2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

SoftNeRF: A Self-Modeling Soft Robot Plugin for Various Tasks

Jiwei Shan[1*] Yirui Li[2*] Qiyu Feng[1] Ditao Li[1] Lijun Han[1] Hesheng Wang[1]

***Abstract*— Building a self-model for robots, enabling them to simulate their physical selves and predict future states without direct interaction with the physical world, is crucial for robot motion planning and control. Existing self-modeling methods primarily focus on rigid robots and typically require significant time, effort, and resources to gather training data. In this study, we introduce SoftNeRF, a self-supervised visual self-model designed for soft robots. We use a hybrid neural shape representation based on the Signed Distance Function (SDF) to capture both the geometry and complex nonlinear motion of soft robots. By leveraging differentiable rendering, our method learns a self-model from readily available RGB images, similar to how humans understand their physical state through reflection. To improve training efficiency and model accuracy, we propose an error-guided adaptive sampling strategy. SoftNeRF can serve as a plug-in for various downstream tasks, even when trained with data unrelated to those tasks. We demonstrate SoftNeRF's ability to support shape prediction and motion planning for robots in both simulated and real-world environments. Furthermore, SoftNeRF excels in detecting and recovering from damage, thereby enhancing machine resilience. Code is available at: https://github.com/IRMVLab/soft-nerf.**

I. INTRODUCTION

Bodily self-awareness is a unique cognitive ability found in humans and a few other species [1], [2]. Throughout growth, humans understand their physical form through various channels and continuously update their self-models [3], [4]. These self-models enable us to visualize future outcomes in different situations, aiding decision-making. For instance, when navigating a narrow alley, we use our self-model to visualize moving through it and assess our physical state. Based on observed information, we assess our ability to pass smoothly, avoiding the embarrassment of getting stuck. Similarly, creating a self-model for robots allows them to simulate and predict their states without direct interaction with the physical world. This capability is vital for planning and controlling robot motion [5], [6]. Developing a robot's self-model detaches it from specific tasks, ensuring versatility and adaptability. Additionally, robots can observe and update their status through multiple channels, enabling continuous self-improvement and lifelong learning [5].

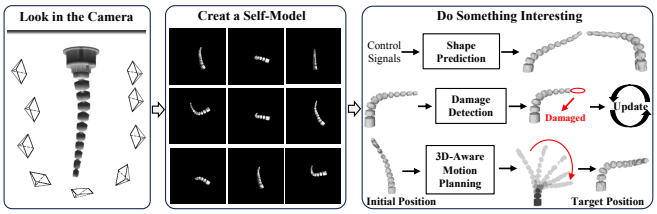

Fig. 1. Visual Self-Modeling of Soft Robots. We equip soft robots with the capability to autonomously model their kinematics and morphology in 3D space using only actuation signals. This process, known as visual self-modeling, learns directly from readily available RGB images. Visual self-models enable soft robots to simulate their states without direct interaction with the physical world. Consequently, they can perform downstream tasks as plug-ins, including damage detection, recovery, and motion planning.

Over the past decade, the challenge of efficiently constructing robot self-models has captivated researchers, leading to significant expansion in related literature [5]–[7]. These studies aim to enable robots to autonomously develop and dynamically update their self-models, similar to active reasoning models from neuroscience [8], [9]. Researchers are also exploring the application of robot self-models in tasks such as motion control and damage detection to assess their versatility [5]–[7]. Despite great progress, challenges persist. Firstly, most self-modeling approaches are designed for robots primarily composed of rigid parts, where component geometry generally remains static [10]. In contrast, soft robots, with dynamically changing shapes due to actuation and external forces, present a different challenge. Furthermore, the near-infinite degrees of freedom associated with soft robots complicate self-modeling efforts [11]. In recent years, data-driven self-modeling has gained popularity as a leading approach [5], [7]. However, collecting the necessary training data and obtaining accurate labels can be resource-intensive, requiring significant time, effort, and expense. Our experiences, such as learning about our physical state by looking in a mirror, suggest a simpler and more cost-effective method for self-modeling. This observation raises an interesting research question:

Is it possible to endow soft robots with morphological self-awareness and adaptability using a straightforward and efficient approach, akin to "looking in a mirror"?

In this study, we introduce SoftNeRF, a novel visual self-modeling approach for soft robots. Our method uses SDF-based neural explicit-implicit representations to accurately capture both the geometry and complex nonlinear behaviors of soft robots. The SoftNeRF learning process requires only control signals and observed RGB images as inputs, similar to how humans learn from reflections, thus eliminating the

*The first two authors contributed equally. This work was supported in part by the Natural Science Foundation of China under Grant 62361166632, 62225309, 62073222, and U21A20480; was partially supported by a grant from the NSFC/RGC Joint Research Scheme sponsored by the Research Grants Council of the Hong Kong Special Administrative Region, China and the National Natural Science Foundation of China (Project No. N_CUHK410/23). Corresponding Author: Hesheng Wang, Lijun Han.

