TwoSquared: 4D Generation from 2D Image Pairs

Lu Sang[*1,2], Zehranaz Canfes[*1], Dongliang Cao[3]
Riccardo Marin[1,2], Florian Bernard[3], Daniel Cremers[1,2]
[1]Technical University of Munich, [2]Munich Center of Machine Learning
[3]University of Bonn

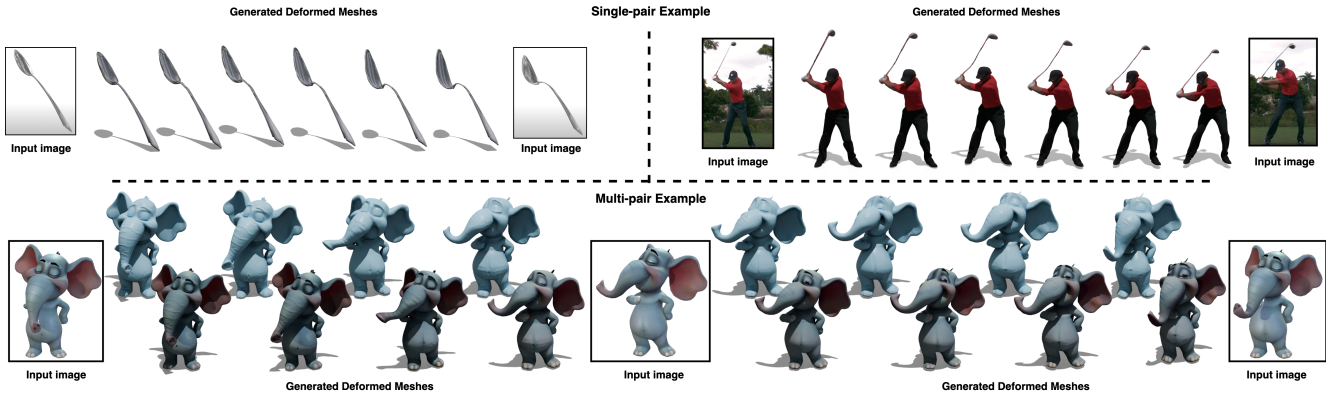

Figure 1. **TwoSquared results:** TwoSquared takes a pair of 2D images representing the initial and final states of an object as input and generates texture-consistent, geometry-consistent 4D continuous sequences. It is designed to be robust to varying input quality, operating without the need for predefined templates or object-class priors. This adaptability enables greater flexibility in processing diverse images while maintaining structural integrity and visual coherence throughout the generated sequences. As demonstrated, our approach effectively handles humans, objects, and inanimate objects. The code is available at https://sangluisme.github.io/TwoSquared/.

Abstract

Recovering a 4D motion from sparse visual information (such as two temporal frames of a subject) is a significant challenge. While humans are able to hallucinate the missing information in a plausible way, generative AI struggles due to a lack of high-quality training data and heavy computing requirements. To overcome these limitations, we propose TwoSquared, a method that obtains a 4D plausible sequence from just two 2D RGB images corresponding to the beginning and the end of the action. We propose to solve the problem in two steps: 1) first, obtaining a 3D reconstruction of the initial and final status, and 2) model the intermediate sequence as a physically plausible deformation. Our method does not require templates or class-specific prior knowledge, and can operate with arbitrary in-the-wild examples. We demonstrate our capabilities in a number of different objects, diverse in terms of nature, class, and deformation, surpassing video-based alternatives, which cannot achieve the same level of consistency.

[*] These authors contributed equally.

1. Introduction

As humans, we naturally reason about our surrounding world as a 4D space, where 3D objects evolve over time, moving and deforming their shape following physical laws. Modeling and analyzing such 4D representations is a crucial step for Spatial AI, and so crucial for a number of fields such as character animation [15], virtual reality (VR) [1, 42], robotics, and autonomous systems [54]. Most of the existing 4D generation methods heavily rely on highly controlled input data, such as synchronized multi-view video recordings [31, 37, 60]. However, deploying such systems is demanding, significantly restricted in terms of capturing volume, and often economically out of reach of labs. Relying on lightweight settings is appealing, but it also opens up to growing levels of uncertainty, making it difficult to provide texture and geometry consistency throughout the 4D sequence and to model deformations that preserve structural integrity. Such attributes are often enforced by relying on category-specific templates [19, 67, 69] or object-class-specific prior knowledge [8, 65], constraining the methods' applicative domains only to popular and well-studied ob-

jects. It is worth mentioning that recent developments in deep learning architectures and data availability have also opened up generalistic approaches [27, 58, 59], which hallucinate 3D assets even from a single image of disparate objects. Despite these advancements, synthesizing temporally coherent moving objects remains an open challenge, and direct synthesis of video [2, 7] produces visually impressive results, but with fundamental inaccuracies due to a lack of explicit physical consistency. In this work, we tackle the ambitious goal of recovering the 4D dynamic of an object observed from the most generic setting possible: two 2D RGB images of the initial and final state. Tackling this problem, we found two sources of ambiguity: those coming from the unknown geometry of the object, and those related to the movement between two different static observations. We advocate treating the two separately, and we develop this intuition in TwoSquared. Our novel method starts solely from two single RGB images and combines 3D Generative AI with a generic physically inspired deformation module. Given the two images, we rely on a state-of-the-art generative AI module to obtain the 3D geometry of the initial and final state. Then, we run a physically-inspired optimization that interpolates between the two, promoting consistency along all the deformation, which is continuous and can be sampled at an arbitrary framerate. Our approach is generic and does not need any template or class-based prior. Compared to state-of-the-art 4D generation methods that rely on temporal input data (e.g., full video sequences) and employ cross-attention mechanisms across temporal, spatial, and multi-view dimensions [63], we obtain more plausible sequences while being significantly cheaper in terms of computation. In summary:

1. We propose TwoSquared, the first method to solve for *4D generation starting only from a pair of 2D frames as input*.
2. TwoSquared achieves physically plausible deformations, while maintaining texture consistency throughout the 4D sequence, being lightweight and class-agnostic;
3. We demonstrate superior performance over state-of-the-art methods, opening up new applications. The release of our code will provide a useful tool to recover a 4D sequence from sparse observations, as well as ease its extension by future work.

2. Related Works

2.1. Single Image to 3D Generation

Generating high quality 3D representations (e.g. meshes [23, 26, 55, 57], NeRF [13, 29, 30], 3DGS [18, 48, 68]) is a popular research direction. Early works [38, 44, 49, 56] attempted to distill prior knowledge from 2D image diffusion models [35, 39] to create 3D models from text or images via Score Distillation Sampling [36]. Despite their compelling results, these methods suffer from two main limitations: efficiency and consistency, due to per-instance optimization and single-view ambiguity [26]. To improve the efficiency, recent works [24–26, 45] separated the generation process into multi-view generation and 3D reconstruction. To generate consistent multi-view images, pretrained 2D image diffusion models are finetuned using large 3D object datasets [9, 10]. Despite the visual appealing results, the reconstructed meshes often fail to meet the requirements for downstream tasks (e.g. shape animation). To guarantee high-quality mesh generation, the most recent methods [27, 50, 61, 66] discard the use of pre-trained 2D image diffusion models but train a 3D shape generation model from scratch, resulting in detailed geometry generation. In our work, we will rely on the most recent single-image to mesh generation method [50].

