Comput. Aided Geom. Des. 119 (2025) 102437

Contents lists available at ScienceDirect

Computer Aided Geometric Design

journal homepage: www.elsevier.com/locate/cagd

Physics and geometry-augmented neural implicit surfaces for rigid bodies

Yuanmu Xu [a,c,1], Guanli Hou [b,1], Jiangbei Hu [d], Tenglong Ren [g], Xiaokun Wang [a], Yalan Zhang [a], Xiaojuan Ban [a], Chen Qian [h], Fei Hou [e,f], Ying He [b,*]

[a] School of Intelligence Science and Technology, University of Science and Technology Beijing, China
[b] College of Computing and Data Science, Nanyang Technological University, Singapore
[c] Shunde Innovation School, University of Science and Technology Beijing, China
[d] School of Software, Dalian University of Technology, China
[e] Key Laboratory of System Software (CAS), State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, China
[f] University of Chinese Academy of Sciences, China
[g] College of Humanities, Arts and Social Sciences, Nanyang Technological University, Singapore
[h] Sensetime, China

ARTICLE INFO

Editor: Shiqing Xin

Dataset link: https://github.com/Raining00/PGA-NeuS

Keywords:
Neural implicit surfaces
Neural rendering
Physics-based simulation
Rigid bodies

ABSTRACT

This paper tackles the challenges of physics-based simulation of rigid bodies in neural rendering, with a focus on 3D model representation and collision handling. We propose Physics and Geometry-Augmented Neural Implicit Surfaces (PGA-NeuS), a novel approach that combines neural implicit surfaces with a differentiable physics solver. In the pre-processing stage, PGA-NeuS reconstructs static scene and object geometry from multi-view images using signed distance fields (SDFs). For dynamic scenes captured in monocular videos, these SDFs, along with the initial position and orientation of moving rigid bodies, are fed into a differentiable rigid body solver to optimize physical parameters, such as initial velocity and friction coefficients. Subsequently, PGA-NeuS leverages color loss, physics loss, and object mask supervision to iteratively refine the neural implicit surface, ensuring the target object's alignment with the predicted motion sequence. We evaluate PGA-NeuS on five real-world scenes, demonstrating its ability to accurately reconstruct realistic motion sequences and estimate physical parameters such as position and velocity. Dataset and source code are available at https://github.com/Raining00/PGA-NeuS.

1. Introduction

Neural Radiance Fields (NeRF) introduce a groundbreaking approach to reconstructing 3D representations from multi-view images (Mildenhall et al., 2020), using neural networks to encode scene attributes such as color and geometry. Since its inception, NeRF has evolved rapidly across various research domains (Sun et al., 2022; Müller et al., 2022; Jain et al., 2021; Long et al., 2022; Xu et al., 2022; Wang et al., 2021; Yang et al., 2022; Wang et al., 2022; Li et al., 2023b), significantly impacting applications in visual computing. Recent studies (Pumarola et al., 2021; Tretschk et al., 2021a) have extended NeRF to dynamic scene reconstruction, achieving high geometric quality. However, these approaches often lack physical interpretability, leading to unnatural outcomes in

* Corresponding author.
E-mail address: yhe@ntu.edu.sg (Y. He).
[1] Equal contribution.

https://doi.org/10.1016/j.cagd.2025.102437
Received 10 March 2025; Accepted 15 April 2025

Available online 29 April 2025

simulations. To address this limitation, recent works (Li et al., 2023a; Chu et al., 2022; Chen et al., 2022; Le Cleac'h et al., 2023) have incorporated physical information into neural field reconstruction, enabling more accurate simulations of physical behaviors. Despite these advancements, most works have focused on synthetic data, as the complexity of real-world motion and collisions remain significant barriers.

To tackle these challenges, we focus on reconstructing scenes with rigid objects undergoing rigid motions, enhancing implicit neural representations to understand rigid body motions and interactions from images through a radiance field reconstruction method augmented by physics and geometry. We introduce PGA-NeuS, a novel framework that combines a differentiable physics solver with neural implicit surfaces to reconstruct dynamic scenes involving rigid body collisions. Our method uses a low-cost data acquisition system with a single camera to capture multi-view images and motion videos, facilitating real-world data collection for simulations. By reconstructing implicit surfaces of objects in various poses, we ensure accurate geometry for rendering and collision detection. PGA-NeuS employs a differentiable rigid body solver to model physical processes, optimizing motion sequences with color loss. This integration of physical and geometric information enables precise reconstruction of geometry and physical parameters, such as velocity and collision responses, in dynamic scenes. We validate our algorithm with real-world datasets, demonstrating its accuracy in fitting physical parameters and detecting collisions using synthetic data. Our contributions include:

- We propose a physics and geometry-augmented approach that reconstructs rigid body motion sequences from monocular video, solving an inverse problem in dynamic scene reconstruction.
- Our method can effectively integrate two scenes represented by neural implicit surfaces in a low-cost manner.
- By combining neural implicit surfaces with a physics solver and separating object reconstruction from the dynamic scene, PGA-NeuS allows objects to be inserted into arbitrary scenes and generates novel motion content.

2. Related work

Neural implicit surface reconstruction. In recent work, surface reconstruction successfully introduces neural implicit representations of 3D geometry (Wang et al., 2021; Yariv et al., 2021; Oechsle et al., 2021; Peng et al., 2020). In subsequent works, hash encoding acceleration methods (Wang et al., 2023; Li et al., 2023c; Rosu and Behnke, 2023) achieve fast and accurate reconstruction. In these 3D reconstruction tasks, geometric representations are performed using occupancy grids (Oechsle et al., 2021; Peng et al., 2020) or signed distance functions (SDF) (Wang et al., 2021; Yariv et al., 2021). The project should be easily reusable, and our physical simulation method requires a continuous representation of the distance from the model to the background and gradient calculations. Therefore, in our work, we use SDF to represent 3D geometry and apply NeuS as a part of the fundamental approach in our pipeline to provide object meshes and background SDF. Recently, Gaussian Splatting (GS) has gained traction as an effective method for reconstructing 3D scenes and performing real-time rendering. This approach utilizes anisotropic Gaussian ellipsoids to achieve high-quality, view-dependent rendering (Kerbl et al., 2023). In efforts to enhance GS's capability to reconstruct surfaces from images, subsequent studies have experimented with integrating geometrical constraints into the optimization process of Gaussian kernels (Lyu et al., 2024; Guédon and Lepetit, 2024; Huang et al., 2024).

Dynamic NeRF. Reconstructing objects from a video sequence has been a perennial challenge in the realms of computer vision and graphics. The advent of NeRF, as proposed by recent works (Mildenhall et al., 2020), has unveiled a novel avenue for reconstructing scenes from multi-view images, achieving commendable performance results. Subsequent endeavors have been made to extend NeRF to reconstruct dynamic scenes. Incorporating an additional neural network to discern the connections and mappings between scenes at disparate time instances has emerged as an efficacious approach (Attal et al., 2021; Du et al., 2021b; Gao et al., 2021; Li et al., 2022a; Park et al., 2021; Pumarola et al., 2021; Wang et al., 2023; Tretschk et al., 2021b; Wu et al., 2023; Johnson et al., 2023; Kairanda et al., 2023). Among these advances, there are two special methods, D-NeRF (Pumarola et al., 2021) and NR-NeRF (Tretschk et al., 2021b), which are highly representative. D-NeRF, for instance, innovates by capturing motion through encircling the subject, employing temporal mapping to accurately reconstruct a diverse array of animations from video sequences. Although theoretically feasible with a single camera, this approach faces practical challenges in real-world data acquisition, highlighting the complexity of capturing dynamic scenes. Building on this, NR-NeRF introduces a novel technique of bending light rays, enabling the successful reconstruction of high-quality dynamic scenes from monocular videos and pushing the boundaries of what's achievable with NeRF in dynamic environments.

Physics-augmented neural rendering. Neural editing techniques have recently gained attention as an effective means to modify neural rendering models (NeRF or 3D-Gaussian) (Chen et al., 2024), enabling intuitive manipulation of scene content, such as object insertion, removal (Peng et al., 2021; Qiao et al., 2023), and deformation (Yang et al., 2022). Subsequent studies further incorporated physics simulation into neural editing workflows. PIE-NeRF (Feng et al., 2024) combines meshless physics simulation with NeRF, allowing physically realistic elastodynamic animations. PhysGaussian (Xie et al., 2024) incorporates Newtonian physics into 3D Gaussian kernels using the Material Point Method (MPM). This integration facilitates a broad spectrum of neural editing capabilities for various materials, encompassing elastic, plastic, non-Newtonian fluids, and granular media. The previously mentioned methods struggle to achieve real-time performance due to the computational demands of physical simulation and volumetric rendering. Thus, VR-GS (Jiang et al., 2024) combines position-based dynamics (PBD) with 3D Gaussian splatting to facilitate real-time neural editing on VR platforms, utilizing a dual-level embedding approach and deformable body simulation for interactive physics-based user experiences in immersive virtual settings.

