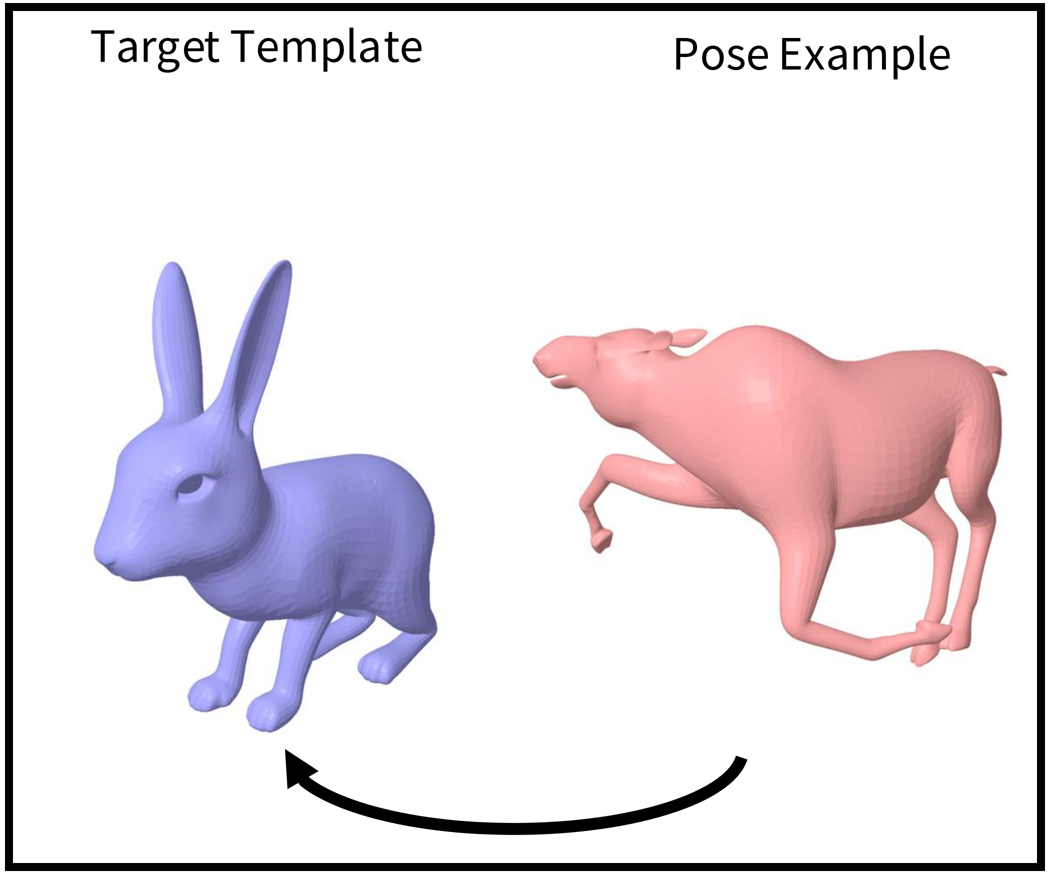
Adobe Mixamo Dataset,
Adobe

Experiment Setup

For quantitative evaluation, we use the following metrics:

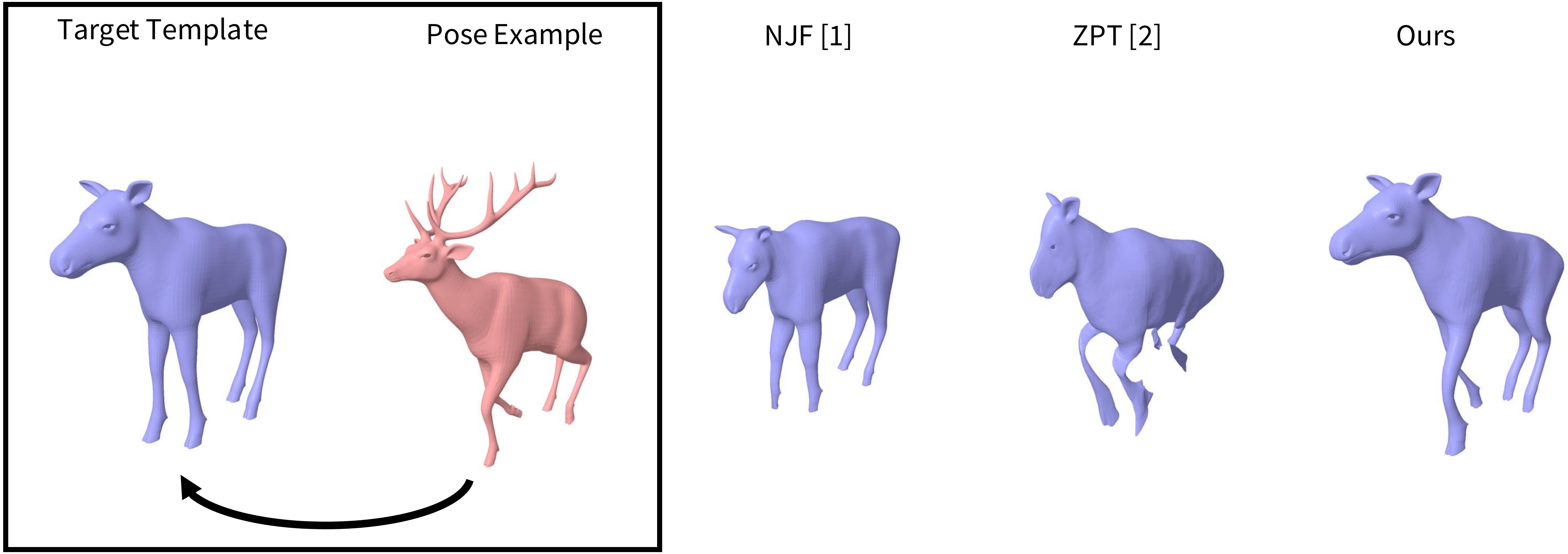
- **DeformingThings4D-Animals (Correspondence ❌)**
  - FID (Fréchet Inception Distance)
  - KID (Kernel Inception Distance)
  - ResNet Classification Accuracy
- **SMPL Human Body (Correspondence ✅)**
  - PMD (Point-wise Mesh Euclidean Distance)
  - FID (Fréchet Inception Distance)
  - KID (Kernel Inception Distance)
  - ResNet Classification Accuracy

Pose Transfer on DeformingThings4D



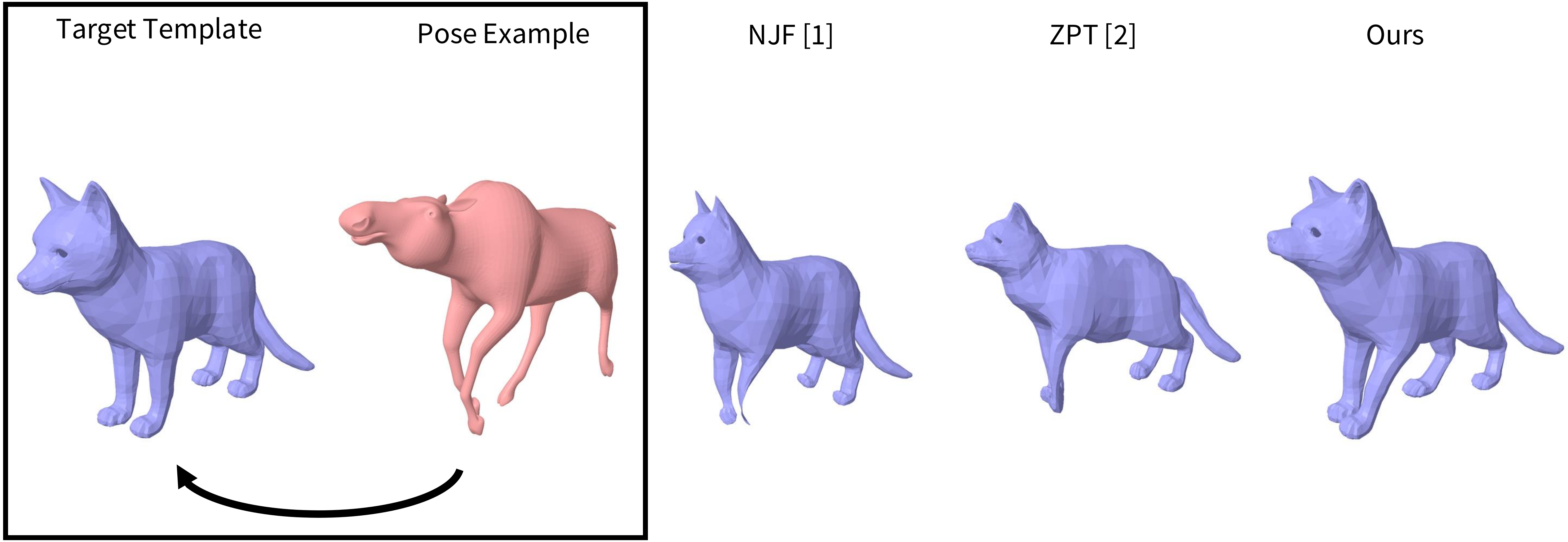
[1] Neural Jacobian Fields: Learning Intrinsic Mappings of Arbitrary Meshes, Aigerman *et al.*, ACM ToG 2022
[2] Zero-shot Pose Transfer for Unrigged Stylized 3D Characters, Wang *et al.*, CVPR 2023

Pose Transfer on DeformingThings4D



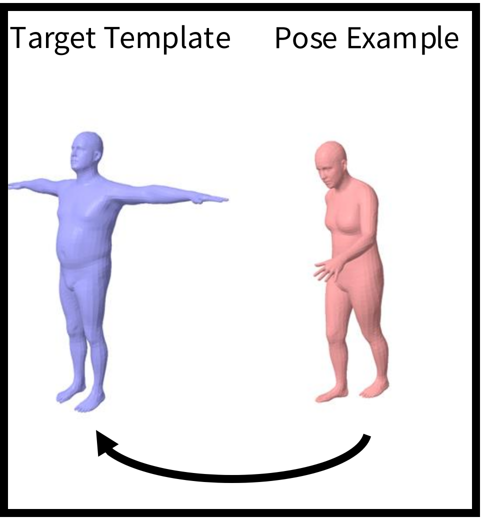
[1] Neural Jacobian Fields: Learning Intrinsic Mappings of Arbitrary Meshes, Aigerman *et al.*, ACM ToG 2022
[2] Zero-shot Pose Transfer for Unrigged Stylized 3D Characters, Wang *et al.*, CVPR 2023

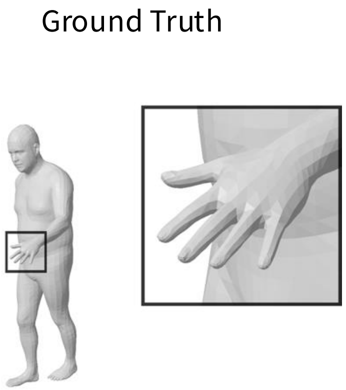
Pose Transfer on DeformingThings4D



[1] Neural Jacobian Fields: Learning Intrinsic Mappings of Arbitrary Meshes, Aigerman *et al.*, ACM ToG 2022
[2] Zero-shot Pose Transfer for Unrigged Stylized 3D Characters, Wang *et al.*, CVPR 2023