2.2. 4D Generation

To generate 4D assets, it is common to rely on predefined templates, such as parametric models of human bodies [8, 33, 34], faces, or animals [19, 69]. Templates provide robust priors on objects' shape and deformation, but they also restrict the applicability, as there are no such deformable templates for the majority of the classes. Instead, an object's motion can also be inferred from videos. Methods such as [20, 21, 62, 67] take a video as input, generating a 3D shape for every frame, which are then combined into 4D sequences. Such input is often unavailable, and generating a 3D shape for every frame accumulates inconsistencies along the sequence. To overcome these inconsistencies, V2M4 [6] first generates 3D meshes for each frame using a 3D generation backbone, then tracks camera motion using dense stereo priors to re-pose and align the meshes with the input video, and applies pairwise registration to enforce consistency across meshes. However, this process requires high computational power. Another option is also to directly rely on 3D input in the form of point clouds [5, 51]. While such methods can successfully reconstruct dynamic 4D shapes, they require access to point clouds or meshes of the same object for the entire sequence. Additionally, [5] is limited to interpolating only within the set of trained objects. Some methods, such as [41], work using only 3D keyframes. While improving robustness in dynamic scenes, it relies on having access to clean or pre-registered training 3D data. Since our approach starts only from RGB images, we cannot assume this. Finally, generating 4D sequences from 2D inputs has also been investigated [22, 47] leveraging diffusion models or text prompts to generate image sequences, which are then used to obtain 4D outputs. Consequently, their results are limited by the backbone diffusion or language models, are hardly controllable, and are mostly achieved on synthetic images.

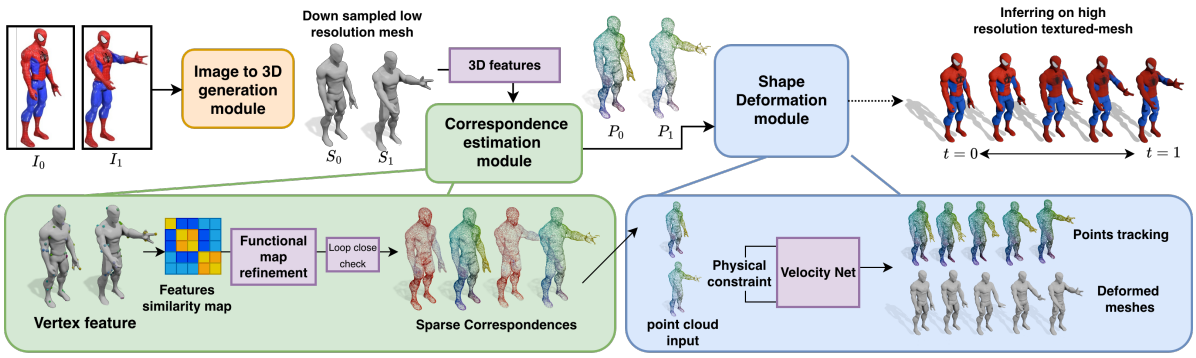

Figure 2. **Pipeline of TwoSquared**: TwoSquared processes two input images through an image-to-3D generation block, producing two 3D meshes. We then extract per-vertex features and compute a cosine similarity map, which is refined using a functional map module and a closed-loop check module to obtain point-to-point correspondences. These registered points are then fed into our shape deformation module, where we model the trajectory of the deformed point cloud. During the inference time, we can directly infer the generated textured mesh from I_0 to obtain the 4D sequence.

3. Method

Motivation and overview. We aim to obtain a 4D sequence of meshes starting solely from two RGB images depicting its initial and final state. We desire a deformation that is consistent, geometrically coherent, and physically plausible, while as general as possible. As illustrated in Fig. 2, TwoSquared consists of three components: a 3D Generation block, which recovers the 3D shapes from the two input images; a Vertex Registration block, which recovers correspondences between the two shapes; and a Shape Deformation block, which recovers the 4D deformation while ensuring smoothness and realism. We remark that our output deformation is continuous, and hence can be sampled at an arbitrary frame rate.

3.1. 3D Generation Block

Given the two frames $\mathcal{I} = \{I_i\},\ i \in \{0, 1\}$ as input, the first step is recovering 3D meshes $\mathcal{S} = \{S_i\},\ i \in \{0, 1\}$ out of them. Our method is not tied to any specific approach, and we decided to adopt Hunyuan3D [50] as our 3D generation block, since it is among the most recent and performing image-to-3D generative AI methods. We found that Hunyuan3D can provide realistic geometries for a disparate class of objects, making it a good fit for our generality purpose. Alongside the detailed geometry, the method also generates high-resolution texture maps using its texture-painting module. In our pipeline, by feeding the network with a pair of keyframe images $\mathcal{I} = \{I_i\},\ i \in \{0, 1\}$, we obtain 3D meshes $\mathcal{S} = \{S_i\},\ i \in \{0, 1\}$ of keyframes.

3.2. Vertex Registration Block

Mesh Downsampling. Our next step is recovering a sparse set of correspondence between S_0 and S_1, which serves as a global guidance on how the semantic parts of the first shape should deform to match the second one. Note that we cannot rely on the assumption that the two shapes are geometrically consistent with each other. Since we recover the geometry using generative AI starting from partial observations, the backbone can hallucinate different local geometries for the two, especially on the unseen parts. Hence, before running the Registration Block, our idea is to downsample the shapes to $\sim$ 4000 vertices. This step makes it simpler to characterize the semantics of local geometry, and we observed in our experiments that it helps to have good-quality correspondence.

Semantic correspondence. Then, we render $N = 100$ depth and normal images and follow the pipeline of Diff3F [11] to obtain the diffusion features for the mesh vertices, unprojecting the per-pixel features to the 3D meshes. For each mesh i with v_i vertices, we obtain a feature matrix with dimension $\mathcal{F}_i \in \mathbb{R}^{v_i \times f}$, where $f = 2048$ is the feature dimension. After that, for mesh S_i and S_j, we compute the cosine similarity matrix $M_{ij} \in \mathbb{R}^{v_i \times v_j}$ from their feature matrices $\mathcal{F}_i, \mathcal{F}_j$.

Thus, we can obtain point-to-point correspondences through M_{ij}. However, the correspondences obtained directly from M_{ij} are noisy and inaccurate, since they only consider the information of each point, leading to errors such as multiple points in S_i being assigned the same point in S_j. To solve this problem, we treat M_{ij} as a functional map [32] between S_i and S_j, then perform a smooth discrete optimization algorithm [28] to refine the point-to-point correspondences. We compute the point-to-point correspondences in both directions, *i.e.*, from shape i to shape j, denoted as M_{ij}, and shape j to shape i, denoted as M_{ji}, then we perform a close-loop check, that is, a vertex in shape i mapped to j using M_{ij} and mapped back using M_{ji} should end up in the same vertex. For $\mathbf{x}$ in shape i, we compute the mis-mapped distance as

$$\mathcal{D}_i^j = \|\mathbf{x}_i - M_{ji}(M_{ij}(\mathbf{x}_i))\|\,,\ . \tag{1}$$

We only select the correspondences that are mapped close

enough, *i.e.*, $\mathcal{D}_i^j < \delta_d$, where δ_d is the distance threshold to filter out mismatched vertices. To increase the robustness, if after the bidirectional check, fewer than 1000 points are left, we initialize point-to-point correspondences of mesh pairs using nearest neighbor search and perform the smooth discrete optimization algorithm again.

