DeformGS: Scene Flow in Highly Deformable Scenes for Deformable Object Manipulation

Bardienus P. Duisterhof[1][0000−0002−2797−4898], Mandi Zhao[2][0009−0005−5877−920X], Yunchao Yao[1][0009−0008−6537−7075], Jia-Wei Liu[4][0000−0002−9766−2419], Jenny Seidenschwarz[1,5][0000−0002−8955−0767], Mike Zheng Shou[4][0000−0002−7681−2166], Deva Ramanan[1][0009−0008−9180−8983], Shuran Song[2][0000−0002−8768−7356], Stan Birchfield[3][1111−2222−3333−4444], Bowen Wen[3][0000−0002−9207−6103], and Jeffrey Ichnowski[1][0000−0003−4874−9478]

[1] Carnegie Mellon University, The Robotics Institute {bduister,jeffi}@cmu.edu
[2] Stanford University
[3] NVIDIA
[4] National University of Singapore
[5] Technical University of Munich

Abstract. Teaching robots to fold, drape, or reposition deformable objects such as cloth will unlock a variety of automation applications. While remarkable progress has been made for rigid object manipulation, manipulating deformable objects poses unique challenges, including frequent occlusions, infinite-dimensional state spaces and complex dynamics. Just as object pose estimation and tracking have aided robots for rigid manipulation, dense 3D tracking (scene flow) of highly deformable objects will enable new applications in robotics while aiding existing approaches, such as imitation learning or creating digital twins with real2sim transfer. We propose DeformGS, an approach to recover scene flow in highly deformable scenes, using simultaneous video captures of a dynamic scene from multiple cameras. DeformGS builds on recent advances in Gaussian splatting, a method that learns the properties of a large number of Gaussians for state-of-the-art and fast novel-view synthesis. DeformGS learns a deformation function to project a set of Gaussians with canonical properties into world space. The deformation function uses a neural-voxel encoding and a multilayer perceptron (MLP) to infer Gaussian position, rotation, and a shadow scalar. We enforce physics-inspired regularization terms based on conservation of momentum and isometry, which leads to trajectories with smaller trajectory errors. We also leverage existing foundation models SAM and XMEM to produce noisy masks, and learn a per-Gaussian mask for better physics-inspired regularization. DeformGS achieves high-quality 3D tracking on highly deformable scenes with shadows and occlusions. In experiments, DeformGS improves 3D tracking by an average of 55.8 % compared to the state-of-the-art. With sufficient texture, DeformGS achieves a median tracking error of 3.3 mm on a cloth of 1.5 × 1.5 m in area. Website: https://deformgs.github.io

Keywords: Perception, Machine Learning in Robotics , Manipulation & Grasping

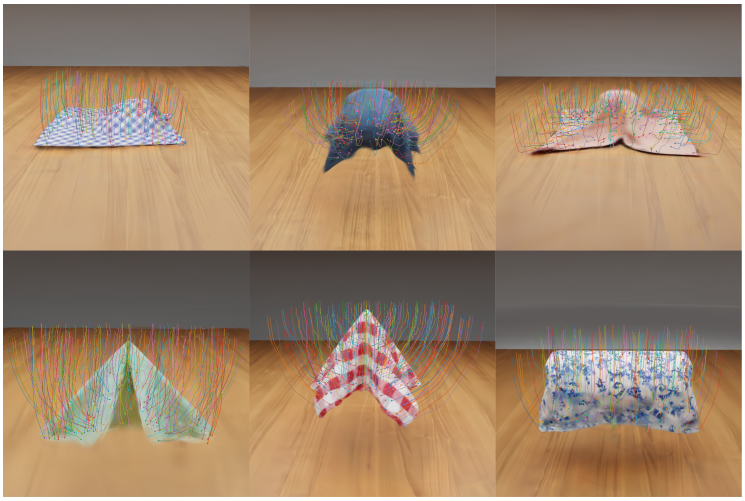


Fig. 1: We propose DeformGS, a method that improves state-of-the-art methods for accurate 3D point tracking in highly deformable scenes. This figure shows the rendering and tracking of DeformGS in the six dynamic Blender [9] scenes used for evaluation. We will refer to the scenes in this Figure as Scenes 1, 2, 3, 4, 5 and 6 ordered from left to right.

1 Introduction

Recent advances in robot learning have demonstrated impressive performance on challenging tasks, including rigid and deformable object manipulation. Scaling these approaches to deployment will require an improvement in robustness and learning from few demonstrations. A promising avenue for improving in robot learning performance are intermediate representations and foundation models, including 6D object pose estimation [10, 11, 27, 42, 50, 51, 55], semantic latent features [35], and 2D pixel-wise tracking [21, 52]. However, perception and representations that will lead to robust manipulation of deformable objects remains an open challenge, due to self-occlusions, shadows, and varying (or lack of) textures.

Three-dimensional dense point tracking, or *3D scene flow*, is a useful representation for robot manipulation, as it provides flexibility to represent high-dimensional dynamic state changes, while the deformable objects drops, deforms, and drapes during manipulation. In particular, dense 3D scene flow can be an input to imitation learning policies [3, 52], can be used to learn a transition model [40], to identify and track task-relevant key points, or to create a digital twin through real2sim transfer. Recent work in monocular tracking has seen improvements in performance on datasets such as TAP-Vid [13], but it remains unclear how to effectively lift from 2D tracking to 3D for robotic spatial understanding in challenging highly deformable scenes.

To overcome these limitations, Gaussian Splatting provides a promising avenue. Recent work demonstrated Gaussian Splatting [22, 23] can yield state-of-the-art novel-view synthesis and rendering speeds exceeding 100 fps. Concretely, 3D Gaussian Splatting uses a fast differentiable renderer to fit the colors, positions, and covariances of a set of Gaussians. An extension of 3D Gaussian splatting [29] showed dynamic scenes can be modeled by explicitly optimizing the properties of Gaussians over time, resulting in novel-view synthesis and scene flow.

Explicitly optimizing the Gaussian pose as in Dynamic 3D Gaussians [29] may result in degraded performance with large deformations and shadows. The Gaussian properties may converge to local optima, especially in scenes with large deformations, strong shadows, and occlusions.