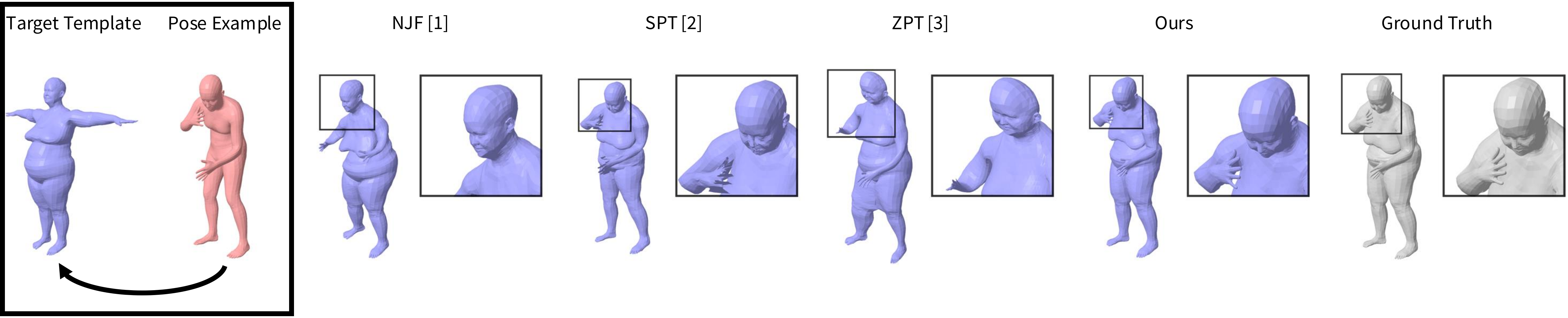
Pose Transfer on SMPL Human Body





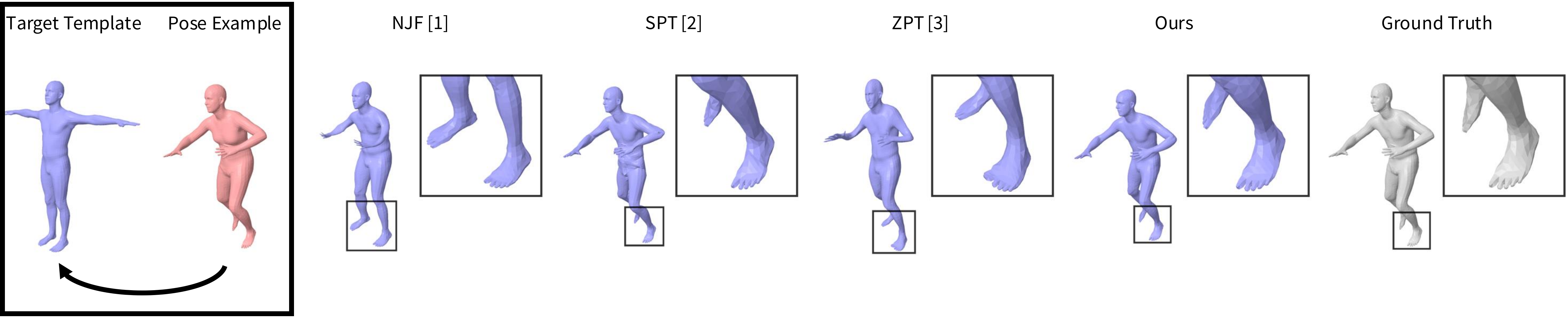
[1] Neural Jacobian Fields: Learning Intrinsic Mappings of Arbitrary Meshes, Aigerman *et al.*, ACM ToG 2022
[2] Skeleton-Free Pose Transfer for Stylized 3D Characters, Liao *et al.*, ECCV 2022
[3] Zero-shot Pose Transfer for Unrigged Stylized 3D Characters, Wang *et al.*, CVPR 2023

Pose Transfer on SMPL Human Body



[1] Neural Jacobian Fields: Learning Intrinsic Mappings of Arbitrary Meshes, Aigerman *et al.*, ACM ToG 2022
[2] Skeleton-Free Pose Transfer for Stylized 3D Characters, Liao *et al.*, ECCV 2022
[3] Zero-shot Pose Transfer for Unrigged Stylized 3D Characters, Wang *et al.*, CVPR 2023

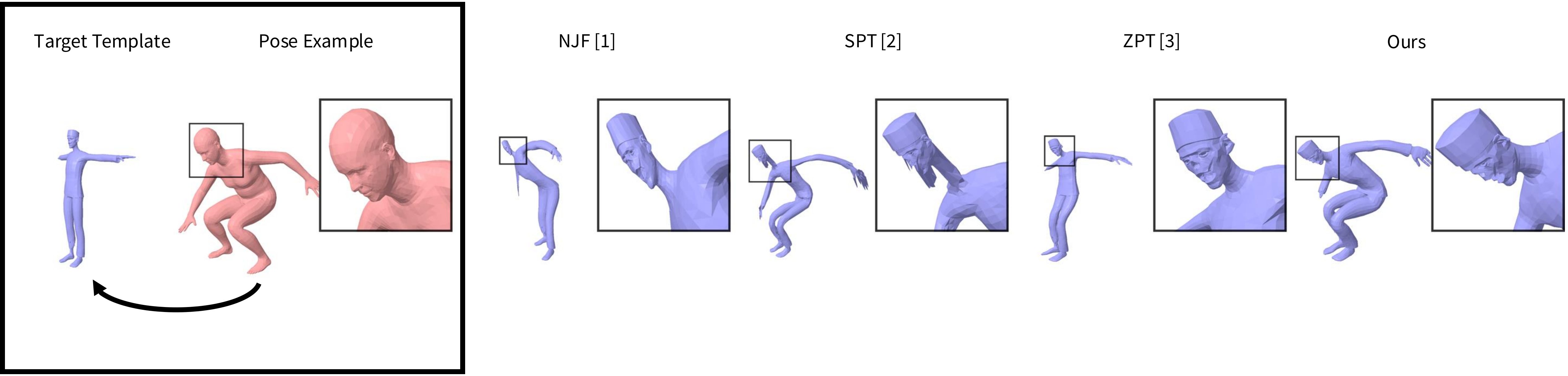
Pose Transfer on SMPL Human Body



[1] Neural Jacobian Fields: Learning Intrinsic Mappings of Arbitrary Meshes, Aigerman *et al.*, ACM ToG 2022
[2] Skeleton-Free Pose Transfer for Stylized 3D Characters, Liao *et al.*, ECCV 2022
[3] Zero-shot Pose Transfer for Unrigged Stylized 3D Characters, Wang *et al.*, CVPR 2023

Pose Transfer on Adobe Mixamo

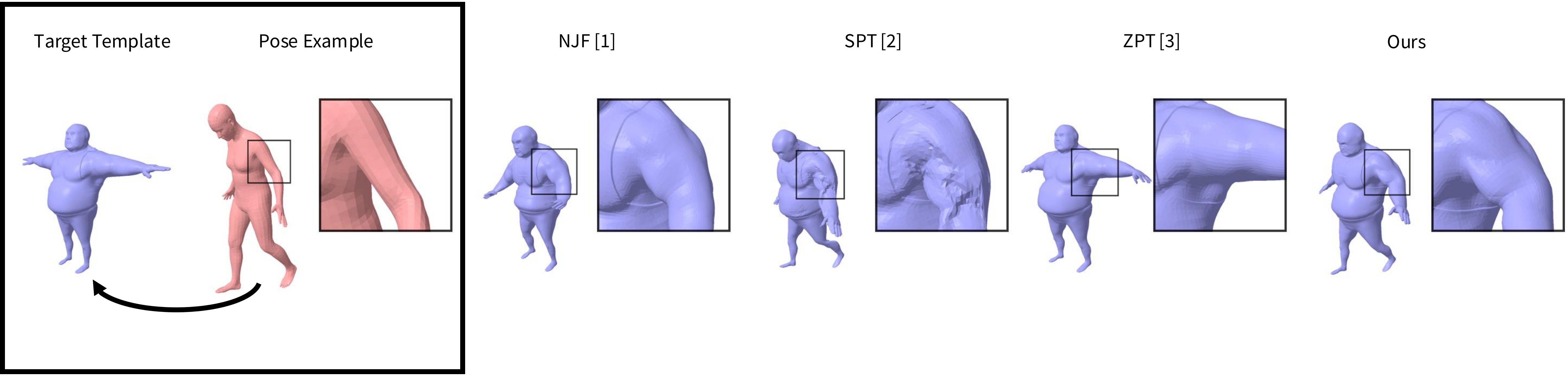
Despite being trained only on SMPL meshes, our model generalizes well to unseen stylized characters from the Adobe Mixamo dataset.



[1] Neural Jacobian Fields: Learning Intrinsic Mappings of Arbitrary Meshes, Aigerman *et al.*, ACM ToG 2022
[2] Skeleton-Free Pose Transfer for Stylized 3D Characters, Liao *et al.*, ECCV 2022
[3] Zero-shot Pose Transfer for Unrigged Stylized 3D Characters, Wang *et al.*, CVPR 2023

Pose Transfer on Adobe Mixamo

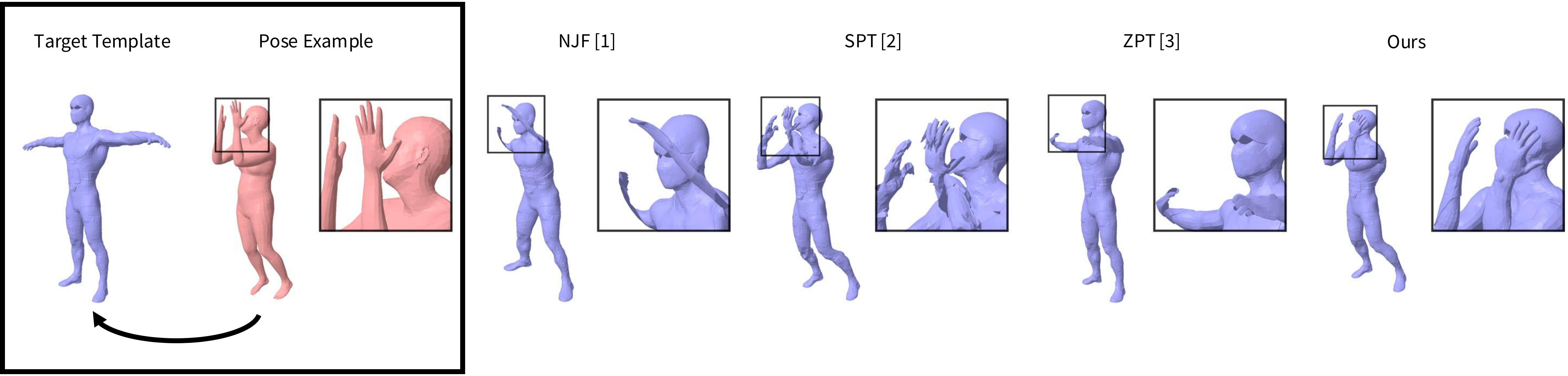
Despite being trained only on SMPL meshes, our model generalizes well to unseen stylized characters from the Adobe Mixamo dataset.



[1] Neural Jacobian Fields: Learning Intrinsic Mappings of Arbitrary Meshes, Aigerman *et al.*, ACM ToG 2022
[2] Skeleton-Free Pose Transfer for Stylized 3D Characters, Liao *et al.*, ECCV 2022
[3] Zero-shot Pose Transfer for Unrigged Stylized 3D Characters, Wang *et al.*, CVPR 2023

Pose Transfer on Adobe Mixamo

Despite being trained only on SMPL meshes, our model generalizes well to unseen stylized characters from the Adobe Mixamo dataset.



[1] Neural Jacobian Fields: Learning Intrinsic Mappings of Arbitrary Meshes, Aigerman *et al.*, ACM ToG 2022
[2] Skeleton-Free Pose Transfer for Stylized 3D Characters, Liao *et al.*, ECCV 2022
[3] Zero-shot Pose Transfer for Unrigged Stylized 3D Characters, Wang *et al.*, CVPR 2023

Quantitative Evaluation

The superior performance of our method compared to the baselines is validated by quantitative metrics.