3.3. Shape Deformation Block

Overview. At the core of our method is the modeling of the continuous deformation between the two shapes. After the 3D generation and registration block, two shapes generated from images and a set of noisy correspondences for (part of) the vertices are given. Now we need to model the deformed intermediate shapes between the pair. We solve this problem by modeling a point trajectory using a Velocity Net $\mathcal{V} : \mathbb{R}^3 \times [0, T] \to \mathbb{R}^3$.

3.3.1 Problem Formulation.

Given two point clouds P_0, $P_1 \subset \mathbb{R}^3$ sampled from two shapes S_0 and S_1, we would like to find the suitable path that not only moves P_0 to P_1, but also meets some physical condition, because the deformation between two shapes needs to be physically plausible and geometrically temporal consistent, *i.e.*, the structural and spatial properties of the surfaces are preserved and the deformation is smooth. For point $\mathbf{x} \in \Omega_i \subset P_i, i \in \{0, 1\}$, where Ω_i is the point domain and $\Omega = \Omega_0 \times \Omega_1$, we want to recover its trajectory $\mathbf{X}(t)$ with minimum kinematic energy [4]. The problem can be formulated as

$$\min_{\mathbf{X}(t)} \int_0^T \int_\Omega \frac{1}{2} \left\| \frac{\mathrm{d}\mathbf{X}(t)}{\mathrm{d}t} \right\|^2 \mathrm{d}\mathbf{x}\mathrm{d}t \ , \tag{2}$$

$$\text{s.t.} \int_{\Omega_0} \mathbf{X}(0)\mathrm{d}\mathbf{x} = P_0, \int_{\Omega_1} \mathbf{X}(T)\mathrm{d}\mathbf{x} = P_1 \ . \tag{3}$$

Instead of directly modeling the trajectory function, we model the velocity function, i.e., we estimate $\mathcal{V} : \mathbb{R}^3 \times \mathbb{R} \to \mathbb{R}^3$, such that

$$\mathcal{V}(\mathbf{x}, t) = \frac{\mathrm{d}\mathbf{X}(t)}{\mathrm{d}t} \ , \tag{4}$$

and we could rewrite Eq. (2) using a velocity formulation:

$$\mathcal{L}_v = \int_0^T \int_\Omega \|\mathcal{V}(\mathbf{x}, t)\|_f^2 \, \mathrm{d}\mathbf{x}\mathrm{d}t \ , \tag{5}$$

where $\|\cdot\|_f$ is a defined norm on functional space.

Remark. One might wonder why formulating the problem as an optimal transport modeled as a velocity field between points instead of relying on mesh deformation. We observe that points as input give more flexibility to infer on different resolution data and free us from dealing with self-intersection when modeling the deformation. and inconsistencies in local resolutions. The use of an explicit and extrinsic velocity field makes the formulation of physically plausible properties simpler, as we are going to show in the next section. Finally, learning a continuous velocity field enables us to optimize our modules on low-resolution point clouds, where we downsampled in the vertex-registration block. This is efficient and adaptable to the available resources, while it does not limit our inference, which can still operate on an arbitrarily high-resolution mesh, as we will show in the following section.

3.3.2 Physical Plausible Velocity Field

We need not only to solve the optimal path problem but also to ensure the path is physically realistic in the 3D world. To further control our path, we add physics-based constraints to our velocity field.

Divergence-free loss and smoothness loss. To ensure a smooth velocity field, i.e., recover a smooth trajectory $\mathbf{X}(t)$, we follow [40] to use the Fobius norm on $\|\mathcal{V}\|_f$ as a functional space norm in Eq. (4). To deal with iso-metric deformation, we also add a divergence-free loss [40] $\mathcal{L}_d$ to ensure the velocity field is volume-perserving:

$$\begin{aligned} \mathcal{L}_v &= \int_\Omega \|(-\alpha\Delta + \gamma\mathbf{I})\mathcal{V}(\mathbf{x}, t)\|_{l^2} \, \mathrm{d}\mathbf{x}, \\ \mathcal{L}_{div} &= \int_\Omega |\nabla \cdot \mathcal{V}(\mathbf{x}, t)| \mathrm{d}\mathbf{x} \ . \end{aligned} \tag{6}$$

Distortion loss. As our training data is a set of points, unlike mesh data, one can use constraints such as As-Rigid-As-Possible (ARAP) [46] on neighboring vertices to enforce rigid movement. To reduce the distortion at each individual point, we follow [41] to incorporate distortion loss, for $\mathbf{D} = \frac{1}{2}(\nabla\mathcal{V} + (\nabla\mathcal{V})^\top)$

$$\mathcal{L}_d = \int_\Omega \left\| \frac{1}{6} \operatorname{Tr}(\mathbf{D})^2 - \frac{1}{2} \operatorname{Tr}(\mathbf{D} \cdot \mathbf{D})^2 \right\|_F \mathrm{d}\mathbf{x} \ . \tag{7}$$

Overlapping loss. To measure how good the point cloud P_0 is moved by our velocity field $\mathcal{V}$ at time T, that is, our velocity field satisfies Eq. (3), we add one overlapping penalty:

$$\mathcal{L}_o = \operatorname{dist}(\int_0^T \int_{\Omega_0} \mathcal{V}(\mathbf{x}, t)\mathrm{d}\mathbf{x}\mathrm{d}t - P_1) \ . \tag{8}$$

The loss requires that after P_0 is moved by the velocity field, it overlaps with P_1. We use the Chamfer distance as the overlapping metric in our case.

Normal loss. As we sample points from meshes, we can also rely on well-behaved normals. We denote the normal of a point $\mathbf{x}$ as $\mathbf{n}(\mathbf{x})$. At each time step, the point is moved by:

$$\mathbf{x}' = \mathbf{x} + \mathcal{V}(\mathbf{x}, t)\Delta t \ . \tag{9}$$

The local deformation near the vertex $\mathbf{x}$ can be approximated by the deformation gradient [16]:

$$\mathbf{F}(\mathbf{x}, t) = \mathbf{I} + \nabla \mathcal{V}(\mathbf{x}, t) \, . \tag{10}$$

The normal can also be updated from $\mathbf{x}$ to $\mathbf{x}'$:

$$\mathbf{n}' = \mathbf{F}^{-\top} \mathbf{n} \, . \tag{11}$$

Such a loss on the normals requires that after moving the point cloud P_0 by our velocity field $\mathcal{V}$, the normals align with those of point cloud P_1 at time T, *i.e.*, for $\mathbf{x} \in \Omega_{0*}$

$$\mathcal{L}_n = \left\| \int_0^T \mathbf{F}(\mathbf{x}, t)^{-\top} \mathbf{n}(\mathbf{x}) \mathrm{d}t - \mathbf{n}(\mathbf{x}_1) \right\| \, , \tag{12}$$

where $\Omega_0 * \subset \Omega$ is the points where the correspondences are known.