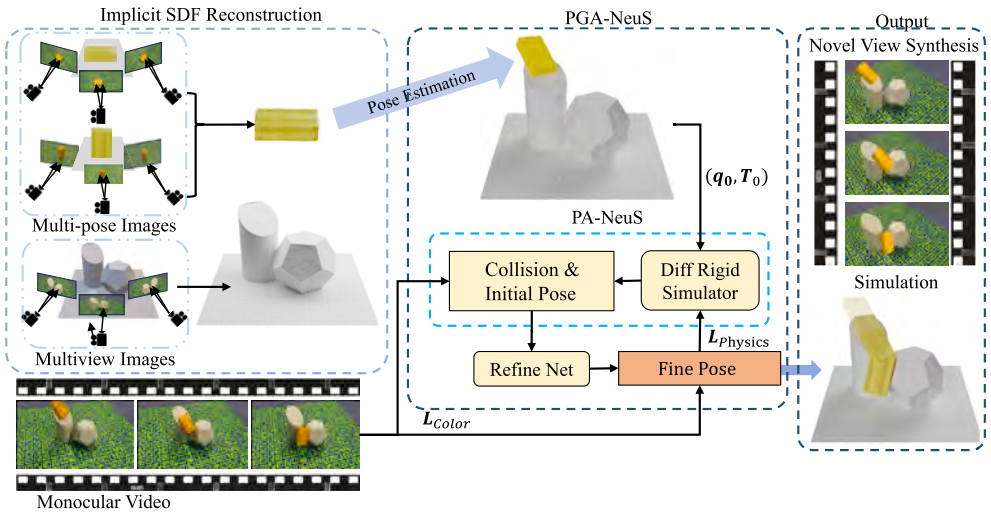

Fig. 1. Overview of PGA-NeuS. PGA-NeuS is designed to reconstruct the motion of rigid objects from monocular videos. Initially, the geometry of the static object and background objects is reconstructed from multi-view images using an SDF-based reconstruction module. This geometry serves as a basis for estimating the initial pose of the movable object. PGA-NeuS then iteratively optimizes the pose parameters, leveraging geometry constraints and a differentiable physics solver. Upon reconstructing the entire motion sequence from monocular video inputs, the neural rendering framework enables the production of visual outcomes through novel view synthesis.

Significant progress has been made in the field of differentiable physical simulation, leading to the creation of solvers for rigid bodies (de Avila Belbute-Peres et al., 2018; Röhrbein and Uchibe, 2021), soft materials (Hu et al., 2019; Hahn et al., 2019; Geilinger et al., 2020; Du et al., 2021a; Li et al., 2022b), and fluid dynamics (McNamara et al., 2004; Schenck and Fox, 2018). These solvers contribute to the advancements in learning and control applications. However, applying these solvers to real-world scenarios is challenging due to data acquisition constraints. Recently, advances in neural editing methodologies, which initially utilized forward physics simulations to integrate dynamic behaviors within neural rendering models, have evolved to enable the smooth incorporation of differentiable physical simulations with neural rendering techniques. NeRF-based methods (Xu et al., 2022; Wang et al., 2021; Yariv et al., 2021) provide necessary geometric structures, enabling the combination of neural fields with physics simulations for more dynamic and meaningful scene modeling (Guan et al., 2022; Chu et al., 2022; Driess et al., 2023). Despite progress, most techniques are limited to synthetic environments and lack interactive motion reconstruction. Approaches like PAC-NeRF (Li et al., 2023a) rely on pre-known collision boundaries unsuitable for real-world data. PI-NeRF (Chu et al., 2022) simulates collisions without true physical underpinnings. DANOs (Le Cleac'h et al., 2023) proposed a Probabilistic Contact Model that allows interaction between rigid bodies but cannot reconstruct trajectory scenes with more than one moving rigid body. Additionally, it does not reconstruct the background and cannot track the trajectory of a rigid body in a scene with static objects, except for the floor. Our method overcomes these gaps by accurately reconstructing real-world rigid body motions, combining SDFs and differentiable physics to simulate objects against diverse backgrounds, and negating the need for specialized setups or conditions.

3. Method

We propose PGA-NeuS, a differentiable, physics-based neural rendering framework. The primary workflow of our method is depicted in Fig. 1.

3.1. Overview

First, we reconstruct the geometry of static objects and backgrounds using neural implicit surfaces (NeuS), as detailed in Sec. 3.2. Next, we incorporate a differentiable rigid body solver with neural radiance fields to obtain rigid body properties, including collision and motion, termed as *PA-NeuS* in Sec. 3.3. We can also generate novel rigid body motion trajectory animation. Finally, we solve inverse problems by refining rigid body trajectories and using them to guide a differentiable solver for inferring physical parameters, termed *PGA-NeuS* in Sec. 3.4.

Note that the complexity of operating several cameras simultaneously creates a barrier to general applicability, so we focus on reconstructing the rigid motion of objects in the real-world with a single camera. In addition, our approach makes the following assumptions: a) The object's mass density is uniform, so its geometric center aligns with the center of mass. b) The static background possesses infinite mass, ensuring that obstacles in the background remain stationary during collisions. c) The background material is homogeneous, resulting in a single set of collision parameters during motion.

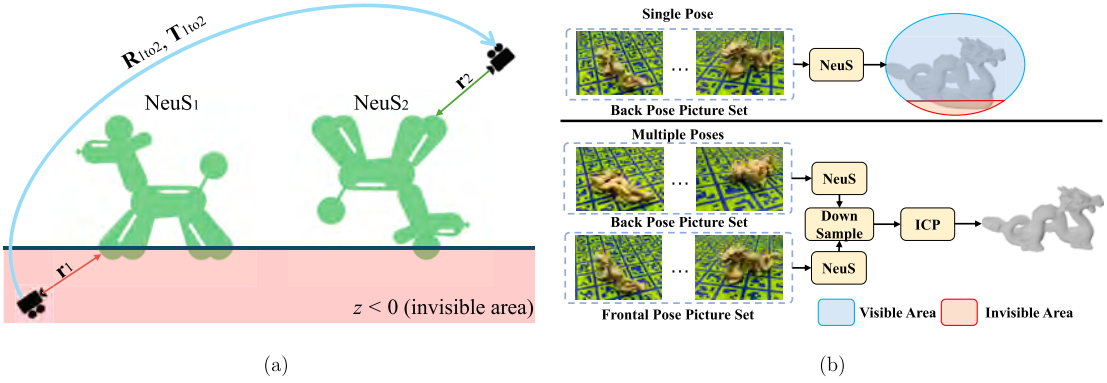

Fig. 2. (a) Switching NeuS: When any ray ($\mathbf{r}_1$) in NeuS_1 starts in the invisible area $z(\mathbf{p}_{\text{cam}}) < 0$, we apply $\mathbf{R}_{1\text{to}2}$, $\mathbf{T}_{1\text{to}2}$ to the ray to switch it in another NeuS ($\mathbf{r}_2$ in NeuS_2) for rendering. (b) Due to the high angle of the camera, the multi-view images captured from a single pose are generally insufficient to recover the geometry of the entire object. To obtain the full geometry, we typically place the object in two orientations-upright and top-down- and reconstruct the partial geometry from each pose separately. Subsequently, we employ the ICP algorithm to align these partial reconstructions, thereby obtaining the complete geometry of the object. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

3.2. Implicit surface reconstruction

As illustrated in Fig. 1, by providing multi-view images as input, we employ NeuS (Wang et al., 2021) to reconstruct the geometry of the objects present in the scenes. Specifically, we build a neural network S_Θ and F_Θ to output the corresponding color $\mathbf{c}$ and the signed distance value s, and opacity α for a query point $\mathbf{p}$ along a sample ray $\mathbf{r}$ as

$$S_\Theta(\mathbf{p}) = \mathbf{s}, F_\Theta(\mathbf{p}, \mathbf{r}, S_\Theta(\mathbf{p})) = (\mathbf{c}, \alpha), \tag{1}$$

where Θ represents the network parameters to be optimized under the supervision of the color loss from the multi-view images. Then, following the formulation of NeuS, we render the pixel color $C(\mathbf{r})$ with N sample points $\{\mathbf{p_i}\}_{i=1}^N$ along the ray $\mathbf{r}$ as:

$$\begin{aligned} C(\mathbf{r}) &= \sum_{i=1}^{N} T_i \alpha_i \mathbf{c}_i, T_i = \prod_{j=1}^{i-1}(1-\alpha_j), \\ \alpha_j &= \max\left(\frac{\Phi_s(s_i) - \Phi_s(s_{i+1})}{\Phi_s(s_i)}, 0\right), \end{aligned} \tag{2}$$

where T is the accumulated transmittance, Φ_s is the cumulative distribution of the logistic distribution, and α is the opacity derived from the SDF values of the adjacent points using Eq. (1).