	DeformingThings4D-Animals			SMPL Human Body			
	FID (↓) ($\times 10^{-2}$)	KID (↓) ($\times 10^{-2}$)	ResNet Acc. (↑) (%)	PMD (↓) ($\times 10^{-3}$)	FID (↓) ($\times 10^{-2}$)	KID (↓) ($\times 10^{-2}$)	ResNet Acc. (↑) (%)
NJF [1]	11.33	5.71	64.43	2.55	1.57	0.82	70.93
SPT [2]	-	-	-	0.28	0.83	0.43	75.38
ZPT [3]	19.88	11.09	48.15	1.28	0.77	0.45	69.88
Ours	**1.11**	**0.42**	**78.72**	**0.13**	**0.30**	**0.19**	**79.09**

[1] Neural Jacobian Fields: Learning Intrinsic Mappings of Arbitrary Meshes, Aigerman *et al.*, ACM ToG 2022
[2] Skeleton-Free Pose Transfer for Stylized 3D Characters, Liao *et al.*, ECCV 2022
[3] Zero-shot Pose Transfer for Unrigged Stylized 3D Characters, Wang *et al.*, CVPR 2023

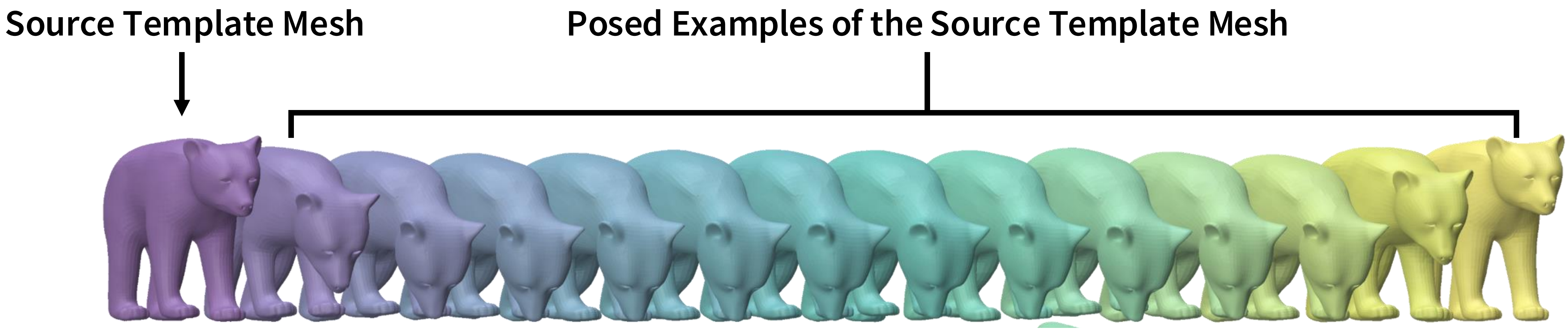
Summary

In this work, we present

- A novel **keypoint-based pose representation** for improved generalizability;
- A pose transfer framework predicting **Jacobian Fields** to preserve local details;
- **Extensive evaluation** using animals, humans, and stylized characters, showing the superior performance of the proposed representation and framework.

Data-Driven Priors Without 3D Examples

So far, we assumed that 3D meshes of a single deformable shape are available to learn a data-driven prior.



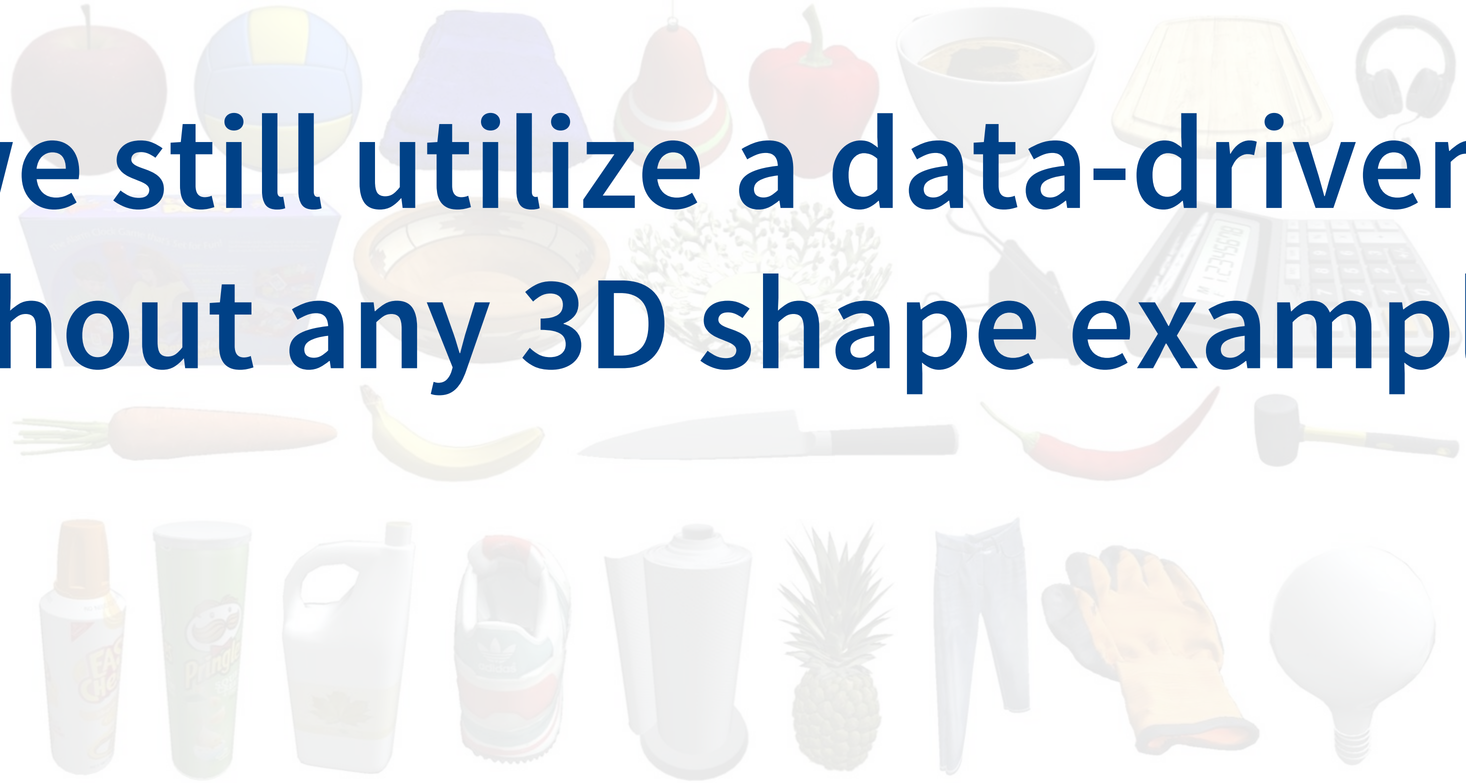
Data-Driven Priors Without 3D Examples

Such exemplars are generally unavailable for **arbitrary objects**.

The BEHAVIOR Dataset of Objects,
Srivastava, CoRL 2021

Data-Driven Priors Without 3D Examples

Such exemplars are generally unavailable for arbitrary objects.

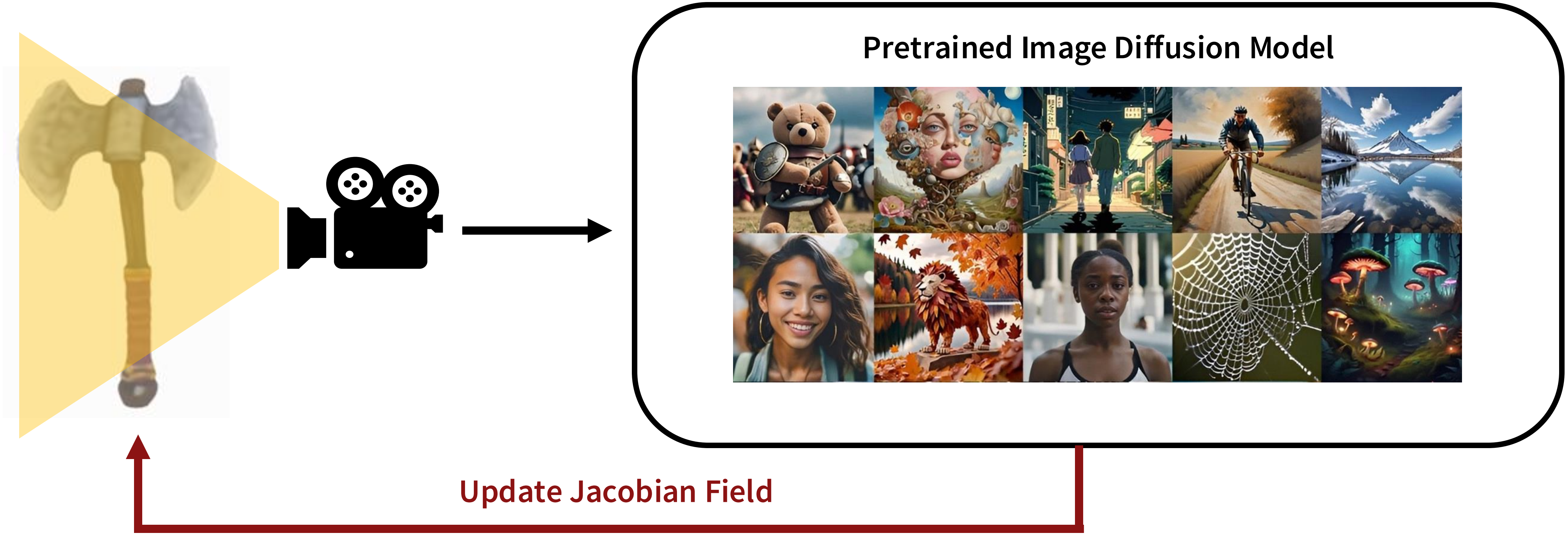


Can we still utilize a data-driven prior without any 3D shape examples?

