석 사 학 위 논 문
Master's Thesis

학습된 사전 지식을 활용한 메쉬 변형

Mesh Deformation Using Learned Priors

2025

유 승 우 (柳 承 佑 Yoo, Seungwoo)

한 국 과 학 기 술 원

Korea Advanced Institute of Science and Technology

석 사 학 위 논 문

학습된 사전 지식을 활용한 메쉬 변형

2025

유 승 우

한 국 과 학 기 술 원

전산학부

학습된 사전 지식을 활용한 메쉬 변형

유 승 우

위 논문은 한국과학기술원 석사학위논문으로
학위논문 심사위원회의 심사를 통과하였음

2024년 11월 26일

심사위원장 성 민 혁 (인)

심 사 위 원 김 민 혁 (인)

심 사 위 원 윤 성 의 (인)

Mesh Deformation Using Learned Priors

Seungwoo Yoo

Advisor: Minhyuk Sung

A thesis submitted to the faculty of
Korea Advanced Institute of Science and Technology in
partial fulfillment of the requirements for the degree of
Master of Science in Computer Science

Daejeon, Korea
November 26, 2024

Approved by

Minhyuk Sung
Professor of Computer Science

The study was conducted in accordance with Code of Research Ethics[1].

[1] Declaration of Ethical Conduct in Research: I, as a graduate student of Korea Advanced Institute of Science and Technology, hereby declare that I have not committed any act that may damage the credibility of my research. This includes, but is not limited to, falsification, thesis written by someone else, distortion of research findings, and plagiarism. I confirm that my thesis contains honest conclusions based on my own careful research under the guidance of my advisor.

MCS 유승우. 학습된 사전 지식을 활용한 메쉬 변형. 전산학부 . 2025년. 41+iv 쪽. 지도교수: 성민혁. (영문 논문)
Seungwoo Yoo. Mesh Deformation Using Learned Priors. School of Computing . 2025. 41+iv pages. Advisor: Minhyuk Sung. (Text in English)

초 록

본 학위논문에서는 데이터로부터 학습된 사전 지식을 활용하여 3차원 메쉬로 표현된 형상들을 변형하는 연구인 Neural Pose Representation Learning과 As-Plausible-As-Possible을 제안한다. 자연어 문장이나 사진과 달리 3차원 메쉬는 제작 과정의 복잡함으로 인해 대량의 데이터를 수집하는 것이 어렵기 때문에, 기존 데이터 기반 방법론은 충분한 데이터가 확보된 형상들에 대해서만 적용 가능하다는 한계를 가지고 있다. 이러한 문제를 해결하기 위해 본 논문에서는 단 하나의 형상과 해당 형상의 여러 변형 예시가 주어졌을 때 이 예시들을 다른 형상으로 전이하는 데 특화된 Neural Pose Representation과 이를 학습, 활용하기 위한 프레임워크를 제시한다. 새로이 제안된 Neural Pose Representation은 3차원 공간 상의 키포인트와 신경망이 예측한 특징 벡터로 구성되어 있으며, 메쉬의 국소적 변형을 표현하는 야코비 행렬을 예측하는 데 활용된다. 나아가 본 논문은 다량의 이미지로부터 학습된 2차원 확산 모델과 야코비장 기반 형상 표현을 결합한 As-Plausible-As-Possible을 통해 3차원 형상 예제 없이 학습 기반의 사전 지식을 활용하는 방법을 제안한다. 이때, As-Plausible-As-Possible은 사전 학습된 확산 모델을 활용해 계산된 손실 함수를 최소화하도록 야코비장을 변화시킴으로써 효과적으로 메쉬를 변형한다. 본 논문에 제시된 실험 결과는 제한된 수의 3차원 형상 데이터나 사진과 같은 타 도메인 데이터를 활용해 학습된 모델로부터 얻은 사전 지식만으로도 메쉬를 사실적으로 변형할 수 있음을 시사한다. 본 학위논문은 학위청구자가 주저자로 작성하여 출판한 논문 [114, 115]에 기초하여 작성되었다.

핵 심 낱 말 메쉬 변형, 자세 전이, 생성 모델, 컴퓨터 그래픽스, 컴퓨터 비전, 딥러닝

Abstract

We introduce Neural Pose Representation and As-Plausible-As-Possible, two novel methods for deforming 3D meshes by leveraging prior knowledge derived from data. Unlike text and image datasets, the creation of large-scale 3D mesh datasets is inherently challenging, which limits existing data-driven techniques to shape categories with sufficient training examples. To address this limitation, we first propose Neural Pose Representation, a framework designed for transferring object poses using only a single shape and its pose variations. This method employs a set of 3D keypoints and their associated neural features to predict local deformations represented as Jacobian matrices. For scenarios where no 3D shape exemplar is available, we introduce As-Plausible-As-Possible, which combines a pretrained 2D diffusion model with a shape representation based on Jacobian fields. This approach enables mesh deformation by leveraging a prior learned from data other than 3D shape examples. As-Plausible-As-Possible deforms the mesh by iteratively updating a Jacobian field representing a mesh while minimizing the loss function computed using a pretrained diffusion model. Experimental results demonstrate the effectiveness of leveraging prior knowledge, either learned from a limited number of 3D shape examples or from a pretrained model trained on data from other domains, such as 2D images, in generating realistic mesh deformations. This thesis is written based on two published papers [115, 114] that the candidate wrote as the first author.

Keywords Mesh Deformation, Pose Transfer, Generative Models, Computer Graphics, Computer Vision, Deep Learning

Contents

List of Tables

List of Figures

Chapter 1. Introduction

Techniques for editing triangular meshes have attracted interest from the research community, given the fundamental role of mesh representation in the computer graphics pipeline. Early studies grounded in geometric priors have led to well-established methods for mesh editing, including mesh deformation [91, 90, 102] and deformation transfer [24, 92, 7]. These techniques are successfully integrated into modern graphics applications, offering human artists efficient and intuitive control over meshes.

While methods based on geometric priors are straightforward to understand and implement, these handcrafted priors often fail to produce the desired deformations due to their inability to account for surface properties that are difficult to capture solely using geometric or physically-based formulations. Motivated by the success of deep learning, recent approaches have proposed learning deformation priors by leveraging large-scale shape datasets [11, 17], which contain hundreds or even thousands of shape examples. These learned priors [39, 61] demonstrate superior performance over those relying on geometric priors, particularly in capturing category-specific shape variations.

However, collecting 3D shapes and their deformed examples for training learning-based frameworks remains a significant challenge, posing a major barrier to replicating the success achieved in domains like text and images. As a result, the application of these techniques is often limited to shape categories with sufficient training examples. This challenge becomes even more severe in real-world scenarios where human artists often craft unique object designs or characters.

To address this, we move toward leveraging learned priors in a data-efficient manner, requiring only a few variations of a single shape or even no 3D shape examples. We begin by introducing Neural Pose Representation (**NPR**) in Ch. 2, a novel pose representation that facilitates pose generation and transfer of non-rigid objects, which can be learned using only a single shape and its multiple pose variations. The proposed representation effectively disentangles poses from identity-specific surface details, representing a pose as explicit 3D keypoints, each augmented with neural features derived from intrinsic surface properties. This hybrid design strengthens the transferability and generalizability of our representation by enabling distance-based queries, which are crucial for predicting implicit deformation fields represented as Jacobian fields [3] that deform a shape according to the given pose exemplars. Furthermore, the compactness of the pose representation facilitates diffusion model training [30, 88, 89, 77, 50], and when combined with its transferability, enables the generative modeling of poses across various shapes. Our extensive evaluations on animal [54] and human body [63] shape datasets demonstrate superior performance compared to learning-based baselines.

As our next step, we address an even more challenging scenario where no 3D shape examples are available for training priors in Ch. 3. Our key idea to this seemingly paradoxical problem is to distill the prior knowledge of a 2D diffusion model [77], trained on an Internet-scale dataset [80] by bridging the gap between the 2D image space and 3D geometry using a differentiable renderer [51]. This approach allows us to replace geometric deformation energies [91, 59, 90] with an image-based loss [71] defined using the 2D diffusion model that assesses the realism of the deformed object's image. Similar to geometric deformation techniques [91, 59, 90], we optimize this loss to produce shape deformations that, when rendered, closely resemble the images used to train the diffusion model. This idea has been infused into the design of As-Plausible-As-Possible (**APAP**), our novel mesh deformation framework that incorporates visual plausibility in shape manipulation. Specifically, **APAP** iteratively updates the given

shape while minimizing the image-based loss [71] to improve visual plausibility, along with a geometric loss that enforces user-defined handle constraints. During deformation, we employ the Jacobian field [3] of the input shape as an optimization variable and update it using gradients from the loss functions to preserve geometric details throughout the process. Comparisons with a baseline [90] that relies solely on geometric energy show the effectiveness of utilizing the prior learned by a 2D diffusion model, as highlighted by the highest k-NN GIQA score [25] and greater preference in a user study.

Chapter 2. Neural Pose Representation Learning

2.1 Introduction

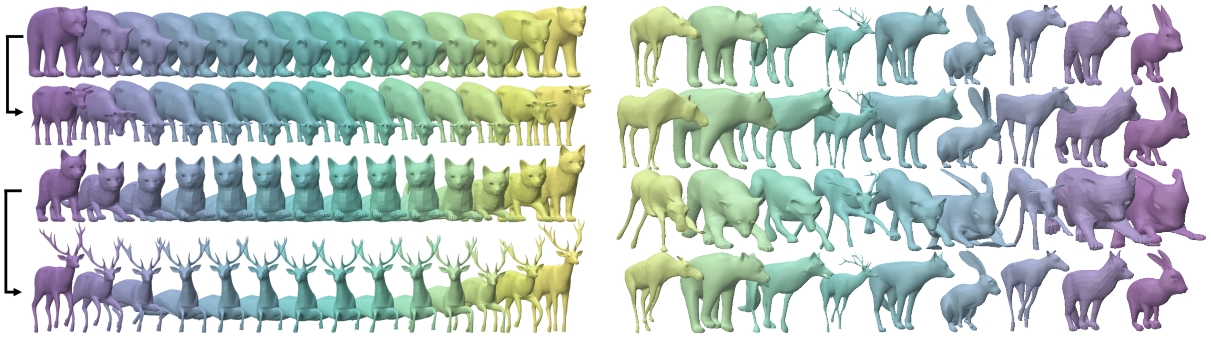

Figure 2.1: Results of motion sequence transfer (left) and shape variation generation (right) using the proposed neural pose representation. On the left, poses from source shapes (first and third rows) are transferred to target shapes (second and fourth rows), preserving intricate details like horns and antlers. On the right, new poses sampled from a cascaded diffusion model, trained with shape variations of the bunny (last column), are transferred to other animal shapes.

The recent great success of generative models [30, 88, 89, 87] has been made possible not only due to advances in techniques but also due to the enormous scale of data that has become available, such as LAION [79] for 2D image generation. For 3D data, the scale has been rapidly increasing, as exemplified by ObjaverseXL [15]. However, it is still far from sufficient to cover all possible 3D shapes, particularly deformable, non-rigid 3D shapes such as humans, animals, and characters. The challenge with deformable 3D shapes is especially pronounced due to the diversity in *both* the *identities* and *poses* of the objects. Additionally, for a new 3D character created by a designer, information about possible variations of the creature does not even exist.

To remedy the requirement of a large-scale dataset for 3D deformable shape generation, we aim to answer the following question: Given variations of a *single* deformable object with its different poses, how can we effectively learn the pose variations while factoring out the object's identity and also make the pose information applicable to other objects? For instance, when we have a variety of poses of a bear (Fig. 2.1 left, first row), our objective is to learn the space of poses without entangling them with the geometric characteristics of the bears. Also, we aim to enable a sample from this space to be applied to a new object, such as a bull, to generate a new shape (Fig. 2.1 left, second row). We believe that such a technique, effectively separating pose from the object's identity and enabling the transfer of poses to other identities, can significantly reduce the need for collecting large-scale datasets covering the diversity of both object identities and poses. This approach can even enable creating variations of a new creature without having seen any possible poses of that specific object.

Transferring poses from one object to another has been extensively studied in computer graphics and vision, with most methods requiring target shape supervision [92, 8, 111, 7, 22] or predefined pose parameterization [24, 14, 52, 93, 4, 110, 75, 18, 99, 40, 56, 98, 13]. Without such additional supervision, our key idea for extracting identity-agnostic pose information and learning their variations is to introduce

a novel pose representation along with associated encoding and decoding techniques. For this, we consider the following three desiderata:

1. **Pose Disentanglement**: The representation should effectively represent the pose only without resembling the source object's identity when applied to the other object.
2. **Compactness**: The representation should be compact enough to effectively learn its variation using a generative model, such as a diffusion model.
3. **Transferability**: The encoded pose information should be applicable to new target objects.

As a representation that meets the aforementioned criteria, we propose an autoencoding framework and a latent diffusion model with three core components. Firstly, we design a *pose extractor* and a *pose applier* to encode an **implicit deformation field** with a **keypoint-based hybrid representation**, comprising 100 keypoints in the space, each associated with a latent feature. Learning the deformation field enables *disentangling* the pose information from the object's identity, while the keypoint-based representation *compactly* encodes it and makes it *transferable* to other objects. However, simply learning the deformation as a new position of the vertex is not sufficient to properly adapt the source object's pose information to others. Hence, secondly, we propose predicting an **intrinsic property** of the deformed mesh, **Jacobian fields** [116, 59, 91, 3], which can successfully apply the pose while preserving the identity of the target shape. To better preserve the target's identity while applying the pose variation from the source, thirdly, we propose a **per-identity refinement step** that fine-tunes the decoder in a *self-supervised* way to adapt to the variations of target shapes, with poses transferred from the source object. Thanks to the compact hybrid representation of pose, a pose generative model can also be effectively learned using **cascaded diffusion models** [31, 50], enabling the generation of varying poses of an object with an arbitrary identity different from the source object.

In our experiments, we compare our framework against state-of-the-art techniques for pose transfer on animals (Sec. 2.4.2) and humans (Sec. 2.4.3). Both qualitative and quantitative analyses underscore the key design factors of our framework, demonstrating its efficiency in capturing identity-agnostic poses from exemplars and its superior performance compared to existing methods. Additionally, we extend the proposed representation to the task of unconditional generation of shape variations. Our representation serves as a compact encoding of poses that can be generated using diffusion models (Sec. 2.4.6) and subsequently transferred to other shapes. This approach facilitates the generation of various shapes, particularly in categories where exemplar collection is challenging.

2.2 Related Work

Due to space constraints, we focus on reviewing the literature on non-rigid shape pose transfer, including methods that operate without parameterizations, those that rely on predefined parameterizations, and recent learning-based techniques that derive parameterizations from data.

Parameterization-Free Pose Transfer. Early works [92, 107, 8, 111] focused on leveraging *point-wise* correspondences between source and target shapes. A seminal work by Sumner and Popović [92] transfers per-triangle affine transforms applied to the target shape by solving an optimization problem. A follow-up work by Ben-Chen *et al.* [8] transfers deformation gradients by approximating source deformations using harmonic bases. On the other hand, a technique proposed by Baran *et al.* [7] instead

employs *pose-wise* correspondences by learning shape spaces from given pairs of poses shared across the source and target identities. The poses are transferred by blending existing exemplars. While these techniques require point-wise or pose-wise correspondence supervision, our method does not require such supervision during training or inference.

Skeleton- or Joint-Based Pose Transfer. Another line of work utilizes handcrafted skeletons, which facilitate pose transfer via motion retargeting [24]. This approach has been extended by incorporating physical constraints [14, 52, 93, 4] or generalizing the framework to arbitrary objects [110, 75]. Several learning-based methods [18, 99, 40, 56, 98] have also been proposed to predict joint transformations involved in forward kinematics from examples. Recently, Chen *et al.* [13] proposed a framework that does not require skeletons during test time by predicting keypoints at joints. The method is trained to predict both relative transformations between corresponding keypoints in two distinct kinematic trees and skinning weights. However, the tasks of rigging and skinning are labor-intensive, and different characters and creatures often require distinct rigs with varying topologies. Liao *et al.* [55] notably presented a representation that comprises character-agnostic deformation parts and a semi-supervised network predicting skinning weights that link each vertex to these deformation parts, although its performance hinges on accurate skinning weight prediction. In this work, we design a more versatile framework that is applicable to various shapes and provides better performance.

Pose Transfer via Learned Parameterization. To bypass the need for correspondence or parameterization supervision, learning-based approaches [22, 113, 101, 122, 12, 3, 55, 100, 85, 86] explore alternative parameterizations *learned* from exemplars. Yifan *et al.* [113] propose to predict source and target cages and their offsets simultaneously, although their method still requires manual landmark annotations. Gao *et al.* [22] introduce a VAE-GAN framework that takes *unpaired* source and target shape sets, each containing its own set of pose variations. The network is trained without direct pose-wise correspondences between samples from these sets, instead enforcing cycle consistency between latent representations. Although this work relaxes the requirement for correspondence supervision, it still requires pose variations for *both* the source and target identities and individual training for each new source-target pair. Numerous works [101, 12, 122, 3] lift the requirement for gathering variations of target shapes by disentangling identities from poses, enforcing cycle consistency [122], or adapting conditional normalization layers [101] from image style transfer [68]. Notably, Aigerman *et al.* [3] train a network that regresses Jacobian fields from SMPL [63] pose parameters. The vertex coordinates are computed by solving Poisson's equation [116], effectively preserving the shapes' local details. Wang *et al.* [100] also train a neural implicit function and retrieve a shape latent from a template mesh of an unseen identity via autodecoding [67]. This method models local deformations through a coordinate-based network that learns continuous deformation fields. However, such methods struggle to generalize to unseen identities due to their reliance on *global* latent embeddings encoding shapes. We propose a representation that not only disentangles poses from identities but also allows for implicit queries using the surface points, thereby improving generalization to new identities.