When an object is placed on the ground, the underside cannot be captured by the camera, as illustrated by the red invisible region ($z < 0$) in Fig. 2 (a). In contrast, we define the area above the calibration board ($z \geq 0$) as a visible region. The invisible regions lead to data deficiencies, preventing NeuS from reconstructing the correct geometry in these areas, as shown in the upper right of Fig. 2. The incorrect geometry affects physics simulation and motion sequence reconstruction. To mitigate this, we capture images of an object from two different poses and reconstruct them separately using two NeuS networks, NeuS_1 and NeuS_2. By aligning and merging the outputs of these two NeuS, we can achieve the complete and correct geometry. Specifically, we extract the zero-level points set in each NeuS and align them using the iterative closest point (ICP) method (Olesik, 1991; Segal et al., 2009). We denote the rotation and translation from NeuS_1 to NeuS_2 as $\mathbf{R}_{1\text{to}2}$ and $\mathbf{T}_{1\text{to}2}$. Then, as shown in Fig. 2 (b), we extract the mesh from NeuS using the marching cube method (Lorensen and Cline, 1998) and remove the part of the mesh in the invisible area. Finally, we apply $\mathbf{R}_{1\text{to}2}$ and $\mathbf{T}_{1\text{to}2}$ to the mesh extracted in the NeuS_1 and then merge two meshes into one mesh, ensuring that a complete surface is reconstructed for simulation.

In addition to obtaining the complete geometry, we also need to achieve full rendering of visible and invisible regions. The two poses have complementary visibility because the invisible area of one NeuS model is visible for the other. As an example in Fig. 2 (a), the foot of the green dog in NeuS_1 and the head of it in NeuS_2 are both in the invisible area, but these parts are visible after switching to the other NeuS. Therefore, we designate NeuS_1 as the primary rendering network and switch to NeuS_2 when the camera is in the invisible area of the primary network. Applying the relative transformation $\mathbf{R}_{1\text{to}2}$ and $\mathbf{T}_{1\text{to}2}$, we generate rays in NeuS_1 and NeuS_2 as follows:

$$\mathbf{r}_1 = \mathbf{p}_{\text{cam}} + r\mathbf{d}, \quad \mathbf{r}_2 = \mathbf{R}_{1\text{to}2}\mathbf{r}_1 + \mathbf{T}_{1\text{to}2}, \tag{3}$$

where $\mathbf{r}_1$ and $\mathbf{r}_2$ are rays generated using principles from classical volume rendering (Kajiya and Von Herzen, 1984), and $\mathbf{d}$ is the view direction. $\mathbf{p}_{\text{cam}}$ is both the camera's 3D position in the calibration board coordinate system and the origin of the ray. r represents the distance of the sample points along the ray from the camera position $\mathbf{p}_{\text{cam}}$ in the direction $\mathbf{d}$. After rendering each ray separately, we choose one of the rendering results $\hat{C}$ using the equation:

Fig. 3. Data acquisition. Left and middle: We use a single camera to capture dynamic scenes involving moving rigid objects in a static environment. As the camera is mounted on a tripod and positioned to shoot the scene from a high angle, given that the objects are relatively small (typically less than 10 cm), some regions may be obscured due to occlusion and self-occlusion. Right: Estimating the initial pose involves using an ArUco marker to determine the object's rough position and orientation. Then, PGA-NeuS refines this estimate by minimizing the color loss between the rendered image and the masked first frame of the input video.

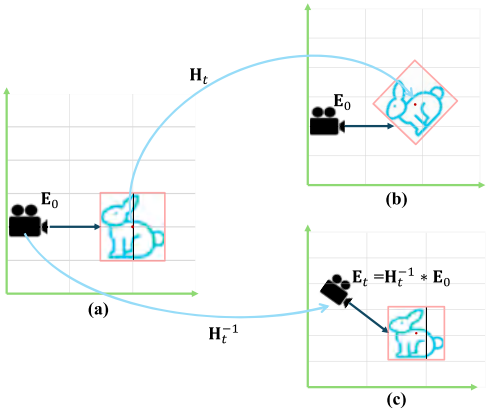

Fig. 4. Camera transformation. (a) E_0 is the camera position of the first frame from estimation. (b) When the object we need to render moves at the time t, we denote its transformation as $\mathbf{H}_t$. (c) the render result is equal to applying the inverse transformation ($\mathbf{H}_t^{-1}$) to the camera.

$$\hat{C} = \begin{cases} C_1(\mathbf{r}_1), & \text{if } z(\mathbf{p}_{\text{cam}}) \geq 0, \\ C_2(\mathbf{r}_2), & \text{if } z(\mathbf{p}_{\text{cam}}) < 0, \end{cases} \tag{4}$$

where C_1 and C_2 represent the rendered pixel colors from the two NeuS models using Eq. (2), $z(\mathbf{p}_{cam})$ is the z-coordinate of the camera position.

3.3. Physics-augmented NeuS

To complete the reconstruction of dynamic sequences, it is essential to address how to combine a statically reconstructed neural radiation field with a rigid solver to generate dynamic motion content. We start by describing the motion of a rigid body. For objects with minimal deformation, treating them as rigid bodies is effective. The motion of a rigid body can be described by the motion of its center of mass:

$$\mathbf{p}_t^c = \mathbf{R}_t \mathbf{p}_0^c + \mathbf{T}_t, \tag{5}$$

where $\mathbf{p}_t^c$ is the spatial coordinate of a rigid body's centroid, while $\mathbf{R}_t$, $\mathbf{T}_t$ represent the rotation and translation at time t. Any point $\mathbf{p}_t$ at time t in the reference configuration space can be described using:

$$\mathbf{p}_t = \mathbf{R}_t(\mathbf{p}_0 - \mathbf{p}_0^c) + \mathbf{T}_t. \tag{6}$$

According to relative motion, the observed motion of objects can be interpreted as the result of an inverse motion of the camera that captures the scene, as shown in Fig. 4. This allows us to generate dynamic motion videos even when only a static reconstruction of the scene is available. In the framework of NeuS, the pose of the camera can be represented by the extrinsic matrix $\mathbf{E}$. We can now write the following formula:

$$\mathbf{E}_t = \mathbf{H}_t^{-1} * \mathbf{E}_0, \tag{7}$$

where $\mathbf{H}_t$ is the homogeneous transformation matrix that represents the combined rotation $\mathbf{R}_t$ and translation $\mathbf{T}_t$ of the body at time t, $\mathbf{E}_0$ means the initial extrinsic matrix. If the sequence of $\mathbf{H}_t$ is provided by a rigid body solver, we can subsequently obtain an animation depicting how this reconstructed object moves under certain conditions defined by the user.

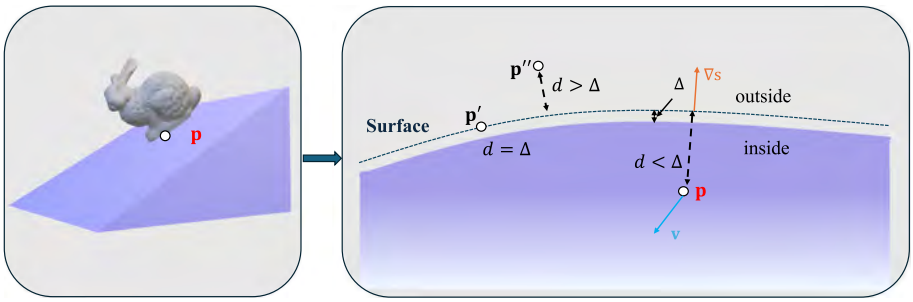

Fig. 5. SDF-based collision detection. Left: The static environment is represented as neural signed distance fields. Right: We determine the spatial relationship between the moving object and the environment by querying the distance value of each point on the moving object for the static environment.

Collision detection. A key challenge in extending NeRF integrated with physical information to real-world applications lies in accurately capturing collision information, which is crucial for realistic simulations. NeuS (Wang et al., 2021) can directly provide SDF and SDF gradient to the physics solver at each time t, which allows efficient collision detection:

$$\mathrm{Col}(\mathbf{p}_t) = \begin{cases} \text{true}, & \text{if } d \le \Delta \text{ and } \mathbf{v}_t \cdot \boldsymbol{\nabla}\mathbf{s} < 0, \\ \text{false}, & \text{otherwise}, \end{cases} \tag{8}$$

where d represents the shortest distance from a point in space to the surface of the colliding object (the SDF value), and $\boldsymbol{\nabla}\mathbf{s}$ is the corresponding SDF gradient, both of which are directly obtained from the background NeuS network. Δ is a pre-defined threshold that adjusts the tolerance for penetration during the detection process. Fig. 5 expresses this process more clearly. In our experiments, we set the threshold to $\Delta = 5 \times 10^{-3}$ meters.