The BEHAVIOR Dataset of Objects, Srivastava, CoRL 2021

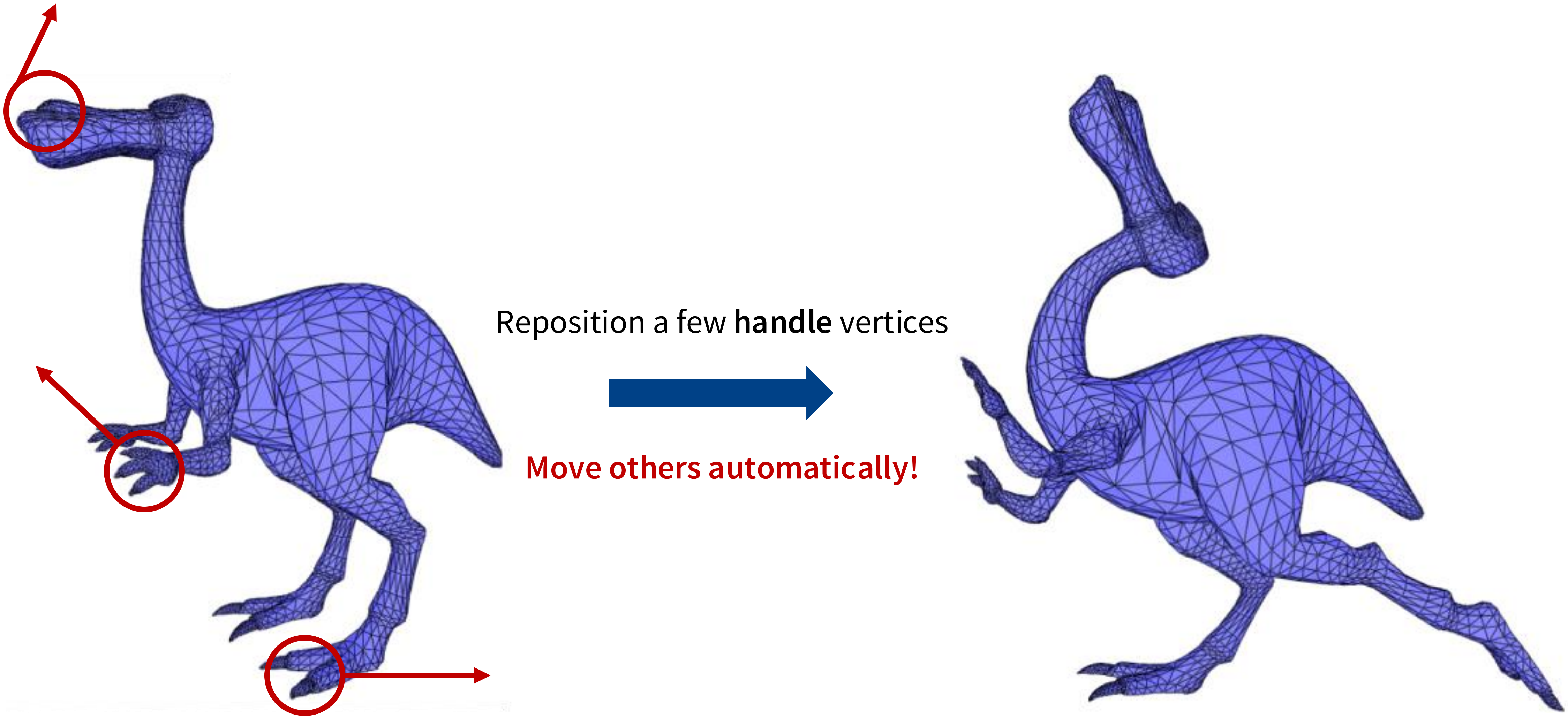
Key Idea

Leverage a 2D diffusion model as a prior and iteratively update the mesh Jacobian field via gradient descent.



SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis, Podell *et al.*, ICLR 2024

Problem Definition



The Computational Geometry Algorithms Library (CGAL)

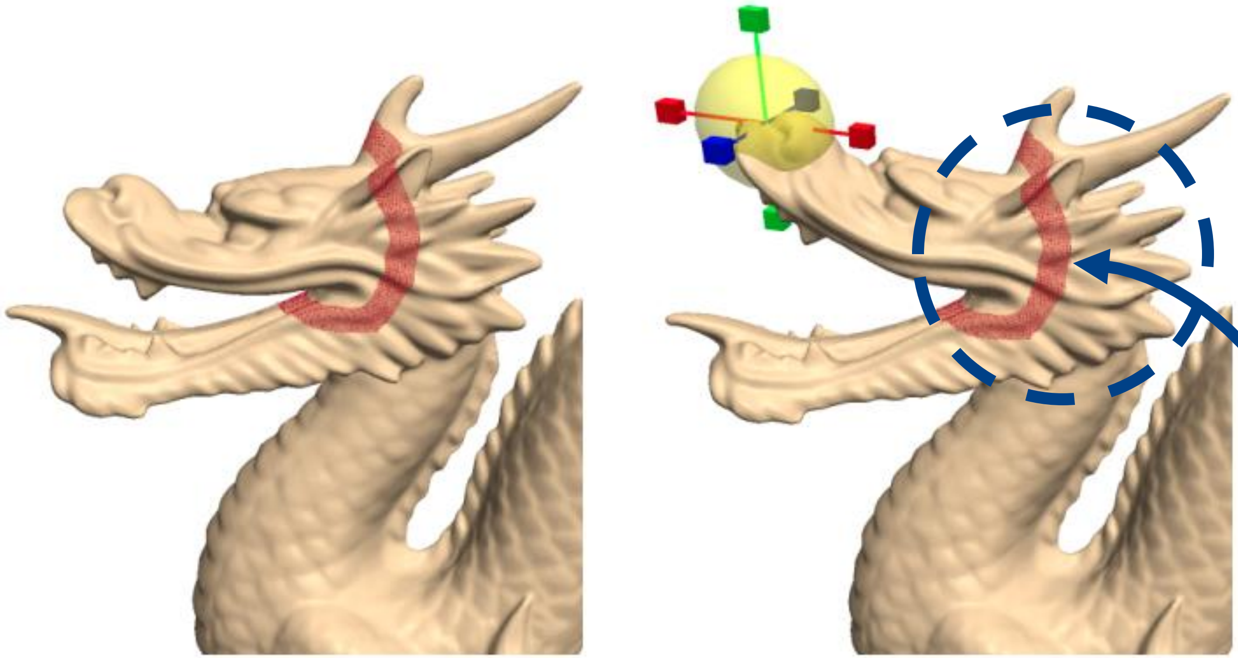
Mesh Deformation

$\mathbf{V} \in \mathbb{R}^{V \times 3}$: Input Vertices
$\hat{\mathbf{V}} \in \mathbb{R}^{V \times 3}$: Deformed Vertices
$\mathbf{S} \in \mathbb{R}^{V_c \times V}$: Indicator Matrix
$\mathbf{C} \in \mathbb{R}^{V_c \times 3}$: Constrained Positions

$$\hat{\mathbf{V}} = \min_{\mathbf{V}} \mathrm{E}(\mathbf{V}) \text{ s.t } \mathbf{S}\mathbf{V} = \mathbf{C}$$

Mesh Deformation

$\mathbf{V} \in \mathbb{R}^{V\times 3}$: Input Vertices
$\hat{\mathbf{V}} \in \mathbb{R}^{V\times 3}$: Deformed Vertices
$\mathbf{S} \in \mathbb{R}^{V_c\times V}$: Indicator Matrix
$\mathbf{C} \in \mathbb{R}^{V_c\times 3}$: Constrained Positions



$$\underbrace{\boxed{\hat{\mathbf{V}}}}_{\text{Deformed Vertex Positions}} = \min_{\mathbf{V}} \underbrace{\boxed{E(\mathbf{V})}}_{\text{Deformation Energy}} \ \text{s.t} \ \underbrace{\boxed{\mathbf{S}\mathbf{V} = \mathbf{C}}}_{\text{Constraints on Fixed Vertices}}$$

Distilling Priors from Diffusion Models

Following DreamFusion [1], we use **the training objective of diffusion models** and optimize it via **gradient descent** to distill deformation priors.

$$\mathcal{L}(\phi, \mathbf{I}) = \left\| \epsilon_{\phi}(\mathbf{I}_t) - \epsilon \right\|^2$$

How can this loss function be applied to deform meshes?

[1] DreamFusion: Text-to-3D using 2D Diffusion, Poole *et al.*, ICLR 2023

Distilling Priors from Diffusion Models

Diffusion models are trained to predict the noise to be removed from a given image.

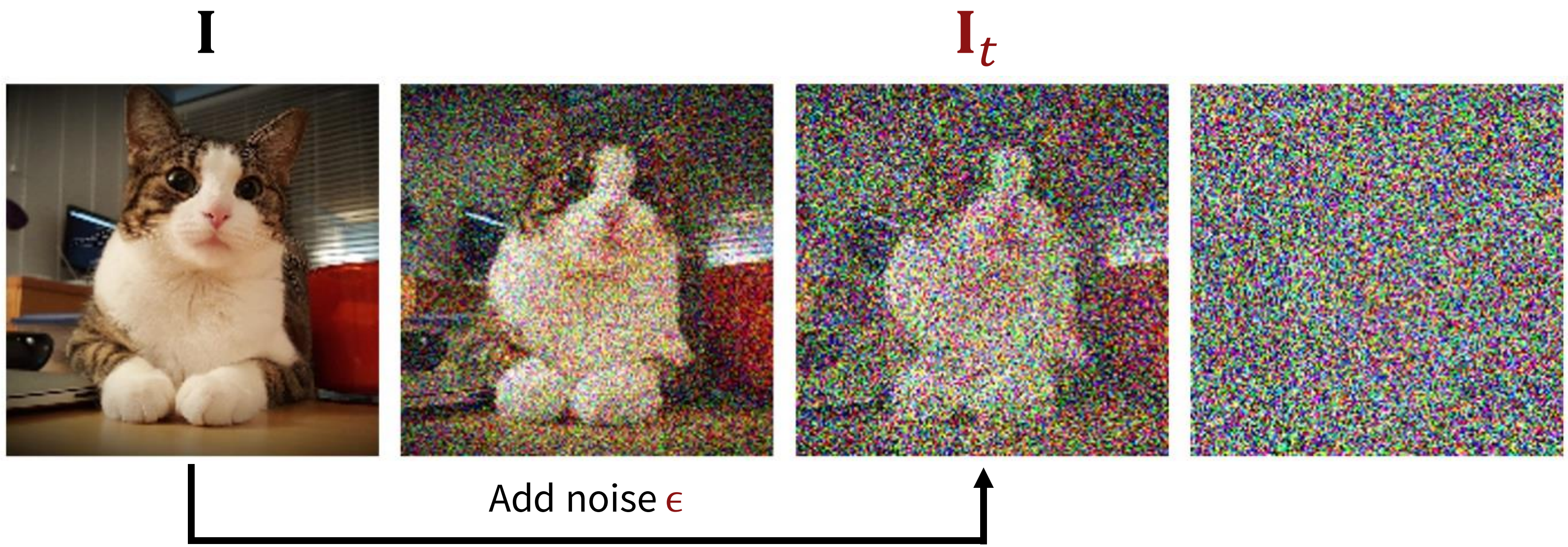
$$\mathcal{L}(\phi, \mathbf{I}) = \left\| \epsilon_\phi(\mathbf{I}_t) - \epsilon \right\|^2$$

$\mathbf{I}$

Distilling Priors from Diffusion Models

Diffusion models are trained to predict the noise to be removed from a given image.

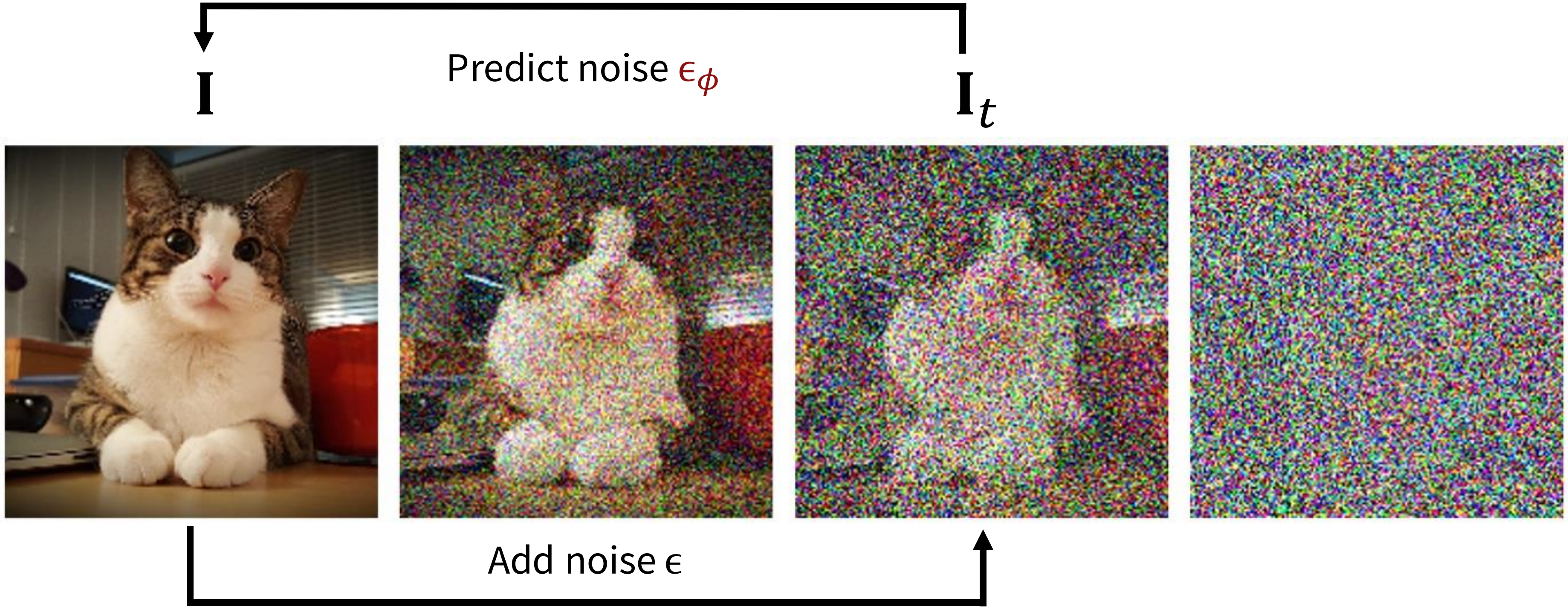
$$\mathcal{L}(\phi, \mathbf{I}) = \left\| \epsilon_\phi(\mathbf{I}_t) - \epsilon \right\|^2$$