We propose DeformGS, a method that uses time-synchronized image frames from a calibrated multi-camera setup to track 3D geometries of deformable objects as they move through shadows and occlusions. DeformGS learns the canonical state of a set of Gaussians and a deformation function that maps the Gaussians into world space. This enables tracking by recovering scene flow, and novel-view rendering (through splatting) using a fast differentiable rasterizer.

We evaluate DeformGS in six photo-realistic synthetic scenes of varying difficulty. The scenes contain large deformations, shadows, and occlusions (Figure 1 shows the scenes and tracking trajectories computed by DeformGS). Empirical results show that DeformGS infers 55.8 % more accurate 3D tracking results compared to previous state-of-the-art [29, 53]. In a scene with a 1.5 m $\times$ 1.5 m cloth (i.e., Scene 1), DeformGS can track cloth deformation with as low as 3.3 mm median tracking error.

We also evaluate DeformGS in the real world on the Robo360 [26] dataset. We show qualitative results for tracking rigid and deformable objects in cluttered scenes, and study two robotics applications: (1) real2sim transfer to create a digital twin, and (2) tracking task-relevant keypoints for downstream grasping applications.

In summary, our contributions are as follows:

- We provide the first approach designed to accurately perform 3D dense tracking for deformable objects using 4D Gaussians.
- We provide experiments that suggest state-of-the-art performance in simultaneous 3D metric tracking and novel view synthesis. DeformGS improves tracking accuracy by an average of 55.8% in synthetic experiments and demonstrates robust 3D tracking in the real world for deformable objects. The latter can be exploited as a representation for imitation learning and represents a new method for building digital twins.
- A set of six synthetic scenes with large deformations, strong shadows, and occlusions. We will open-source the scenes and as well as the source code.

2 Related Work

2.1 Neural Rendering for Novel View Synthesis

DeformGS builds on prior work in novel-view synthesis, and uses photometric consistency as a signal to achieve 3D tracking. A popular novel view synthesis approach is NeRF [30]. It uses neural networks to learn scene representations that are capable of photo-realistic novel view reconstruction. Particle-based methods use a more explicit representation than typical NeRF-based approaches. DeformGS builds on 3D Gaussian Splatting [22, 23] which belongs to the latter caregory. [22] proposed a differential rasterizer to render a large number of Gaussian 'splats,' each with their state including color, position, and covariance matrix. Contrary to the NeRF-based approaches, Gaussian splatting achieves real-time rendering of novel views with state-of-the-art performance.

2.2 Dynamic Novel View Synthesis

The assumption of static scenes in neural rendering approaches prevents application to real-world scenarios with moving objects or humans, such as the dynamic and deformable scenes in this work. One line of work to address this assumption is adding a time dimension to NeRF modeling [15, 18, 25, 54]. Prior works either condition the neural field on explicit time input or a time embedding. Another line of work learns a deformation field to map 4D points into a canonical space [36, 37], i.e., every 4D point in space and time maps to a 3D point in a canonical NeRF. DeVRF [28] proposed to model the 3D canonical space and 4D deformation field of a dynamic, non-rigid scene with explicit and discrete voxel-based representations.

Several recent works extend the above approaches to 3D Gaussian splatting. Dynamic 3D Gaussians [29] explicitly model the position and covariance matrix of each Gaussian at each time step. This method struggles in dynamic scenes with large deformations, strong shadows, or occlusions. We build on another recent work, 4D Gaussian splatting [53], which uses feature encoding techniques proposed in HexPlanes [6] and K-planes [17], and learns a deformation field instead.

2.3 Point Tracking

Point tracking methods, usually trained on large amounts of data, aided previous 3D tracking approaches by providing a strong prior [28]. We also construct several baselines that include point tracking methods (Section 6). Prior work on point tracking often studies tracking 2D points across video frames, where a dominant approach is training models on large-scale synthetic datasets containing ground-truth point trajectories [12, 14, 20, 57] or dense optical flows [48]. Optical flow [2, 43] or scene flow [1, 19, 45, 46] can also be viewed as single-step point-tracking in 2D and 3D, respectively.

Another relevant line of work tightly couples dynamic scene reconstruction and motion estimation of non-rigid objects. A predominant setup is fusing RGBD frames from videos of dynamic scenes or objects [33]. Tracking or correspondence-matching methods see a progression from template-based tracking of objects with known shape or kinematics priors (such as human hand, face or body poses) [7, 34,39], to more general shapes or scenes [4,5,58]. The main difference from these works is that we do not use depth input, and perform more rigorous quantitative evaluations on tracking specific points.

Most related to ours is the more recent methods that obtain tracking from neural scene rendering. DCT-NeRF [47] learns a coordinate-based neural scene representation that outputs continuous 3D trajectories across the whole input sequence. PREF [41] optimizes a dynamic space-time neural field with self-supervised motion prediction loss. Most recently, Luiten et al. [29] models dynamic 3D Gaussians explicitly across timestamps to achieve tracking. While our work also leverages 3D Gaussians, in contrary to the explicit modeling in Dynamic 3D Gaussians [29], we learn a deformation function that scales much better with video length, and we focus on deformable objects that are more challenging than the ball-throwing videos used in [29].

2.4 Tracking for Robotics

A core motivation for studying point tracking is the potential it can unlock for robotics applications: for example, RoboTAP [44] shows pre-trained point-tracking models improve sample efficiency of visual imitation learning. It detects task-relevant keypoints, infers where those points should move to, and computes an action that moves them there. Any-point [52] learns to predict keypoint tracks, but conditioned on language inputs. Track2Act [3] builds on Any-point by learning a generalizable zero-shot policy, which only needs a few embodiment-specific demonstrations.

Rigid-body, or 6D, pose tracking and estimation has a rich history in robotics due to it foundational ability to model the world for a robot to manipulate [10, 11, 27, 31, 42, 49–51, 55]. In this work, we propose a deformable object analog of 6D pose tracking with the aim of extending successes to deformable object manipulation.

While existing methods leverage 2D tracking, and learn an additional policy to output robot actions, DeformGS provides a more powerful representation that allows for reasoning directly in 3D, instead of in the 2D image space.

3 Problem Statement

Given a set of timed image sequences captured from multiple cameras with known intrinsics and extrinsics, the objective is to learn a model that performs 3D tracking and novel view synthesis. Each image sequence is captured over the same time interval $t \in [0, H]$.