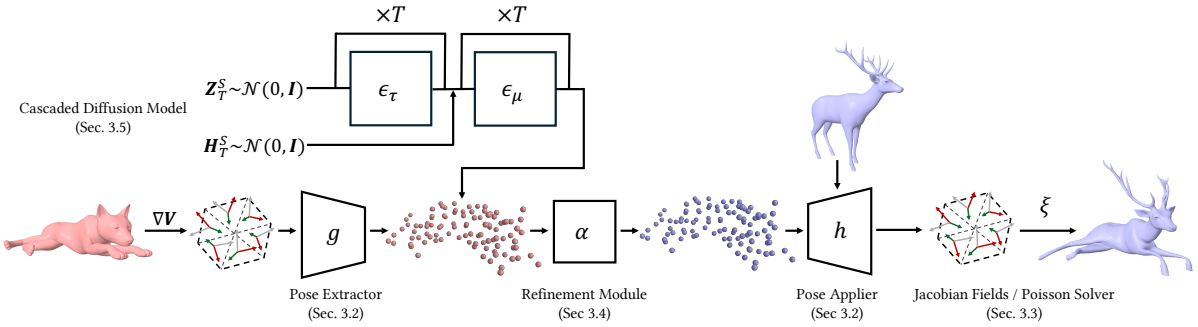

Figure 2.2: Method overview. Our framework extracts keypoint-based hybrid pose representations from Jacobian fields. These fields are mapped by the pose extractor g and mapped back by the pose applier h. The pose applier, conditioned on the extracted pose, acts as an implicit deformation field for various shapes, including those unseen during training. A refinement module α, positioned between g and h, is trained in a self-supervised manner, leveraging the target's template shape. The compactness of our latent representations facilitates the training of a diffusion model, enabling diverse pose variations through generative modeling in the latent space.

2.3 Method

2.3.1 Problem Definition

Consider a *source template mesh* $\overline{\mathcal{M}}^S = (\overline{\mathbf{V}}^S, \mathbf{F}^S)$, given as a 2-manifold triangular mesh. The mesh comprises vertices $\overline{\mathbf{V}}^S$ and faces $\mathbf{F}^S$. Suppose there exist N variations of the source template mesh, $\{\mathcal{M}_1^S, \ldots, \mathcal{M}_N^S\}$, where each $\mathcal{M}_i^S = (\mathbf{V}_i^S, \mathbf{F}^S)$ is constructed with a different *pose*, altering the vertex positions while sharing the same mesh connectivity $\mathbf{F}^S$.

Assume a *target template mesh* $\overline{\mathcal{M}}^T = (\overline{\mathbf{V}}^T, \mathbf{F}^T)$ is given without any information about its variations or existing pose parameterization (e.g., , skeletons or joints). Our goal is to define functions g and h that can *transfer* the pose variations from the source meshes to the target template mesh. For each variation of the source shape $\mathcal{M}_i^S$, its corresponding mesh $\mathcal{M}_i^T$ for the target is obtained as:

$$\mathcal{M}_i^T = (h(g(\mathcal{M}_i^S), \overline{\mathcal{M}}^T), \mathbf{F}^T), \quad \text{for } i = 1, 2, \cdots, N. \tag{2.1}$$

Specifically, we design g as a *pose extractor* that embeds a source object mesh $\mathcal{M}_i^S$ into a *pose latent representation* $\mathcal{Z}_i^S = g(\mathcal{M}_i^S)$. This representation disentangles the pose information from the object's identity in $\mathcal{M}_i^S$ and facilitates transferring the pose to the target template mesh $\overline{\mathcal{M}}^T$. Given this pose representation, the *pose applier* h then applies the pose to $\overline{\mathcal{M}}^T$, yielding the corresponding variation of the target object $\mathcal{M}_i^T = h(\mathcal{Z}_i^S, \overline{\mathcal{M}}^T)$. Note that our method is not limited to transferring the pose of a *given* variation of the source object to the target mesh but can also apply a pose *generated* by a diffusion model to the target mesh. In Sec. 2.3.5, we explain how a diffusion model can be trained with the latent pose representation extracted from source object variations. In the following, we first describe the key design factors of the functions g and h to tackle the problem.

2.3.2 Keypoint-Based Hybrid Pose Representation

To encode a source shape $\mathcal{M}^S$ into a latent representation $\mathcal{Z}^S$, we consider its vertices $\mathbf{V}^S$ as its geometric representation and use them as input to the pose extractor g, which is designed as a sequence of Point

Transformer [120, 94] layers. These layers integrate vector attention mechanisms [119] with progressive downsampling of input point clouds. The output of the pose extractor g is a set of unordered tuples $\mathcal{Z}^S = \{(\mathbf{z}_k^S, \mathbf{h}_k^S)\}_{k=1}^K$ where $\mathbf{z}_k^S \in \mathbb{R}^3$ represents a 3D coordinate of a keypoint subsampled from $\mathbf{V}^S$ via farthest point sampling (FPS) and $\mathbf{h}_k^S$ is a learned feature associated with $\mathbf{z}_k^S$. This set $\mathcal{Z}^S$ forms a sparse point cloud of keypoints in 3D space, augmented with latent features. We set $K = 100$ in our experiments.

This keypoint-based hybrid representation, visualized in Fig. 2.2, is designed to exclusively transfer pose information from the source to the target while preventing leakage of the source shape's identity characteristics. Since the keypoints $\{\mathbf{z}_k^S\}$ are sampled from $\mathbf{V}^S$ using FPS, they effectively capture the overall pose structure of $\mathbf{V}^S$ while also supporting geometric queries with the vertices of a new mesh. This property is essential during the decoding phase, where the pose applier h predicts the pose-applied mesh from the input template as an *implicit deformation field*.

The pose applier h is implemented with a neural network that takes the 3D coordinates of a vertex from the input template mesh as a query, along with the hybrid pose latent representation $\mathcal{Z}$, and outputs the new position of the vertex in the pose-applied deformed mesh. Note that h indicates a function that collectively maps all vertices in the input template mesh to their new positions using the network. Like the pose extractor g, the implicit deformation network is also parameterized as Point Transformer layers [120]. It integrates the pose information encoded in $\mathcal{Z}^S$ by combining vector attention mechanisms with nearest neighbor queries to aggregate features of the keypoints $\mathbf{z}_k^S$ around each query point. The aggregated features are then decoded by an MLP to predict the vertex coordinates of the deformed shape. (This is a base network, and we also introduce a *better* way to design the implicit deformation network in Sec. 2.3.3.)

Given only the variations of the source object $\{\mathcal{M}_1^S, \ldots, \mathcal{M}_N^S\}$, we jointly learn the functions g and h by reconstructing the variations of the source object as a deformation of its template:

$$\mathcal{L}_V = \|\mathbf{V}_i^S - h(g(\mathcal{M}_i^S), \overline{\mathcal{M}}^S)\|^2. \tag{2.2}$$

While g and h are trained using the known variations of the source object $\mathcal{M}^S$, the latent representation $\mathcal{Z}^S = g(\mathcal{M}^S)$, when queried and decoded with the target template mesh, effectively transfers the pose extracted by g from $\mathcal{M}^S$. However, we also observe that g and h, when trained using the loss $\mathcal{L}_V$, often result in geometry with noticeable imperfections and noise on the surfaces. To address this, we explore an alternative representation of a mesh that better captures and preserves geometric details, which will be discussed in the following section.

2.3.3 Representing Shapes as Jacobian Fields

In this work, we advocate employing the differential properties of surfaces as dual representations of a mesh. Of particular interest are *Jacobian fields*, a gradient-domain representation noted for its efficacy in preserving local geometric details during deformations [3], while ensuring that the resulting surfaces maintain smoothness [23].

Given a mesh $\mathcal{M} = (\mathbf{V}, \mathbf{F})$, a Jacobian field $\mathbf{J}$ represents the spatial derivative of a scalar-valued function $\phi : \mathcal{M} \to \mathbb{R}$ defined over the surface. We discretize ϕ as $\phi_{\mathbf{V}} \in \mathbb{R}^{|\mathbf{V}|}$, sampling its value at each vertex $\mathbf{v}$ of the vertex set $\mathbf{V}$. The spatial derivative of ϕ at each triangle $\mathbf{f} \in \mathbf{F}$ is computed as $\boldsymbol{\nabla}_{\mathbf{f}}\phi_{\mathbf{V}}$ using the per-triangle gradient operator $\boldsymbol{\nabla}_{\mathbf{f}}$. Given that each dimension of vertex coordinates $\mathbf{V}$ is such a function, we compute its spatial gradient at each triangle $\mathbf{f}$ as $\mathbf{J}_{\mathbf{f}} = \boldsymbol{\nabla}_{\mathbf{f}}\mathbf{V}$. Iterating this process for all triangles yields the Jacobian field $\mathbf{J} = \{\mathbf{J}_{\mathbf{f}} | \mathbf{f} \in \mathbf{F}\}$.

To recover $\mathbf{V}$ from a given Jacobian field $\mathbf{J}$, we solve a least-squares problem, referred to as Poisson's equation:

$$\mathbf{V}^* = \underset{\mathbf{V}}{\operatorname{argmin}} \|\mathbf{L}\mathbf{V} - \boldsymbol{\nabla}^T \mathcal{A}\mathbf{J}\|^2, \tag{2.3}$$

where $\mathbf{L} \in \mathbb{R}^{|\mathbf{V}|\times|\mathbf{V}|}$ is the cotangent Laplacian of $\mathcal{M}$, $\boldsymbol{\nabla}$ is the stack of gradient operators defined at each $\mathbf{f} \in \mathbf{F}$, and $\mathcal{A} \in \mathbb{R}^{3|\mathbf{F}|\times 3|\mathbf{F}|}$ is the diagonal mass matrix, respectively. Since the rank of $\mathbf{L}$ is at most $|\mathbf{V}|-1$, we can obtain the solution by fixing a single point, which is equivalent to eliminating one row of the system in Eqn. 2.3. Since $\mathbf{L}$ in Eqn. 2.3 remains constant for a given shape $\mathcal{M}$, we can prefactorize the matrix (e.g., using Cholesky decomposition) and quickly solve the system for different Jacobian fields $\mathbf{J}$'s. Furthermore, the upstream gradients can be propagated through the solver since it involves only matrix multiplications [3].

Employing Jacobian fields as shape representations, we now modify the implicit deformation network described in Sec. 2.3.2 to take *face center coordinates* as input instead of vertex coordinates and to output a new *face Jacobian* for a query face instead of a new vertex position. This results in decomposing h into two functions $h = (\xi \circ h')$, where h' is a function that collectively maps all the faces in the input mesh to the new Jacobian, and ξ is a differentiable Poisson solver layer. Both g and h' are then trained by optimizing the following loss:

$$\mathcal{L}_J = \|\mathbf{V}_i^S - \xi(h'(g(\mathcal{M}_i^S), \overline{\mathcal{M}}^S))\|^2. \tag{2.4}$$

2.3.4 Per-Identity Refinement using Geometric Losses

While the latent pose representation $\mathcal{Z}$ learned by g and h exhibits promising generalization capabilities in transferring poses, the quality of the transferred shapes can be further improved by incorporating a trainable, identity-specific refinement module into our system. This module is trained in a *self-supervised* manner with the set of pose-applied target meshes. Similarly to techniques for personalized image generation [32, 112], we introduce a shallow network α between g and h, optimizing its parameters while keeping the rest of the pipeline frozen.

The optimization of α is driven by geometric losses, aiming to minimize the geometric discrepancies in terms of the object's identity between the target template mesh $\overline{\mathcal{M}}^T$ and the pose-transferred meshes $\mathcal{M}_i^T$. In particular, we first extract poses $\{\mathcal{Z}_1^S, \ldots, \mathcal{Z}_N^S\}$ corresponding to the known shapes $\{\mathcal{M}_1^S, \ldots, \mathcal{M}_N^S\}$ of the source object. A transformer-based network α, which maps a latent representation $\mathcal{Z}^S$ to $\mathcal{Z}^{S\prime}$, is plugged in between the pose extractor g and the pose applier h :

$$\mathbf{V}_i^T = h(\alpha(\mathcal{Z}_i^S), \overline{\mathcal{M}}^T). \tag{2.5}$$

The parameters of α are updated by optimizing the following loss function:

$$\mathcal{L}_{\text{ref}} = \lambda_{\text{lap}}\mathcal{L}_{\text{lap}}(\mathbf{V}_i^T, \overline{\mathbf{V}}^T) + \lambda_{\text{edge}}\mathcal{L}_{\text{edge}}(\mathbf{V}_i^T, \overline{\mathbf{V}}^T) + \lambda_{\text{reg}}(\sum_k \|\mathbf{z}_k^S - \mathbf{z}_k^{S\prime}\|^2 + \|\mathbf{h}_k^S - \mathbf{h}_k^{S\prime}\|^2), \tag{2.6}$$

where $\mathcal{L}_{\text{lap}}(\cdot)$ is the mesh Laplacian loss [61], $\mathcal{L}_{\text{edge}}(\cdot)$ is the edge length preservation loss [55], and λ_{lap}, λ_{edge}, and λ_{reg} are the weights of the loss terms. Specifically, $\mathcal{L}_{\text{lap}}(\cdot)$ encourages the preservation of Laplacian energy by minimizing the discrepancy between the Laplacians of the target template $\overline{\mathcal{M}}^T$ and the pose-transferred mesh $\mathcal{M}_i^T$:

$$\mathcal{L}_{\text{lap}}\left(\mathbf{V}_i^T, \overline{\mathbf{V}}^T\right) = \mathbf{L}_T\left(\mathbf{V}_i^T - \overline{\mathbf{V}}^T\right), \tag{2.7}$$

where $\mathbf{L}_T$ is the cotangent Laplacian matrix of $\overline{\mathcal{M}}^T$. Simultaneously, $\mathcal{L}_{\text{edge}}(\cdot)$ mitigates abrupt shape changes by penalizing the differences in edge lengths between $\overline{\mathcal{M}}^T$ and $\mathcal{M}_i^T$:

$$\mathcal{L}_{\text{edge}}(\mathbf{V}_i^T, \overline{\mathbf{V}}^T) = \sum_{\{j,k\}\in\mathcal{E}} |\|\mathbf{V}_{i,j}^T - \mathbf{V}_{i,k}^T\|_2 - \|\overline{\mathbf{V}}_j^T - \overline{\mathbf{V}}_k^T\|_2|, \tag{2.8}$$

where $\mathcal{E}$ is the set of edges defining the target shapes, and $\mathbf{V}_{i,j}^T$ and $\overline{\mathbf{V}}_j^T$ represent the coordinates of the j-th vertex in $\mathbf{V}_i^T$ and $\overline{\mathbf{V}}^T$, respectively. Note that this refinement step leverages only the originally provided template shape $\overline{\mathcal{M}}^T$ and does not require its given variations or any other additional supervision.

2.3.5 Learning Latent Diffusion via Cascaded Training

The use of the keypoint-based hybrid representation discussed in Sec. 2.3.2 offers a compact latent space suitable for generative modeling using diffusion models [88, 30, 89, 87]. Unlike the Jacobian fields with dimensionality $|\mathbf{F}| \times 9$, the keypoints and their feature vectors that comprise the pose representation $\mathcal{Z}^S$ lie in significantly lower dimensional space, facilitating generative modeling with latent diffusion models [77].

We employ a cascaded diffusion framework [31, 50] to separately capture the layouts of keypoints and the associated feature vectors. Given a set $\{\mathcal{Z}_1^S, \ldots, \mathcal{Z}_N^S\}$ of N latent embeddings extracted from the known source shape variations $\{\mathcal{M}_1^S, \ldots, \mathcal{M}_N^S\}$, we first learn the distribution over $\mathbf{Z}^S = \{\mathbf{z}_k^S\}_{k=1}^K$. To handle unordered sets with small cardinality, we employ a transformer-based network to facilitate interactions between each element within the noise prediction network $\boldsymbol{\epsilon}_\tau(\mathbf{Z}_t^S, t)$ where t is a diffusion timestep and $\mathbf{Z}_t^S$ is a noisy 3D point cloud obtained by perturbing a clean keypoint set $\mathbf{Z}_0^S\ (= \mathbf{Z}^S)$ via forward diffusion process [30]. We train the network by optimizing the denoising loss:

$$\mathcal{L}_{\mathbf{Z}^S} = \mathbb{E}_{\mathbf{Z}^S, \boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I}), t\sim\mathcal{U}(0,1)}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\tau(\mathbf{Z}_t^S, t)\|^2\right]. \tag{2.9}$$

Likewise, the distribution of the set of latent features $\mathbf{H}^S = \{\mathbf{h}_k^S\}_{k=1}^K$ is modeled as a conditional diffusion model $\boldsymbol{\epsilon}_\mu$, which takes $\mathbf{Z}^S$ as an additional input to capture the correlation between $\mathbf{Z}^S$ and $\mathbf{H}^S$. The network is trained using the same denoising loss:

$$\mathcal{L}_{\mathbf{H}^S} = \mathbb{E}_{\mathbf{H}^S, \boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I}), t\sim\mathcal{U}(0,1)}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\mu(\mathbf{H}_t^S, \mathbf{Z}^S, t)\|^2\right]. \tag{2.10}$$

Once trained, the models can sample new pose representations through the reverse diffusion steps [30]:

$$\mathbf{Z}_{t-1}^S = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{Z}_t^S - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\tau\left(\mathbf{Z}_t^S, t\right)\right), \tag{2.11}$$

$$\mathbf{H}_{t-1}^S = \frac{1}{\sqrt{\alpha_t}}\left(\mathbf{H}_t^S - \frac{\beta_t}{\sqrt{1-\bar{\alpha}_t}}\boldsymbol{\epsilon}_\mu\left(\mathbf{H}_t^S, \mathbf{Z}_0^S, t\right)\right), \tag{2.12}$$

where α_t, $\bar{\alpha}_t$, and β_t are the diffusion process coefficients.

2.4 Experiments

2.4.1 Experiment Setup

Datasets. In our experiments, we consider animal and human shapes that are widely used in various applications. For the animal shapes, we utilize animation sequences from the DeformingThings4D-Animals dataset [54]. Specifically, we extract 300 meshes from the animation sequences of each of 9

different animal identities, spanning diverse species: Bear, Bunny, Canine, Deer, Dog, Elk, Fox, Moose, and Puma. We use the first frame of the first animation sequence (alphabetically ordered) as the template for each identity, and the last frame of randomly sampled animation sequences for pose variations. For humanoids, we use SMPL [63, 69], which facilitates easy generation of synthetic data for both training and testing. We sample 300 random pose parameters from VPoser [69] to generate variations of an unclothed human figure using the *default* body shape parameters, which are used to train our networks. For testing, we keep the pose parameters constant and sample 40 different body shapes from the parametric space covered by the unit Gaussian. This produces 40 different identities, each in 300 poses. The generated meshes serve as the ground truth for pose transfer. To assess the generalization capability to unusual identities that deviate significantly from the default body shape, we increase the standard deviation to 2.5 when sampling SMPL body parameters for 30 of the 40 identities. Additionally, we collect 9 stylized character meshes from the Mixamo dataset [1] to test the generalizability of different methods. In both SMPL [63] and Mixamo [1] datasets, T-posed humanoid shapes are used as templates. For diffusion model training, the extracted keypoints and their associated features from the given set of source meshes are used as the training data for our cascaded diffusion model, which is trained separately for each identity.