Stretching loss. After we establish the normal update equation Eq. (11), we follow the idea presented in [41] to compute a loss to penalize the stretching. Different from them, as they obtain the normals from the implicit surface representation function, our normals are passed through the deformation gradient operator $\mathbf{F}$, which only works with our normal update term Eq. (11). We define the tangent projection operator as $\mathbf{P} = \mathbf{I} - \mathbf{n}^\top \mathbf{n}$, and

$$\mathcal{L}_s = \int_{\Omega_0} \left\| \mathbf{P}^\top (\mathbf{F}^\top \mathbf{F} - \mathbf{I}) \mathbf{P} \right\|_F \mathrm{d}\mathbf{x} \, , \tag{13}$$

where $\|\cdot\|_F$ is the Frobenius norm [12] of a matrix.

Matching loss. Finally, we integrate into the whole point cloud chamfer loss Eq. (8) the set of sparse correspondences obtained by the vertex registration block. We use them to supervise the velocity $\mathcal{V}$ as well, and to provide semantic information next to the geometrical one. The matching loss is defined as:

$$\mathcal{L}_m = \left\| \mathbf{x}_0 + \int_0^T \mathcal{V}(\mathbf{x}, t) \mathrm{d}t - \mathbf{x}_1 \right\|^2 \, . \tag{14}$$

3.4. Training

Training. Given a pair of keyframe images I_0, I_1, we first generate 3D meshes S_0, S_1 for each of them and the per-vertex features. Then we compute the pairwise correspondences. Then we sample $k = 20,000$ points from meshes together with estimated correspondences, obtaining two point clouds. We refer to the ordered set of the two as $\{P_0, P_1\}$. The total loss then is:

$$\begin{aligned} \mathcal{L} = & \lambda_v \mathcal{L}_v + \lambda_{div} \mathcal{L}_{div} + \lambda_o \mathcal{L}_o + \lambda_n \mathcal{L}_n \\ & + \lambda_s \mathcal{L}_s + \lambda_m \mathcal{L}_m + \lambda_d \mathcal{L}_d \, , \end{aligned} \tag{15}$$

where λ_v, λ_{div} λ_o, λ_n, λ_s, λ_m, and λ_d are hyperparameters, serve as weights for each loss term.

Extended to multi-pairs. If for a sequence more than two keyframes are available, TwoSquared naturally extends, with almost no modification to the procedure described above. An example is shown in 1, while we point to supplementary material for the minor details.

Inference. At inference time, we can input S_0 vertices into our velocity neural network to evolve the shape in a continuous 4D mesh sequence. We highlight that such a continuous velocity field provides tracking of points along time, and so a 4D continuous dense correspondence for all the intermediate shapes. Also, we are not bound by any time discretization, and the deformation can be produced at an arbitrary framerate. Finally, by deforming the first shape along the whole sequence, we are able to automatically provide geometry and structure consistency.

4. Experiments

4.1. Setting

Network architecture. We use Hunyuan3D-2 [50] as the 3D generation backbone to obtain meshes of the keyframes. Then, we pass the generated mesh pair to Diff3F [11] to get per-vertex features. Our Velocity-Net contains only an 8-layer MLP with 256 nodes, enabling fast training and near real-time inference. Our method is implemented using purely JAX [3].

Comparison baseline. As we are the first method to address 4D generation for pairs of images, we propose a set of competitive baselines in two different categories. First, we consider the deformation as 2D image morphing. *DiffMorpher* [64] and *DreamMover* [43] are state-of-the-art methods which take a source image I_0 and a target image I_1 and morph them to generate an intermediate image sequence. Both methods leverage pre-trained text-to-image diffusion models (Stable Diffusion [39]) and train LoRAs [14] to fine-tune. We treat the generated image sequence as a time-dependent 2D sequence, and we plug the intermediate frames into Hunyuan3D [50], obtaining a 4D sequence. Second, we compare with the newest video-to-4D reconstruction method *V2M4* [6]. It is worth mentioning that V2M4 assumes a static monocular camera sequence as input, a much more demanding input than ours.

Validation datasets. We perform a quantitative evaluation on the 4D-DRESS [53] dataset, composed of real-scanned human motions equipped with a high frame rate RGB image sequences and ground truth textured 3D mesh sequences. We select two sequences and we pick every 5th image as an input image to generate the deformed 4D sequences and compare them against the ground truth intermediate shapes. For qualitative comparison, we use a variety of image sources, such as video keyframes in *Consistent4D*[17], the dataset used by V2M4 [6], daily photographs, and images collected from the web, demonstrating the robustness

of our method.

Training and Inferencing Time. To train our network on an image pair, we require around 15 minutes, which mainly involves running Hunyuan3D [50] inference (3 minutes per shape), computing per-vertex features (5 minutes per shape), and the vertex registration block and then the deformation optimizing block (both less than 1 minute). Our training is performed on point clouds sampled from low-resolution meshes, which is efficient, especially for recovering the correspondence, while at inference time it operates at arbitrary resolution without additional overhead. In contrast, image-morphing-based methods have a generation time that increases linearly with the desired frame rate. For instance, using *DiffMorpher* [64] plus Hunyuan3D [50] to obtain a deformed 4D sequence with 30 frames will need around 60 minutes only for generating 4D meshes. Additionally, a change in the frame rate is possible only by retraining the whole method. The video-to-4D reconstruction method V2M4 [6] is computationally more expensive; it needs a high-end NVIDIA GPU with at least 40 GB of memory, and it takes approximately 50 minutes for a video with 32 frames. Finally, V2M4 generates only one mesh per input frame, and hence, when more frames are needed, it linearly interpolates the generated meshes.

Metrics. To quantitatively evaluate the generated sequence quality and physical plausibility, we compute the Chamfer Distance (CD), and Hausdorff Distance (HD) of deformed mesh sequences, and we also report the surface area standard deviation SAσ across the sequence to evaluate the distortion of generated meshes [53].

4.2. Validation

In this section, we evaluate our method on multiple datasets to demonstrate its effectiveness and compare it against baseline approaches. We structure the evaluation into three parts: quantitative analysis, qualitative analysis, and ablation studies for each proposed loss.