3D Tracking The primary goal is to recover the trajectory of any point in a dynamic scene by modeling the deformation of Gaussians over time. Thus, the

objective is to find a function $x_t = Q(x_0, t_0, t)$, where $x_0 \in \mathbb{R}^3$ is the location of a point of interest at a chosen time $t_0 \in [0, H]$, while $x_t \in \mathbb{R}^3$ is the location of the same point at another chosen time $t \in [0, H]$. The function Q is valid for any point x_0 and any $t \in [0, H]$, allowing for tracking of any point in space.

Novel View Synthesis The secondary goal is to achieve accurate scene flow by using photometric consistency as a supervision signal. To achieve this, the objective is to recover novel views from arbitrary viewpoints. The extrinsics at any viewpoint can be captured by matrix P, with $P = K[R|T]$. Here K is the intrinsics matrix, R is the rotation matrix of a camera with respect to the world frame, and T is the translation vector with respect to the world frame. Concretely, the goal is to learn a function V such that $I_{P,t} = V(P, t)$, where $I_{P,t}$ is an image rendered from a camera with extrinsics P at time t. As with the tracking objective, the time parameter is valid for any $t \in [0, H]$.

4 Preliminary

4.1 Gaussian Splatting

3D Gaussian Splatting [22] deploys an explicit scene representation by rendering a large set of Gaussians each defined by their mean position μ and covariance matrix Σ. Given $x \in \mathbb{R}^3$, its Gaussian is

$$G(x) = e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)},$$

Directly optimizing the covariance matrix Σ would lead to infeasible covariance matrices, as they must be positive semi-definite to have a physical meaning. Instead, Gaussian Splatting [22] proposes to decompose Σ into a rotation R and scale S for each Gaussian:

$$\Sigma = RSS^T R^T,$$

and optimize R, S, and the mean position.

Given the transformation W of a camera, the covariance matrix can be projected into image space as

$$\Sigma' = JW\Sigma W^T J^T,$$

where J is the Jacobian of the affine approximation of the projective transformation.

During rendering, we compute the color C of a pixel by blending N ordered Gaussians overlapping the pixel :

$$C = \sum_{i \in N} c_i \alpha_i \prod_{j=1}^{i-1} (1 - \alpha_j).$$

where c_i is the color of each Gaussian and α_i is given by evaluating a 2D Gaussian with covariance multiplied with a learned per-Gaussian opacity σ [22, 56]. This representation allows for fast rendering of novel views, and aims to reconstruct the geometry of the scene.

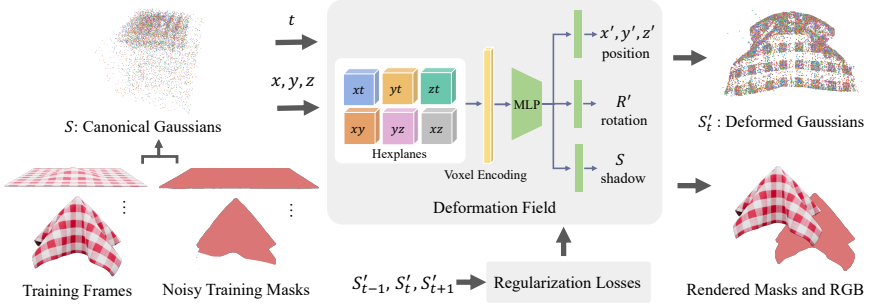


Fig. 2: DeformGS maps a set of Gausians with canonical properties to metric space using a deformation function F. The deformation function takes in the position of a Gaussian (x, y, z) and a queried timestamp t, to infer shadow s, rotation r' and metric position x'. During training, we use the metric positions and rotations to regularize the deformation function, considering the state at $t = \{i-1, i, i+1\}$ with Gaussian metric states P'_{t-1}, P'_t, P'_{t+1}

4.2 Deformation Fields for Dynamic Scenes

Prior work showed that a deformation function combined with a static NeRF in a canonical space can enable novel view synthesis in dynamic scenes. The deformation function $F_{\text{NeRF}} : \mathbb{R}^3 \rightarrow \mathbb{R}^3$ deforms a point in world coordinates (x') into a point in canonical space (x).

Prior work formulated F_{NeRF} as an MLP [38] and a multi-resolution voxel grid [16]. Wu et al. [53] applied a similar approach to arrive at Gaussian splatting of dynamic scenes. Given the state of a single canonical Gaussian, defined by $P = [\mu, S, R, \sigma, C]$ at time t, a deformation function is

$$P' = F_{\text{4DGS}}(P, t),$$

where F_{4DGS}, similar to the Hexplanes [6], contains a neural-voxel encoding in space and time. 4D-GS [53] starts with a *coarse* stage for initializing the canonical space, by setting $P' = P$, bypassing the deformation field and learning canonical properties directly. During the *fine* stage we learn the deformation function.

We propose DeformGS (Figure 2), based on 4D-Gaussians [53], to render novel views in dynamic scenes. The key differences with 4D-GS are: (1) we propose an intuitive method to track canonical Gaussians in world coordinates using a continuous deformation function, (2) the output of the deformation function is different, e.g., DeformGS infers shadows and does not alter opacity or scale over time, and (3) using the method shown in Figure 3, we enforce physics-inspired regularization losses on the 3D trajectories of Gaussians.

5 Method

DeformGS achieves novel-view synthesis and high-quality 3D tracking using a canonical space of Gaussians and a deformation function to deform them to

world space (Section 5.1). To incentivize learning physically plausible deformations, DeformGS introduces several regularization terms (Section 5.2). Finally, DeformGS learns 3D masks to focus regularization and Gaussian deformation on dynamic parts of the scene (Section 5.3).

5.1 4D Gaussian Splatting

Canonical Neural Voxel Encoding. As with prior work, DeformGS learns a deformation function F from a canonical space. We use a neural-voxel encoding to ensure F has sufficient capacity to capture complex deformations. Prior work [16,17,32,53] showed that neural-voxel encodings improve the speed and accuracy of novel-view synthesis in dynamic scenes. We leverage HexPlanes [6, 53] to increase capacity for simultaneous 3D tracking and novel-view synthesis.