Baselines. To assess the performance of pose transfer, we compare our method against NJF [3], SPT [55], ZPT [100], and various modifications of our framework. For NJF [3], we use the official code from the *Morphing Humans* experiment, employing a PointNet [72] encoder to map input shapes to global latents, and we train the model on our datasets. For SPT [55], we use the official code and pretrained model on humanoid shapes. Since a pretrained model for animal shapes is not provided, the comparison with SPT on the DeformingThings4D-Animals dataset is omitted. For ZPT [100], the official implementation is not provided, so we implemented the model based on the description in the paper. In our ablation study, we explore different variations of our method to assess their impact on performance, including: (1) using vertex coordinates as shape representations (as described in Sec. 2.3.2), and (2) omitting the per-identity refinement module (Sec. 2.3.4). All models (except for SPT [55], for which we employ a pretrained model) are trained for *each* shape identity.

Evaluation Metrics. For pose transfer, when the corresponding shapes of the same pose are given for both source and target shapes, in the SMPL case, we measure accuracy using Point-wise Mesh Euclidean Distance (PMD) [122, 101], following our baselines[55, 100]. Note that this measurement cannot be applied in the DeformingThings4D-Animals case since pose-wise correspondences are not provided. For both pose transfer and shape generation (via pose generation), we measure the visual plausibility of the output meshes using FID [29], KID [9], and ResNet classification accuracy with images rendered from four viewpoints (front, back, left, and right) without texture. For the latter, we train a ResNet-18 [26] network using 10,800 images rendered from the four viewpoints of all ground truth shape variations of each animal.

Implementation Details. We utilize Point Transformer layers from Zhao *et al.* [120] and Tang *et al.* [94] for implementing the pose extractor g and the pose applier h. The network architectures for our cascaded diffusion models, as detailed in Sec. 2.4.6, are adapted from Koo *et al.* [50]. These models operate over $T = 1000$ timesteps with a linear noise schedule ranging from $\beta_1 = 1 \times 10^{-4}$ to $\beta_T = 5 \times 10^{-2}$. For model training, we employ the ADAM optimizer at a learning rate of 1×10^{-3}

and standard parameters. Our experiments are conducted on RTX 3090 GPUs (24 GB VRAM) and A6000 GPUs (48 GB VRAM). For per-identity refinement modules, we set $\lambda_{\text{lap}} = 1.0$, $\lambda_{\text{edge}} = 1.0$, and $\lambda_{\text{reg}} = 5 \times 10^{-2}$ during training.

2.4.2 Pose Transfer on DeformingThings4D-Animals

We begin our experiments by transferring pose variations across different animals, a challenging task that necessitates strong generalization capabilities due to the diverse shapes of the animals involved.

	DeformingThings4D-Animals [54]			SMPL [63]			
	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 [3]	11.33	5.71	64.43	2.55	1.57	0.82	70.93
SPT [55]	-	-	-	0.28	0.83	0.43	75.38
ZPT [100]	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**

Table 2.1: Quantitative results on the experiments using the DeformingThings4D-Animals dataset [54] and the human shape dataset populated using SMPL [63].

We summarize the quantitative metrics in Tab. 2.1 (left). Our method outperforms NJF [3] and ZPT [100], both of which use global latent codes to encode shapes, while ours uses a keypoint-based hybrid representation. Note that SPT [55] is not compared in this experiment since the pretrained model is not provided for animal shapes. Qualitative results are also shown in Fig. 2.3, demonstrating the transfer of poses from a source mesh $\mathcal{M}^S$ (second and seventh column, *red*) to a target template mesh $\overline{\mathcal{M}}^T$ (first and sixth column, *blue*). Both NJF [3] and ZPT [100] introduce significant distortions to the results and often fail to properly align the pose extracted from the source to the target. Our method, on the other hand, effectively transfers poses to the targets while preserving local geometric details.

2.4.3 Pose Transfer on SMPL and Mixamo

We further test our method and baselines using humanoid shapes ranging from SMPL to stylized characters from the Mixamo [1] dataset. While we employ the parametric body shape and pose model of SMPL [63, 69], it is important to note that this is only for evaluation purposes; our method does not assume any parametric representations, such as skeletons, for either training or inference.

Tab. 2.1 (right) summarizes the evaluation metrics measured across the 40 target shapes. Notably, our method achieves lower PMD than SPT [55], which is trained on a large-scale dataset consisting of diverse characters and poses, while ours is trained using only 300 pose variations of the default human body. This is further illustrated in the qualitative results in Fig. 2.4, where we demonstrate pose transfer from source meshes (*red*) to target template meshes (*blue*) not seen during training. As shown, the shapes transferred by our method accurately match the overall poses. Our method benefits from combining a keypoint-based hybrid representation with Jacobian fields, outperforming the baselines in preserving local details, especially in areas with intricate geometric features such as the hands. See the zoomed-in views in Fig. 2.4.

Furthermore, we apply our model to a more challenging setup involving stylized characters. In Fig. 2.5, we present qualitative results using shapes from the Mixamo [1] dataset. Despite being trained

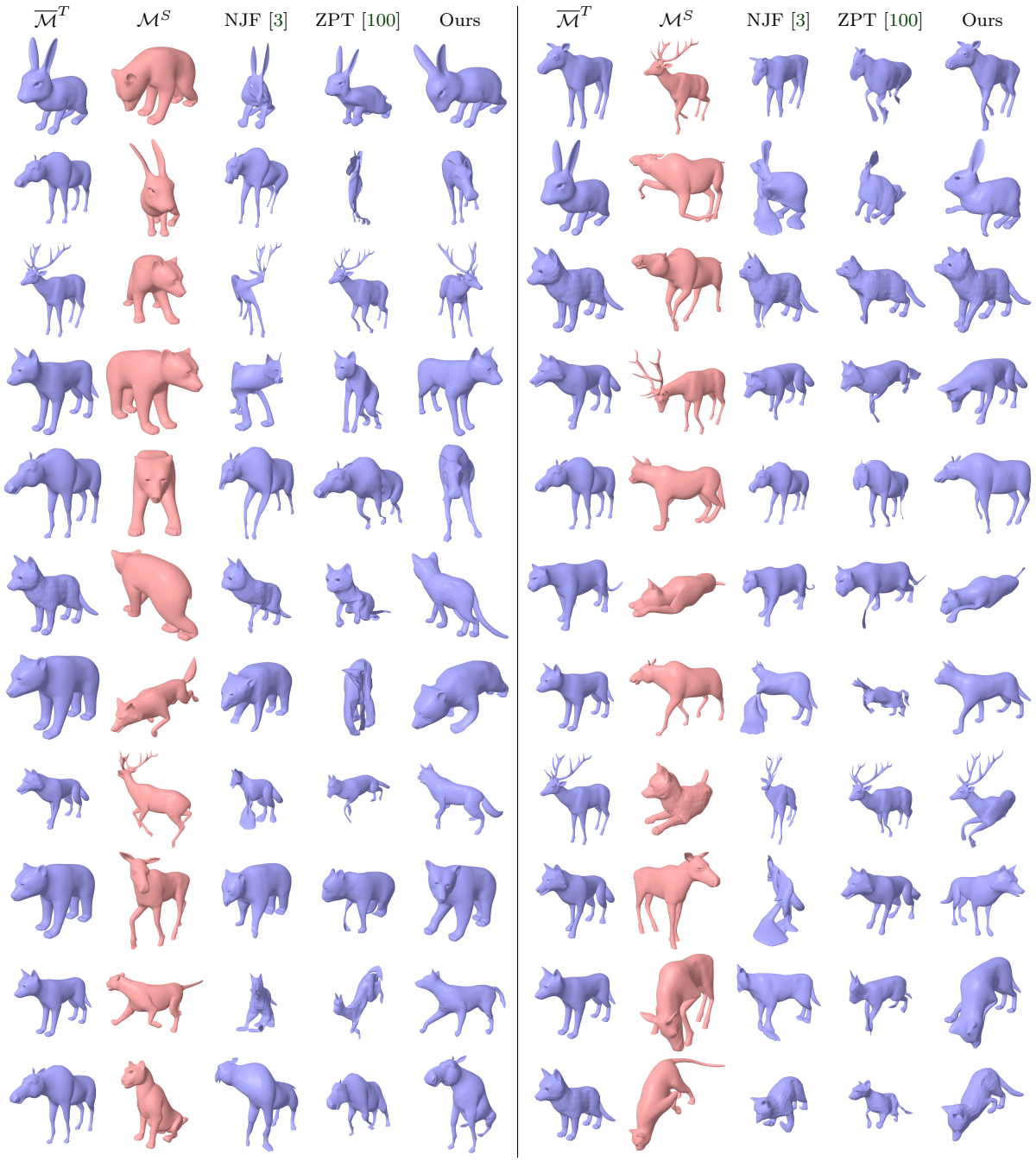

Figure 2.3: Qualitative results of transferring poses of the source meshes $\mathcal{M}^S$'s (*red*) in the DeformingThings4D animals [54] to target templates $\overline{\mathcal{M}}^T$'s (*blue*). Best viewed when zoomed in.

on a single, unclothed SMPL body shape, our method generalizes well to stylized humanoid characters with detailed geometry (first row) and even to a character missing one arm (second row).

2.4.4 Ablation Study

Our framework design is further validated by comparisons against different variations of our framework, as listed in Sec. 2.4.1, in the pose transfer experiment on animal shapes discussed in Sec. 2.4.2. Tab. 2.2 (left) summarizes the image plausibility metrics measured using the results from our internal baselines.

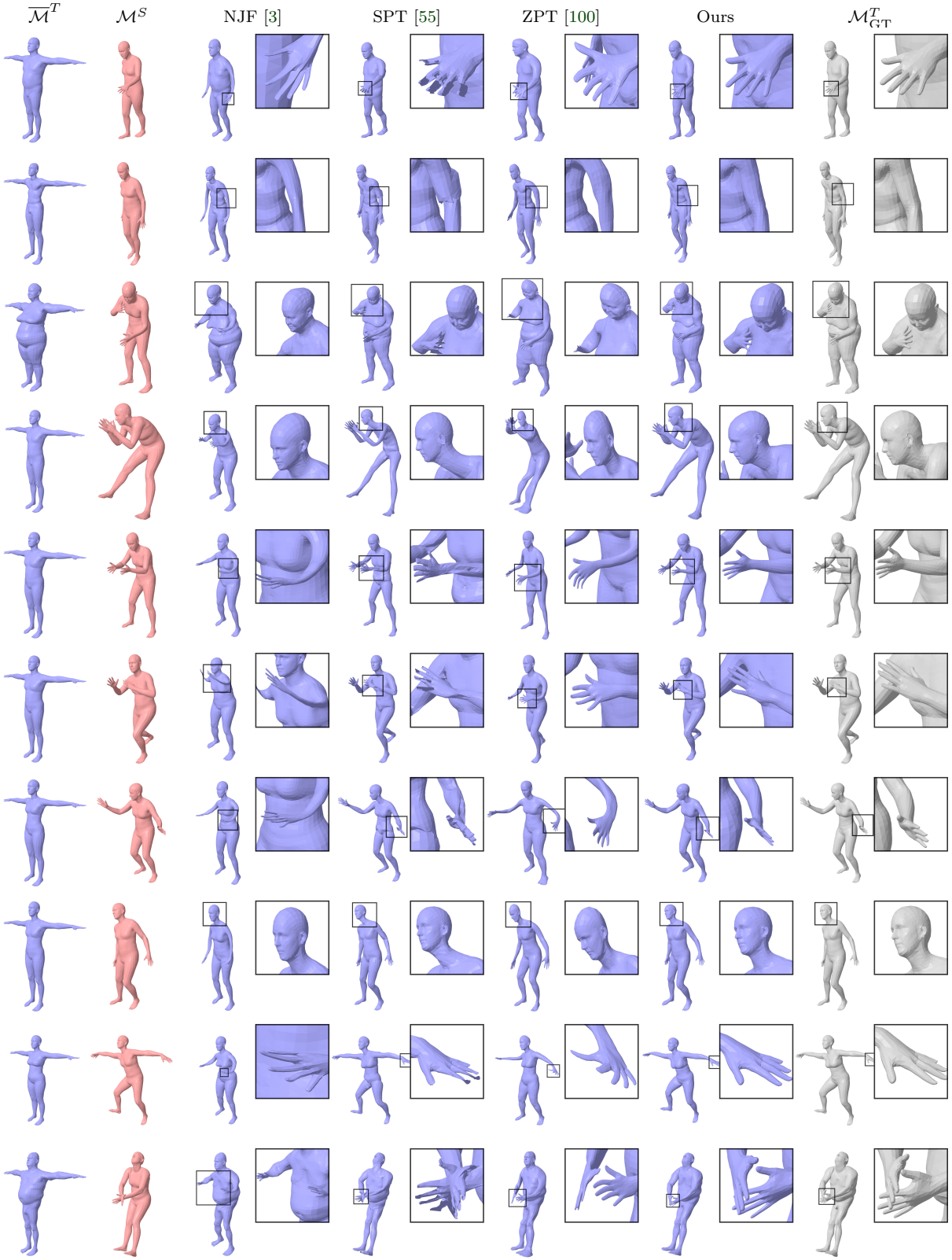

Figure 2.4: Qualitative results of transferring poses of the default human meshes $\mathcal{M}^S$'s (*red*) to different target template meshes $\overline{\mathcal{M}}^T$'s (*blue*). The ground truth targets $\mathcal{M}^T_{\text{GT}}$'s (*grey*) are displayed for reference. Best viewed when zoomed in.

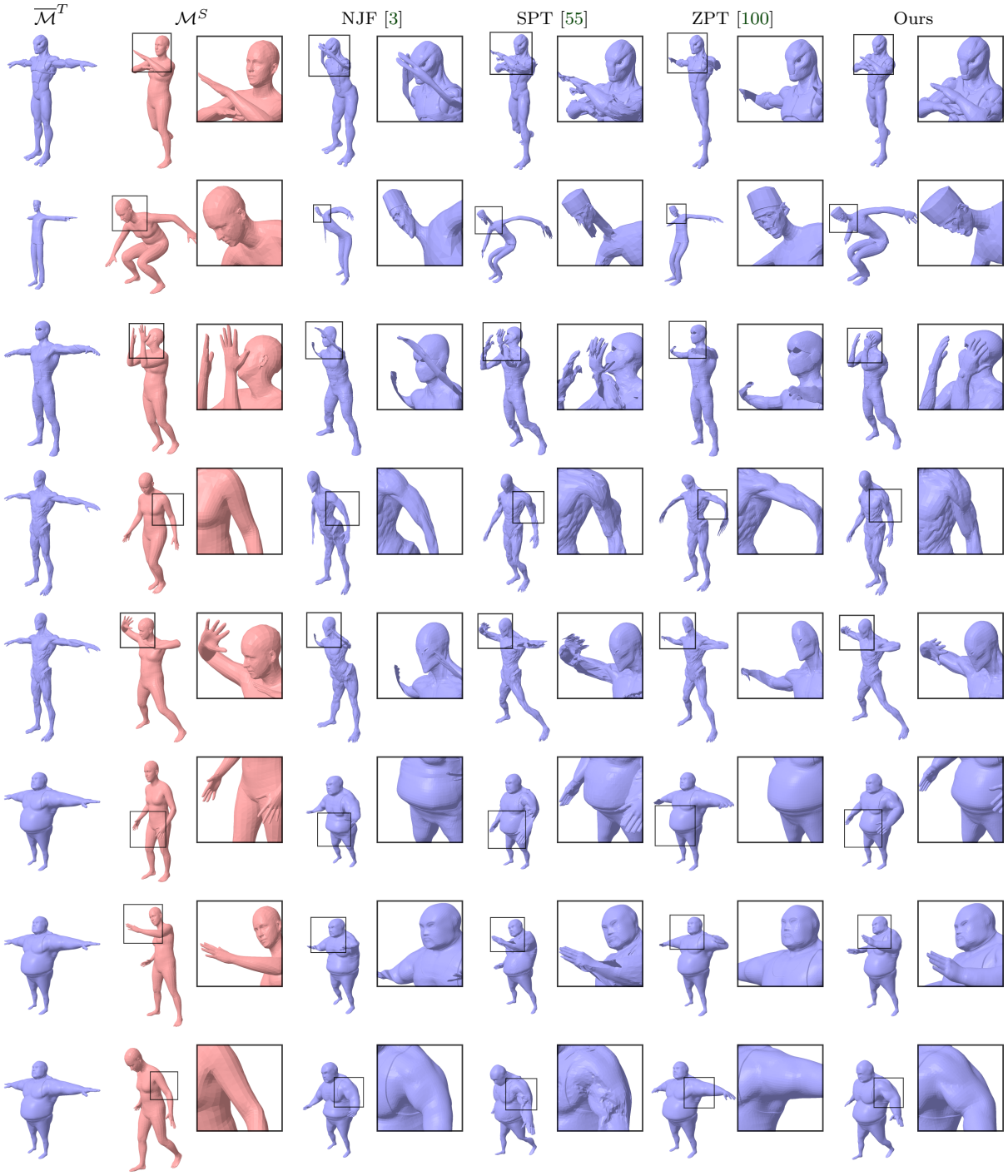

Figure 2.5: Qualitative results of transferring poses of the default human meshes $\mathcal{M}^S$'s (*red*) to target template meshes $\overline{\mathcal{M}}^T$'s (*blue*) of Mixamo characters [1]. Best viewed when zoomed in.

Qualitative results are presented in Fig. 2.6. Our method, which extracts pose representations from Jacobian fields and leverages the per-identity refinement module, achieves the best performance among all the variations.