Distilling Priors from Diffusion Models

Diffusion models are trained to predict the noise to be removed from a given image.

$$\mathcal{L}(\phi, \mathbf{I}) = \left\| \epsilon_\phi(\mathbf{I}_t) - \epsilon \right\|^2$$



Distilling Priors from Diffusion Models

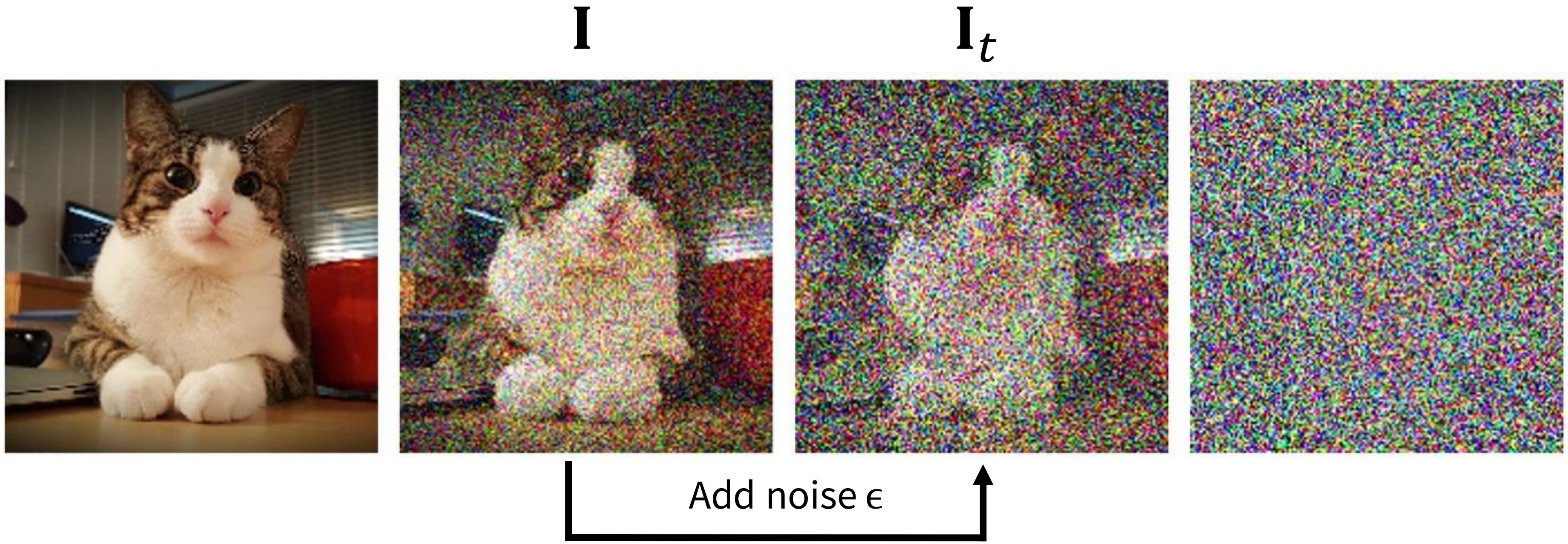
Given **a fully converged model** ϕ, we can instead compute the gradient with respect to the image $\mathbf{I}$:

$$\nabla_{\mathbf{I}} \mathcal{L}(\phi, \mathbf{I}) = (\epsilon_\phi(\mathbf{I}_\mathrm{t}) - \epsilon) \frac{\partial \mathbf{I}_t}{\partial \mathbf{I}}$$

Distilling Priors from Diffusion Models

Given **a fully converged model** ϕ, we can instead compute the gradient with respect to the image $\mathbf{I}$:

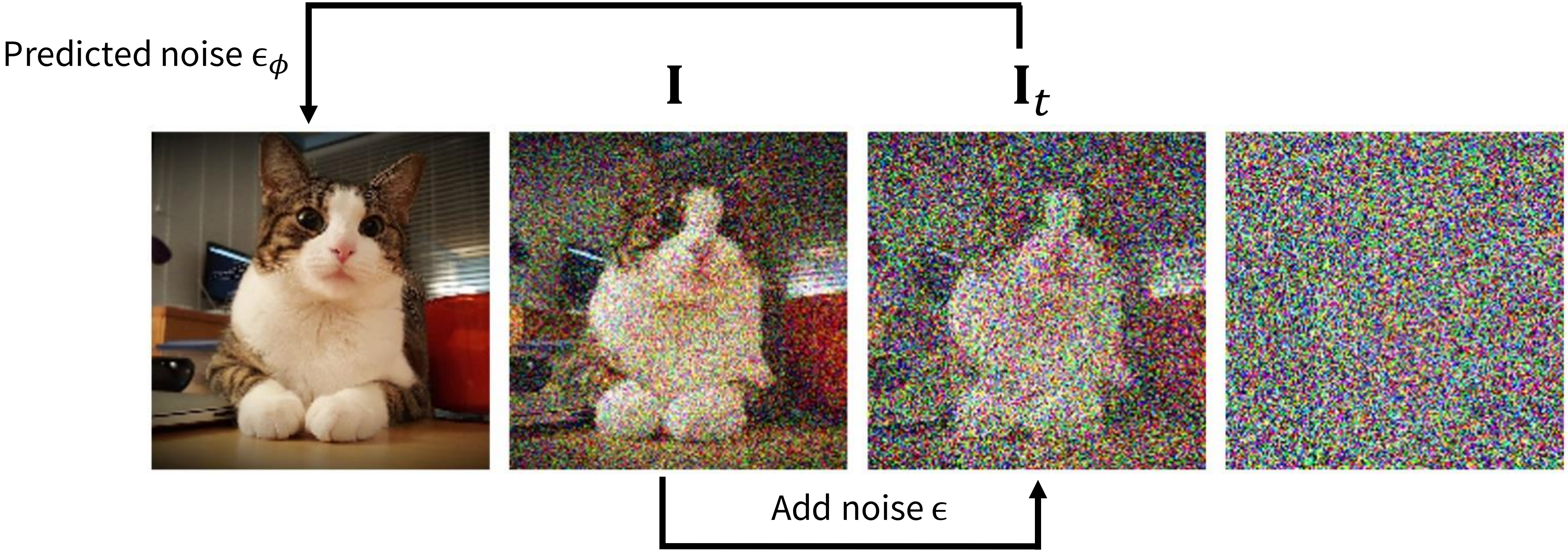
$$\nabla_{\mathbf{I}}\mathcal{L}(\phi, \mathbf{I}) = (\epsilon_\phi(\mathbf{I}_\mathrm{t}) - \epsilon)\frac{\partial \mathbf{I}_t}{\partial \mathbf{I}}$$



Distilling Priors from Diffusion Models

Given **a fully converged model** ϕ, we can instead compute the gradient with respect to the image $\mathbf{I}$:

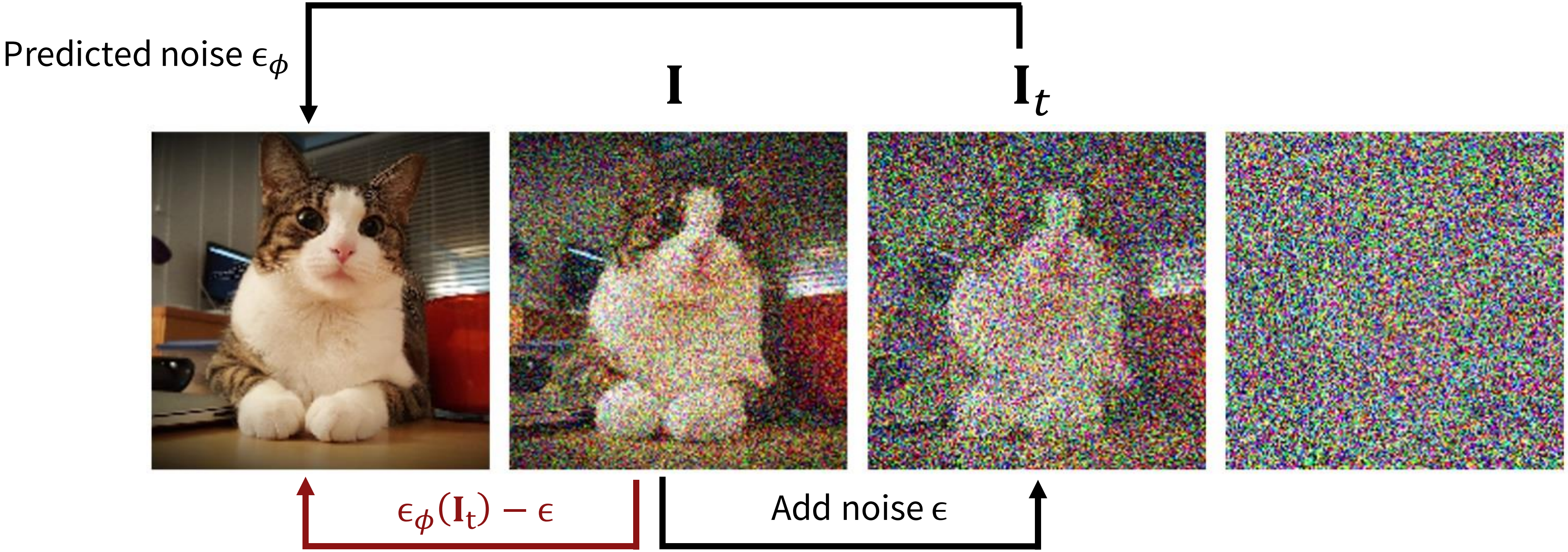
$$\nabla_{\mathbf{I}}\mathcal{L}(\phi, \mathbf{I}) = (\epsilon_\phi(\mathbf{I}_t) - \epsilon)\frac{\partial \mathbf{I}_t}{\partial \mathbf{I}}$$



Distilling Priors from Diffusion Models

Given **a fully converged model** ϕ, we can instead compute the gradient with respect to the image $\mathbf{I}$:

$$\nabla_{\mathbf{I}}\mathcal{L}(\phi, \mathbf{I}) = (\epsilon_{\phi}(\mathbf{I}_t) - \epsilon)\frac{\partial \mathbf{I}_t}{\partial \mathbf{I}}$$