Figure 2 shows an overview of the canonical neural-voxel encoding. Each of the six voxel modules can be defined as $R(i,j) \in \mathbb{R}^{h \times lN_i \times lN_j}$. Here $\{i,j\} \in \{(x,y),(x,z),(y,z),(x,t),(y,t),(z,t)\}$, i.e., we adopt HexPlanes in all possible combinations. h is the size of each feature vector in the voxel, N_i, N_j are the sizes of the HexPlanes in each dimension, l is the upsampling scale. In every module, each plane has a different upsampling scale l. To query the multi-resolution voxel grids, we query each plane using bilinear interpolation to finally arrive at a feature vector used by the deformation MLP.

Deformation MLP. The deformation MLP takes in the voxel encoding and uses the encoding to deform the canonical Gaussians into world coordinates. Figure 2 shows the deformation MLP, which infers position, rotation, and a shadow scalar, given a feature vector from the neural voxel encoding. We choose this set of outputs to model rigid-body transformations of each Gaussian and changes in illumination. Modeling changes in illumination is critical in the presence of shadows. We multiply the RGB color of each Gaussian by the shadow scalar $s \in [0,1]$, and the shadow scalar is in the range $[0,1]$ by feeding the output of the MLP through a sigmoid activation function.

Next, we deform the Gaussians, modifying their mean positions μ and rotation R, and arrive at a set of Gaussians in the world space each with state P. The differentiable rasterizer from Gaussian Splatting [22] then renders the Gaussians to retrieve gradients for regressing both the canonical Gaussian states and the parameters of the deformation function.

Unlike 4D-Gaussians [53], we propose to not infer opacity or scale using the deformation field. Optimizing for opacity and scale over time would allow Gaussians to disappear or appear instead of following the motion, which would make tracking less accurate. This design choice reduces the capacity of the deformation function, hence a lower view reconstruction quality as compared to 4D-Gaussians might be expected.

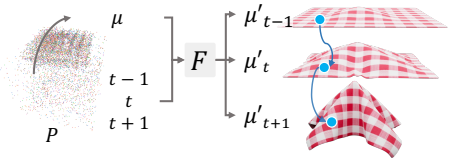


Fig. 3: DeformGS uses three adjacent timesteps at every iteration to enforce physics-inspired regularization terms. All Gaussians are deformed to world space using the deformation function F, and rasterized to compute the photometric loss and its gradients. The positions of the Gaussians are used to compute the regularization terms based on local isometry and conservation of momentum (Section 5.2).

5.2 3D Tracking using 4D Gaussians

Physics-Inspired Losses. Figure 3 shows the process of tracking Gaussians from the canonical space in world space. By querying the deformation function F, we can track the position of a Gaussian along the entire trajectory.

Without additional supervision, this approach will not necessarily converge to physically plausible deformations. Especially when objects include areas with little texture and uniform color, the solution space for all deformations is underconstrained by photometric consistency alone. To learn a more grounded deformation function, we propose regularization terms inspired by physics.

After empirically evaluating several combinations of regularization terms, we adopt the isometry loss proposed in [29] and add a conservation of momentum term. The first term captures a local isometry loss, which we compute based on the state of the k nearest neighboring (KNN) Gaussians.

Local Isometry Loss We incentivize the Gaussians to keep the relative position of the k nearest neighbors constant w.r.t. $t = 0$. With sufficient deformation, this assumption will be broken at a larger scale, but at a local scale, this regularization avoids drift from the ground-truth trajectory. The isometry loss is

$$\mathcal{L}_t^{\text{iso}} = \frac{1}{k|\mathcal{P}|} \sum_{i \in \mathcal{P}} \sum_{j \in \text{knn}_i} w_{i,j} \left| \|\mu_{j,0} - \mu_{i,0}\|_2 - \|\mu_{j,t} - \mu_{i,t}\|_2 \right| .$$

with

$$w_{i,j} = \exp\left(-\lambda_w \|\mu_{j,0} - \mu_{i,0}\|_2^2\right),$$

Here $\mathcal{P}$ is the set of all Gaussians.

Conservation of Momentum We add a term to incentivize conservation of momentum. Newton's first law states objects without external forces applied, given some mass m and velocity vector $\mathbf{v}$, maintain their momentum $m \cdot \mathbf{v}$. We introduce the regularization term

$$\mathcal{L}_{i,t}^{\text{momentum}} = \|\mu_{i,t+1} + \mu_{i,t-1} - 2\mu_{i,t}\|_1.$$

This term incentivizes a constant-velocity vector and has the effect of imposing a low-pass filter on the 3D trajectories. It smooths out trajectories with many sudden changes of direction and magnitude (momentum).

5.3 Learning 3D Masks

Learning accurate 3D tracking in scenes with a mix of static and dynamic objects and rich textures poses significant challenges, mainly: (1) imposing physics-inspired regularization terms on all Gaussians may cause issues when dynamic and static objects interact, and (2) modeling millions of dynamic Gaussians can become a significant computational burden.

To address this, DeformGS takes noisy masks of dynamic scene components such as cloth, and learns what Gaussians are dynamic. More formally, we render a mask M by

$$M = \sum_{i \in N} m_i \alpha_i \prod_{j=1}^{i-1} (1 - \alpha_j).$$

where m_i is a per-Gaussian property, with $m_i \in [0, 1]$. We then add a regularization term to the loss function s.t. m_i is regressed to best reconstruct M. Finally, DeformGS uses m_i to select a subset of Gaussians to be dynamic, and applies regularization terms only to those Gaussians.

6 Experiments

We evaluate DeformGS on synthetic and real-world datasets of scenes with highly deformable objects. Section 6.1 provides details on the simulation experiment setup, evaluation metrics, and baseline methods. Section 6.2 reports evaluation results from the compared methods and provides analysis. Section 6.3 lists the real-world evaluation setup, and finally in Section 6.4 we provide a qualitative evaluation of the performance of DeformGS in the real-world.

6.1 Simulation Experiment Setup

Dataset Preparation We use Blender to model dynamic cloth sequences and render photo-realistic images. We create 6 distinct scenes, each containing a different cloth with distinct visual and physical properties, and we render images from 100 different camera views and 40 consecutive time steps for training for a total of 4,000 images. The cloth deformations are introduced by dropping each cloth over one or a few invisible balls onto a ground plane or by constraining the cloth at an attachment point. We obtain ground-truth trajectories by tracking the mesh vertices of deformable objects in Blender. Every scene contains a single deformable object and a rendered background.