Jacobian (Sec. 2.3.3)	Refinement (Sec. 2.3.4)	Poses from $\mathcal{M}_i^S$ FID ↓ ($\times 10^{-2}$)	KID ↓ ($\times 10^{-2}$)	ResNet Acc. ↑ (%)	Generated Poses (Sec. 2.3.5) FID ↓ ($\times 10^{-2}$)	KID ↓ ($\times 10^{-2}$)	ResNet Acc. ↑ (%)
✗	✗	3.52	2.13	55.67	9.52	4.69	44.34
✓	✗	1.17	0.47	75.13	4.40	2.39	75.82
✓	✓	**1.11**	**0.42**	**78.72**	**4.22**	**2.24**	**78.81**

Table 2.2: Ablation study using the poses from the source shapes in DeformingThings4D-Animals [54] dataset (left) and the poses generated from our cascaded diffusion model.

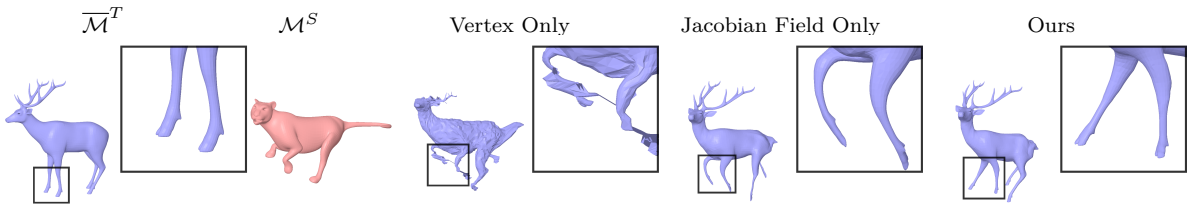

Figure 2.6: Qualitative results from the ablation study where a pose of the source shape $\mathcal{M}^S$ (*red*) in the DeformingThings4D-Animals [54] is transferred to the target template shape $\overline{\mathcal{M}}^T$ (*blue*).

2.4.5 Sensitivity to Number of Keypoints

We examine the sensitivity of our method to the number of keypoints by testing different variants of our framework while varying the number of keypoints extracted by the pose extractor to 50, 25, and 10, respectively. These variants are trained using the same SMPL [63] human body shapes and animal shapes from the DeformingThings4D-Animals [54] dataset. Our per-identity refinement stage (Sec. 2.3.4) is omitted to focus exclusively on the impact of keypoint counts on performance. Tab. 2.3 summarizes FID and PMD measured using the DeformingThings4D-Animals and SMPL dataset, respectively. We showcase qualitative results in Fig. 2.7 and Fig. 2.8. As reflected in both quantitative and qualitative results, reducing the number of keypoints does not significantly affect pose transfer accuracy.

DeformingThings4D-Animals [54]					SMPL [63]				
Method	Ours–10	Ours-25	Ours-50	**Ours-100**	Method	Ours–10	Ours-25	Ours-50	**Ours-100**
FID ($\times 10^{-2}$)	1.25	0.87	0.83	**0.72**	PMD ($\times 10^{-3}$)	0.20	0.17	0.17	**0.13**

Table 2.3: Quantitative results from the variants of our framework trained to extract different number of keypoints. Ours-N denotes a variant of our network trained to extract N keypoints.

2.4.6 Pose Variation Generation Using Diffusion Models

We evaluate the generation capabilities of our diffusion models trained using different pose representations. Since no existing generative model can learn pose representations transferable across various shapes, we focus on analyzing the impact of using Jacobian fields on generation quality. We use shapes obtained by applying 300 generated poses to both $\overline{\mathcal{M}}^S$'s (*red*) and various $\overline{\mathcal{M}}^T$'s (*blue*). The quantitative and qualitative results are summarized in Tab. 2.2 (right) and Fig. 2.9, respectively. While the poses are

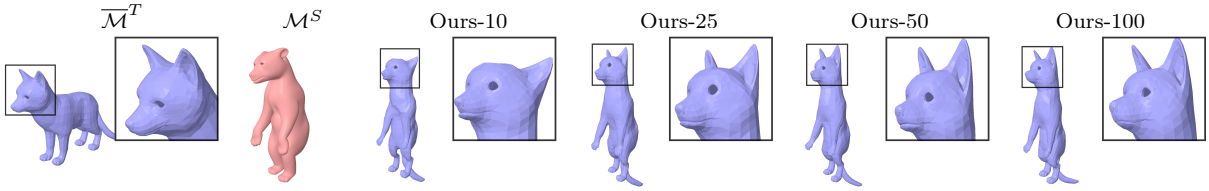

Figure 2.7: Qualitative results of transferring a pose of the source shape $\mathcal{M}^S$ (*red*) in DeformingThings4D–Animals [54] to the target template shape $\overline{\mathcal{M}}^T$ (*blue*) using variants of our framework (Ours-N), trained to extract N keypoints.

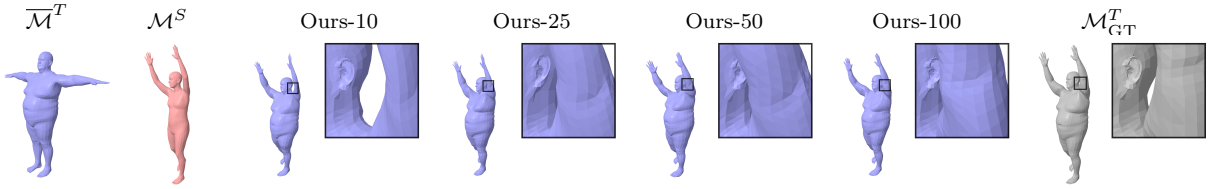

Figure 2.8: Qualitative results of transferring a pose of the default human mesh $\mathcal{M}^S$ (*red*) to the target template mesh $\overline{\mathcal{M}}^T$ (*blue*) using variants of our framework (Ours-N), trained to extract N keypoints.

generated using the diffusion model, our model still achieves ResNet classification accuracy comparable to the pose transfer experiment (Tab. 2.2, left). This tendency is also reflected in the qualitative results shown in Fig. 2.9. These results validate that the latent space learned from variations of Jacobian fields is more suitable for generating high-quality shape and pose variations compared to the one based on vertices.

2.5 Conclusion

We have presented a method for learning a novel neural representation of the pose of non-rigid 3D shapes, which facilitates: 1) the disentanglement of pose and object identity, 2) the training of a generative model due to its compactness, and 3) the transfer of poses to other objects' meshes. In our experiments, we demonstrated the state-of-the-art performance of our method in pose transfer, as well as its ability to generate diverse shapes by applying the generated poses to different identities.

Limitations. Our method leverages differential operators to compute the Jacobian field of the given template mesh, requiring additional preprocessing when dealing with meshes that have multiple disconnected components or defects in the triangulation. Our framework also assumes that a template mesh of the shape is known for pose transfer. We plan to extend our framework for transferring poses between arbitrary shapes in future work.

Societal Impacts. Our generative model for poses and the pose transfer technique could potentially be misused for deepfakes. Developing robust guidelines and techniques to prevent such misuse is an important area for future research.

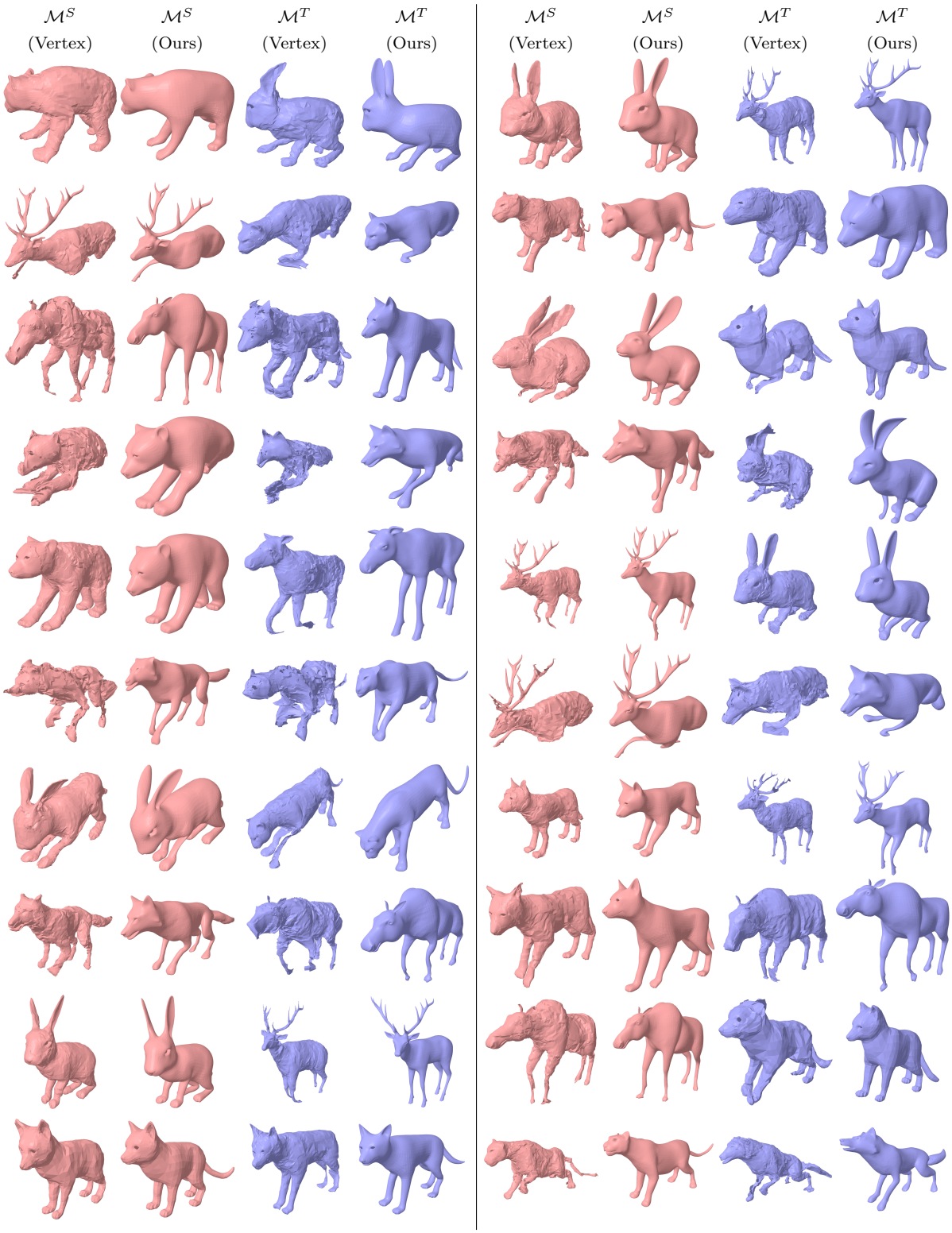

Figure 2.9: Pose variation generation results. Each row illustrates the outcomes of applying a generated pose to a source template mesh $\overline{\mathcal{M}}^S$ (*red*) and a target template mesh $\overline{\mathcal{M}}^T$ (*blue*).

Chapter 3. Plausibility-Aware Mesh Deformation

3.1 Introduction

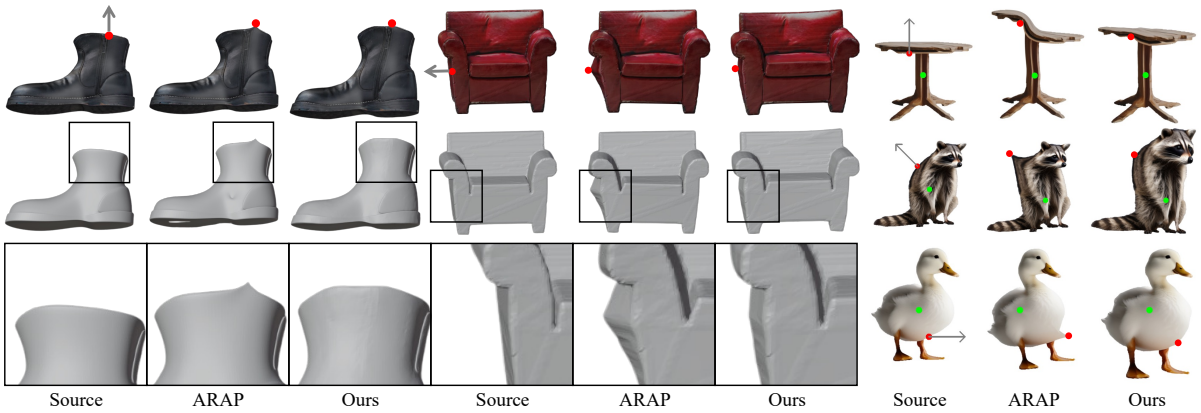

Figure 3.1: **APAP**, our novel shape deformation method, enables plausibility-aware mesh deformation and preservation of fine details of the original mesh offering an interface that alters geometry by directly displacing a handle (*red*) along a direction (*gray*), fixing an anchor vertex (*green*). Using a diffusion prior results in smoother geometry around the armchair handle, as seen in the example (middle column).

For 2D and 3D content, mesh is the most prevalent representation, thanks to its efficiency in storage, simplicity in rendering and also compatibility in common graphics pipelines, versatility in diverse applications such as design, physical simulation, and 3D printing, and flexibility in terms of decomposing geometry and appearance information, with widespread adoption in the industry.

For the creation of 2D and 3D meshes, recent breakthroughs in generative models [81, 84, 50, 71, 103, 82, 62, 95] have demonstrated significant advances. These breakthroughs enable users to easily generate content from a text prompt [71, 103, 82, 62, 95], or from photos [82, 74]. However, visual content creation typically involves numerous editing processes, deforming the content to satisfy users' desires through interactions such as mouse clicks and drags. Facilitating such interactive editing has remained relatively underexplored in the context of recent generative techniques.

Mesh deformation is a subject that has been researched for decades in computer graphics. Over time, researchers have established well-defined methodologies, characterizing mesh deformation as an optimization problem that aims to preserve specific geometric properties, such as the Mesh Laplacian [59, 91, 60], local rigidity [34, 90], and mesh surface Jacobians [3, 23], while satisfying given constraints. To facilitate user interaction, these methodologies have been extended to introduce specific user-interactive deformation handles, such as keypoints [36, 102, 46], cage mesh [43, 42, 58, 104, 113, 39], and skeleton [6, 108, 109], with the blending functions defined based on the preservation of geometric properties.

Despite the widespread use of classical mesh deformation methods, they often fail to meet users' needs because they do not incorporate the perceptual plausibility of the outputs. For example, as illustrated in Fig. 3.1, when a user intends to drag a point on the top of a table image, the classical deformation technique may introduce unnatural bending instead of lifting the tabletop. This limitation arises because deformation techniques solely based on geometric properties do not incorporate such

semantic and perceptual priors, resulting in the mesh editing process becoming more tedious and time-consuming.

Recent learning-based mesh deformation techniques [113, 39, 46, 108, 61, 3, 94] have attempted to address this problem in a data-driven way. However, they are also limited by relying on the existence of certain variations in the training data. Even recent large-scale 3D datasets [11, 106, 17, 16] have not reached the scale that covers all possible visual content users might intend to create.

To this end, we introduce our novel mesh deformation framework, dubbed APAP (As-Plausible-As-Possible), which exploits 2D image priors from a diffusion model pretrained on an Internet-scale image dataset to enhance the plausibility of deformed 2D and 3D meshes while preserving the geometric priors of the given shape. Recently, score distillation sampling (SDS) [71] has demonstrated great success in generating plausible 2D and 3D content, such as NeRF [123, 47, 41] and vector images [38, 35], using the distilled 2D image priors from a diffusion model. We incorporate these diffusion-model-based 2D priors into the optimization-based deformation framework, achieving the best synergy between geometry-based optimization and distilled-prior-based optimization.

To achieve this optimal synergy between geometric and perceptual priors within a unified framework, we introduce an alternative optimization approach. At each step, we first update the Jacobian of each mesh face using the SDS loss and user-provided constraints. Subsequently, the mesh vertex positions are recalculated by solving Poisson's equation with the updated face Jacobians. The direct application of the 2D diffusion prior via SDS, however, tends to compromise the identity of the given objects—an essential aspect in deformation. We thus enhance the identity awareness of the diffusion prior by finetuning it with the provided source image. The model is integrated into our two-stage pipeline that initiates deformation without the perceptual prior (SDS) and refines it with SDS and the given constraints afterward to create deformations that adhere to user-defined editing instructions while remaining visually plausible.

In experiments, we examine **APAP** using APAP-Bench consisting of 3D and 2D triangular meshes and editing instructions. The proposed method produces plausible deformations of 3D meshes compared to its baseline [90] based exclusively on a geometric prior. Evaluation in the task of 2D mesh editing further verifies the effectiveness of **APAP** as illustrated by the highest k-NN GIQA score [25] in quantitative analysis, and the higher preference over the baseline in a user study.

3.2 Related Work

3.2.1 Geometric Mesh Deformation

Mesh deformation has been one of the central problems in geometry processing and is thus addressed by a wide range of techniques. Cage-based methods [43, 42, 58, 104] let users alter meshes by manipulating cages enclosing them, calculating a point inside as a weighted sum of cage vertices. Skeleton-based approaches [105, 6, 108, 109] offer animation control by mapping surface points to underlying joints and bones, ideal for animating human/animal-like figures. Unlike the previous techniques that require the manual cage or skeleton construction, biharmonic coordinates-based methods [36, 102] automate establishing mappings from control points to vertices by formulating optimization problems. Other types of works instead allow users to manipulate shapes via direct vertex displacement while imposing constraints on local surface geometry, including rigidity [34, 90] and Laplacian smoothness [59, 91, 60]. Such hand-crafted deformation priors often lack consideration of visual plausibility, necessitating careful control point placement and iterative manual refinement to achieve satisfactory results.

3.2.2 Data-Driven Mesh Deformation

Data-driven approaches to mesh deformation [113, 39, 46, 108, 61, 3, 94] learn from shape collections, utilizing neural networks to infer parameters for classical deformation techniques, such as cage vertex coordinates and displacements [113], keypoints [39, 102, 46], subspaces of keypoint arrangements [61], differential coordinates [3], etc. However, these methods assume the availability of large-scale category-specific shape collection [113, 39, 102, 46, 108] or require dense correspondences between them [94, 3], limiting their applicability to new, out-of-sample shapes. We instead propose to directly mine deformation priors from pretrained diffusion models. Leveraging a generic (category-agnostic) image generative model trained on an Internet-scale image dataset, we devise a method that easily generalizes to novel 2D and 3D shapes while lifting the requirement for shape collections.