Quantitative Comparison. To quantitatively evaluate our method, we extract five keyframes from two sequences in the 4D-DRESS [53] dataset and generate four 4D deformation sequences using different methods. We then compute errors using the ground truth intermediate meshes. As shown in Tab. 1, our method achieves lower error rates compared to other approaches. To compare with V2M4 [6], we provide it with *all the ground truth frames* as a video input, which is the setting in which they achieve the best results. In contrast, our method uses much less data and still achieves better geometric results, without relying on ground truth information. As shown in Fig. 3, most methods can generate high-quality starting and ending meshes. However, methods that require per-step mesh generation often suffer from texture inconsistencies, suggesting that relying on 3D generative AI for all the frames accumulates inconsistencies. We also remark that previous approaches have computational limitations, as adjusting the 4D sequence frame rate requires regenerating morphing images and regenerating meshes for each frame, making the process inefficient. V2M4 [6] often introduces large global rotations when generating meshes across frames. We suspect this occurs because, in its attempt to enforce global consistency, the model includes a block for estimating camera extrinsics, which mistakenly interprets human motion as pure camera transformation.

Seq.	Method	CD ($\times 10^3$) ↓	HD ($\times 10^2$) ↓	SAσ($\times 10$) ↓
Take19	GT image	1.503	1.998	0.201
	DiffMorpher [64]	1.678	2.029	1.161
	DreamMover [43]	1.683	2.033	1.116
	V2M4 [6]	3.300	5.489	**0.114**
	Ours	**1.451**	**1.996**	0.201
Take7	GT image	0.099	0.145	1.079
	DiffMorpher [64]	0.110	0.710	1.228
	DreamMover [43]	0.126	0.186	1.359
	V2M4 [6]	12.390	13.961	0.260
	Ours	**0.074**	**0.117**	**0.139**

Table 1. **Quantitative comparison:** Our method achieves the best quantitative results compared to competitors, even when they rely on more demanding input (video) and ground truth information.

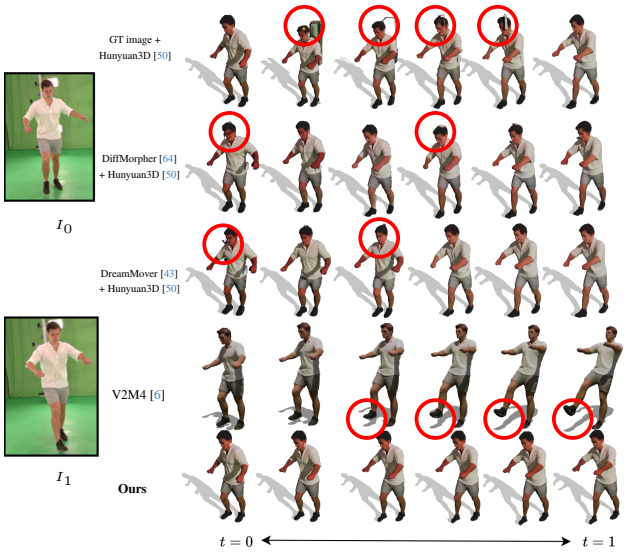

Figure 3. **Comparison with other methods:** TwoSquared generates texture-consistent, physically plausible 4D sequences, and it is more robust than 4Deform [41] to correspondence noise. In contrast, other methods show artifacts in the intermediate shapes.

Qualitative comparison with video-based methods. To further stress the robustness of our method, we show qualitative comparisons with V2M4 [6] on long sequences, providing them with the same or more information than the one used by us. In Fig. 4 we give only 7 key frames to both methods. Our method outputs a physically plausible deformation and structure, while V2M4 [6] creates strong

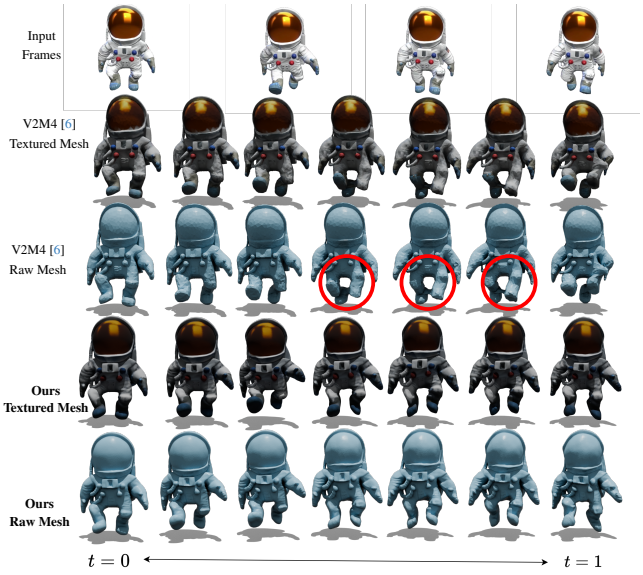

Figure 4. **Multi-pairs example:** We present the textured meshes (second and fourth rows) alongside their corresponding untextured meshes (third and fifth rows) and compare them with V2M4 [6]. Our method produces both physically realistic deformations and consistently maintains high-fidelity meshes across all frames.

artifacts between movements. Despite the assumption that adjacent video frames are similar, V2M4 still requires such a dense supervision signal. However, even with that, severe artifacts can arise. In Fig. 5 V2M4 [6] takes the whole set of ground truth frames as input, while our method takes only 7 keyframes. As shown, our method produces deformations that remain physically plausible and texture-consistent across all frames, while also preserving finer texture and geometric details. Due to space constraints, for both cases we report only a subset of the output here, while we include the complete ones in the supplementary material.

Ablations. To demonstrate the contribution of the physical constraints, we ablate the proposed losses on *4D-DRESS*. We report quantitative analysis in Tab. 2 and a qualitative support in Fig. 6. The overlapping loss $\mathcal{L}_o$ enforces global alignment between the deformed point cloud and the target shape, making it beneficial in cases where correspondences are sparse or entirely absent in local regions. The normal loss $\mathcal{L}_n$ ensures that the surface normals of the deformed point cloud align with those of the target. It can be observed that the impact of $\mathcal{L}_n$ appears marginal in certain cases, which we attribute to suboptimal correspondence quality, since normal alignment can be enforced only where correspondences exist. The stretching loss $\mathcal{L}_s$ generally enhances the overall quality of the deformation results. In most real-world scenarios, correspondences are neither uniformly distributed nor perfectly accurate. Compared to synthetic data, estimated correspondences in real-world cases tend to exhibit significantly lower quality, often leading to missing correspondences in whole regions.

Figure 5. **Multi-pairs example:** We present the textured meshes (second and fourth rows) alongside the corresponding untextured meshes (third and fifth rows), comparing them with V2M4 [6]. Our method achieves comparable results using only sparse keyframes. Our approach generates high-resolution meshes that capture fine details in texture and surface geometry.

While the spatial continuity of the velocity field is enforced through Eq. (5), this constraint alone is sometimes insufficient to prevent local distortions. The stretching loss $\mathcal{L}_s$ complements the spatial smoothness constraint by penalizing excessive local shear and stretching. As demonstrated in Tab. 2, omitting $\mathcal{L}_s$ frequently results in substantial surface area deviation (SAσ). This effect is further corroborated by the visualizations in Fig. 6, where the absence of stretching loss leads to unrealistic elongation of the leg region. In summary, both quantitative and qualitative ablation studies confirm that each loss term contributes.

4.3. Applications

We aim to demonstrate the generality of TwoSquared, which makes it suitable to work on in-the-wild images such as frames from the web. Such robustness also enables new applications, such as editing from hand sketches or pose transfer from different subjects.