Oracle Baselines We compare DeformGS to 2D tracking oracle models which have access to ground truth depth and trajectory information. While these methods were not designed for 3D tracking, they are well-known for their

impressive 2D tracking performance. Their numbers aid in putting the tracking performance of the other baselines into context. We run RAFT [43] on all views, project tracking to 3D using ground truth depth, and report the mean results as the RAFT model. We provide two additional oracle methods which have access to the ground-truth trajectories as well. *RAFT Oracle* first evaluates on all views, to then output only the trajectories from the view with the lowest median trajectory error. We also report *OmniMotion Oracle*, which runs OmniMotion [48] on the viewpoint with the lowest MTE for RAFT. Training OmniMotion takes roughly 12–13 hours on an Nvidia RTX 4090 GPU, making inference on all 100 views impractical. The numbers from *RAFT Oracle* and *OmniMotion Oracle* are not an apples-to-apples comparison with the other methods, as to obtain their result they had to access privileged ground-truth trajectories.

Gaussian Splatting Baselines (1) Dynamic 3D Gaussians (*DynaGS*) [29], which also builds on 3D Gaussian splatting for dynamic novel-view synthesis, except it explicitly models the positions and rotations of each Gaussian at each time-step. This results in straightforward tracking of any point via finding the trajectory of the learned Gaussian closest to a queried point. Although the original paper assumes a known point cloud at the first frame, we instead use a randomly sampled point cloud for a fair comparison, with DynaGS and DeformGS both not using depth information.

(2) Finally, we compare to tracking using 4D-Gaussians [53] (*4D-GS*). We add the approach for 3D tracking of canonical Gaussian, as shown in Figure 3, to extract 3D trajectories from a learning view synthesis model. Comparing to *4D-GS* serves to show the impact of the changes made in the model architecture, the regularization terms, and using learning per-Gaussian masks to arrive at DeformGS.

Training and Evaluation Setup We create a dataset of 6 dynamic cloth scenes, each with varying physical and visual properties (Figure 1). For DeformGS and 4D Gaussians, we perform 30,000 training iterations, and set point cloud pruning interval to 100, voxel plane resolution to [64, 64], and multi-resolution upsampling to levels $L = \{1, 2, 4, 8\}$. We set the regularization hyper parameters (Section 5.2) to $\lambda_w = 2{,}000$, $\lambda^{\text{momentum}} = 0.03$, $\lambda^{\text{iso}} = 0.3$, and $k = 20$ for KNN. We generate the masks with segment anything (SAM) [24] for the initial frame, and use XMem [8] to propagate to future frames.

For DynaGS, we set $\lambda^{\text{rigid}} = 4$, $\lambda_w = 2{,}000$,$\lambda^{\text{iso}} = 2.0$, and $k = 20$, as in the open-source code.

We evaluate each compared method on 1,000 randomly sampled points on each cloth.

6.2 Simulation Results

3D Point Tracking Following prior work [29, 57], we report median trajectory error (MTE), position accuracy (δ), and the survival rate with a threshold of 0.5 [m] [29].

The results are summarized in Table 1. We make the following observations: (1) DeformGS outperforms baselines RAFT, DynaGS, and 4D-GS, by achieving

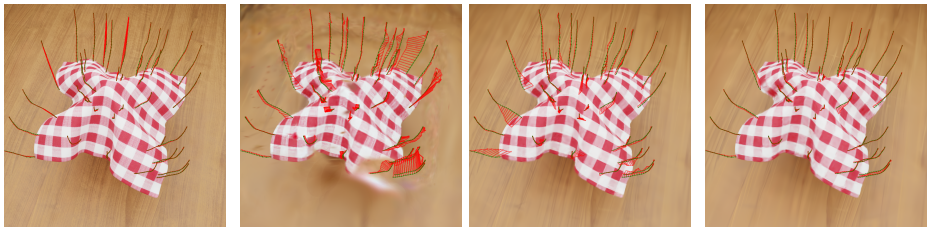
Metric	Method	Scene 1	Scene 2	Scene 3	Scene 4	Scene 5	Scene 6	Mean
3D MTE [mm] ↓	RAFT [43] [a]	67.264	89.944	220.125	177.909	84.593	23.422	110.543
	RAFT Oracle [43][ab]	3.381	26.956	58.971	12.481	3.930	3.192	18.152
	OmniMotion Oracle [48] [ab]	0.535	14.513	39.958	4.556	2.487	2.011	10.677
	DynaGS [29]	24.233	81.119	464.64	54.074	34.985	36.101	115.859
	4D-GS [53]	4.645	95.032	223.839	27.441	14.091	11.619	62.778
	DeformGS	**3.373**	**45.33**	**88.369**	**14.022**	**7.257**	**8.173**	**27.754**
3D δ_{avg} ↑	RAFT [43] [a]	0.553	0.379	0.222	0.533	0.586	0.619	0.482
	RAFT Oracle [43][ab]	0.926	0.577	0.411	0.703	0.833	0.808	0.710
	OmniMotion Oracle [48] [ab]	0.987	0.693	0.549	0.864	0.885	0.849	0.805
	DynaGS [29]	0.624	0.31	0.042	0.435	0.533	0.527	0.4118
	4D-GS [53]	0.902	0.339	0.164	0.583	0.697	0.715	0.5667
	DeformGS	**0.929**	**0.522**	**0.322**	**0.71**	**0.856**	**0.787**	**0.688**
3D Survival ↑	RAFT [43] [a]	0.945	0.792	0.779	0.822	0.792	0.854	0.831
	RAFT Oracle [43][ab]	0.986	0.833	0.957	0.872	0.929	0.903	0.913
	OmniMotion Oracle [48] [ab]	1	0.963	1	0.985	0.963	0.933	0.977
	DynaGS [29]	0.999	0.99	0.483	0.988	0.992	0.992	0.907
	4D-GS [53]	**1**	0.967	0.834	**1**	0.999	**1**	0.967
	DeformGS	**1**	**1**	**1**	**1**	**1**	**1**	**1**

[a] Method had access to ground truth depth. [b] Method had access to ground truth trajectories to pick the best camera view.