3.2.3 Pretrained 2D Priors for Shape Manipulation

Image analysis [73] and generation [77, 117, 5, 53] techniques can serve as effective visual priors for image editing tasks [10, 28, 97, 118, 83]. In addition, recent work [21, 78] and their adaption [20], enable personalized image generation and editing by learning a text embedding [21] or fine-tuning additional parameters, such as LoRA [33] to preserve and replicate the identities of given exemplars during editing. One interesting work is DragDiffusion [83], akin to DragGAN [66], which introduces a drag-based user interface for image editing through the manipulation of latent representations. However, it is not extendable to the deformation of parametric images, such as 2D meshes, and also 3D shapes. Another interesting line of works [64, 45, 23] extends the idea further to manipulate shapes by propagating image-based gradients to the underlying shape representations. They maximize CLIP [73] similarity between the renderings and text prompts to either add geometric textures [64], jointly update both vertices and texture [45], or deform a shape parameterized by per-triangle Jacobians [23]. In contrast to such text-driven editing techniques, we build on Score Distillation Sampling (SDS) [71] to enable direct manipulation of shapes via handle displacement, ensuring visual plausibility. While the technique is prevalent in various problems ranging from text-to-3D [71, 103, 82, 62, 95], image editing [27] and neural field editing [123], it has not been adopted for shape deformation.

3.3 Method

We present **APAP**, a novel handle-based mesh deformation framework capable of producing visually plausible deformations of either 2D or 3D triangular meshes. To achieve this goal, we integrate powerful 2D diffusion priors into a learnable Jacobian field representation of shapes.

We emphasize that leveraging 2D priors, such as latent diffusion models (LDMs) [77] trained on large-scale datasets [80], for shape deformation poses challenges that require meticulous design choices. The following sections will delve into the details of shape representation (Sec. 3.3.1) and diffusion prior (Sec. 3.3.2), offering a rationale for the design decisions underpinning our framework (Sec. 3.3.3).

3.3.1 Representing Shapes as Jacobian Fields

Let $\mathcal{M}_0 = (\mathbf{V}_0, \mathbf{F}_0)$ denote a source mesh to be deformed, represented by vertices $\mathbf{V}_0 \in \mathbb{R}^{V\times 3}$ and faces $\mathbf{F}_0 \in \mathbb{R}^{F\times 3}$. Users are allowed to select a set of vertices used as movable handles designated by an indicator matrix $\mathbf{K}_h \in \{0,1\}^{V_h \times V}$. We also require users to select a set of anchors, represented as

another indicator matrix $\mathbf{K}_a \in \{0,1\}^{V_a \times V}$, to avoid trivial solutions (i.e., global translations). Then, the handle and anchor vertices become $\mathbf{V}_h = \mathbf{K}_h \mathbf{V}_0$ and $\mathbf{V}_a = \mathbf{K}_a \mathbf{V}_0$.

Our framework also expects a set of vectors $\mathbf{D}_h \in \mathbb{R}^{V_h \times 3}$ that indicate the directions along which the handles will be displaced. Furthermore, we let $\mathbf{T}_h = \mathbf{V}_h + \mathbf{D}_h$ and $\mathbf{T}_a = \mathbf{V}_a$ denote the target positions of the user-specified handles and anchors, respectively.

In this work, we employ a Jacobian field $\mathbf{J}_0 = \{\mathbf{J}_{0,f} | f \in \mathbf{F}_0\}$, a dual representation of $\mathcal{M}_0$, defined as a set of per-face Jacobians $\mathbf{J}_{0,f} \in \mathbb{R}^{3\times 3}$ where

$$\mathbf{J}_{0,f} = \boldsymbol{\nabla}_f \mathbf{V}_0, \tag{3.1}$$

and $\boldsymbol{\nabla}_f$ is the gradient operator of triangle f.

Conversely, we compute a set of *deformed* vertices $\mathbf{V}^*$ from a given Jacobian field $\mathbf{J}$ by solving a Poisson's equation

$$\mathbf{V}^* = \arg\min_{\mathbf{V}} \|\mathbf{L}\mathbf{V} - \boldsymbol{\nabla}^T \mathcal{A}\mathbf{J}\|^2, \tag{3.2}$$

where $\boldsymbol{\nabla}$ is a stack of per-face gradient operators, $\mathcal{A} \in \mathbb{R}^{3F\times 3F}$ is the mass matrix and $\mathbf{L} \in \mathbb{R}^{V\times V}$ is the cotangent Laplacian of $\mathcal{M}_0$, respectively. Since $\mathbf{L}$ is rank-deficient, the solution of Eqn. 3.2 cannot be uniquely determined unless we impose constraints. We thus consider a constrained optimization problem

$$\mathbf{V}^* = \arg\min_{\mathbf{V}} \|\mathbf{L}\mathbf{V} - \boldsymbol{\nabla}^T \mathcal{A}\mathbf{J}\|^2 + \lambda \|\mathbf{K}_a \mathbf{V} - \mathbf{T}_a\|^2, \tag{3.3}$$

where $\lambda \in \mathbb{R}^+$ is a weight for the constraint term. Note that we solve Eqn. 3.3 with the user-specified anchors as constraints to determine $\mathbf{V}^*$.

Taking the derivative with respect to $\mathbf{V}$, the problem in Eqn. 3.3 turns into a system of equations

$$\left(\mathbf{L}^T\mathbf{L} + \lambda \mathbf{K}_a^T \mathbf{K}_a\right) \mathbf{V} = \mathbf{L}^T \boldsymbol{\nabla}^T \mathcal{A}\mathbf{J} + \lambda \mathbf{K}_a^T \mathbf{T}_a, \tag{3.4}$$

which can be efficiently solved using a differentiable solver [3] implementing Cholesky decomposition.

We let g denote a functional representing the aforementioned differentiable solver for notational convenience, $\mathbf{V}^* = g\left(\mathbf{J}, \mathbf{K}_a, \mathbf{T}_a\right)$. Since g is differentiable, we can deform $\mathcal{M}_0$ by propagating upstream gradients from various loss functions to the underlying parameterization $\mathbf{J}$. For instance, one may impose a *soft* constraint on the locations of selected handles during optimization with the objective of the form:

$$\mathcal{L}_h = \|\mathbf{K}_h \mathbf{V}^* - \mathbf{T}_h\|^2. \tag{3.5}$$

We will discuss how such a soft constraint can be blended into our framework in Sec. 3.3.3. Next, we describe how to incorporate a pretrained diffusion model as a prior for visual plausibility.

3.3.2 Score Distillation for Shape Deformation

While traditional mesh deformation techniques make variations that match the given *geometric* constraints, their lack of consideration on *visual plausibility* results in unrealistic shapes. Motivated by recent success in text-to-3D literature, we harness a powerful 2D diffusion prior [77] in our framework as a critic that directs deformation by scoring the realism of the current shape.

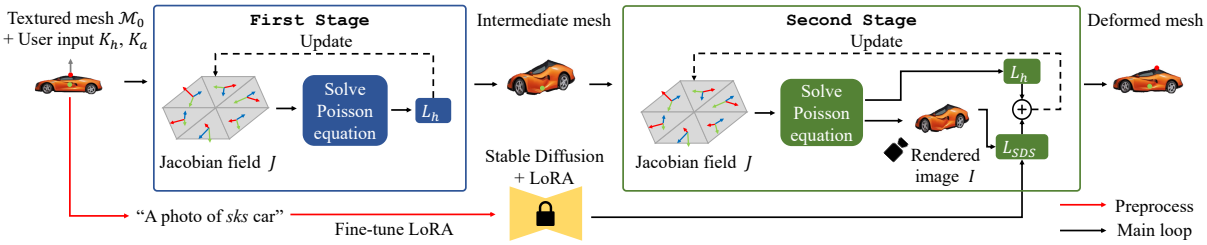

Figure 3.2: The overview of APAP. APAP parameterizes a triangular mesh as a per-face Jacobian field that can be updated via gradient-descent. Given a textured mesh and user inputs specifying the handle(s) and anchor(s), our framework initializes a Jacobian field as a trainable parameter. During the first stage, the Jacobian field is updated via iterative optimization of $\mathcal{L}_h$, a soft constraint that initially deforms the shape according to the user's instruction. In the following stage, the mesh is rendered using a differentiable renderer $\mathcal{R}$ and the rendered image is provided as an input to a diffusion prior finetuned with LoRA [33] that computes the SDS loss $\mathcal{L}_{\text{SDS}}$. The joint optimization of $\mathcal{L}_h$ and $\mathcal{L}_{\text{SDS}}$ improves the visual plausibility of the mesh while conforming to the given edit instruction.

Specifically, we distill its prior knowledge via Score Distillation Sampling (SDS) [71]. Let $\mathbf{J}$ denote the current Jacobian field and $\mathbf{V}^*$ be the set of vertices computed from $\mathbf{J}$ following the procedure described in Sec. 3.3.1.

We render $\mathcal{M}^* = (\mathbf{V}^*, \mathbf{F})$ from a viewpoint defined by camera extrinsic parameters $\mathbf{C}$ using a differentiable renderer $\mathcal{R}$, producing an image $\mathcal{I} = \mathcal{R}(\mathcal{M}^*, \mathbf{C})$. The diffusion prior $\hat{\epsilon}_\phi$ then rates the realism of $\mathcal{I}$, producing a gradient

$$\nabla_{\mathbf{J}} \mathcal{L}_{\text{SDS}}(\phi, \mathcal{I}) = \mathbb{E}_{t,\epsilon}\left[w(t) \left(\hat{\epsilon}_\phi(\mathbf{z}_t; y, t) - \epsilon \right) \frac{\partial \mathcal{I}}{\partial \mathbf{J}} \right], \tag{3.6}$$

where $t \sim \mathcal{U}(0,1)$, $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and $\mathbf{z}_t$ is a noisy latent embedding of $\mathcal{I}$. The propagated gradient alters the geometry of $\mathcal{M}$ by modifying $\mathbf{J}$.

To increase the instance-awareness of the diffusion model, we follow recent work [78, 83] on personalized image editing and finetune the model using LoRA [33]. In particular, we first render $\mathcal{M}$ from n different viewpoints to obtain a set $\mathcal{I} = \{\mathcal{I}_1, \ldots, \mathcal{I}_n\}$ of training images and inject additional parameters to the model, resulting in an expanded set of network parameters ϕ'. The parameters are then optimized with a denoising loss [77]

$$\mathcal{L} = \mathbb{E}_{t,\epsilon,\mathbf{z}}\left[\| \hat{\epsilon}_{\phi'}(\mathbf{z}_t; y, t) - \epsilon \|^2 \right], \tag{3.7}$$

where $\mathbf{z}_t$ denotes a latent of a training image perturbed with noise at timestep t.

The finetuned diffusion prior, together with a learnable Jacobian field representation of the source mesh $\mathcal{M}_0$, comprises the proposed framework described in the following section.

3.3.3 As-Plausible-As-Possible (APAP)

APAP tackles the problem of plausibility-aware shape deformation by harmonizing the best of both worlds: a learnable shape representation founded on classical geometry processing, robust to noisy gradients, and a powerful 2D diffusion prior finetuned with the image(s) of the source mesh for better instance-awareness.

We provide an overview of the proposed pipeline in Fig. 3.2 and the algorithm in Alg. 1. We will delve into details in the following. Provided with a textured mesh $\mathcal{M}_0$, handles $\mathbf{K}_h$, anchors $\mathbf{K}_a$, as well

Algorithm 1 As-Plausible-As-Possible

Parameters: g, $\mathcal{R}$, ϕ, γ, M, N
Inputs: $\mathcal{M}_0 = (\mathbf{V}_0, \mathbf{F}_0)$, $\mathbf{K}_a$, $\mathbf{K}_h$, $\mathbf{T}_a$, $\mathbf{T}_h$, $\{\mathbf{C}_i\}_{i=1}^n$
Output: $\mathcal{M}$

procedure FIRSTSTAGE($\mathbf{J}$, $\mathbf{K}_a$, $\mathbf{K}_h$, $\mathbf{T}_a$, $\mathbf{T}_h$, g)
  for $i = 1, 2, \ldots, M$ **do**
    $\mathbf{V}^* \leftarrow g(\mathbf{J}, \mathbf{K}_a, \mathbf{T}_a)$ ▷ Solving Eqn. 3.4
    $\mathbf{J} \leftarrow \mathbf{J} - \gamma \nabla_{\mathbf{J}} \mathcal{L}_h(\mathbf{V}^*, \mathbf{K}_h, \mathbf{T}_h)$
  end for
  return J
end procedure
procedure SECONDSTAGE($\mathbf{J}$, $\mathbf{F}_0$, $\mathbf{K}_a$, $\mathbf{K}_h$, $\mathbf{T}_a$, $\mathbf{T}_h$, g, ϕ, $\{\mathbf{C}_i\}$)
  for $i = 1, 2, \ldots, N$ **do**
    $\mathbf{V}^* \leftarrow g(\mathbf{J}, \mathbf{K}_a, \mathbf{T}_a)$ ▷ Solving Eqn. 3.4
    $\mathcal{M}^* \leftarrow (\mathbf{V}^*, \mathbf{F}_0)$
    $\mathbf{C} \sim \mathcal{U}(\{\mathbf{C}_i\})$ ▷ Viewpoint Sampling
    $\mathcal{I} \leftarrow \mathcal{R}(\mathcal{M}^*, \mathbf{C})$ ▷ Rendering
    $\mathbf{J} \leftarrow \mathbf{J} - \gamma \nabla_{\mathbf{J}} (\mathcal{L}_{\text{SDS}}(\phi, \mathcal{I}) + \mathcal{L}_h(\mathbf{V}^*, \mathbf{K}_h, \mathbf{T}_h))$
  end for
  return J
end procedure

$\phi \leftarrow \text{LoRA}(\phi, \mathcal{M}_0, \mathcal{R}, \{\mathbf{C}_i\})$
$\mathbf{J} \leftarrow \{\mathbf{J}_{0,f} | f \in \mathbf{F}_0\}$
$\mathbf{J} \leftarrow$ FIRSTSTAGE($\mathbf{J}$, $\mathbf{K}_a$, $\mathbf{K}_h$, $\mathbf{T}_a$, $\mathbf{T}_h$, g)
$\mathbf{J} \leftarrow$ SECONDSTAGE($\mathbf{J}$, $\mathbf{F}_0$, $\mathbf{K}_a$, $\mathbf{K}_h$, $\mathbf{T}_a$, $\mathbf{T}_h$, g, ϕ, $\{\mathbf{C}_i\}$)
$\mathbf{V} \leftarrow g(\mathbf{J}, \mathbf{K}_a, \mathbf{T}_a)$
$\mathcal{M} \leftarrow (\mathbf{V}, \mathbf{F}_0)$
return $\mathcal{M}$

as their target positions $\mathbf{T}_h$ and $\mathbf{T}_a$ as inputs, **APAP** yields a plausible deformation $\mathcal{M}$ of $\mathcal{M}_0$ that conforms to the given handle-target constraints. Before deforming $\mathcal{M}_0$, we render $\mathcal{M}_0$ from a single view in the case of 2D meshes and four canonical views (i.e., front, back, left, and right) for 3D meshes and use the images to finetune Stable Diffusion [77] by optimizing LoRA [33] parameters injected to the model (the *red* line in Fig. 3.2). Simultaneously, **APAP** computes the Jacobian field $\mathbf{J}_0$ of the input mesh $\mathcal{M}_0$ and initializes it as a trainable parameter $\mathbf{J}$.

APAP deforms the input mesh through two stages. In the `FirstStage`, it first deforms the input mesh according to instructions from users without taking visual plausibility into account. The subsequent `SecondStage` integrates a 2D diffusion prior into the optimization loop, simultaneously enforcing user constraints and visual plausibility.

At every iteration of the `FirstStage` illustrated as the *blue* box in Fig. 3.2, we compute the vertex positions $\mathbf{V}^*$ corresponding to the current Jacobian field $\mathbf{J}$ by solving Eqn. 3.3 using the anchors specified by $\mathbf{K}_a$ as hard constraints. Then, we compute the soft constraint $\mathcal{L}_h$ defined as Eqn. 3.5 that drags a set of handle vertices $\mathbf{K}_h\mathbf{V}^*$ toward the corresponding targets $\mathbf{T}_h$. The interleaving of differentiable Poisson solve and optimization of $\mathcal{L}_h$ via gradient-descent is repeated for M iterations. This progressively updates $\mathbf{J}$, treated as a learnable black box in our framework, deforming $\mathcal{M}_0$. Consequently, the edited mesh $\mathcal{M}^* = (\mathbf{J}, \mathbf{F}_0)$ follows user constraints at the cost of the degraded plausibility, mitigated in the following stage through the incorporation of a diffusion prior.

The result of `FirstStage` then serves as an initialization for the `SecondStage`, illustrated as the *green* box in Fig. 3.2 guided by plausibility constraint $\mathcal{L}_{\text{SDS}}$. Unlike the `FirstStage` where the update of $\mathbf{J}$ was purely driven by the geometric constraint $\mathcal{L}_h$, we aim to steer the optimization based on the visual plausibility of the current mesh $\mathcal{M}^*$. To achieve this, we render $\mathcal{M}^*$ using a differentiable renderer $\mathcal{R}$ using the same viewpoint(s) from which the training image(s) for finetuning was rendered. When deforming 3D meshes, we randomly sample one viewpoint at each iteration. The rendered image $\mathcal{I}$ is used to evaluate $\mathcal{L}_{\text{SDS}}$ which is optimized jointly with $\mathcal{L}_h$ for N iterations. The combination of geometric and plausibility constraints improves the visual plausibility of the output while encouraging it to conform to the given constraints.

We note that the iterative approach in the `FirstStage` leads to better results than alternative update strategies such as deforming the source mesh $\mathcal{M}_0$ by minimizing ARAP energy [90] or, solving Eqn. 3.3 using both $\mathbf{K}_h$ and $\mathbf{K}_a$ as hard constraints. In our experiments (Sec. 3.4), we show that both methods produce distortions that cannot be corrected by the diffusion prior in the subsequent stage. Specifically, directly solving Eqn. 3.3 using all available constraints only yields the least squares solution $\mathbf{V}^*$ without updating the underlying Jacobians $\mathbf{J}$, resulting in the aforementioned distortions.