Distilling Priors from Diffusion Models

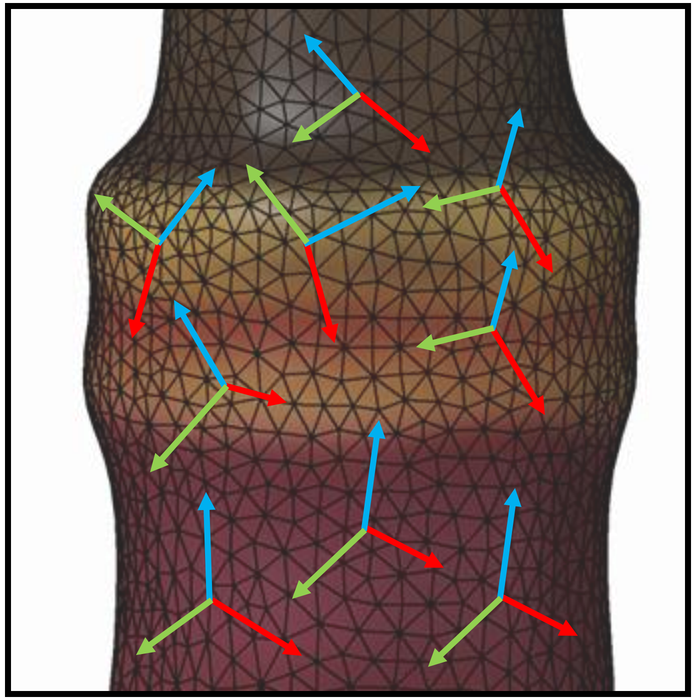
If the image $\mathbf{I}$ was rendered from a mesh whose vertex coordinates are $\mathbf{V}$, we can apply the chain rule to obtain:

$$\nabla_{\mathbf{V}}\mathcal{L}(\phi, \mathbf{I}) = (\epsilon_\phi(\mathbf{I}_t) - \epsilon)\frac{\partial \mathbf{I}_t}{\partial \mathbf{I}}\frac{\partial \mathbf{I}}{\partial \mathbf{V}}$$

Gradient Through the Renderer

Representing a Shape as a Jacobian Field

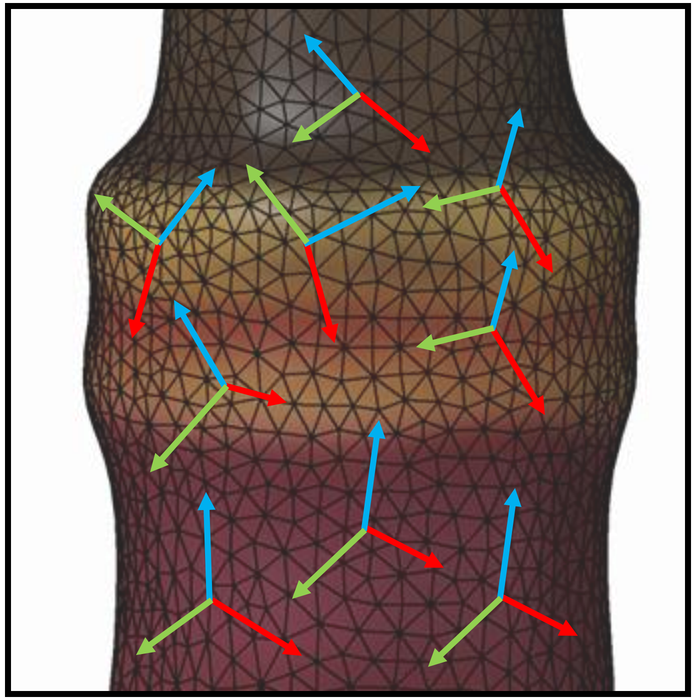
Instead of optimizing $\mathbf{V}$, we update the **Jacobian field** to deform the shape using the distilled prior.



- Local Detail Preservation
- Differentiability
- Prefactorization for Fast Forward Passes

Representing a Shape as a Jacobian Field

During optimization, we compute $\mathbf{V}$ from a given Jacobian field $\mathbf{J}$ by solving the Poisson's equation.



$$\mathbf{J} = \nabla \mathbf{V}$$

$$\mathbf{V}^* = \underset{\mathbf{V}}{\operatorname{argmin}} \parallel \mathbf{L}\mathbf{V} - \nabla^T \mathcal{A} \mathbf{J} \parallel^2$$

Mesh Laplacian ($\mathbb{R}^{|\mathbf{V}| \times |\mathbf{V}|}$)

Mass Matrix ($\mathbb{R}^{3|\mathbf{F}| \times 3|\mathbf{F}|}$)

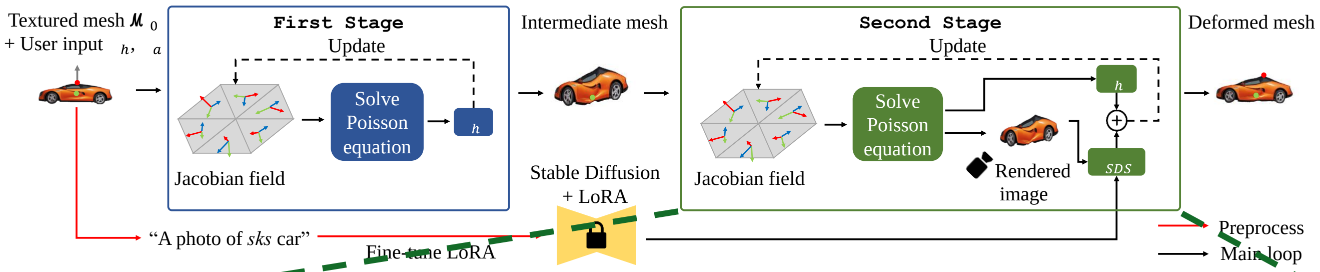
Representing a Shape as a Jacobian Field

Then, we apply the chain rule again to compute the gradient with respect to the Jacobian field serving as the **optimization variable**:

$$\nabla_{\mathbf{J}} \mathcal{L}(\phi, \mathbf{I}) = (\epsilon_\phi(\mathbf{I}_t) - \epsilon) \frac{\partial \mathbf{I}_t}{\partial \mathbf{I}} \frac{\partial \mathbf{I}}{\partial \mathbf{V}} \frac{\partial \mathbf{V}}{\partial \mathbf{J}}$$

Gradient Through
the Poisson Solver

As-Plausible-As-Possible Framework



for i = 1, …, N:

$\mathbf{V}^* \leftarrow \underset{\mathbf{V}}{\text{argmin}} \parallel \mathbf{LV} - \boldsymbol{\nabla}^T \mathcal{A} \mathbf{J} \parallel^2$ // Solve Poisson's Equation

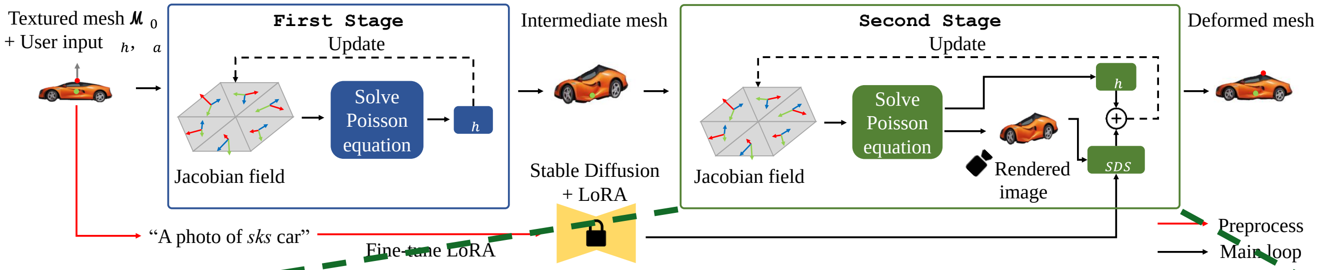
$\mathbf{I} \leftarrow \mathcal{R}(\mathbf{V}^*)$ // Render Mesh

$\mathbf{J} \leftarrow \mathbf{J} - \boldsymbol{\nabla}_{\mathbf{J}} \|\mathbf{K}_h \mathbf{V}^* - \mathbf{T}_h\|^2$ // Move Handle Vertices

$\mathbf{J} \leftarrow \mathbf{J} - \boldsymbol{\nabla}_{\mathbf{J}} \mathcal{L}(\mathbf{I}, \boldsymbol{\epsilon}_{\phi})$ // Follow Image Model Guidance

return $\mathbf{J}$

As-Plausible-As-Possible Framework



for i = 1, … , N:

$\mathbf{V}^* \leftarrow \underset{\mathbf{V}}{\operatorname{argmin}} \parallel \mathbf{L}\mathbf{V} - \boldsymbol{\nabla}^T \mathcal{A}\mathbf{J} \parallel^2$ // Solve Poisson's Equation

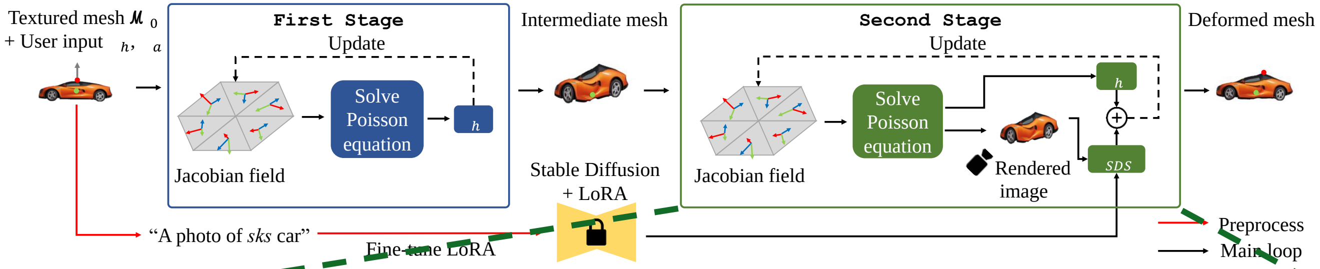
$\mathbf{I} \leftarrow \mathcal{R}(\mathbf{V}^*)$ // Render Mesh

$\mathbf{J} \leftarrow \mathbf{J} - \boldsymbol{\nabla}_{\mathbf{J}} \|\mathbf{K}_h \mathbf{V}^* - \mathbf{T}_h\|^2$ // Move Handle Vertices

$\mathbf{J} \leftarrow \mathbf{J} - \boldsymbol{\nabla}_{\mathbf{J}} \mathcal{L}(\mathbf{I}, \boldsymbol{\epsilon}_{\phi})$ // Follow Image Model Guidance

return **J**

As-Plausible-As-Possible Framework



for i = 1, … , N:

$$\mathbf{V}^* \leftarrow \underset{\mathbf{V}}{\operatorname{argmin}} \parallel \mathbf{LV} - \boldsymbol{\nabla}^T \mathcal{A}\mathbf{J} \parallel^2$$ // Solve Poisson's Equation