Table 1: 3D tracking results on the deformable cloth dataset (Figure 1). For each metric, the methods above the solid line had access to privileged information, see [ab] and Section 6.1 for more details. The results suggest that DeformGS outperforms the baselines in all averaged metrics, and is competitive with the oracle models. The results also suggest our novel deformation function architecture, learning per-Gaussian masks, and physics-inspired regularization losses improve the tracking performance compared to 4D-GS [53]. We do not consider the oracle methods to be fair baselines and therefore do not bold their results.

a MTE of 55.8% - 76.0% lower compared to the baselines. (2) The discrepancy between the RAFT oracle model and its averaged result demonstrates the difficulty arising from frequent self-occlusions. This also points to future research avenues for additional supervision through optic flow and 2D tracking algorithms such as RAFT. (3) The oracle models perform very well, this is in part thanks to the falling and short-horizon nature of these sequences, limiting self-occlusions. In the real-world we expect much larger errors due to noisy depth and more challenging occlusions in long-horizon tasks. It would also be unclear what viewpoint to choose without access to ground truth trajectories. (4) Scenes with less texture such as scene 3 perform significantly worse than scenes with strong texture.

Qualitative Results Figure 4 shows ground truth and inferred trajectories in scene 5. The results show that especially DynaGS and 4D-GS introduce large errors as the cloth drapes down. RAFT improves over DynaGS and 4D-GS but requires accurate depth estimation.



(a) RAFT [43] (b) DynaGS [29] (c) 4D-GS [53] (d) DeformGS

Fig. 4: **Results on Scene 5**: randomly sampled ground-truth trajectories in green, inferred trajectories in red, and the error of corresponding points in red lines. Compared to the baseline methods, DeformGS results in fewer errors in 3D tracking.

6.3 Real-World Experiment Setup

Robo360 Data The Robo360 dataset [26] is a 3D omnispective multi-material robotic manipulation dataset. It covers many different scenario's, including manipulation by robot manipulators and humans captured by 86 calibrated cameras. These properties make it an ideal dataset to evaluate the effectiveness of DeformGS in the real world. We select two scenes: (1) a human folding a larger duvet and (2) a human folding a smaller cloth. In the cloth folding scene, we exclude viewpoints where the entire person's body is visible to eliminate unnecessary complexity.

We also subsample the data to demonstrate DeformGS performance with fewer views. The duvet folding scene contains 17 training views and the cloth folding scene contains 20 training views.

6.4 Real-World Experiment Results

Real2Sim for Digital Twins Figure 5 shows the 3D tracking overlaid on rendered images, as well as the Gaussian points at each time step. The results suggest that DeformGS is able to successfully infer smooth and meaningful trajectories in the real world. While no ground truth is available, the trajectories appear to follow their geometry closely except for a few floating Gaussians. Hyperparameter tuning of the regularization functions, as well as discarding Gaussians with a low opacity, might help resolve this.

The point cloud included in this Figure can be used to create a digital twin after recording the sequence. The digital twin of the duvet, and the entire environment, can then be used to create more dense supervision for imitation learning approaches.

Task-Relevant Keypoint Tracking Robotic manipulators can benefit from tracking task-relevant keypoints, such as the corner of a cloth or the edge of a jacket. Figure 6 shows a comparison between 4D-GS [53] and DeformGS in 3D point tracking, evaluated on both duvet and cloth scenes. The results suggest DeformGS leads to more smooth and overall useful trajectories. The trajectories

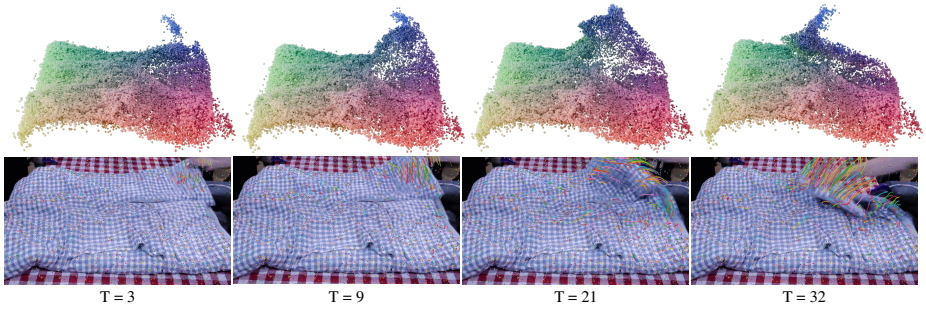


Fig. 5: A person manipulating a duvet in the Robo360 [26] dataset, reconstructed using DeformGS. The top row shows the 4D Gaussians as point clouds, where the color represents dense correspondences. The bottom row shows rendered views overlaid with 3D trajectories projected to image space.

from 4D-GS intertwine into more messy trajectories, and appear less physically plausible. This would hinder the adoption of 3D tracking into robot applications.

Fig. 6: Real-world results comparing our proposed DeformGS against 4D-GS [53]. The 3D trajectories inferred by DeformGS appear more smooth and accurate, whereas 4D-GS displays more cluttered trajectories.

7 Conclusions

In this work, we address the challenging problem of 3D point-tracking in dynamic scenes with deformable objects. We introduced DeformGS, the first approach that learns continuous deformations for 3D tracking of deformable scenes. We empirically demonstrate that DeformGS outperforms baseline methods and achieves both high-quality dynamic scene reconstruction and high-accuracy 3D tracking on highly deformed cloth objects with occlusions and shadows, both in simulation and the real world. We also contribute a dataset of six synthetic scenes to facilitate future research.

Limitations and Future Work DeformGS, similar to prior work on dynamic novel view reconstruction, requires a setup of multiple synchronized and calibrated cameras, which may require a significant engineering effort in real-world scenarios. Additionally, significant innovation will be required to achieve the demonstrated results in real-time, as will be beneficial for scalable robot applications.

While DeformGS improves upon prior methods, we do observe Gaussians wandering off in some cases. This might be resolved by adding supervision from state-of-the-art point-tracking algorithms. These limitations point to promising directions for future research.

Disclosure of Interests. The authors have no competing interests to declare that are relevant to the content of this article.