3.4 Experiments

We evaluate **APAP** in downstream applications involving manipulation of 3D and 2D meshes.

3.4.1 Experiment Setup

Benchmark. To evaluate the plausibility of a mesh deformation we propose a novel benchmark APAP-Bench of textured 3D and 2D triangular meshes spanning both human-made and organic objects annotated with handle vertices and their editing directions, and anchor vertices. The set of 3D meshes, APAP-Bench 3D, is constructed using meshes from ShapeNet [11] and *Genie* [2]. The meshes are

normalized to fit in a unit cube. Each mesh is manually annotated with editing instructions, including a set of anchors, handles, and corresponding targets to simulate editing scenarios. APAP-Bench offers another subset called APAP-Bench 2D, a collection of 80 textured, planar meshes of various objects, to facilitate quantitative analysis and user study described later in this section. To create APAP-Bench 2D, we first generate 2 images of real-world objects for each of the 20 categories using Stable Diffusion-XL [70]. We employ the following template prompt `"a photo of [category name] in a white background"` for all categories to facilitate foreground object segmentation in the following step. The list of all categories, spanning human-made and organic objects, is shown in Tab. 3.1.

Human-Made	Organic
backpack	flying bird
bike	side view of cat
chair	side view of dog
high-heeled shoes	runway model
purse	sitting bird
side view of car	standing cheetah
sneakers	standing dragon
table	standing raccoon
airplane	standing sheep
	standing white duck
	starfish

Table 3.1: Object categories of 2D meshes in APAP-Bench 2D. APAP-Bench 2D includes 2D triangle meshes depicting various objects, including both human-made and organic objects.

We extract foreground masks from the generated images using SAM [49] and sample pixels that lie on the boundary and interior. The sampled pixels are used for Delaunay triangulation, constrained with the edges along the main contour of the masks, that produces 2D triangular meshes with texture. We assign two handle and anchor pairs to each mesh that imitate user instructions. Specifically, we choose vertices on the shape boundaries instead of internal vertices to induce deformations that alter object silhouettes. For instance, users would try to drag the bottom of a backpack downward to enlarge the shape, instead of dragging an interior point which may flip triangles, distorting the appearance. As an anchor, we use the vertex closest to the center of mass of each mesh. For evaluation purposes, we populate the reference set by sampling $1,000$ images for each object category using Stable Diffusion-XL. The generated images are used to evaluate a perceptual metric to assess the plausibility of 2D mesh editing results as described in Sec. 3.4.3.

Baselines. We compare our method (**APAP**) and As-Rigid-As-Possible (ARAP) [90] since it is one of the widely used mesh deformation techniques that permits shape manipulation via direct vertex displacement. Throughout the experiments, we use the implementation in `libigl` [37] with default parameters.

Evaluation Metrics. In 2D experiments, we conduct quantitative analysis based on k-NN GIQA score [25] as an evaluation metric to assess the plausibility of instance-specific editing results. The metric quantifies the perceptual proximity between the edited image and its k nearest neighbors in the reference set included in APAP-Bench 2D. As our objective is to make plausible variations of 2D meshes via deformation, an edited object should remain perceptually similar to other objects in the same

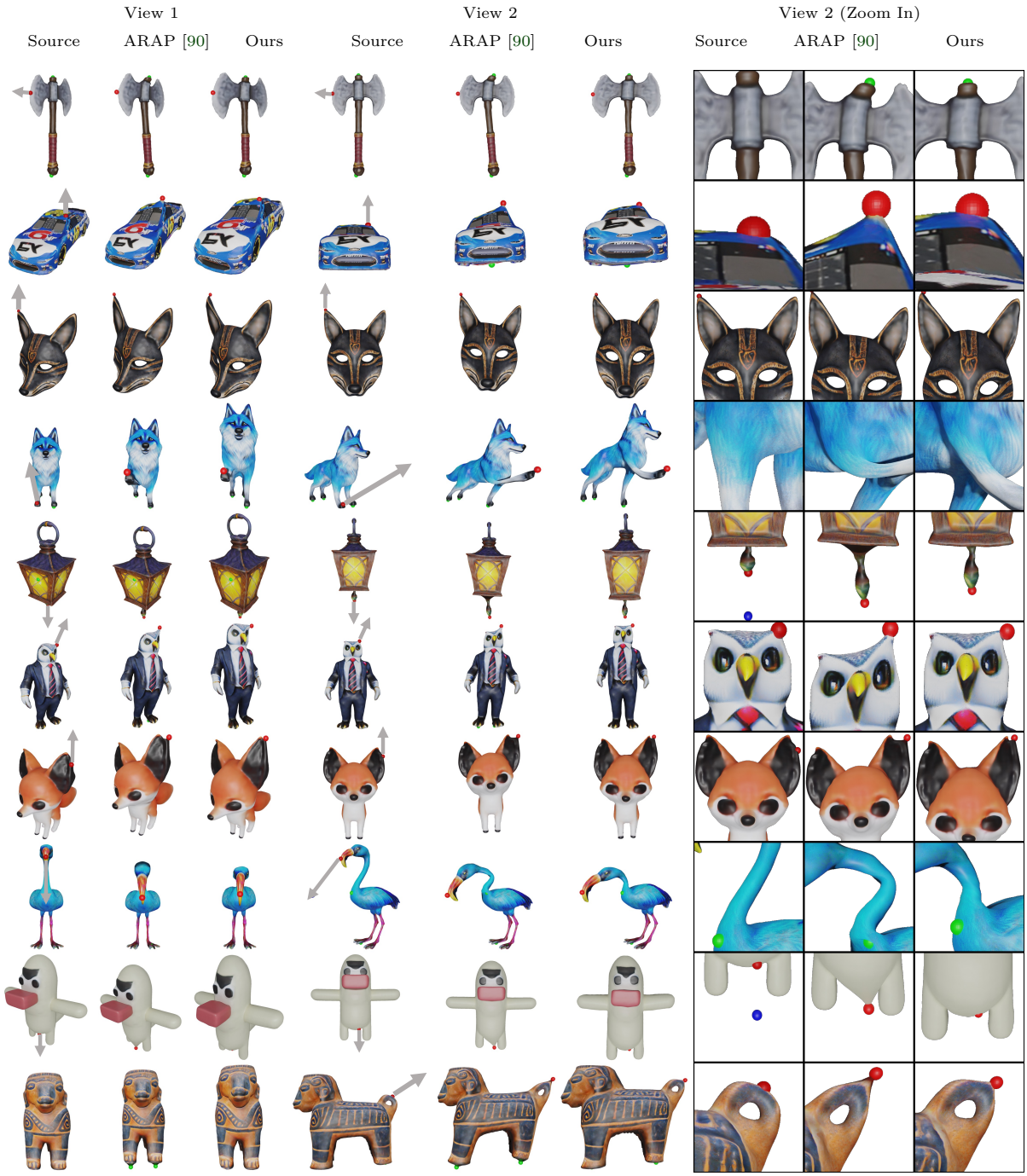

Figure 3.3: Qualitative results from 3D shape deformation. We visualize the source shapes and their deformations made using ARAP [90] and ours by following the instructions each of which specifies a handle (*red*), an edit direction denoted with an arrow (*gray*), and an anchor (*green*). We showcase the rendered images captured from two different viewpoints, as well as one zoom-in view highlighting local details.

category. We use $k = 12$ throughout the experiments.

Implementation Details. When implementing Alg. 1, we used a modified version of the differentiable Poisson solver from [3], denoted by g in Alg. 1, and `nvdiffrast` [51] when implementing the differentiable

renderer $\mathcal{R}$ in our pipeline. We render 2D/3D meshes at a resolution of 512×512.

When editing 2D meshes, we optimize $\mathcal{L}_h$ for $M = 300$ iterations in the `FirstStage` and jointly optimize $\mathcal{L}_h$ and $\mathcal{L}_{\text{SDS}}$ for $N = 700$ iterations in the `SecondStage`. For experiments involving the optimization of 3D meshes with increased geometric complexity, we use $M = 300$ and $N = 1000$ for each stage, respectively. We use ADAM [48] with a learning rate $\gamma = 1 \times 10^{-3}$ throughout the optimization. We use the Classifier-Free Guidance (CFG) scale of 100.0 and randomly sample $t \in [0.02, 0.98]$ when evaluating $\mathcal{L}_{\text{SDS}}$ following DreamFusion [71].

We also utilize neighboring vertices of handles and anchors during deformation to maintain smooth geometry near the handles. Along with the given handles and anchors, vertices within a sphere of radius $r = 0.01$ around the handles and anchors are included to form extended sets called region handles and region anchors, respectively. When deforming 3D meshes, we use region anchors and a single handle, while for 2D mesh editing, we employ both region anchors and region handles. Note that the same sets of handles and anchors are used when deforming shapes with our baseline methods to ensure fair comparisons.

We use a script from `diffusers` [19] to finetune Stable Diffusion [77] with LoRA [33]. We employ `stabilityai/stable-diffusion-2-1-base` as our base model and augment its cross-attention layers in the U-Net with rank decomposition matrices of rank 16. For the task of 2D mesh editing, we train the injected parameters for 60 iterations, utilizing a rendering of a mesh as a training image. In the 3D shape deformation, where renderings from 4 canonical viewpoints (front, back, left, and right) are available, we finetune the model for 200 iterations. In both cases, we use the learning rate $\gamma = 5 \times 10^{-4}$.

3.4.2 3D Shape Deformation

Qualitative Results. We showcase examples of 3D shape deformation where each deformation is specified by a handle (*red*), an edit direction (*gray*), and an anchor (*green*). As shown in Fig. 3.3, **APAP** is capable of manipulating 3D shapes to improve visual plausibility which is not achievable by solely relying on geometric prior such as ARAP [90]. For instance, given a user input that drags a handle on one blade of an axe (the first row) along an arrow, **APAP** simultaneously expands both blades of the axe whereas ARAP [90] produces distortions near the head. Similar examples that demonstrate symmetry-awareness of **APAP** can be found in other cases such as a car (the second row), and an owl (the sixth row) where a user lifts only one side of the shape upward and the symmetry is recovered by **APAP** which cannot be achieved by ARAP [90]. Also, note that **APAP** is capable of making a smooth articulation at the leg of the wolf (the fourth row) by adjusting the overall posture in comparison to ARAP which creates an excess bending.

3.4.3 2D Mesh Editing

Qualitative Evaluation. We present qualitative results using the baselines and our method in Fig. 3.5. Each row shows two different results obtained by editing an image based on a handle moved from the original position (*red*) along a direction indicated by an arrow (*gray*) while fixing an anchor (*green*), similar to the 3D experiments discussed in the previous section.

As shown in Fig. 3.5, ARAP [90] enforces local rigidity and often results in implausible deformations. For example, it does not account for the mechanics of the human body and introduces an unrealistic articulation of a human arm (the fourth row). In addition, it twists the body of a sports car (the fifth row). Both of them originate from the lack of understanding of the appearance of objects. **APAP**

Figure 3.4: Failure cases of DragDiffusion. DragDiffusion [83] can easily compromise the identity of edited instances as it manipulates their latents without an explicit parameterization, the identity of instances can be broken during editing.

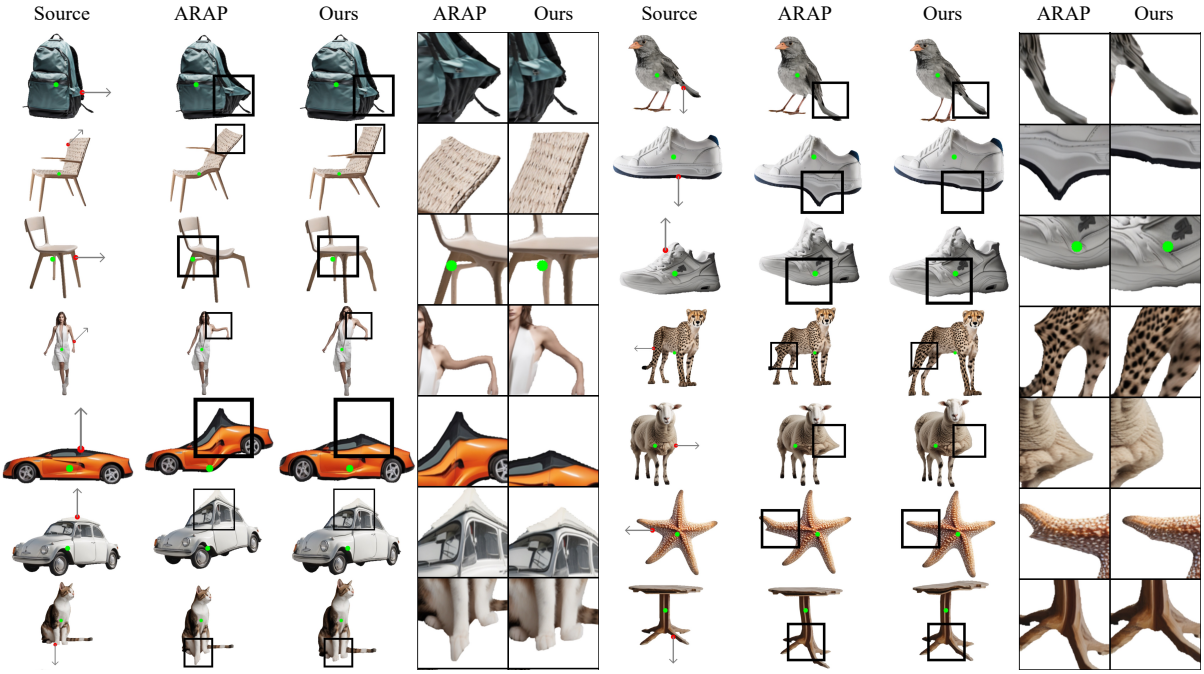

Figure 3.5: Qualitative results from 2D mesh deformation. 2D meshes are edited using ARAP [90] and the proposed method following the edit instruction consisting of a handle (*red*), a target direction (*gray*), and an anchor (*green*). We showcase the rendered images of the edited meshes, as well as a zoom-in view highlighting local details.

alleviates this issue by incorporating a visual prior into shape deformation producing a bending near the elbow and preserving the smooth silhouette of the car, respectively.

While **APAP** is designed for meshes not images, we provide an additional qualitative comparison against DragDiffusion [83], an image editing technique that operates in pixel space, to demonstrate the effectiveness of mesh-based parameterization in applications where identity preservation is crucial. As shown in Fig. 3.4, DragDiffusion [83] may corrupt the identity of the instances depicted in input images during the encoding and decoding procedure. **APAP**, on the other hand, makes plausible variations of the given objects while maintaining their originality, benefiting from an explicit mesh representation it

Methods	k-NN GIQA ($\times 10^{-2}$) ↑
ARAP [90]	4.753
DragDiffusion [83]	4.545
Ours ($\mathcal{L}_h$ Only)	4.797
Ours (ARAP Init.)	4.740
Ours (Poisson Init.)	4.316
Ours	**4.887**

Table 3.2: Quantitative analysis for 2D mesh editing. APAP outperforms its baselines in quantitative evaluation using k-NN GIQA [25].

is grounded.

Methods	Preference (%) ↑
ARAP [90]	41.7
Ours	**58.3**

Table 3.3: User study preference for 3D mesh deformation. In a user study targeting users on Amazon Mechanical Turk (MTurk), the results produced using ours were preferred over the outputs from the baseline.

Quantitative Evaluation. Tab. 3.2 summarizes k-NN GIQA scores measured on the outputs from ARAP [90] (the first row) and **APAP** (the sixth row) using APAP-Bench 2D. As shown, **APAP** demonstrates superior performance over ARAP [90]. This again verifies the observations from qualitative evaluation where ARAP [90] introduces distortions that harm visual plausibility. As in qualitative evaluation, we also report the k-NN GIQA score of DragDiffusion [83], degraded due to artifacts caused during direct manipulation of latents.

User Study. We further conduct user studies for a more accurate perceptual analysis. We follow Ritchie [76] and recruit participants on Amazon Mechanical Turk (MTurk).

For 3D mesh deformation, we asked user study participants to compare rendered images of meshes deformed using ARAP [90] and **APAP**. Each participant is provided with 20 image pairs and asked to select one image at each time given the question: "Which edited image is more realistic and plausible? Choose one of the following images." An example of a questionnaire displayed to the participants is shown in Fig. 3.6 (left). To check whether the response from a participant is reliable, all questionnaires include vigilance tests requiring participants to select the more visually plausible image between two options, one of which is an edited result from DragDiffusion [83] containing noticeable visual artifacts. An example of a vigilance test is shown in Fig. 3.6 (right). We present 5 vigilance tests and collect 47 responses from the participants who passed the vigilance test. As summarized in Tab. 3.3, the deformation produced by **APAP** is preferred over the results from ARAP [90].

Similarly to 3D mesh deformation, we evaluate the quality of 2D mesh editing through a user study in which participants compare the editing results of ARAP [90] and **APAP**. Each participant is given a set of 20 randomly selected images of the source meshes, as well as meshes edited using ARAP [90] and **APAP**. In particular, we instructed participants to select the most anticipated outcome when the displayed source image is edited by the dragging operation visualized as an arrow with the question: "A

Methods	Preference (%) ↑
ARAP [90]	40.83
Ours	**59.17**

Table 3.4: User study preference for 2D image editing. In a user study targeting users on Amazon Mechanical Turk (MTurk), the results produced using ours were preferred over the outputs from the baseline.