By inserting objects in different scenes and combining them with Eq. (6) and Eq. (8), we can generate animations of different objects with collisions in different scenes. The generated animation is based on the initial pose (translation $\mathbf{T}_0$, quaternion $\mathbf{q}_0$), the initial velocity (linear velocity $\mathbf{v}_t$ and angular velocity $\boldsymbol{\omega}_0$), and the collision parameters (normal recovery coefficient k_n and friction coefficient μ).

3.4. Physics and geometry-augmented NeuS

Since we are already capable of inserting meshes into the scenes and generating animations, reconstructing the motion trajectory of an object from a captured video becomes possible, even if its collision boundary is an irregular object. Specifically, we can compute the color loss by comparing the rendering results with each video frame to align the rigid body with its correct geometric position. However, real-world complexities make direct optimization via pixel color less effective. Thus, we try to obtain the geometric pose sequence at first and then incorporate it into the loss function of the differentiable physics process, eliminating the need for indirect interactions between the physics solver and volume rendering.

Geometry augmentation. To simplify the separation of static background reconstruction from moving objects, we use an ArUco marker-based approach for initial pose estimation. As shown in Fig. 3 (right), we use an ArUco marker to represent the object and place it at the approximate position of the photographing scene (the upper-left part of Implicit SDF Reconstruction in Fig. 1) and the t_0 moment of the motion scene (the bottom-left part of Implicit SDF Reconstruction in Fig. 1). Then we roughly estimate the spatial transformation relationship between the two poses of the ArUco markers to get a coarse estimation of the rigid body within the background. This initial estimation is refined in a subsequent optimization phase, where the pose is iteratively adjusted for better accuracy, as expressed by the following equation:

$$\min_{\mathbf{q}_t, \mathbf{T}_t} \quad \mathcal{L}_{\text{color}} = \sum |\hat{C}(\mathbf{q}_t, \mathbf{T}_t) - C_{gt}(t)| + L_{penalty}, \tag{9}$$

where $\hat{C}$ denotes the pixel color rendered using Eq. (4) and $C_{gt}(t)$ means the captured frame RGB at time t. Symbol $\mathbf{q}_t$ is the quaternion which represents the object's rotation information. Additionally, $L_{penalty}$ represents the penalty term that is incurred if the constraints are violated during the optimization process. This term consists of two components: one enforces the constraint $\|\mathbf{q}_t\| = 1$, and the other is called contact constraint penalty $L_{contact}$, which is specified as follows:

$$L_{contact} = \begin{cases} \lambda_{in} \exp\left(\frac{1}{M} \frac{\sum_{j=1}^{M} s(\mathbf{p}_j)}{s(\mathbf{p}_c)} \right) - 1, & \text{if } M > 0, \\ \lambda_{out} \exp\left(\frac{\tilde{s}(\mathbf{p}_c^t) - s(\mathbf{p}_c^t)}{s(\mathbf{p}_c^t)} \right) - 1, & \text{if } M = 0, \end{cases} \tag{10}$$

where M indicates the number of points that are detected using Eq. (8), and $s(\mathbf{p})$ is SDF value that can infer from S_Θ using Eq. (1). Additionally, λ_{in} and λ_{out} are pre-defined hyper-parameters. We will discuss this in detail later, but for initial pose estimation, $L_{contact}$ can be disregarded. The optimization process yields the transformation between the two NeuS networks, providing the pose of the rigid

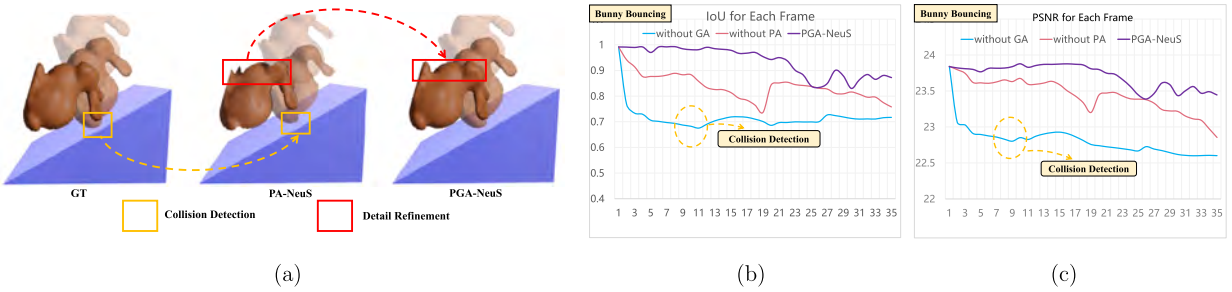

Fig. 6. Synthetic scene. (a) PAC-NeuS detects collisions and reconstructs motion sequences using a differentiable physics solver. Moreover, PGA-NeuS, which incorporates both physical and geometric constraints, produces more accurate outcomes. We evaluate the consistent IoU (b) and PSNR (c) metrics throughout the video sequence to analysis the accuracy.

body at that moment. By extending this to subsequent frames, we recover the pose for each frame and use it to guide the optimization of the rigid body's trajectory in the differentiable physics solver. This process is referred to as the geometric augmentation of PA-NeuS.

Contact constraint. We observed that the optimized poses from monocular videos can lead to floating objects during collisions (Fig. 11) rather than adhering to the contact points. The physics solver can detect this abnormal state. Although the poses from the PA process are slightly inaccurate, the PA process can reliably indicate which frames have collisions. Therefore, we store the collision status and SDF values of each frame to penalize locally optimal results caused by monocular ambiguity during pose optimization, as detailed in Eq. (10).

When optimizing object poses in monocular videos using geometric augmentation, as shown in the upper of Fig. 11, the optimization direction of the object poses always tends to move toward the camera. From the training view, where the position of the camera is fixed, this results in the optimized poses visually appearing to float. However, this state is difficult to detect in the training view but becomes visible when switching to a new perspective (the lower part of Fig. 11). This is because during the color optimization process within the mask, the optimization tends to fill the entire mask, and moving towards the camera significantly accomplishes this process, leading to erroneous optimal solutions. With the assistance of collision constraint information provided by the physics solver, we successfully solve this issue.

Physics and geometry-augmentation. We initially guide the optimization of the physics solver with the color loss for a few iterations to obtain rough poses of the rigid body for each frame $\{\{\mathbf{q}_1^*, \mathbf{T}_1^*\}, \cdots, \{\mathbf{q}_{N-1}^*, \mathbf{T}_{N-1}^*\}\}$. Subsequently, we employ the same strategy used in the initial pose estimation process, utilizing the rough pose information obtained in the previous step as initial positions to optimize and solve for accurate poses of each frame $\{\{\mathbf{q}_1, \mathbf{T}_1\}, \cdots, \{\mathbf{q}_{N-1}, \mathbf{T}_{N-1}\}\}$ as an optimization problem in Eq. (9). Finally, we use this pose information to reformulate the target loss function of the differentiable physics solver:

$$\min_{k_n, \mu, \mathbf{v}_0^c, \boldsymbol{\omega}_0^c} \quad \mathcal{L}_{\text{physics}} = \sum_{i=1}^{N-1} \boldsymbol{\omega}_i (|\mathbf{q}(k_n, \mu, \mathbf{v}_0^c, \boldsymbol{\omega}_0^c - \mathbf{q}_i|) + |\mathbf{T}(k_n, \mu, \mathbf{v}_0^c, \boldsymbol{\omega}_0^c - \mathbf{T}_i|, \tag{11}$$
$$\text{s.t.} \quad 0 < k_n < 1 \quad \text{and} \quad 0 < \mu < 1,$$

where $\mathbf{v}_0^c$ and $\boldsymbol{\omega}_0^c$ represent the initial velocity and angular velocity of the rigid body. Iterating a few steps ahead with a differentiable solver to estimate rough positions for each frame is crucial. Relying solely on the previous frame's results for the next frame's input can be unstable, especially in real-world scenarios involving gravitational acceleration, where objects can quickly gain speed and experience significant spatial displacement. This can disrupt the continuity of spatiotemporal information, leading to failures in dynamic reconstruction. Pre-establishing this relationship with a differentiable solver helps mitigate the issue.