References

1. Basha, T., Moses, Y., Kiryati, N.: Multi-view scene flow estimation: A view centered variational approach. International Journal of Computer Vision **101**, 6–21 (2010), https://api.semanticscholar.org/CorpusID:1284146
2. Beauchemin, S.S., Barron, J.L.: The computation of optical flow. ACM computing surveys (CSUR) **27**(3), 433–466 (1995)
3. Bharadhwaj, H., Mottaghi, R., Gupta, A., Tulsiani, S.: Track2act: Predicting point tracks from internet videos enables diverse zero-shot robot manipulation (2024)
4. Božič, A., Palafox, P., Zollhöfer, M., Dai, A., Thies, J., Nießner, M.: Neural non-rigid tracking (2021)
5. Božič, A., Zollhöfer, M., Theobalt, C., Nießner, M.: Deepdeform: Learning non-rigid rgb-d reconstruction with semi-supervised data (2020)
6. Cao, A., Johnson, J.: Hexplane: A fast representation for dynamic scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 130–141 (2023)
7. Cao, C., Weng, Y., Lin, S., Zhou, K.: 3d shape regression for real-time facial animation. ACM Transactions on Graphics (TOG) **32**, 1 – 10 (2013), https://api.semanticscholar.org/CorpusID:2818777
8. Cheng, H.K., Schwing, A.G.: Xmem: Long-term video object segmentation with an atkinson-shiffrin memory model (2022)
9. Community, B.O.: Blender - a 3D modelling and rendering package. Blender Foundation, Stichting Blender Foundation, Amsterdam (2018), http://www.blender.org
10. Deng, X., Xiang, Y., Mousavian, A., Eppner, C., Bretl, T., Fox, D.: Self-supervised 6d object pose estimation for robot manipulation. In: 2020 IEEE International Conference on Robotics and Automation (ICRA). pp. 3665–3671. IEEE (2020)
11. Devgon, S., Ichnowski, J., Balakrishna, A., Zhang, H., Goldberg, K.: Orienting novel 3d objects using self-supervised learning of rotation transforms. In: 2020 IEEE 16th International Conference on Automation Science and Engineering (CASE). pp. 1453–1460. IEEE (2020)
12. Doersch, C., Gupta, A., Markeeva, L., Recasens, A., Smaira, L., Aytar, Y., Carreira, J., Zisserman, A., Yang, Y.: Tap-vid: A benchmark for tracking any point in a video. Advances in Neural Information Processing Systems **35**, 13610–13626 (2022)

13. Doersch, C., Gupta, A., Markeeva, L., Recasens, A., Smaira, L., Aytar, Y., Carreira, J., Zisserman, A., Yang, Y.: Tap-vid: A benchmark for tracking any point in a video (2023)
14. Doersch, C., Yang, Y., Vecerik, M., Gokay, D., Gupta, A., Aytar, Y., Carreira, J., Zisserman, A.: Tapir: Tracking any point with per-frame initialization and temporal refinement. arXiv preprint arXiv:2306.08637 (2023)
15. Du, Y., Zhang, Y., Yu, H.X., Tenenbaum, J.B., Wu, J.: Neural radiance flow for 4d view synthesis and video processing. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (2021)
16. Fang, J., Yi, T., Wang, X., Xie, L., Zhang, X., Liu, W., Nießner, M., Tian, Q.: Fast dynamic radiance fields with time-aware neural voxels. In: SIGGRAPH Asia 2022 Conference Papers (2022)
17. Fridovich-Keil, S., Meanti, G., Warburg, F.R., Recht, B., Kanazawa, A.: K-planes: Explicit radiance fields in space, time, and appearance. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12479–12488 (2023)
18. Gao, C., Saraf, A., Kopf, J., Huang, J.B.: Dynamic view synthesis from dynamic monocular video. In: Proceedings of the IEEE International Conference on Computer Vision (2021)
19. Guo, X., Sun, J., Dai, Y., Chen, G., Ye, X., Tan, X., Ding, E., Zhang, Y., Wang, J.: Forward flow for novel view synthesis of dynamic scenes. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 16022–16033 (October 2023)
20. Harley, A.W., Fang, Z., Fragkiadaki, K.: Particle video revisited: Tracking through occlusions using point trajectories. In: European Conference on Computer Vision. pp. 59–75. Springer (2022)
21. Karaev, N., Rocco, I., Graham, B., Neverova, N., Vedaldi, A., Rupprecht, C.: Cotracker: It is better to track together (2023)
22. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics **42**(4) (July 2023), https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/
23. Keselman, L., Hebert, M.: Approximate differentiable rendering with algebraic surfaces. In: European Conference on Computer Vision (ECCV) (2022)
24. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.Y., Dollár, P., Girshick, R.: Segment anything (2023)
25. Li, Z., Niklaus, S., Snavely, N., Wang, O.: Neural scene flow fields for space-time view synthesis of dynamic scenes. In: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 6494–6504. IEEE Computer Society, Los Alamitos, CA, USA (jun 2021). https://doi.org/10.1109/CVPR46437.2021.00643, https://doi.ieeecomputersociety.org/10.1109/CVPR46437.2021.00643
26. Liang, L., Bian, L., Xiao, C., Zhang, J., Chen, L., Liu, I., Xiang, F., Huang, Z., Su, H.: Robo360: A 3d omnispective multi-material robotic manipulation dataset (2023)
27. Lin, J., Liu, L., Lu, D., Jia, K.: Sam-6d: Segment anything model meets zero-shot 6d object pose estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 27906–27916 (2024)
28. Liu, J.W., Cao, Y.P., Mao, W., Zhang, W., Zhang, D.J., Keppo, J., Shan, Y., Qie, X., Shou, M.Z.: Devrf: Fast deformable voxel radiance fields for dynamic scenes. Advances in Neural Information Processing Systems **35**, 36762–36775 (2022)