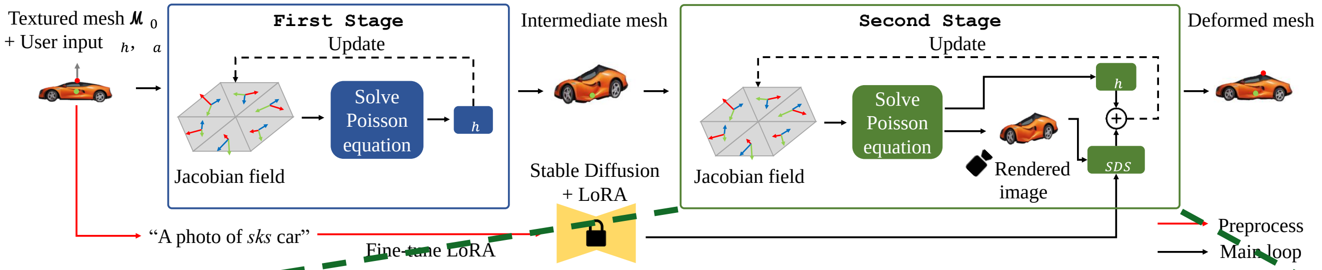
$$\mathbf{I} \leftarrow \mathcal{R}(\mathbf{V}^*)$$ // Render Mesh

$$\mathbf{J} \leftarrow \mathbf{J} - \boldsymbol{\nabla}_{\mathbf{J}} ||\mathbf{K}_h \mathbf{V}^* - \mathbf{T}_h||^2$$ // Move Handle Vertices

$$\mathbf{J} \leftarrow \mathbf{J} - \boldsymbol{\nabla}_{\mathbf{J}} \mathcal{L}(\mathbf{I}, \boldsymbol{\epsilon}_{\phi})$$ // Follow Image Model Guidance

return $\mathbf{J}$

As-Plausible-As-Possible Framework



for i = 1, ..., N:

$$\mathbf{V}^* \leftarrow \underset{\mathbf{V}}{\operatorname{argmin}} \parallel \mathbf{LV} - \nabla^T \mathcal{A}\mathbf{J} \parallel^2$$ // Solve Poisson's Equation

$$\mathbf{I} \leftarrow \mathcal{R}(\mathbf{V}^*)$$ // Render Mesh

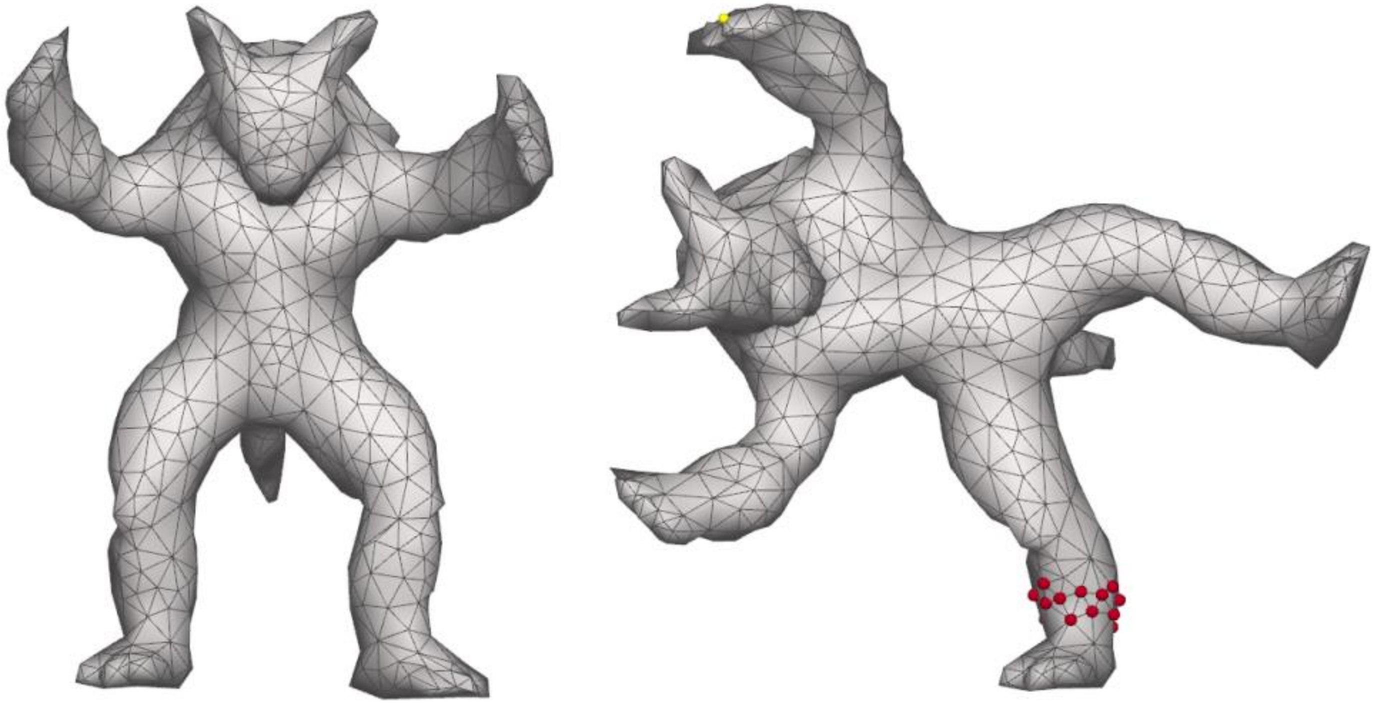
$$\mathbf{J} \leftarrow \mathbf{J} - \nabla_{\mathbf{J}} \|\mathbf{K}_h \mathbf{V}^* - \mathbf{T}_h\|^2$$ // Move Handle Vertices

$$\mathbf{J} \leftarrow \mathbf{J} - \nabla_{\mathbf{J}} \mathcal{L}(\mathbf{I}, \boldsymbol{\epsilon}_\phi)$$ // Follow Image Model Guidance

return **J**

Experiment Setup

We compare our method against ARAP, the most widely used mesh deformation technique based on rigidity energy.



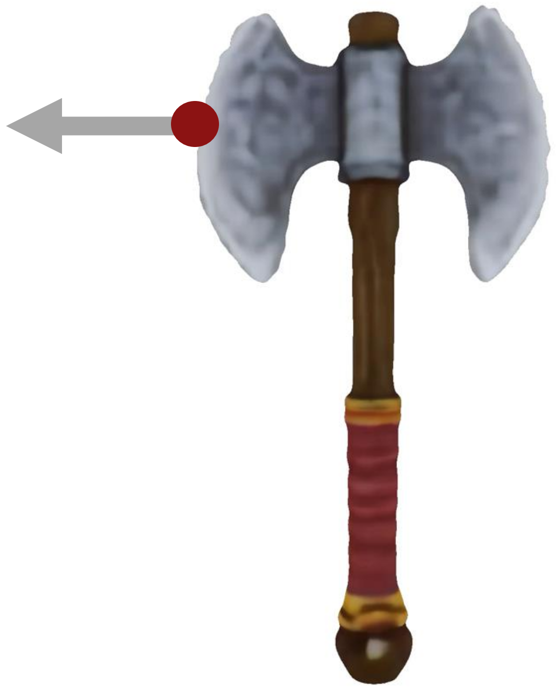
As-Rigid-As-Possible Surface Modeling,
Sorkine and Alexa, SGP 2007

Experiment Setup

We built **APAP-Bench**, a benchmark consisting of textured 2D and 3D triangular meshes spanning various object categories:

- **APAP-Bench 3D**
  - 10 textured 3D meshes from the ShapeNet dataset and LumaAI Genie.
- **APAP-Bench 2D**
  - 40 textured 2D meshes spanning 20 (non)organic categories;
  - Each category contains 1,000 images generated using Stable Diffusion-XL.

3D Mesh Deformation



Source

3D Mesh Deformation

Source

ARAP [1]

Ours

[1] As-Rigid-As-Possible Surface Modeling, Sorkine and Alexa, SGP 2007

3D Mesh Deformation

Source

3D Mesh Deformation

Source | ARAP [1] | **Ours**

[1] As-Rigid-As-Possible Surface Modeling, Sorkine and Alexa, SGP 2007

2D Mesh Deformation

We leverage 2D meshes from **APAP-Bench 2D** to measure a perceptual metric and conduct a user study using the results.

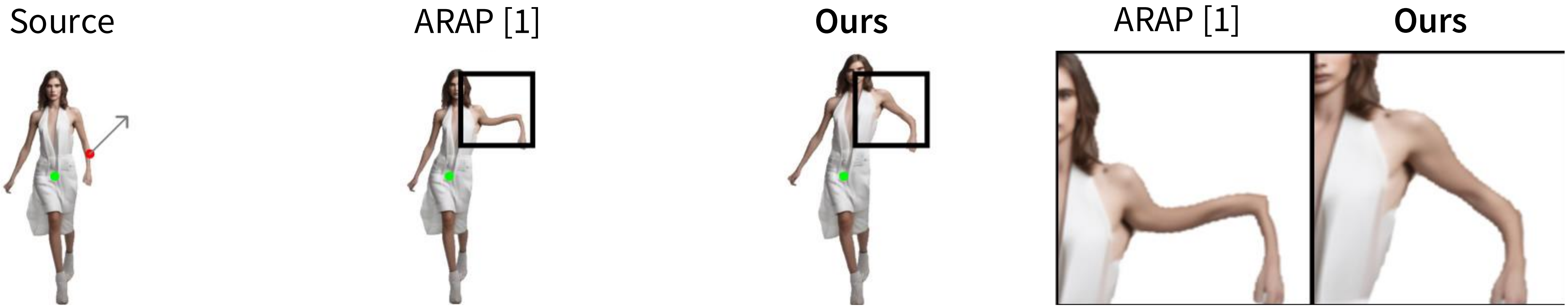
- k-NN GIQA Score [1]
  - The average of the inverse distances in InceptionNet feature space:

$$\mathrm{S}(\mathbf{x}) = \frac{1}{K}\sum_{k=1}^{K}\frac{1}{\|\mathbf{x}-\mathbf{x}_k\|^2}$$

- User Study
  - Binary selection between ARAP and ours measuring preference.

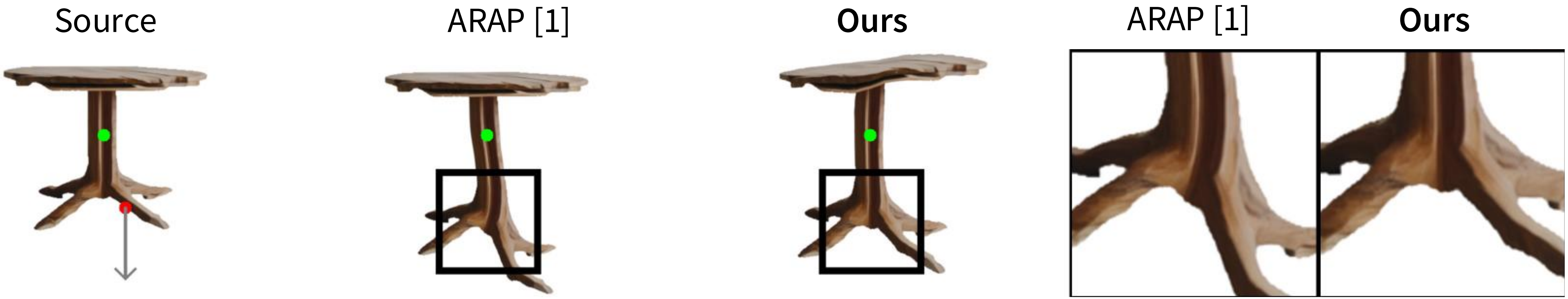
[1] Generated Image Quality Assessment, Gu *et al.*, ECCV 2020

2D Mesh Deformation



[1] As-Rigid-As-Possible Surface Modeling, Sorkine and Alexa, SGP 2007

2D Mesh Deformation



[1] As-Rigid-As-Possible Surface Modeling, Sorkine and Alexa, SGP 2007

2D Mesh Deformation

[1] As-Rigid-As-Possible Surface Modeling, Sorkine and Alexa, SGP 2007

2D Mesh Deformation

Our method surpasses the baseline in quantitative evaluations using a perceptual metric and user study.

Perceptual Metric

Methods	GIQA Score ($\times 10^{-2}$) ($\uparrow$)
ARAP [1]	4.753
Ours	**4.887**

User Study

Methods	Preference (%)
ARAP [1]	40.83
Ours	**59.17**

[1] As-Rigid-As-Possible Surface Modeling, Sorkine and Alexa, SGP 2007

2D Image Editing

Our method better preserves the identity while the SotA baseline suffers from artifacts caused by editing latent encodings directly.