Reconstruction of multiple rigid bodies. Trajectory reconstruction for multiple rigid bodies is a very challenging task because of the sensitivity of the collision physical parameters; small changes have a significant impact on the results of the entire experiment. In practice, the collision between two objects typically occurs within one single frame, representing only a small fraction of the video sequence, making the reconstruction process even harder. Furthermore, without an appropriate initial velocity to trigger collisions, gradient information cannot be obtained and the optimization would fail. Due to the characteristics of PGA-NeuS, we transformed the color loss into the approximate motion trajectory of the object in the geometry augmentation process. Even if there is a large deviation between the initial velocity of the object and the optimal value, we can use this trajectory information to estimate the correct initial velocity of each object, so that the collision can occur at the right moment.

Specifically, we first extract the initial trajectory ($traj_{\text{init}}$) from the first few frames and optimize the initial velocity to align it with this trajectory. Using the PA process, we then identify the collision frame, $t_{\text{collision}}$, and extract the post-collision trajectory, $traj_{\text{post}}$, to compute the velocity changes of the two rigid bodies. This information is subsequently employed to optimize the momentum exchange at the collision frame, effectively mitigating the issue of insufficient gradient information over extended motion sequences. Notably, by segmenting the trajectory in this manner, the optimization of collision parameters is confined to the moment of collision, simplifying the process and enhancing its efficiency and accuracy.

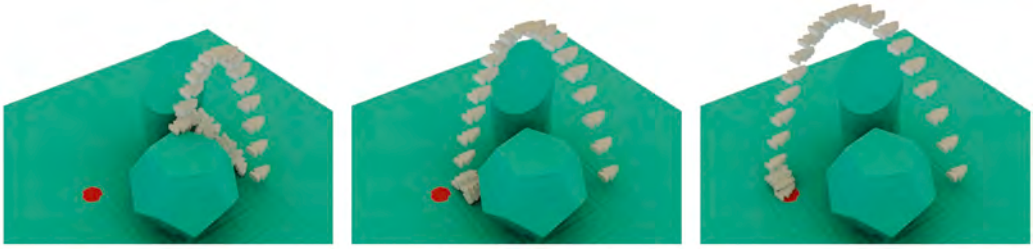

Fig. 7. Initialization (left): At the start, the object lacks the proper velocity and fails to reach the target (marked as a red point). Intermediate Iterations (middle): After several optimization steps, the resulting trajectory still can not precisely reach the red dot. Final Optimization (right): Following full optimization, the object's initial velocity is adjusted to perfection, allowing it to reach the predefined target position accurately.

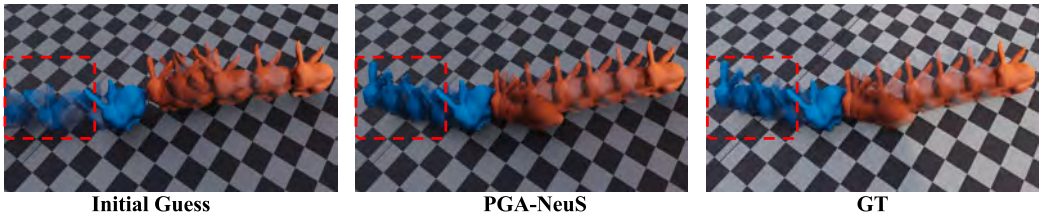

Fig. 8. Trajectories reconstruction of two rigid bodies. Initial guess (left): At the time of initialization, the initial velocity of the object and the collision parameters are wrong, the collision occurs at the wrong time (4th frame), and the trajectory after the collision is completely wrong. Optimization (middle): We first optimize the initial velocity so that the collision occurs at the right time (6th frame), and then we split the trajectory according to that frame. After optimizing the post-collision velocity, we optimize the collision parameters on the 5th frame. Finally, a trajectory consistent with the ground truth (right) is obtained.

4. Experiments

Implementation details and evaluation protocol. We implemented the PGA-NeuS pipeline using Python and PyTorch (Paszke et al., 2019), directly leveraging PyTorch's automatic differentiation capabilities for the differentiable rigid body solver. We tested PGA-NeuS on both synthetic and real-world scenes involving rigid body motion. The experiments were carried out on a single NVIDIA RTX 4090 GPU. We used NeuS (Wang et al., 2021) for the reconstruction of the static neural radiation field. Additionally, in the ablation experiments, we demonstrated the necessity of each part of our method by removing the physics augmentation process and the geometry augmentation process. We evaluated our pipeline using Peak Signal-to-Noise Ratio (PSNR) and Intersection over Union (IoU) metrics.

Datasets. The dataset for our experiment contains two components: the static component consists of multi-view images, while the dynamic component comprises monocular video, encompassing both composite and real-world captured data. Synthetic data were generated using Blender, whereas real-world data were captured through a single camera. Since our objects and scenes were reconstructed independently, we not only facilitated the reconstruction of dynamic video but also solved some other kinds of inverse problems. Our real-world dataset contains different types of rigid body movements: sliding, rolling, and collision. For some cases, we applied an initial external force to start the motion and skipped the first few frames, beginning the optimization once the rigid body reached a stable speed, with initial velocity as one of the optimization targets.

4.1. Application

In our framework, both the rendering and physics components are differentiable. This allows the rendered images to be used for computing color loss, with the resulting gradient information passed back to the object solver to tackle inverse problems involving parameter optimization.

Target Reaching: In the experiment shown in Fig. 7, we placed an object on one side of an obstacle within the scene and selected a target point using the method illustrated in Fig. 3 (right). The target point, represented by a red dot in the Fig. 7, is separated from the object by the obstacle. Our goal is to optimize the initial velocity of the object so that it successfully crosses over the obstacle and reaches the target point within 50 frames. In the initial state (Fig. 7 left), the object could not cross over the obstacle. After a few iterations (Fig. 7 middle), the object managed to clear the obstacle but misses the target. Finally, after 20 optimization steps, the object successfully reached the target position (Fig. 7 right).

Trajectory reconstruction: We tested our method on all the videos in the dataset. In order to keep time in the differentiable physics solver synchronized with the videos, the time step of each simulation is set to the frame rate of the corresponding video. Fig. 9 shows the reconstruction results of all the single-body movements. As illustrated in the figure, our method successfully reconstructs the dynamics of the objects across various scenes, accurately capturing their trajectories and interactions with the environment.

We also tested our method on a scene created in Blender, containing two objects (Fig. 9.(h)). In this experiment, we set the collision parameters of the background as ground truth and focus on optimizing the collision parameters between the two objects.