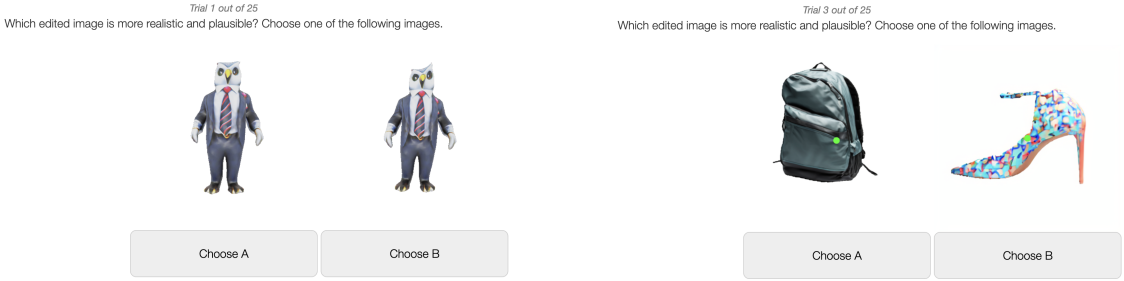

Figure 3.6: Examples of questionnaires displayed during the user study (3D shape deformation). During the user study, we asked the participants to evaluate 20 different result pairs from ARAP [90] and ours as shown on the left. To check whether a participant is focusing on the user study, we included 5 items for the vigilance test. As shown on the right, a vigilance test asks a participant to compare two images, with one of them containing noticeable artifacts.

visual designer wants to modify the object by clicking on a red point and dragging it in the direction of the arrow. Please choose a result that best satisfies the designer's edit, while retaining the characteristics and plausibility of the object." Fig. 3.7 (left) shows an example of a questionnaire provided to the participants. For vigilance tests, we included an editing result from DragDiffusion [83] depicting an object irrelevant to the source image in each question. The participants were asked to answer the same question. We illustrate an example questionnaire of a vigilance test in Fig. 3.7 (right). We collect 102 responses from the participants who passed 5 vigilance tests. Tab. 3.4 shows a higher preference of the participants on our method over ARAP [90] implying that our method produces more visually plausible deformations.

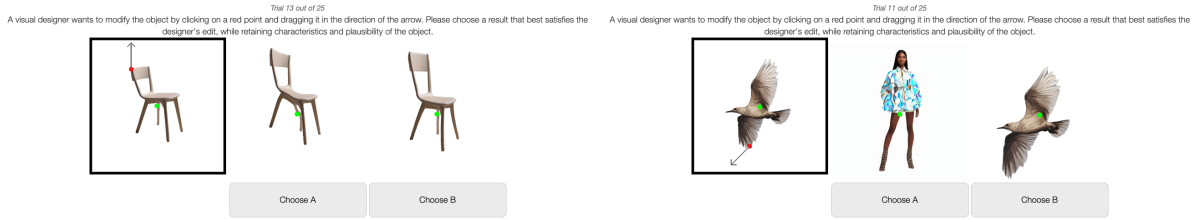

Figure 3.7: Examples of questionnaires displayed during the user study (2D mesh editing). During the user study, we asked the participants to evaluate 20 different result pairs from ARAP [90] and ours as shown on the left. To check whether a participant is focusing on the user study, we included 5 items for the vigilance test. As shown on the right, a question for the vigilance test includes an image of an object that is not related to the source image.

Ablation Study. While designing the algorithm illustrated in Alg. 1, we considered other options for `FirstStage`. Instead of optimizing $\mathcal{L}_h$ to initially deform a shape, we used a shape produced by ARAP [90] or by solving a Poisson's equation constrained not only on anchor positions but also on handles at their target positions reached by following the given edit directions. We report k-NN GIQA scores of the alternatives in the fourth and fifth row of Tab. 3.2, respectively. Both initialization strategies degrade the plausibility of results due to large distortions introduced by either solely enforcing local rigidity or, finding least square solutions without updating Jacobians.

In Fig. 3.8, we summarize the qualitative results obtained by (1) optimizing only $\mathcal{L}_h$, (2) $\mathcal{L}_h$ and $\mathcal{L}_{\text{SDS}}$ without LoRA finetuning, (3) skipping the `FirstStage`, (4) using ARAP initialization, (5) using Poisson initialization, and (6) Ours. Optimizing only $\mathcal{L}_h$ (the second column) either distorts texture (the fifth row) or inflates or shrinks other parts of the given shape (the seventh and twelfth row). This demonstrates the necessity of a visual prior during deformation. Also, we observe the cases where skipping the `FirstStage` (the fourth column) does not lead to intended deformation as our diffusion prior is reluctant to modify shapes from their original states (the first, second, and fifth row). On the other hand, deformations initialized with the meshes produced by ARAP [90] (the fifth column) or Poisson solve (the sixth column) suffer from distortions that could not be resolved by optimizing $\mathcal{L}_{\text{SDS}}$ in the `SecondStage`.

3.5 Conclusion

We presented **APAP**, a novel deformation framework that tackles the problem of plausibility-aware shape deformation while offering intuitive controls over a wide range of shapes represented as triangular meshes. To this end, we carefully orchestrate two core components, a learnable Jacobian-based parameterization that originates from geometry processing and powerful 2D priors acquired by text-to-image diffusion models trained on Internet-scale datasets. We assessed the performance of the proposed method against an existing geometric-prior-based deformation technique and also thoroughly investigated the significance of our design choices through experiments.

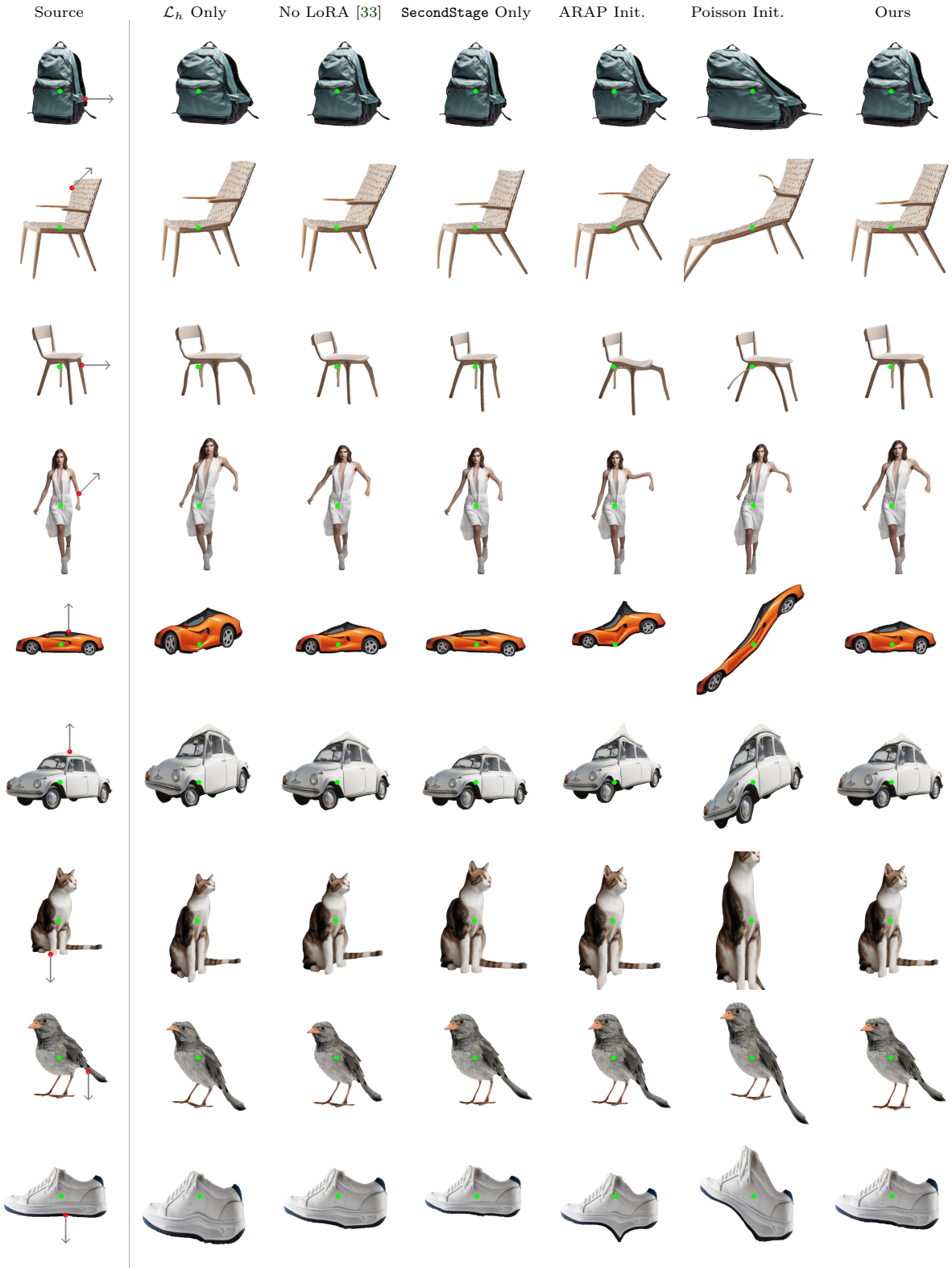

Figure 3.8: Ablation study for 2D mesh editing. We examine the impact of each design choice on deformation outputs, including the use of diffusion prior (the second column), LoRA finetuning (the third column), two-stage pipeline (the fourth column), and initialization strategies during the `FirstStage` (the fifth and sixth column).

Chapter 4. Conclusion

Throughout this thesis, we have discussed a series of works focused on integrating learned priors into mesh deformation, addressing the challenges posed by the lack of large 3D shape collections, which are costly to populate.

In Ch. 2, we introduced Neural Pose Representation (**NPR**), a novel hybrid representation designed for pose generation and transfer of non-rigid object poses, along with its learning framework. This representation enables accurate pose transfer to different objects, even when trained solely on the pose variations of a single object. Its transferability and generalizability further support pose generation for diverse objects. This is achieved by training a diffusion model on pose representations extracted from one object and transferring the generated representations to others.

Taking one step further, we explored an even more challenging scenario in Ch. 3, where no 3D shape examples are available. To address this, we proposed As-Plausible-As-Possible (**APAP**), a plausibility-aware mesh deformation technique that leverages the prior knowledge of 2D diffusion models in place of 3D shape examples. **APAP** orchestrates a powerful 2D diffusion model and a Jacobian field parameterization of the given shape and performs an iterative optimization to satisfy *both* the plausibility constraints imposed by the diffusion model and the handle constraints defined by the user. This combination enables **APAP** to outperform an existing baseline that relies solely on geometric priors, highlighting the effectiveness of learned priors even in the absence of direct supervision through 3D shape examples.

Our work demonstrates the effectiveness of learned priors for mesh deformation, even when trained on a limited number of 3D shape examples or only 2D images. Additionally, we aim to pursue the following promising directions for future exploration. For instance, extending our approach to alternative 3D shape representations, such as point clouds and implicit functions, could enable intuitive control over emerging formats like neural radiance fields [65] and Gaussian splats [44]. Furthermore, recent advancements in text-to-video generative models [96, 121, 57] present an opportunity for text-driven motion generation. These models could eliminate the need for motion sequence data, which are even more challenging to collect than 3D shape data.

Bibliography

[1] Adobe. Mixamo. https://www.mixamo.com/.

[2] Luma AI. Genie.

[3] Noam Aigerman, Kunal Gupta, Vladimir G Kim, Siddhartha Chaudhuri, Jun Saito, and Thibault Groueix. Neural Jacobian Fields: Learning Intrinsic Mappings of Arbitrary Meshes. *ACM TOG*, 2022.

[4] Mazen Al Borno, Ludovic Righetti, Michael J. Black, Scott L. Delp, Eugene Fiume, and Javier Romero. Robust physics-based motion retargeting with realistic body shapes. *Computer Graphics Forum*, 2018.

[5] Omer Bar-Tal, Lior Yariv, Yaron Lipman, and Tali Dekel. MultiDiffusion: Fusing Diffusion Paths for Controlled Image Generation. In *ICML*, 2023.

[6] Ilya Baran and Jovan Popović. Automatic Rigging and Animation of 3D Characters. *ACM TOG*, 2007.

[7] Ilya Baran, Daniel Vlasic, Eitan Grinspun, and Jovan Popović. Semantic deformation transfer. *ACM TOG*, 2009.

[8] Mirela Ben-Chen, Ofir Weber, and Craig Gotsman. Spatial deformation transfer. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2009.

[9] Mikołaj Bińkowski, Danica J. Sutherland, Michael Arbel, and Arthur Gretton. Demystifying mmd gans. In *ICLR*, 2018.

[10] Mingdeng Cao, Xintao Wang, Zhongang Qi, Ying Shan, Xiaohu Qie, and Yinqiang Zheng. MasaCtrl: Tuning-Free Mutual Self-Attention Control for Consistent Image Synthesis and Editing. In *ICCV*, 2023.

[11] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. ShapeNet: An Information-Rich 3D Model Repository. *arXiv preprint arXiv:1512.03012*, 2015.

[12] Haoyu Chen, Hao Tang, Shi Henglin, Wei Peng, Nicu Sebe, and Guoying Zhao. Intrinsic-extrinsic preserved gans for unsupervised 3d pose transfer. In *ICCV*, 2021.

[13] Jinnan Chen, Chen Li, and Gim Hee Lee. Weakly-supervised 3d pose transfer with keypoints. In *ICCV*, 2023.

[14] Kwang-Jin Choi and Hyeong-Seok Ko. On-line motion retargetting. In *Proceedings. Seventh Pacific Conference on Computer Graphics and Applications (Cat. No.PR00293)*, 1999.

[15] Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, et al. Objaverse-xl: A universe of 10m+ 3d objects. In *NeurIPS*, 2023.

[16] Matt Deitke, Ruoshi Liu, Matthew Wallingford, Huong Ngo, Oscar Michel, Aditya Kusupati, Alan Fan, Christian Laforte, Vikram Voleti, Samir Yitzhak Gadre, Eli VanderBilt, Aniruddha Kembhavi, Carl Vondrick, Georgia Gkioxari, Kiana Ehsani, Ludwig Schmidt, and Ali Farhadi. Objaverse-XL: A Universe of 10M+ 3D Objects. In *NeurIPS*, 2023.

[17] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A Universe of Annotated 3D Objects. In *CVPR*, 2023.

[18] Brian Delhaisse, Domingo Esteban, Leonel Rozo, and Darwin Caldwell. Transfer learning of shared latent spaces between robots with similar kinematic structure. In *International Joint Conference on Neural Networks (IJCNN)*, 2017.

[19] Hugging Face. Diffusers: State-of-the-art diffusion models for image and audio generation in PyTorch.

[20] Hugging Face. DreamBooth fine-tuning with LoRA.

[21] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H. Bermano, Gal Chechik, and Daniel Cohen-Or. An Image is Worth One Word: Personalizing Text-to-Image Generation using Textual Inversion. In *ICLR*, 2023.

[22] Lin Gao, Jie Yang, Yi-Ling Qiao, Yu-Kun Lai, Paul L Rosin, Weiwei Xu, and Shihong Xia. Automatic unpaired shape deformation transfer. *ACM TOG*, 2018.

[23] William Gao, Noam Aigerman, Groueix Thibault, Vladimir Kim, and Rana Hanocka. TextDeformer: Geometry Manipulation using Text Guidance. *ACM TOG*, 2023.

[24] Michael Gleicher. Retargetting motion to new characters. *ACM TOG*, 1998.

[25] Shuyang Gu, Jianmin Bao, Dong Chen, and Fang Wen. GIQA: Generated Image Quality Assessment. In *ECCV*, 2020.

[26] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[27] Amir Hertz, Kfir Aberman, and Daniel Cohen-Or. Delta Denoising Score. In *ICCV*, 2023.

[28] Amir Hertz, Ron Mokady, Jay Tenenbaum, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Prompt-to-Prompt Image Editing with Cross-Attention Control. In *ICLR*, 2023.

[29] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NeurIPS*, 2017.

[30] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *NeurIPS*, 2019.

[31] Jonathan Ho, Chitwan Saharia, William Chan, David J Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *arXiv preprint arXiv:2106.15282*, 2021.

[32] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *ICLR*, 2022.

[33] Yelong Hu, Edward J.and Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. LoRA: Low-Rank Adaptation of Large Language Models. In *ICLR*, 2022.

[34] Takeo Igarashi, Tomer Moscovich, and John F. Hughes. As-Rigid-as-Possible Shape Manipulation. *ACM TOG*, 2005.

[35] Shir Iluz, Yael Vinker, Amir Hertz, Daniel Berio, Daniel Cohen-Or, and Ariel Shamir. Word-As-Image for Semantic Typography. *ACM TOG*, 2023.

[36] Alec Jacobson, Ilya Baran, Jovan Popović, and Olga Sorkine. Bounded Biharmonic Weights for Real-Time Deformation. *ACM TOG*, 2011.

[37] Alec Jacobson, Daniele Panozzo, et al. libigl: A simple C++ geometry processing library, 2018.

[38] Ajay Jain, Amber Xie, and Pieter Abbeel. VectorFusion: Text-to-SVG by Abstracting Pixel-Based Diffusion Models. In *CVPR*, 2023.

[39] Tomas Jakab, Richard Tucker, Ameesh Makadia, Jiajun Wu, Noah Snavely, and Angjoo Kanazawa. KeypointDeformer: Unsupervised 3D Keypoint Discovery for Shape Control. In *CVPR*, 2020.

[40] Hanyoung Jang, Byungjun Kwon, Moonwon Yu, Seong Uk Kim, and Jongmin Kim. A variational u-net for motion retargeting. In *SIGGRAPH Asia 2018 Posters*, 2018.

[41] Jong Chul Ye Jangho Park, Gihyun Kwon. ED-NeRF: Efficient Text-Guided Editing of 3D Scene using Latent Space NeRF. *arXiv*, 2023.

[42] Pushkar Joshi, Mark Meyer, Tony DeRose, Brian Green, and Tom Sanocki. Harmonic Coordinates for Character Articulation. *ACM TOG*, 2007.

[43] Tao Ju, Scott Schaefer, and Joe Warren. Mean Value Coordinates for Closed Triangular Meshes. *ACM TOG*, 2005.

[44] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM TOG*, 2023.

[45] Nasir Mohammad Khalid, Tianhao Xie, Eugene Belilovsky, and Popa Tiberiu. CLIP-Mesh: Generating Textured Meshes from Text Using Pretrained Image-Text Models. *SIGGRAPH ASIA*, 2022.

[46] Kunho Kim, Mikaela Angelina Uy, Despoina Paschalidou, Alec Jacobson, Leonidas J. Guibas, and Minhyuk Sung. OptCtrlPoints: Finding the Optimal Control Points for Biharmonic 3D Shape Deformation. *Computer Graphics Forum*, 2023.

[47] Subin Kim, Kyungmin Lee, June Suk Choi, Jongheon Jeong, Kihyuk Sohn, and Jinwoo Shin. Collaborative Score Distillation for Consistent Visual Synthesis. In *NeurIPS*, 2023.

[48] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[49] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment Anything. In *ICCV*, 2023.

[50] Juil Koo, Seungwoo Yoo, Minh Hieu Nguyen, and Minhyuk Sung. SALAD: Part-Level Latent Diffusion for 3D Shape Generation and Manipulation. In *ICCV*, 2023.