Motion transfer for web image pairs. Our method enables the creation of dynamic 4D sequences from web images, generating smooth and realistic animations that adapt to various styles and contexts. For example, the two input images do not need to depict the exact same object. Hence, we can use our method to perform pose transfer by deform-

Seq.	Method	CD ($\times 10^2$) ↓	HD ($\times 10$) ↓	SAσ($\times 10$) ↓
Take19	w/o $\mathcal{L}_o$	1.462	1.998	**0.201**
	w/o $\mathcal{L}_n$	1.539	**1.965**	0.202
	w/o $\mathcal{L}_s$	1.458	2.004	0.223
	Ours	**1.451**	1.996	**0.201**
Take7	w/o $\mathcal{L}_o$	0.075	0.119	0.140
	w/o $\mathcal{L}_n$	**0.074**	0.119	**0.137**
	w/o $\mathcal{L}_s$	0.081	0.126	0.486
	Ours	**0.074**	**0.117**	0.139

Table 2. **Quantitative ablation:** We show the effect of every loss by isolating its contribution. Stratching loss provides help in limiting area distortion, while the loss on normals provides a less evident impact, being bound to the correspondence quality.

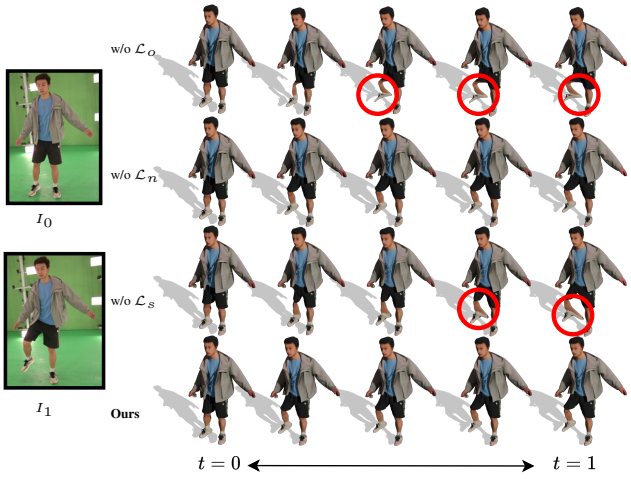

Figure 6. **Visualization of ablations:** The stretching loss and overlapping loss help the shape to remain physically plausible. While the normal loss has little qualitative impact, it does lead to quantitative improvements, as reported in Table 2.

ing the subject of the initial image I_0 (preserving its identity) into the pose of the ending image I_1. Fig. 7 shows an example of a deformation between two horse images. Our method creates temporally consistent 4D sequences, while DiffMorpher [64] fails to interpolate this image pair and so the 3D generation task. DreamMover [43] successfully deformed the image from I_0 to I_1. However, since the following 3D generations are based on the deformed 2D images, it results in texture blending, losing subject identity.

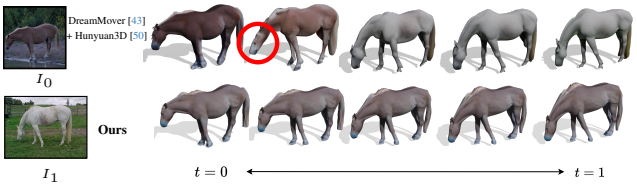

Figure 7. **4D reconstruction from web images:** Our method can take a pair of images as input and generate the temporally consistent 4D deformation between these two objects.

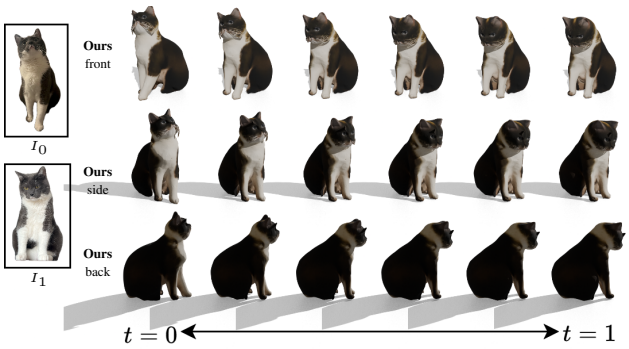

Figure 8. **Daily photo example:** Our method can directly take images that are taken in daily life and generate continuous 4D sequences for different objects, animals, and humans. In the figure, we demonstrate the different angles of the 4D mesh sequences to show that our method generates high-quality meshes for each step.

4D reconstruction from photos. We also show in Fig. 8 how our approach can directly process real-world images, which often contain inconsistencies such as variations in brightness. These inconsistencies lead to changes in subject's appearance across multiple photos. Image-morphing-based 4D generation methods struggle in such cases, often failing outright or transferring such inconsistencies onto the generated meshes. Another challenge with real-world objects is their structural complexity. For example, fine details such as a cat's whiskers are difficult to preserve using image-morphing methods (see the first red circle), which often fail to reconstruct such intricate features (see supplementary material for a comparison). In contrast, our method effectively addresses these challenges, producing 4D sequences that maintain both texture consistency and geometric integrity. To further demonstrate the quality of our results, Fig. 8 presents the generated meshes from multiple angles, highlighting the high fidelity of our reconstructions.

5. Conclusion & Future work

We presented TwoSquared as the first approach that generates a complete 4D sequence of an arbitrary object from just a pair of images, by combining the latest advances in 3D reconstruction with physically plausible modeling of the deformation. While we show promising results on this new challenge, there remain significant open problems. We address a broad class of objects (non-rigid deformation of articulated shapes), but whether our deformations model would scale to more intricate scenarios where thousands of separate deformations happen simultaneously, e.g., hair, is yet to be explored. Such challenges would require highly precise tracking, which is not available at present, and would be an exciting future direction. Ultimately, TwoSquared opens up the interesting new challenge of 4D reconstruction from minimal input, serving as a fundamental step to 4D AI generations.

Acknowledgements

This work was supported by the ERC Advanced Grant "SIMULACRON" (agreement No. 884679), the ERC starting grant No. 101160648 (Harmony), and the DFG project CR 250/26-1 "4DYoutube".