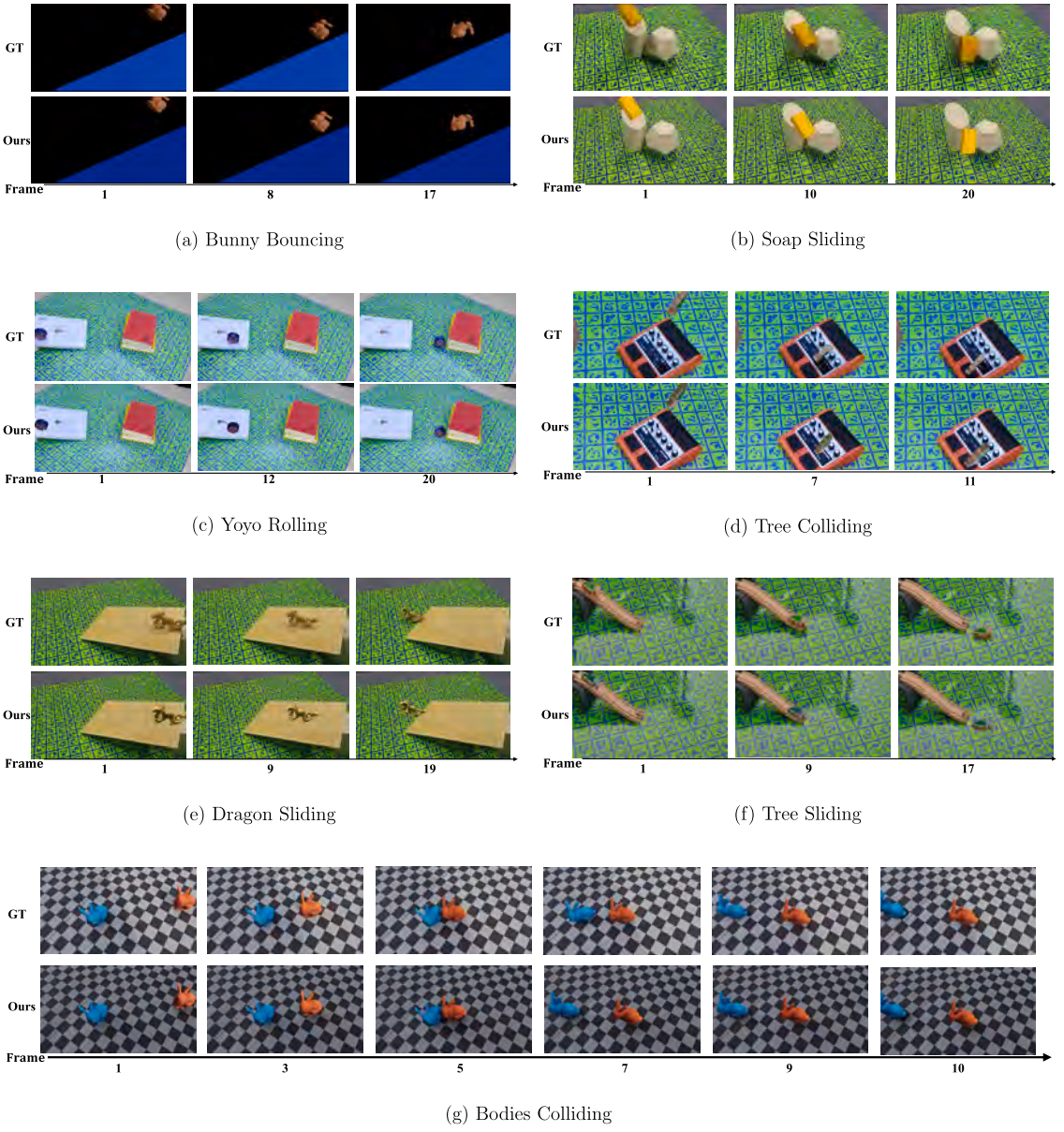

Fig. 9. We validated our method in both synthetic and real-world scenes. Each scene has a different composition of rigid body motion. Compared with reference frames, our method is capable of reconstructing and simulating the whole rigid body motion sequence.

Fig. 8 shows the experiment in more detail. We used one object to hit another object, and the collision detection between the two is completed by the SDF information provided by NeuS. We started with an initial guess to optimize the trajectories of the two bodies. In our PGA-NeuS method, we prioritized the use of object collision information to recover the motion trajectory through the GA process, thereby avoiding gradient propagation between the physical simulator and the neural renderer. We optimized the object's initial velocity. Without optimization, the frame in which the collision happened between the two objects was notably inconsistent with the ground truth. After optimizing the initial velocity data, we segmented the trajectory into two segments using the collision frame as a reference point and estimated the post-collision velocity in the same way. Finally, we optimized the momentum change of both objects at the moment of collision, thereby completing the motion trajectory.

4.2. Comparisons

PAC-NeRF (Li et al., 2023a) is a recent work that also combines a differentiable physical model with NeRF-based representations. Its physics numerical solver is built on the Material Point Method (MPM), which is specifically designed for elastic bodies. Additionally, PAC-NeRF does not model the background, making it incapable of simulating collisions with objects that lack an analytical SDF

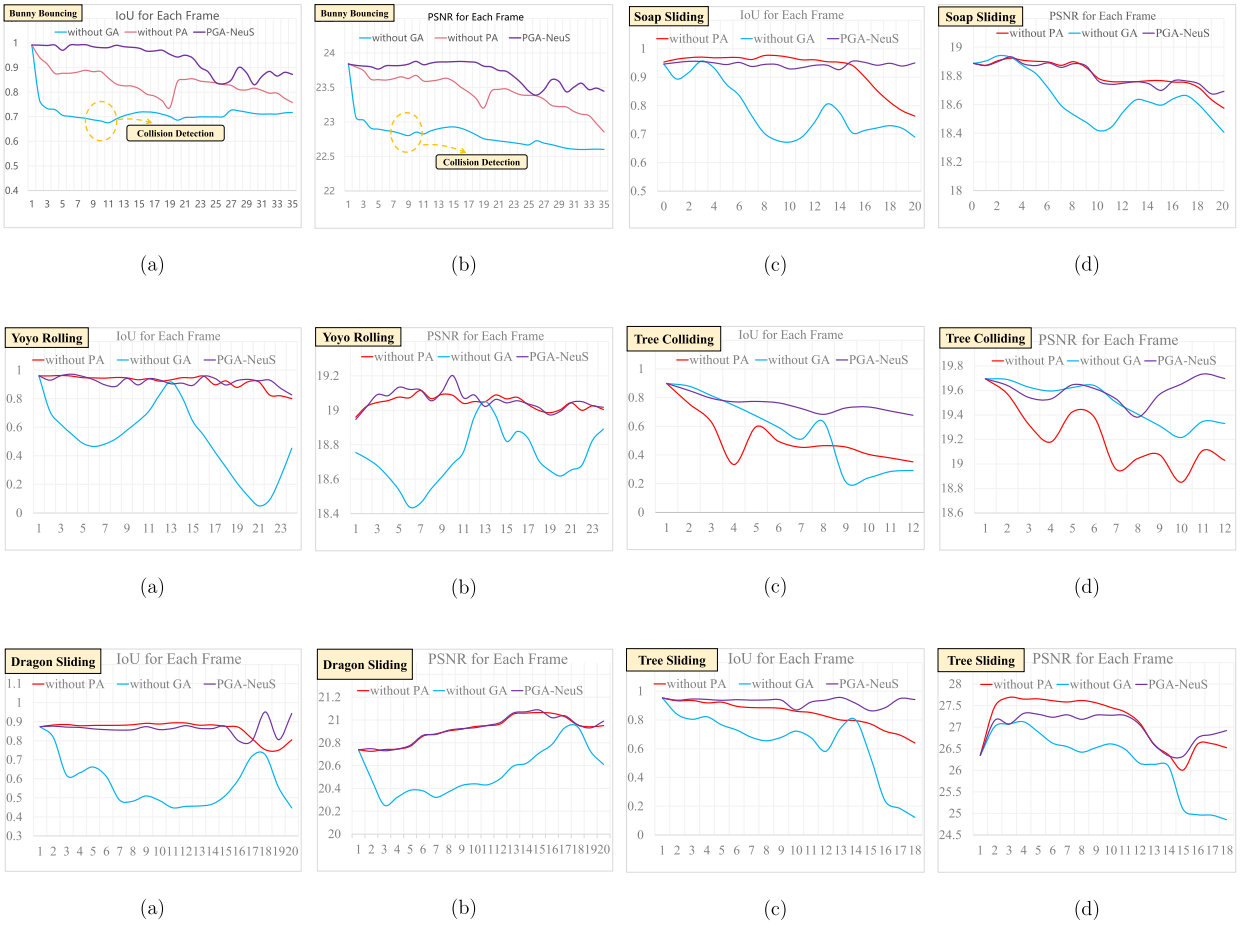

Fig. 10. The PSNR and IoU values over time for three real-world testing models show that errors accumulate as motion progresses, leading to a downward trend. The method without PA achieves good evaluation scores but produces visually incorrect results due to local minima. Conversely, without geometric augmentation, the PA method struggles to reconstruct the correct trajectory accurately.

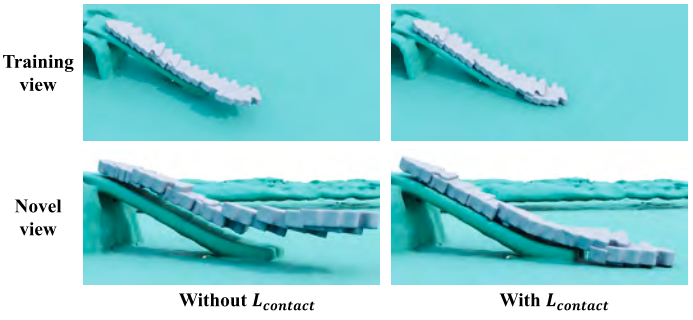

Fig. 11. Due to the tendency of geometric pose optimization to fill the entire mask, the color loss reaches a local minimum. Consequently, the object's position tends to move closer to the camera, resulting in erroneous floating outcomes. Utilizing the collision object information provided by the physics solver can ultimately optimize a more accurate and physically meaningful result.

representation. Thus, we made slight modifications to PAC-NeRF so that we can compare it. Specifically, we replaced its diff-MPM solver with a differentiable rigid body solver and also enhanced it by utilizing the SDF information from NeuS for collision handling. In the following figures, we refer to this adapted version as PAC-NeuS, which is the same as PA-NeuS in our framework. We mark it as "Without GA" in the figures.

In *synthetic data*, the scene lighting and object materials are relatively simple and ideal. As shown in Fig. 6.(a), PAC-NeuS already has sufficient capability to recover a visually reasonable and complete motion. Fig. 6. (b)-(c) shows that PAC-NeuS can still be further improved by our GA process.