[51] Samuli Laine, Janne Hellsten, Tero Karras, Yeongho Seol, Jaakko Lehtinen, and Timo Aila. Modular Primitives for High-Performance Differentiable Rendering. *ACM TOG*, 2020.

[52] Jehee Lee and Sung Yong Shin. A hierarchical approach to interactive motion editing for human-like figures. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques*, 1999.

[53] Yuseung Lee, Kunho Kim, Hyunjin Kim, and Minhyuk Sung. SyncDiffusion: Coherent Montage via Synchronized Joint Diffusions. In *NeurIPS*, 2023.

[54] Yang Li, Hikari Takehara, Takafumi Taketomi, Bo Zheng, and Matthias Nießner. 4dcomplete: Non-rigid motion estimation beyond the observable surface. In *ICCV*, 2021.

[55] Zhouyingcheng Liao, Jimei Yang, Jun Saito, Gerard Pons-Moll, and Yang Zhou. Skeleton-free pose transfer for stylized 3d characters. In *ECCV*, 2022.

[56] Jongin Lim, Hyung Jin Chang, and Jin Young Choi. Pmnet: Learning of disentangled pose and movement for unsupervised motion retargeting. In *BMVC*, 2019.

[57] Bin Lin, Yunyang Ge, Xinhua Cheng, Zongjian Li, Bin Zhu, Shaodong Wang, Xianyi He, Yang Ye, Shenghai Yuan, Liuhan Chen, et al. Open-sora plan: Open-source large video generation model. *arXiv preprint arXiv:2412.00131*, 2024.

[58] Yaron Lipman, David Levin, and Daniel Cohen-Or. Green Coordinates. *ACM TOG*, 2008.

[59] Yaron Lipman, Olga Sorkine, Daniel Cohen-Or, David Levin, Christian Rössl, and Hans-Peter Seidel. Differential Coordinates for Interactive Mesh Editing. In *Proceedings of Shape Modeling International*, pages 181–190, 2004.

[60] Yaron Lipman, Olga Sorkine, David Levin, and Daniel Cohen-Or. Linear Rotation-Invariant Coordinates for Meshes. *ACM TOG*, 2005.

[61] Minghua Liu, Minhyuk Sung, Radomir Mech, and Hao Su. DeepMetaHandles: Learning Deformation Meta-Handles of 3D Meshes with Biharmonic Coordinates. In *CVPR*, 2021.

[62] Yuan Liu, Cheng Lin, Zijiao Zeng, Xiaoxiao Long, Lingjie Liu, Taku Komura, and Wenping Wang. SyncDreamer: Learning to Generate Multiview-consistent Images from a Single-view Image. *arXiv*, 2023.

[63] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *ACM TOG*, 2015.

[64] Oscar Michel, Roi Bar-On, Richard Liu, Sagie Benaim, and Rana Hanocka. Text2Mesh: Text-Driven Neural Stylization for Meshes. In *CVPR*, 2022.

[65] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

[66] Xingang Pan, Ayush Tewari, Thomas Leimkühler, Lingjie Liu, Abhimitra Meka, and Christian Theobalt. Drag Your GAN: Interactive Point-based Manipulation on the Generative Image Manifold. *ACM TOG*, 2023.

[67] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *CVPR*, 2019.

[68] Taesung Park, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. Semantic image synthesis with spatially-adaptive normalization. In *CVPR*, 2019.

[69] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed A. A. Osman, Dimitrios Tzionas, and Michael J. Black. Expressive body capture: 3d hands, face, and body from a single image. In *CVPR*, 2019.

[70] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving Latent Diffusion Models for High-Resolution Image Synthesis. *arXiv*, 2023.

[71] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. DreamFusion: Text-to-3D using 2D Diffusion. In *ICLR*, 2023.

[72] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *CVPR*, 2017.

[73] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning Transferable Visual Models from Natural Language Supervision. In *ICML*, 2021.

[74] Amit Raj, Srinivas Kaza, Ben Poole, Michael Niemeyer, Ben Mildenhall, Nataniel Ruiz, Shiran Zada, Kfir Aberman, Michael Rubenstein, Jonathan Barron, Yuanzhen Li, and Varun Jampani. DreamBooth3D: Subject-Driven Text-to-3D Generation. In *ICCV*, 2023.

[75] Helge Rhodin, James Tompkin, Kwang In Kim, Edilson de Aguiar, Hanspeter Pfister, Hans-Peter Seidel, and Christian Theobalt. Generalizing wave gestures from sparse examples for real-time character control. *ACM TOG*, 2015.

[76] Daniel Ritchie. Rudimentary framework for running two-alternative forced choice (2afc) perceptual studies on mechanical turk.

[77] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022.

[78] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. DreamBooth: Fine Tuning Text-to-image Diffusion Models for Subject-Driven Generation. In *CVPR*, 2023.

[79] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. In *NeurIPS*, 2022.

[80] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. LAION-5B: An open large-scale dataset for training next generation image-text models. In *NeurIPS*, 2022.

[81] Tianchang Shen, Jun Gao, Kangxue Yin, Ming-Yu Liu, and Sanja Fidler. Deep Marching Tetrahedra: a Hybrid Representation for High-Resolution 3D Shape Synthesis. In *NeurIPS*, 2021.

[82] Yichun Shi, Peng Wang, Jianglong Ye, Long Mai, Kejie Li, and Xiao Yang. MVDream: Multi-view Diffusion for 3D Generation. *arXiv*, 2023.

[83] Yujun Shi, Chuhui Xue, Jiachun Pan, Wenqing Zhang, Vincent YF Tan, and Song Bai. DragDiffusion: Harnessing Diffusion Models for Interactive Point-based Image Editing. *arXiv*, 2023.

[84] J. Ryan Shue, Eric Ryan Chan, Ryan Po, Zachary Ankner, Jiajun Wu, and Gordon Wetzstein. 3D Neural Field Generation using Triplane Diffusion. In *CVPR*, 2023.

[85] Chaoyue Song, Jiacheng Wei, Ruibo Li, Fayao Liu, and Guosheng Lin. 3d pose transfer with correspondence learning and mesh refinement. In *NeurIPS*, 2021.

[86] Chaoyue Song, Jiacheng Wei, Ruibo Li, Fayao Liu, and Guosheng Lin. Unsupervised 3d pose transfer with cross consistency and dual reconstruction. *IEEE TPAMI*, 2023.

[87] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *ICLR*, 2021.

[88] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *NeurIPS*, 2019.

[89] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *ICLR*, 2021.

[90] Olga Sorkine and Marc Alexa. As-Rigid-As-Possible Surface Modeling. *Proceedings of EUROGRAPHICS/ACM SIGGRAPH Symposium on Geometry Processing*, pages 109–116, 2007.

[91] Olga Sorkine, Daniel Cohen-Or, Yaron Lipman, Marc Alexa, Christian Rössl, and Hans-Peter Seidel. Laplacian Surface Editing. *Proceedings of the EUROGRAPHICS/ACM SIGGRAPH Symposium on Geometry Processing*, pages 179–188, 2004.

[92] Robert W. Sumner and Jovan Popović. Deformation transfer for triangle meshes. *ACM TOG*, 2004.

[93] Seyoon Tak and Hyeong-Seok Ko. A physically-based motion retargeting filter. *ACM TOG*, 2005.

[94] Jiapeng Tang, Markhasin Lev, Wang Bi, Thies Justus, and Matthias Nießner. Neural Shape Deformation Priors. In *NeurIPS*, 2022.

[95] Jiaxiang Tang, Jiawei Ren, Hang Zhou, Ziwei Liu, and Gang Zeng. DreamGaussian: Generative Gaussian Splatting for Efficient 3D Content Creation. *arXiv*, 2023.

[96] Genmo Team. Mochi 1. https://github.com/genmoai/models, 2024.

[97] Narek Tumanyan, Michal Geyer, Shai Bagon, and Tali Dekel. Plug-and-Play Diffusion Features for Text-Driven Image-to-Image Translation. In *CVPR*, 2023.

[98] Ruben Villegas, Duygu Ceylan, Aaron Hertzmann, Jimei Yang, and Jun Saito. Contact-aware retargeting of skinned motion. In *ICCV*, 2021.

[99] Ruben Villegas, Jimei Yang, Duygu Ceylan, and Honglak Lee. Neural kinematic networks for unsupervised motion retargetting. In *CVPR*, 2018.

[100] Jiashun Wang, Xueting Li, Sifei Liu, Shalini De Mello, Orazio Gallo, Xiaolong Wang, and Jan Kautz. Zero-shot pose transfer for unrigged stylized 3D characters. In *CVPR*, 2023.

[101] Jiashun Wang, Chao Wen, Yanwei Fu, Haitao Lin, Tianyun Zou, Xiangyang Xue, and Yinda Zhang. Neural pose transfer by spatially adaptive instance normalization. In *CVPR*, 2020.

[102] Yu Wang, Alec Jacobson, Jernej Barbič, and Ladislav Kavan. Linear Subspace Design for Real-Time Shape Deformation. *ACM TOG*, 2015.

[103] Zhengyi Wang, Cheng Lu, Yikai Wang, Fan Bao, Chongxuan Li, Hang Su, and Jun Zhu. Prolificdreamer: High-fidelity and diverse text-to-3d generation with variational score distillation. In *NeurIPS*, 2023.

[104] Ofir Weber, Mirela Ben-Chen, and Craig Gotsman. Complex barycentric coordinates with applications to planar shape deformation. *Computer Graphics Forum*, 2009.

[105] Ofir Weber, Olga Sorkine, Yaron Lipman, and Craig Gotsman. Context-Aware Skeletal Shape Deformation. *Computer Graphics Forum*, 2007.

[106] Tong Wu, Jiarui Zhang, Xiao Fu, Yuxin Wang, Jiawei Ren, Liang Pan, Wayne Wu, Lei Yang, Jiaqi Wang, Chen Qian, Dahua Lin, and Ziwei Liu. OmniObject3D: Large-Vocabulary 3D Object Dataset for Realistic Perception, Reconstruction and Generation. In *CVPR*, 2023.

[107] Weiwei Xu, Kun Zhou, Yizhou Yu, Qifeng Tan, Qunsheng Peng, and Baining Guo. Gradient domain editing of deforming mesh sequences. In *ACM TOG*, 2007.

[108] Zhan Xu, Yang Zhou, Evangelos Kalogerakis, Chris Landreth, and Karan Singh. RigNet: Neural Rigging for Articulated Characters. *ACM TOG*, 2020.

[109] Zhan Xu, Yang Zhou, Li Yi, and Evangelos Kalogerakis. Morig: Motion-Aware Rigging of Character Meshes from Point Clouds. In *SIGGRAPH ASIA*, 2022.

[110] Katsu Yamane, Yuka Ariki, and Jessica Hodgins. Animating non-humanoid characters with human motion data. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2010.

[111] Jie Yang, Lin Gao, Yu-Kun Lai, Paul L. Rosin, and Shihong Xia. Biharmonic deformation transfer with automatic key point selection. *Graphical Models*, 2018.

[112] Hu Ye, Jun Zhang, Sibo Liu, Xiao Han, and Wei Yang. Ip-adapter: Text compatible image prompt adapter for text-to-image diffusion models. *arXiv preprint arxiv:2308.06721*, 2023.

[113] Wang Yifan, Noam Aigerman, Vladimir G. Kim, Chaudhuri Siddhartha, and Olga Sorkine. Neural Cages for Detail-Preserving 3D Deformations. In *CVPR*, 2020.

[114] Seungwoo Yoo, Kunho Kim, Vladimir G. Kim, and Minhyuk Sung. As-Plausible-As-Possible: Plausibility-Aware Mesh Deformation Using 2D Diffusion Priors. In *CVPR*, 2024.

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

[116] Yizhou Yu, Kun Zhou, Dong Xu, Xiaohan Shi, Hujun Bao, Baining Guo, and Heung-Yeung Shum. Mesh editing with poisson-based gradient field manipulation. *ACM TOG*, 2004.

[117] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding Conditional Control to Text-to-Image Diffusion Models. In *ICCV*, 2023.

[118] Yuechen Zhang, Jinbo Xing, Eric Lo, and Jiaya Jia. Real-World Image Variation by Aligning Diffusion Inversion Chain. In *NeurIPS*, 2023.

[119] Hengshuang Zhao, Jiaya Jia, and Vladlen Koltun. Exploring self-attention for image recognition. In *CVPR*, 2020.

[120] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip Torr, and Vladlen Koltun. Point transformer. In *ICCV*, 2021.

[121] Zangwei Zheng, Xiangyu Peng, Tianji Yang, Chenhui Shen, Shenggui Li, Hongxin Liu, Yukun Zhou, Tianyi Li, and Yang You. Open-sora: Democratizing efficient video production for all, March 2024.

[122] Keyang Zhou, Bharat Lal Bhatnagar, and Gerard Ons-Moll. Unsupervised shape and pose disentanglement for 3d meshes. In *ECCV*, 2020.

[123] Jingyu Zhuang, Chen Wang, Lingjie Liu, Liang Lin, and Guanbin Li. DreamEditor: Text-Driven 3D Scene Editing with Neural Fields. *ACM TOG*, 2023.

Acknowledgments in Korean

제가 학부 연구생으로 연구실 생활을 시작했던 2021년 여름 이후로 어느덧 3년 반이라는 시간이 흘렀습니다. 컴퓨터 그래픽스와 비전에 대한 호기심으로 가득 차, 당시로서는 난해하게만 느껴졌던 논문들을 이해해 보려 애쓰던 어리숙한 저의 모습이 아직도 연구실 곳곳에 남아있는 듯합니다. 석사 과정의 마침표를 찍는 지금이 앞으로 제가 연구자로서 걸어야 할 긴 여정의 출발선임을 알기에, 제가 여기에 서기까지 도움을 주신 분들에 대한 감사의 마음도 각별합니다. 제가 받은 도움에 비하면 턱없이 부족하겠지만, 이 글을 통해 제 진심이 온전히 전해지기를 바랍니다.

연구 경험이 전무했던 제가 어엿한 연구자로 성장할 수 있도록 오랜 시간 가장 가까이에서 지켜봐 주시고 이끌어 주신 성민혁 교수님께 진심으로 감사드립니다. 지금까지의 성취는 학부 연구생 시절부터 석사 과정에 이르기까지 교수님께서 아낌없이 주신 격려와 조언, 그리고 가르침이 없었다면 결코 이루어낼 수 없었을 것입니다. 매 순간 학문과 후학 양성에 힘쓰시는 교수님의 모습을 보며, 참된 연구자의 길이 무엇인지 깊이 깨달을 수 있었습니다. 제가 진심으로 존경할 수 있는 분을 스승으로 만난 것은 제게 큰 축복이었습니다.

또한, 본 학위논문의 심사를 위해 귀중한 시간을 내주시고, 심사 과정에서 소중한 조언과 통찰을 주신 전산학부 김민혁 교수님과 윤성의 교수님께 깊이 감사드립니다. 학부 시절 두 분의 컴퓨터 그래픽스와 비전 강의를 통해 기초를 다졌던 저로서는, 두 분을 심사위원으로 모실 수 있었던 것이 큰 영광이었습니다.

연구실 생활을 함께하며 힘이 되어준 연구실 동료들에게도 감사의 마음을 전합니다. 학문적인 논의를 넘어, 때로는 즐거운 순간들을, 때로는 힘든 순간들을 스스럼없이 나눌 수 있는 사람들과 함께 일할 수 있어 참으로 행복했습니다. 특히 연구실 초창기부터 함께해 준 주일이 형, 현진이 형, 건호 형, 은지 누나, 유승이 형, 찬혁이, Charlie를 비롯해 유쾌하고 따뜻한 형이 되어준 재훈이 형과 찬호 형, 탁월함과 엉뚱함을 두루 갖춘 동갑 친구 지성이와 태훈이 형, 어느덧 반년을 함께한 민규 형과, 배울 점이 많은 후배 경민이까지, 모두가 뛰어난 실력만큼이나 따뜻한 마음씨를 가졌기에, 이들과 함께할 수 있었던 시간은 제게 과분한 행운이었다고 생각합니다. 이와 더불어, 연구에 매진하는 동안 묵묵히 곁을 지켜준 오랜 친구들에게도 진심으로 감사의 마음을 전합니다. 일이 바빠 몇 달에 한 번씩 얼굴을 비추는 부족한 친구 곁에서 늘 변함없이 힘이 되어주어 고마울 뿐입니다.

끝으로, 지금까지 제가 걸어온 길을 응원해 주신 어머니, 아버지께 이 논문을 바칩니다. 무뚝뚝하고 표현이 서툰 큰아들에게 종종 서운하지는 않으셨을까 하는 걱정과 미안한 마음이 앞섭니다. 제가 긴 고민과 방황에 지쳐 있을 때면 말없이 다가와 안락한 울타리가 되어 저를 품어주신 부모님, 사랑합니다. 그리고 하나뿐인 내 동생 시현이, 앞으로는 늘 건강하고 행복하자.

Curriculum Vitae in Korean

이 름: 유 승 우

생 년 월 일: 2001년 01월 15일

출 생 지: 전라북도 전주시 (現 전북특별자치도 전주시)

학 력

2017. 3. – 2019. 2. 전북과학고등학교

2019. 2. – 2023. 8. KAIST 전산학부 (학사)

2023. 8. – 2025. 2. KAIST 전산학부 (석사)

경 력

2019 – 2022 대통령과학장학생

2021 – 2023 KAIST Visual AI Group 학부 인턴

2024 대학원대통령과학장학생

연 구 업 적

1. Juil Koo*, **Seungwoo Yoo***, Minh Hieu Nguyen*, and Minhyuk Sung. SALAD: Part-Level Latent Diffusion for 3D Shape Generation and Manipulation. In *ICCV*, 2023.
2. **Seungwoo Yoo***, Kunho Kim*, Vladimir G. Kim, and Minhyuk Sung. As-Plausible-As-Possible: Plausibility-Aware Mesh Deformation Using 2D Diffusion Priors. In *CVPR*, 2024.
3. **Seungwoo Yoo**, Juil Koo, Kyeongmin Yeo, and Minhyuk Sung. Neural Pose Representation Learning for Generating and Transferring Non-Rigid Object Poses. In *NeurIPS*, 2024.