29. Luiten, J., Kopanas, G., Leibe, B., Ramanan, D.: Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. arXiv preprint arXiv:2308.09713 (2023)
30. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. In: ECCV (2020)
31. Morgan, A.S., Wen, B., Liang, J., Boularias, A., Dollar, A.M., Bekris, K.: Vision-driven compliant manipulation for reliable, high-precision assembly tasks. RSS (2021)
32. Müller, T., Evans, A., Schied, C., Keller, A.: Instant neural graphics primitives with a multiresolution hash encoding. ACM Trans. Graph. **41**(4), 102:1–102:15 (Jul 2022). https://doi.org/10.1145/3528223.3530127, https://doi.org/10.1145/3528223.3530127
33. Newcombe, R.A., Fox, D., Seitz, S.M.: Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 343–352 (2015). https://doi.org/10.1109/CVPR.2015.7298631
34. Oikonomidis, I., Kyriazis, N., Argyros, A.A.: Efficient model-based 3d tracking of hand articulations using kinect. In: British Machine Vision Conference (2011), https://api.semanticscholar.org/CorpusID:8677556
35. Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., Assran, M., Ballas, N., Galuba, W., Howes, R., Huang, P.Y., Li, S.W., Misra, I., Rabbat, M., Sharma, V., Synnaeve, G., Xu, H., Jegou, H., Mairal, J., Labatut, P., Joulin, A., Bojanowski, P.: Dinov2: Learning robust visual features without supervision (2024)
36. Park, K., Sinha, U., Barron, J.T., Bouaziz, S., Goldman, D.B., Seitz, S.M., Martin-Brualla, R.: Nerfies: Deformable neural radiance fields. ICCV (2021)
37. Pumarola, A., Corona, E., Pons-Moll, G., Moreno-Noguer, F.: D-NeRF: Neural Radiance Fields for Dynamic Scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (2021)
38. Pumarola, A., Corona, E., Pons-Moll, G., Moreno-Noguer, F.: D-nerf: Neural radiance fields for dynamic scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 10318–10327 (2021)
39. Schmidt, T., Newcombe, R.A., Fox, D.: Dart: dense articulated real-time tracking with consumer depth cameras. Autonomous Robots **39**, 239 – 258 (2015), https://api.semanticscholar.org/CorpusID:254251877
40. Shi, H., Xu, H., Huang, Z., Li, Y., Wu, J.: Robocraft: Learning to see, simulate, and shape elasto-plastic objects with graph networks (2022)
41. Song, L., Gong, X., Planche, B., Zheng, M., Doermann, D., Yuan, J., Chen, T., Wu, Z.: Pref: Predictability regularized neural motion fields. In: European Conference on Computer Vision (2022)
42. Taher, M., Alzugaray, I., Davison, A.J.: Fit-ngp: Fitting object models to neural graphics primitives. arXiv preprint arXiv:2401.02357 (2024)
43. Teed, Z., Deng, J.: Raft: Recurrent all-pairs field transforms for optical flow. In: Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16. pp. 402–419. Springer (2020)
44. Vecerik, M., Doersch, C., Yang, Y., Davchev, T., Aytar, Y., Zhou, G., Hadsell, R., Agapito, L., Scholz, J.: Robotap: Tracking arbitrary points for few-shot visual imitation. arXiv preprint arXiv:2308.15975 (2023)
45. Vedula, S., Baker, S., Rander, P., Collins, R., Kanade, T.: Three-dimensional scene flow. In: Proceedings of the Seventh IEEE International Conference on Computer Vision. vol. 2, pp. 722–729. IEEE (1999)

46. Vogel, C., Schindler, K., Roth, S.: 3d scene flow estimation with a piecewise rigid scene model. International Journal of Computer Vision **115**, 1–28 (2015), https://api.semanticscholar.org/CorpusID:3358790
47. Wang, C., Eckart, B., Lucey, S., Gallo, O.: Neural trajectory fields for dynamic novel view synthesis (2021)
48. Wang, Q., Chang, Y.Y., Cai, R., Li, Z., Hariharan, B., Holynski, A., Snavely, N.: Tracking everything everywhere all at once. In: International Conference on Computer Vision (2023)
49. Wen, B., Lian, W., Bekris, K., Schaal, S.: You only demonstrate once: Category-level manipulation from single visual demonstration. RSS (2022)
50. Wen, B., Mitash, C., Ren, B., Bekris, K.E.: se (3)-tracknet: Data-driven 6d pose tracking by calibrating image residuals in synthetic domains. In: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 10367–10373. IEEE (2020)
51. Wen, B., Yang, W., Kautz, J., Birchfield, S.: Foundationpose: Unified 6d pose estimation and tracking of novel objects (2024)
52. Wen, C., Lin, X., So, J., Chen, K., Dou, Q., Gao, Y., Abbeel, P.: Any-point trajectory modeling for policy learning (2023)
53. Wu, G., Yi, T., Fang, J., Xie, L., Zhang, X., Wei, W., Liu, W., Tian, Q., Wang, X.: 4d gaussian splatting for real-time dynamic scene rendering. arXiv preprint arXiv:2310.08528 (2023)
54. Xian, W., Huang, J., Kopf, J., Kim, C.: Space-time neural irradiance fields for free-viewpoint video. In: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). pp. 9416–9426. IEEE Computer Society, Los Alamitos, CA, USA (jun 2021). https://doi.org/10.1109/CVPR46437.2021.00930, https://doi.ieeecomputersociety.org/10.1109/CVPR46437.2021.00930
55. Xiang, Y., Schmidt, T., Narayanan, V., Fox, D.: Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. arXiv preprint arXiv:1711.00199 (2017)
56. Yifan, W., Serena, F., Wu, S., Öztireli, C., Sorkine-Hornung, O.: Differentiable surface splatting for point-based geometry processing. ACM Transactions on Graphics **38**(6), 1–14 (Nov 2019). https://doi.org/10.1145/3355089.3356513, http://dx.doi.org/10.1145/3355089.3356513
57. Zheng, Y., Harley, A.W., Shen, B., Wetzstein, G., Guibas, L.J.: Pointodyssey: A large-scale synthetic dataset for long-term point tracking. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 19855–19865 (2023)
58. Zollhöfer, M., Nießner, M., Izadi, S., Rhemann, C., Zach, C., Fisher, M., Wu, C., Fitzgibbon, A.W., Loop, C.T., Theobalt, C., Stamminger, M.: Real-time non-rigid reconstruction using an rgb-d camera. ACM Transactions on Graphics (TOG) **33**, 1 – 12 (2014), https://api.semanticscholar.org/CorpusID:9616070