Summary

In this work, we present

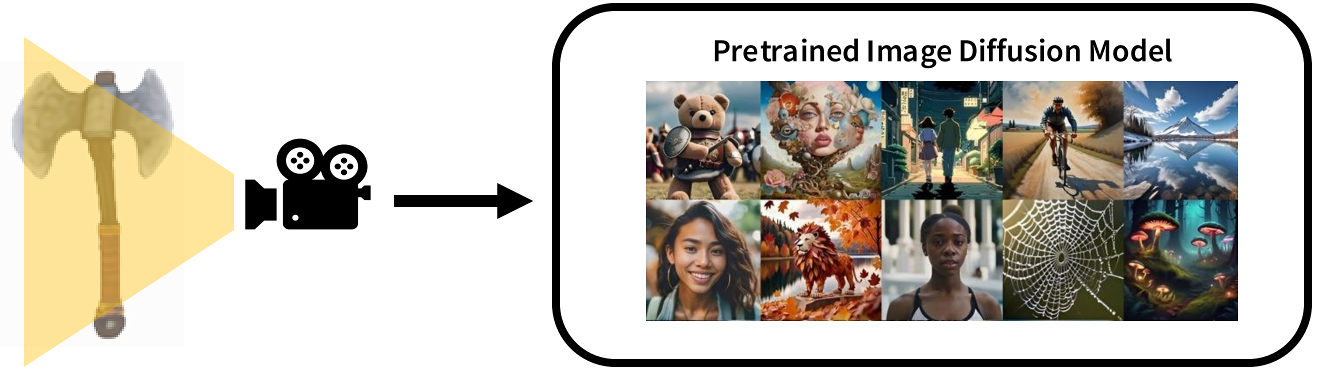
- APAP, a novel **mesh deformation technique using 2D diffusion priors**;
- An **iterative, gradient-based optimization** algorithm for **Jacobian fields**;
- APAP-Bench, a benchmark setup for **assessing visual plausibility** in deformation;
- Experimental analysis, including user study for **human-level assessment**.

Conclusion & Future Work

Conclusion

We discussed how learned priors can be incorporated into mesh deformation:

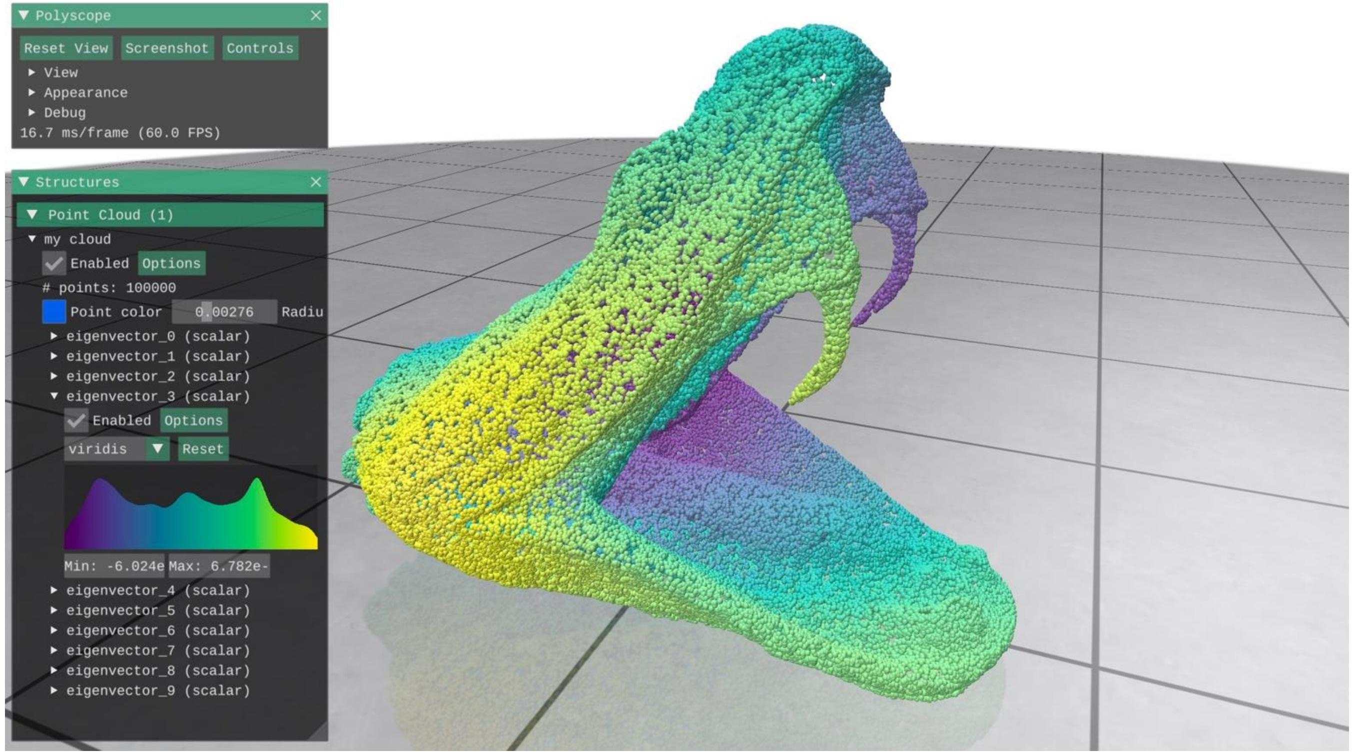
Representation Learning for Non-Rigid Object Pose Transfer



Plausibility-Aware Mesh Deformation Using 2D Diffusion Priors

Future Work

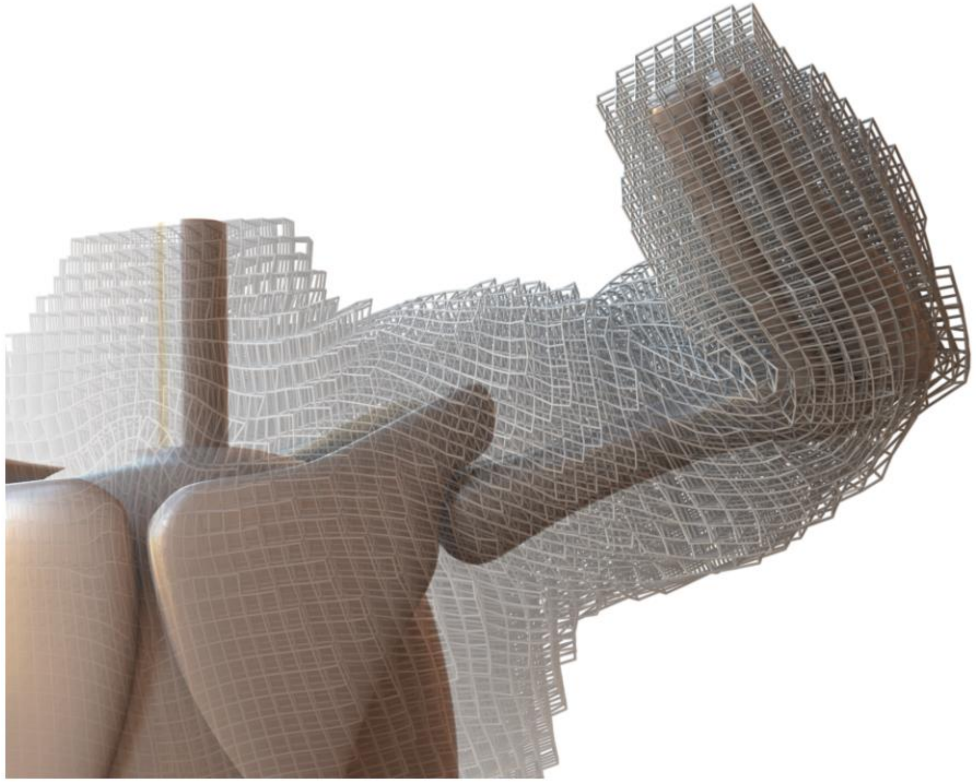
Develop versatile editing techniques for **other 3D representations**, including implicit functions, point clouds, and Gaussian Splats.



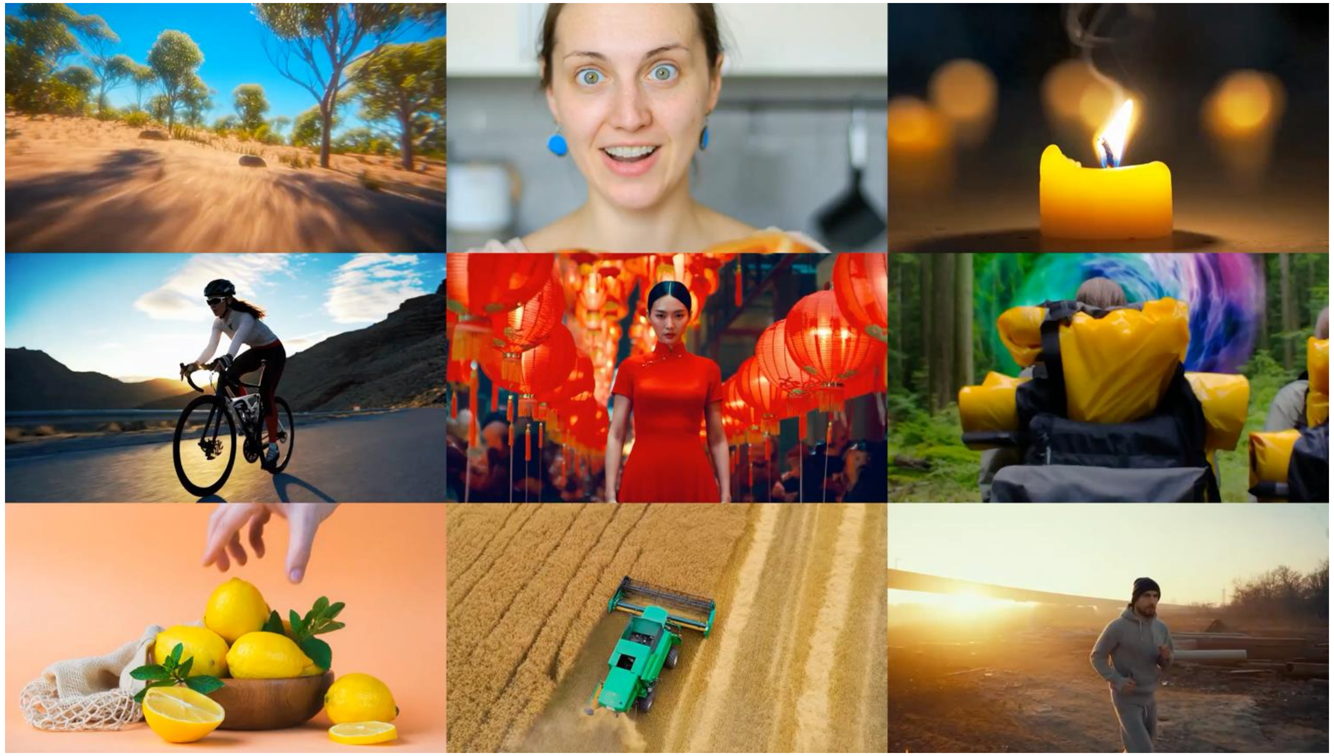
A Laplacian for Nonmanifold Triangle Meshes,
Sharp and Crane, SGP 2020

Future Work

Discover and learn **unknown physical models and parameters** governing deformations using **video generation models**.



Stable Neo-Hookean Flesh Simulation,
Smith *et al.*, ACM ToG 2018



Mochi 1, Genmo

Thank You