References

[1] *Virtual Reality*. MIT Press, 2019. ISBN: 9780262354684. 1

[2] Hritik Bansal, Zongyu Lin, Tianyi Xie, Zeshun Zong, Michal Yarom, Yonatan Bitton, Chenfanfu Jiang, Yizhou Sun, Kai-Wei Chang, and Aditya Grover. Videophy: Evaluating physical commonsense for video generation. *arXiv preprint arXiv:2406.03520*, 2024. 2

[3] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. 5

[4] Yann Brenier. Optimal transportation of particles, fluids and currents. In *Variational methods for evolving objects*, pages 59–86. Mathematical Society of Japan, 2015. 4

[5] Wei Cao, Chang Luo, Biao Zhang, Matthias Nießner, and Jiapeng Tang. Motion2vecsets: 4d latent vector set diffusion for non-rigid shape reconstruction and tracking, 2024. 2

[6] Jianqi Chen, Biao Zhang, Xiangjun Tang, and Peter Wonka. V2m4: 4d mesh animation reconstruction from a single monocular video. In *ICCV*, 2025. 2, 5, 6, 7, 1, 3

[7] Yongfan Chen, Xiuwen Zhu, Tianyu Li, Hao Chen, and Chunhua Shen. A physical coherence benchmark for evaluating video generation models via optical flow-guided frame prediction. *arXiv preprint arXiv:2502.05503*, 2025. 2

[8] C Curreli, D Muhle, A Saroha, Z Ye, R Marin, and D Cremers. Nonisotropic gaussian diffusion for realistic 3d human motion prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025. 1, 2

[9] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13142–13153, 2023. 2

[10] 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. *Advances in Neural Information Processing Systems*, 36, 2024. 2

[11] Niladri Shekhar Dutt, Sanjeev Muralikrishnan, and Niloy J. Mitra. Diffusion 3d features (diff3f): Decorating untextured shapes with distilled semantic features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4494–4504, 2024. 3, 5, 1

[12] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, MD, USA, 3rd edition, 1996. 5

[13] Jiatao Gu, Alex Trevithick, Kai-En Lin, Joshua M Susskind, Christian Theobalt, Lingjie Liu, and Ravi Ramamoorthi. Nerfdiff: Single-image view synthesis with nerf-guided distillation from 3d-aware diffusion. In *International Conference on Machine Learning*, pages 11808–11826. PMLR, 2023. 2

[14] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. Lora: Low-rank adaptation of large language models. *ICLR*, 1(2):3, 2022. 5

[15] Li Hu, Xin Gao, Peng Zhang, Ke Sun, Bang Zhang, and Liefeng Bo. Animate anyone: Consistent and controllable image-to-video synthesis for character animation. *arXiv preprint arXiv:2311.17117*, 2023. 1

[16] Fridtjov Irgens. *Continuum Mechanics in Curvilinear Coordinates*, pages 599–624. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008. 5

[17] Yanqin Jiang, Li Zhang, Jin Gao, Weiming Hu, and Yao Yao. Consistent4d: Consistent 360° dynamic object generation from monocular video. In *The Twelfth International Conference on Learning Representations*, 2024. 5, 1

[18] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 2

[19] Ci Li, Yi Yang, Zehang Weng, Elin Hernlund, Silvia Zuffi, and Hedvig Kjellström. Dessie: Disentanglement for articulated 3d horse shape and pose estimation from images. In *Asian Conference on Computer Vision*, 2024. 1, 2

[20] Zhiqi Li, Yiming Chen, and Peidong Liu. Dreammesh4d: Video-to-4d generation with sparse-controlled gaussian-mesh hybrid representation. *Advances in Neural Information Processing Systems*, 37:21377–21400, 2025. 2

[21] Hanwen Liang, Yuyang Yin, Dejia Xu, Hanxue Liang, Zhangyang Wang, Konstantinos N. Plataniotis, Yao Zhao, and Yunchao Wei. Diffusion4d: Fast spatial-temporal consistent 4d generation via video diffusion models, 2024. 2

[22] Jiajing Lin, Zhenzhong Wang, Shu Jiang, Yongjie Hou, and Min Jiang. Phys4dgen: A physics-driven framework for controllable and efficient 4d content generation from a single image. *arXiv preprint arXiv:2411.16800*, 2024. 2

[23] Minghua Liu, Chao Xu, Haian Jin, Linghao Chen, Mukund Varma T, Zexiang Xu, and Hao Su. One-2-3-45: Any single image to 3d mesh in 45 seconds without per-shape optimization. *Advances in Neural Information Processing Systems*, 36, 2024. 2

[24] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9298–9309, 2023. 2

[25] Yuan Liu, Cheng Lin, Zijiao Zeng, Xiaoxiao Long, Lingjie Liu, Taku Komura, and Wenping Wang. Syncdreamer: Generating multiview-consistent images from a single-view image. *arXiv preprint arXiv:2309.03453*, 2023.

[26] Xiaoxiao Long, Yuan-Chen Guo, Cheng Lin, Yuan Liu, Zhiyang Dou, Lingjie Liu, Yuexin Ma, Song-Hai Zhang,

Marc Habermann, Christian Theobalt, et al. Wonder3d: Single image to 3d using cross-domain diffusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9970–9980, 2024. 2

[27] Zhang Longwen, Wang Ziyu, Zhang Qixuan, Qiu Qiwei, Pang Anqi, Jiang Haoran, Yang Wei, Xu Lan, and Yu Jingyi. Clay: A controllable large-scale generative model for creating high-quality 3d assets. *arXiv preprint arXiv:2406.13897*, 2024. 2

[28] Robin Magnet, Jing Ren, Olga Sorkine-Hornung, and Maks Ovsjanikov. Smooth non-rigid shape matching via effective dirichlet energy optimization. In *2022 International Conference on 3D Vision (3DV)*, pages 495–504. IEEE, 2022. 3

[29] Luke Melas-Kyriazi, Iro Laina, Christian Rupprecht, and Andrea Vedaldi. Realfusion: 360deg reconstruction of any object from a single image. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8446–8455, 2023. 2

[30] 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. *Communications of the ACM*, 65(1):99–106, 2021. 2

[31] Martin Ralf Oswald, Jan Stühmer, and Daniel Cremers. Generalized connectivity constraints for spatio-temporal 3d reconstruction. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part IV 13*, pages 32–46. Springer, 2014. 1

[32] Maks Ovsjanikov, Mirela Ben-Chen, Justin Solomon, Adrian Butscher, and Leonidas Guibas. Functional maps: a flexible representation of maps between shapes. *ACM Transactions on Graphics (TOG)*, 31(4):30:1–30:11, 2012. 3

[33] Ilya A Petrov, Vladimir Guzov, Riccardo Marin, Emre Aksan, Xu Chen, Daniel Cremers, Thabo Beeler, and Gerard Pons-Moll. Echo: Ego-centric modeling of human-object interactions. *arXiv preprint arXiv:2508.21556*, 2025. 2

[34] Ilya A Petrov, Riccardo Marin, Julian Chibane, and Gerard Pons-Moll. Tridi: Trilateral diffusion of 3d humans, objects, and interactions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5523–5535, 2025. 2

[35] 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 preprint arXiv:2307.01952*, 2023. 2

[36] Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022. 2

[37] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10318–10327, 2021. 1

[38] Guocheng Qian, Jinjie Mai, Abdullah Hamdi, Jian Ren, Aliaksandr Siarohin, Bing Li, Hsin-Ying Lee, Ivan Skorokhodov, Peter Wonka, Sergey Tulyakov, et al. Magic123: One image to high-quality 3d object generation using both 2d and 3d diffusion priors. *arXiv preprint arXiv:2306.17843*, 2023. 2

[39] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 2, 5

[40] Lu Sang, Zehranaz Canfes, Dongliang Cao, Florian Bernard, and Daniel Cremers. Implicit neural surface deformation with explicit velocity fields. In *ICLR*, 2025. 4

[41] L Sang, Z Canfes, D Cao, R Marin, F Bernard, and D Cremers. 4deform: Neural surface deformation for robust shape interpolation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025. 2, 4, 5, 6