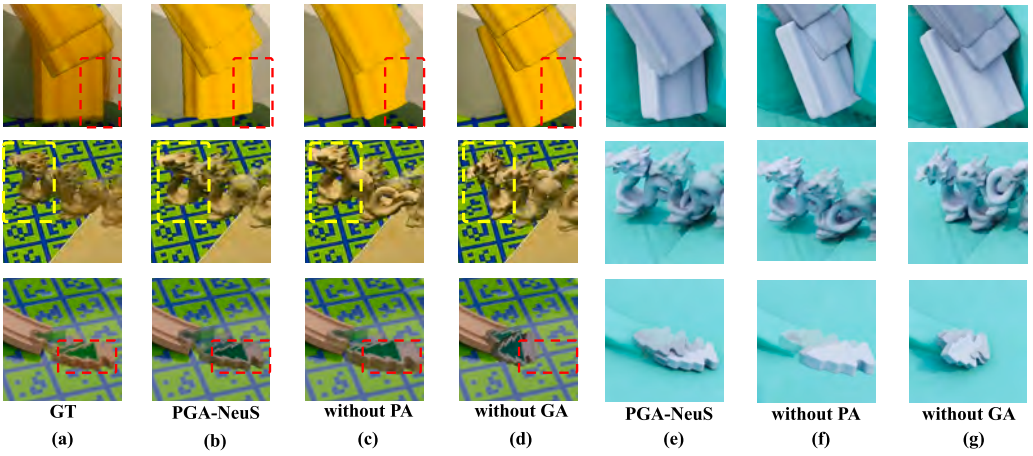

Fig. 12. We compared the object trajectories reconstructed by three methods on real-world data. PAC-NeuS generates physically realistic trajectories but struggles with accurate motion reconstruction. The method without PA achieves poses closer to the ground truth but can produce physically inaccurate results. PGA-NeuS combines the strengths of both, delivering the best performance overall.

In *real-world scenarios*, complex lighting conditions, such as highlights, reflections, and shadows, significantly hinder the accuracy of color loss calculations. This limitation reduces the effectiveness of PAC-NeuS, which relies on color loss for backpropagation, in accurately recovering motion trajectories, as shown in Fig. 12.(d) and (g). We suspect motion blur and lighting variations during capture contribute to this, as they cause pixel colors to change across frames, making it challenging to extract pose information from color loss. This leads to amplified errors in the physical process, as seen in Fig. 10. While geometrically optimized poses may outperform those from PAC-NeuS in metrics, they lack continuity across frames and can result in incorrect depth estimations, leading to physically implausible outcomes like interpenetration with background objects, as shown in Fig. 12.(c) and (f). By combining both approaches, the collision constraints provided by the physics solver can correct these physically implausible results from geometric pose optimization while also supplying more accurate pose information for the differentiable physics solver.

Penalty formulation. When optimizing object poses in monocular videos using geometric augmentation, as shown in the upper of Fig. 11, the optimization direction of the object poses always tends to move toward the camera. From the training view (the fixed position of the camera), this results in the optimized poses visually appearing to float. However, this state is difficult to detect in the training view but becomes clearly visible when switching to a new perspective (the bottom of Fig. 11). We believe this is because, during the color optimization process within the mask, the optimization tends to fill the entire mask, and moving towards the camera significantly accomplishes this process, leading to erroneous optimal solutions. With the assistance of collision constraint information provided by the physics solver, we successfully resolved this issue.

5. Conclusion and limitations

We propose PGA-NeuS, a method for reconstructing moving rigid objects from monocular videos by integrating differentiable physics and geometric constraints into neural implicit surfaces. Experiments on both synthetic and real-world scenes demonstrate that PGA-NeuS can effectively reconstruct object geometry and simulate rigid body motion from monocular video inputs.

Despite its effectiveness, the current approach has two limitations. First, it does not distinguish between different materials, which leads to reduced accuracy when objects composed of diverse materials move through varying media. This limitation could be overcome by segmenting the SDF field and assigning distinct collision parameters to each material region. Second, our multi-object experiments are still preliminary. As the number of interacting objects increases, the collision dynamics become significantly more complex, making the learning process more challenging and potentially unstable.

CRediT authorship contribution statement

Yuanmu Xu: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Project administration, Resources, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Guanli Hou:** Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Project administration, Resources, Software, Validation, Visualization, Writing – original draft, Writing – review & editing. **Jiangbei Hu:** Supervision, Writing – original draft, Writing – review & editing. **Tenglong Ren:** Conceptualization, Writing – original draft, Writing – review & editing. **Xiaokun Wang:** Supervision, Writing – review & editing. **Yalan Zhang:** Supervision, Writing – review & editing. **Xiaojuan Ban:** Supervision, Writing – review & editing. **Chen Qian:** Conceptualization, Funding acquisition, Supervision. **Fei Hou:** Project administration, Supervision, Validation. **Ying He:** Supervision, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

This work was supported in part by the Ministry of Education, Singapore, under its Academic Research Fund Grant (RT19/22) and the RIE2020 Industry Alignment Fund–Industry Collaboration Projects (IAF-ICP) Funding Initiative, as well as cash and in-kind contributions from industry partner(s). It was also supported by the Guangdong Basic and Applied Basic Research Foundation (No. 2023A1515030177), the National Natural Science Foundation of China (Nos. 62376025, 62332017), and the National Science and Technology Major Project of the Ministry of Science and Technology of China (2024D0608100).

Data availability

Dataset and source code are available at https://github.com/Raining00/PGA-NeuS.

References

Attal, B., Laidlaw, E., Gokaslan, A., Kim, C., Richardt, C., Tompkin, J., O'Toole, M., 2021. Törf: time-of-flight radiance fields for dynamic scene view synthesis. Adv. Neural Inf. Process. Syst. 34, 26289–26301.

Chen, H.-y., Tretschk, E., Stuyck, T., Kadlecek, P., Kavan, L., Vouga, E., Lassner, C., 2022. Virtual elastic objects. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 15827–15837.

Chen, Y., Chen, Z., Zhang, C., Wang, F., Yang, X., Wang, Y., Cai, Z., Yang, L., Liu, H., Lin, G., 2024. Gaussianeditor: Swift and controllable 3d editing with Gaussian splatting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 21476–21485.

Chu, M., Liu, L., Zheng, Q., Franz, E., Seidel, H.-P., Theobalt, C., Zayer, R., 2022. Physics informed neural fields for smoke reconstruction with sparse data. ACM Trans. Graph., 1–14.

de Avila Belbute-Peres, F., Smith, K., Allen, K., Tenenbaum, J., Kolter, J.Z., 2018. End-to-end differentiable physics for learning and control. Adv. Neural Inf. Process. Syst. 31.

Driess, D., Huang, Z., Li, Y., Tedrake, R., Toussaint, M., 2023. Learning multi-object dynamics with compositional neural radiance fields. In: Conference on Robot Learning. PMLR, pp. 1755–1768.

Du, T., Wu, K., Ma, P., Wah, S., Spielberg, A., Rus, D., Matusik, W., 2021a. Diffpd: differentiable projective dynamics. ACM Trans. Graph. 41, 1–21.

Du, Y., Zhang, Y., Yu, H.-X., Tenenbaum, J.B., Wu, J., 2021b. Neural radiance flow for 4d view synthesis and video processing. In: 2021 IEEE/CVF International Conference on Computer Vision (ICCV).

Feng, Y., Shang, Y., Li, X., Shao, T., Jiang, C., Yang, Y., 2024. Pie-nerf: physics-based interactive elastodynamics with nerf. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4450–4461.

Gao, C., Saraf, A., Kopf, J., Huang, J.-B., 2021. Dynamic view synthesis from dynamic monocular video. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 5712–5721.

Geilinger, M., Hahn, D., Zehnder, J., Bächer, M., Thomaszewski, B., Coros, S., 2020. Add: analytically differentiable dynamics for multi-body systems with frictional contact. ACM Trans. Graph. 39, 1–15.

Guan, S., Deng, H., Wang, Y., Yang, X., 2022. Neurofluid: fluid dynamics grounding with particle-driven neural radiance fields. In: International Conference on Machine Learning. PMLR, pp. 7919–7929.

Guédon, A., Lepetit, V., 2024. Sugar: surface-aligned Gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5354–5363.

Hahn, D., Banzet, P., Bern, J.M., Coros, S., 2019. Real2sim: visco-elastic parameter estimation from dynamic motion. ACM Trans. Graph. 38, 1–13.

Hu, Y., Anderson, L., Li, T.-M., Sun, Q., Carr, N., Ragan-Kelley, J., Durand, F., 2019. Difftaichi: differentiable programming for physical simulation. ArXiv preprint. arXiv:1910.00935.

Huang, B., Yu, Z., Chen, A., Geiger, A., Gao, S., 2024. 2d Gaussian splatting for geometrically accurate radiance fields. In: ACM SIGGRAPH 2024 Conference Papers, pp. 1–11.