[42] Dieter Schmalstieg and Tobias Hollerer. *Augmented Reality: Principles and Practice*. Addison-Wesley Professional, 2016. 1

[43] Liao Shen, Tianqi Liu, Huiqiang Sun, Xinyi Ye, Baopu Li, Jianming Zhang, and Zhiguo Cao. Dreammover: Leveraging the prior of diffusion models for image interpolation with large motion. *arXiv preprint arXiv:2409.09605*, 2024. 5, 6, 8, 1, 3, 4

[44] Qiuhong Shen, Xingyi Yang, and Xinchao Wang. Anything-3d: Towards single-view anything reconstruction in the wild. *arXiv preprint arXiv:2304.10261*, 2023. 2

[45] Ruoxi Shi, Hansheng Chen, Zhuoyang Zhang, Minghua Liu, Chao Xu, Xinyue Wei, Linghao Chen, Chong Zeng, and Hao Su. Zero123++: a single image to consistent multi-view diffusion base model. *arXiv preprint arXiv:2310.15110*, 2023. 2

[46] Olga Sorkine and Marc Alexa. As-rigid-as-possible surface modeling. In *Symposium on Geometry processing*, pages 109–116. Citeseer, 2007. 4

[47] Wenqiang Sun, Shuo Chen, Fangfu Liu, Zilong Chen, Yueqi Duan, Jun Zhang, and Yikai Wang. Dimensionx: Create any 3d and 4d scenes from a single image with controllable video diffusion. *arXiv preprint arXiv:2411.04928*, 2024. 2

[48] Stanislaw Szymanowicz, Chrisitian Rupprecht, and Andrea Vedaldi. Splatter image: Ultra-fast single-view 3d reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10208–10217, 2024. 2

[49] Junshu Tang, Tengfei Wang, Bo Zhang, Ting Zhang, Ran Yi, Lizhuang Ma, and Dong Chen. Make-it-3d: High-fidelity 3d creation from a single image with diffusion prior. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 22819–22829, 2023. 2

[50] Tencent Hunyuan3D Team. Hunyuan3d 2.0: Scaling diffusion models for high resolution textured 3d assets generation, 2025. 2, 3, 5, 6, 8, 1

[51] Tuan-Anh Vu, Duc Thanh Nguyen, Binh-Son Hua, Quang-Hieu Pham, and Sai-Kit Yeung. Rfnet-4d++: Joint object reconstruction and flow estimation from 4d point clouds with cross-attention spatio-temporal features. 2024. 2

[52] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt:

Visual geometry grounded transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025. 1

[53] Wenbo Wang, Hsuan-I Ho, Chen Guo, Boxiang Rong, Artur Grigorev, Jie Song, Juan Jose Zarate, and Otmar Hilliges. 4d-dress: A 4d dataset of real-world human clothing with semantic annotations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 5, 6, 1

[54] Felix Wimbauer, Nan Yang, Christian Rupprecht, and Daniel Cremers. Behind the scenes: Density fields for single view reconstruction. *arXiv preprint arXiv:2301.07668*, 2023. 1

[55] Kailu Wu, Fangfu Liu, Zhihan Cai, Runjie Yan, Hanyang Wang, Yating Hu, Yueqi Duan, and Kaisheng Ma. Unique3d: High-quality and efficient 3d mesh generation from a single image. *arXiv preprint arXiv:2405.20343*, 2024. 2

[56] Dejia Xu, Yifan Jiang, Peihao Wang, Zhiwen Fan, Yi Wang, and Zhangyang Wang. Neurallift-360: Lifting an in-the-wild 2d photo to a 3d object with 360deg views. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4479–4489, 2023. 2

[57] Jiale Xu, Weihao Cheng, Yiming Gao, Xintao Wang, Shenghua Gao, and Ying Shan. Instantmesh: Efficient 3d mesh generation from a single image with sparse-view large reconstruction models. *arXiv preprint arXiv:2404.07191*, 2024. 2

[58] Yuxuan Xue, Xianghui Xie, Riccardo Marin, and Gerard Pons-Moll. Human 3diffusion: Realistic avatar creation via explicit 3d consistent diffusion models. *Arxiv*, 2024. 2

[59] Yuxuan Xue, Xianghui Xie, Riccardo Marin, and Gerard Pons-Moll. Gen-3diffusion: Realistic image-to-3d generation via 2d & 3d diffusion synergy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–17, 2025. 2

[60] Zhiwen Yan, Chen Li, and Gim Hee Lee. Nerf-ds: Neural radiance fields for dynamic specular objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8285–8295, 2023. 1

[61] Biao Zhang, Jiapeng Tang, Matthias Niessner, and Peter Wonka. 3dshape2vecset: A 3d shape representation for neural fields and generative diffusion models. *ACM Transactions on Graphics (TOG)*, 42(4):1–16, 2023. 2

[62] Haiyu Zhang, Xinyuan Chen, Yaohui Wang, Xihui Liu, Yunhong Wang, and Yu Qiao. 4diffusion: Multi-view video diffusion model for 4d generation. *arXiv preprint arXiv:2405.20674*, 2024. 2

[63] Haiyu Zhang, Xinyuan Chen, Yaohui Wang, Xihui Liu, Yunhong Wang, and Yu Qiao. 4diffusion: Multi-view video diffusion model for 4d generation. *Advances in Neural Information Processing Systems*, 37:15272–15295, 2025. 2

[64] Kaiwen Zhang, Yifan Zhou, Xudong Xu, Xingang Pan, and Bo Dai. Diffmorpher: Unleashing the capability of diffusion models for image morphing. *arXiv preprint arXiv:2312.07409*, 2023. 5, 6, 8, 1, 3, 4

[65] Zechuan Zhang, Zongxin Yang, and Yi Yang. Sifu: Side-view conditioned implicit function for real-world usable clothed human reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9936–9947, 2024. 1

[66] Zibo Zhao, Wen Liu, Xin Chen, Xianfang Zeng, Rui Wang, Pei Cheng, Bin Fu, Tao Chen, Gang Yu, and Shenghua Gao. Michelangelo: Conditional 3d shape generation based on shape-image-text aligned latent representation. *Advances in Neural Information Processing Systems*, 36, 2024. 2

[67] Hanxin Zhu, Tianyu He, Xiqian Yu, Junliang Guo, Zhibo Chen, and Jiang Bian. Ar4d: Autoregressive 4d generation from monocular videos. *arXiv preprint arXiv:2501.01722*, 2025. 1, 2

[68] Zi-Xin Zou, Zhipeng Yu, Yuan-Chen Guo, Yangguang Li, Ding Liang, Yan-Pei Cao, and Song-Hai Zhang. Triplane meets gaussian splatting: Fast and generalizable single-view 3d reconstruction with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10324–10335, 2024. 2

[69] Silvia Zuffi, Ylva Mellbin, Ci Li, Markus Hoeschle, Hedvig Kjellström, Senya Polikovsky, Elin Hernlund, and Michael J. Black. VAREN: Very accurate and realistic equine network. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. 1, 2