Jain, A., Tancik, M., Abbeel, P., 2021. Putting nerf on a diet: semantically consistent few-shot view synthesis. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 5885–5894.

Jiang, Y., Yu, C., Xie, T., Li, X., Feng, Y., Wang, H., Li, M., Lau, H., Gao, F., Yang, Y., et al., 2024. Vr-gs: a physical dynamics-aware interactive Gaussian splatting system in virtual reality. In: ACM SIGGRAPH 2024 Conference Papers, p. 1.

Johnson, E., Habermann, M., Shimada, S., Golyanik, V., Theobalt, C., 2023. Unbiased 4d: monocular 4d reconstruction with a neural deformation model. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 6598–6607.

Kairanda, N., Habermann, M., Theobalt, C., Golyanik, V., 2023. Neuralclothsim: neural deformation fields meet the Kirchhoff-Love thin shell theory. ArXiv preprint. arXiv:2308.12970.

Kajiya, J.T., Von Herzen, B.P., 1984. Ray tracing volume densities. In: Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques - SIGGRAPH '84.

Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G., 2023. 3d Gaussian splatting for real-time radiance field rendering. ACM Trans. Graph. 42, 139.

Le Cleac'h, S., Yu, H.-X., Guo, M., Howell, T., Gao, R., Wu, J., Manchester, Z., Schwager, M., 2023. Differentiable physics simulation of dynamics-augmented neural objects. IEEE Robot. Autom. Lett. 8, 2780–2787.

Li, T., Slavcheva, M., Zollhoefer, M., Green, S., Lassner, C., Kim, C., Schmidt, T., Lovegrove, S., Goesele, M., Newcombe, R., et al., 2022a. Neural 3d video synthesis from multi-view video. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5521–5531.

Li, X., Qiao, Y.-L., Chen, P.Y., Jatavallabhula, K.M., Lin, M., Jiang, C., Gan, C., 2023a. Pac-nerf: physics augmented continuum neural radiance fields for geometry-agnostic system identification. ArXiv preprint. arXiv:2303.05512.

Li, Y., Du, T., Wu, K., Xu, J., Matusik, W., 2022b. Diffcloth: differentiable cloth simulation with dry frictional contact. ACM Trans. Graph. 42, 1–20.

Li, Z., Fu, K., Wang, H., Wang, M., 2023b. Pi-nerf: a partial-invertible neural radiance fields for pose estimation. In: Proceedings of the 31st ACM International Conference on Multimedia, MM '23. Association for Computing Machinery, New York, NY, USA, pp. 7826–7836.

Li, Z., Müller, T., Evans, A., Taylor, R.H., Unberath, M., Liu, M.-Y., Lin, C.-H., 2023c. Neuralangelo: high-fidelity neural surface reconstruction. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

Long, X., Lin, C., Wang, P., Komura, T., Wang, W., 2022. Sparseneus: fast generalizable neural surface reconstruction from sparse views. In: European Conference on Computer Vision. Springer, pp. 210–227.
Lorensen, W.E., Cline, H.E., 1998. Marching cubes: a high resolution 3d surface construction algorithm. In: Seminal Graphics: Pioneering Efforts That Shaped the Field, pp. 347–353.
Lyu, X., Sun, Y.-T., Huang, Y.-H., Wu, X., Yang, Z., Chen, Y., Pang, J., Qi, X., 2024. 3dgsr: implicit surface reconstruction with 3d Gaussian splatting. ACM Trans. Graph. 43, 1–12.
McNamara, A., Treuille, A., Popović, Z., Stam, J., 2004. Fluid control using the adjoint method. ACM Trans. Graph. 23, 449–456.
Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R., 2020. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis, pp. 405–421.
Müller, T., Evans, A., Schied, C., Keller, A., 2022. Instant neural graphics primitives with a multiresolution hash encoding. ACM Trans. Graph. 41, 1–15.
Oechsle, M., Peng, S., Geiger, A., 2021. Unisurf: unifying neural implicit surfaces and radiance fields for multi-view reconstruction. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 5589–5599.
Olesik, J.W., 1991. Elemental analysis using icp-oes and icp/ms. Anal. Chem. 63, 12A–21A.
Park, K., Sinha, U., Barron, J.T., Bouaziz, S., Goldman, D.B., Seitz, S.M., Martin-Brualla, R., 2021. Nerfies: deformable neural radiance fields. In: 2021 IEEE/CVF International Conference on Computer Vision (ICCV).
Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al., 2019. Pytorch: An Imperative Style, High-Performance Deep Learning Library. Advances in Neural Information Processing Systems, vol. 32, pp. 8026–8037.
Peng, S., Niemeyer, M., Mescheder, L., Pollefeys, M., Geiger, A., 2020. Convolutional occupancy networks. In: Computer Vision–ECCV 2020: 16th European Conference, Proceedings, Part III 16. Glasgow, UK, August 23–28, 2020. Springer, pp. 523–540.
Peng, S., Zhang, Y., Xu, Y., Wang, Q., Shuai, Q., Bao, H., Zhou, X., 2021. Neural body: implicit neural representations with structured latent codes for novel view synthesis of dynamic humans. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 9054–9063.
Pumarola, A., Corona, E., Pons-Moll, G., Moreno-Noguer, F., 2021. D-nerf: neural radiance fields for dynamic scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10318–10327.
Qiao, Y.-L., Gao, A., Xu, Y., Feng, Y., Huang, J.-B., Lin, M.C., 2023. Dynamic mesh-aware radiance fields. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 385–396.
Röhrbein, F., Uchibe, E., 2021. A differentiable physics engine for deep learning in robotics. In: Frontiers in Neurorobotics–Editor's Pick 2021.
Rosu, R.A., Behnke, S., 2023. Permutosdf: fast multi-view reconstruction with implicit surfaces using permutohedral lattices. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 8466–8475.
Schenck, C., Fox, D., 2018. Spnets: differentiable fluid dynamics for deep neural networks. In: Conference on Robot Learning. PMLR, pp. 317–335.
Segal, A., Haehnel, D., Thrun, S., 2009. Generalized-icp. In: Robotics: Science and Systems, vol. 2. Seattle, WA, p. 435.
Sun, C., Sun, M., Chen, H.-T., 2022. Direct voxel grid optimization: super-fast convergence for radiance fields reconstruction. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5459–5469.
Tretschk, E., Tewari, A., Golyanik, V., Zollhofer, M., Lassner, C., Theobalt, C., 2021a. Non-rigid neural radiance fields: reconstruction and novel view synthesis of a dynamic scene from monocular video. In: 2021 IEEE/CVF International Conference on Computer Vision (ICCV).
Tretschk, E., Tewari, A., Golyanik, V., Zollhöfer, M., Lassner, C., Theobalt, C., 2021b. Non-rigid neural radiance fields: reconstruction and novel view synthesis of a dynamic scene from monocular video. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 12959–12970.
Wang, C., Wu, X., Guo, Y.-C., Zhang, S.-H., Tai, Y.-W., Hu, S.-M., 2022. Nerf-sr: high quality neural radiance fields using supersampling. In: MM '22. Association for Computing Machinery, New York, NY, USA, pp. 6445–6454.
Wang, P., Liu, L., Liu, Y., Theobalt, C., Komura, T., Wang, W., 2021. Neus: learning neural implicit surfaces by volume rendering for multi-view reconstruction. In: NeurIPS.
Wang, Y., Han, Q., Habermann, M., Daniilidis, K., Theobalt, C., Liu, L., 2023. Neus2: fast learning of neural implicit surfaces for multi-view reconstruction. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 3295–3306.
Wu, G., Yi, T., Fang, J., Xie, L., Zhang, X., Wei, W., Liu, W., Tian, Q., Wang, X., 2023. 4d Gaussian splatting for real-time dynamic scene rendering. ArXiv preprint. arXiv:2310.08528.
Xie, T., Zong, Z., Qiu, Y., Li, X., Feng, Y., Yang, Y., Jiang, C., 2024. Physgaussian: physics-integrated 3d Gaussians for generative dynamics. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 4389–4398.
Xu, Q., Xu, Z., Philip, J., Bi, S., Shu, Z., Sunkavalli, K., Neumann, U., 2022. Point-nerf: point-based neural radiance fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 5438–5448.
Yang, B., Bao, C., Zeng, J., Bao, H., Zhang, Y., Cui, Z., Zhang, G., 2022. Neumesh: learning disentangled neural mesh-based implicit field for geometry and texture editing. In: European Conference on Computer Vision. Springer, pp. 597–614.
Yariv, L., Gu, J., Kasten, Y., Lipman, Y., 2021. Volume rendering of neural implicit surfaces. Adv. Neural Inf. Process. Syst. 34, 4805–4815.