[1]Department of Automation, Key Laboratory of System Control and Information Processing of Ministry of Education, Key Laboratory of Marine Intelligent Equipment and System of Ministry of Education, Shanghai Engineering Research Center of Intelligent Control and Management, Shanghai Jiao Tong University, Shanghai 200240, China

[2]School of mechanical engineering, Shanghai Jiao Tong University, Shanghai 200240, China

need for manual annotations. This reduces training costs and improves the scalability and generalizability of training samples. Although our model training uses task-independent data, SoftNeRF can be easily integrated into various downstream tasks as a plug-in. We verify SoftNeRF's accuracy in simulation environments and with real soft robots, demonstrating excellent performance in three downstream tasks: shape reconstruction, damage detection and recovery, and motion planning. Our contributions are as follows:

1) We present a novel method for creating a self-model of soft robots using a hybrid representation. This approach accurately captures the robot's geometry and complex nonlinear motion through a combination of explicit grids and neural implicit networks.
2) We develop a self-supervised learning framework for this visual self-model using differentiable rendering. This framework learns directly from RGB images without requiring manual annotations.
3) We introduce an error-guided adaptive sampling strategy that enhances training efficiency and improves the accuracy of modeling small terminal structures.
4) We demonstrate the effectiveness of our proposed visual self-model in both simulated and real-world soft robotics scenarios across various tasks.

II. PROPOSED METHOD

For the soft robot, our system initially receives control signals from the actuators and corresponding RGB images $\{I_i\}_{i=1}^{M}$, where M is the number of RGB images taken from various views under a specific control signal. In this paper, we use cable-driven soft robots, so the control signals can be represented as $\{l_i\}_{i=1}^{N}$, where l_i denotes the lengths of the cables, with N representing the total number of cables. Our goal is to develop a query-based visual self-model named SoftNeRF. This model allows us to estimate the occupancy of any point in three-dimensional space based on the actuators' control signals, thus achieving the robot's morphological self-awareness. Once trained, SoftNeRF can serve as a plug-in for various downstream tasks.

Fig. 2 provides an overview of SoftNeRF. For system input, we preprocess the multi-view RGB images $\{I_i\}_{i=1}^{M}$ and estimate each camera's pose using COLMAP [12]. The captured images contain many background pixels, but our objective is to model the soft robot. Therefore, we use either color segmentation or a pre-trained segmentation model [13] to create a binary mask, effectively isolating the robot from its background. Our methodology leverages the foundational principles of neural radiance fields (NeRF) [14]. We first review the key technologies behind NeRF in Sec. II-A. Then, we introduce our kinematic-aware SDF hash grids module in Sec. II-B. The error-guided adaptive sampling strategy is explained in Sec. II-C. Finally, we provide a detailed description of the model's optimization process in Sec. II-D.

A. Preliminaries: Neural Radiance Fields

Neural Radiance Fields (NeRF) [14], a pioneering work in neural rendering, have experienced rapid development in recent years. Specifically, NeRF employs a Multi-Layer Perceptron (MLP) to model a scene continuously from a set of multi-view images with known camera intrinsics and poses. It maps each 3D point x and its viewing direction d to its corresponding directional emitted radiance c and volume density σ, via a function $F_\Theta : (x, d) \rightarrow (c, \sigma)$, where Θ denotes the learnable parameters of the model. The color $C(r)$ of a pixel along a camera ray r is determined by integrating the radiance from sampled points along the ray through volume rendering. Given the camera's origin o and direction d, points are sampled at $p_i = o + h_i d$, to calculate the expected color $C(r)$:

$$C(r) = \sum_{i=1}^{N} T_i \alpha_i c_i. \tag{1}$$

Here, N indicates the number of sampling points along the ray, T_i represents the transmittance to point i, and α_i is the activation function. The formulas for calculating these are:

$$T_i = \prod_{j=1}^{i-1}(1 - \alpha_j), \quad \alpha_i = 1 - \exp(-\sigma_i \delta_i), \tag{2}$$

where δ_i signifies the distance between adjacent sampling points. NeRF parameters Θ are optimized by minimizing the color reconstruction loss:

$$L_{color} = \frac{1}{R} \sum_{r \in R} \left\| \hat{C}(r) - C_{gt}(r) \right\|_2^2. \tag{3}$$

Here, R denotes the set of sampled pixels/rays in the minibatch, and $C_{gt}(r)$ is the ground truth color of ray r.

B. Kinematic-Aware SDF Hash Grids

The implicit representation used in NeRF [14] offers advantages due to its continuous and compact nature. However, the high computational demands lead to extended training and rendering times. While hybrid representations accelerate the training and inference processes of NeRF [15]–[18], they fail to comprehensively capture vital kinematic information for soft robots. To address these issues, we propose kinematic-aware SDF hash grids that integrate a spatial geometry encoder, kinematic encoder, SDF decoder, and color decoder. The module is illustrated in Fig. 2.(c).

In the kinematic encoding module, control signals A from the actuators are input into a kinematic encoder network K_ϕ, which extracts the corresponding kinematic feature, denoted as $F_k(A)$. Inspired by Instant-NGP [15], the spatial geometry encoding module employs multi-resolution hash feature grids $G_\alpha = \{V_\alpha^l\}_{l=1}^{L}$ to encode the robot's geometric attributes. The spatial resolution of each layer is progressively set from the coarsest R_{min} to the finest R_{max}. For a 3D point x, we first identify the eight adjacent feature vertices of this point at each resolution. The feature vector at each resolution l is then derived through trilinear interpolation. Finally, we concatenate the features obtained at each resolution to form the final spatial geometric feature $F_g(x)$.

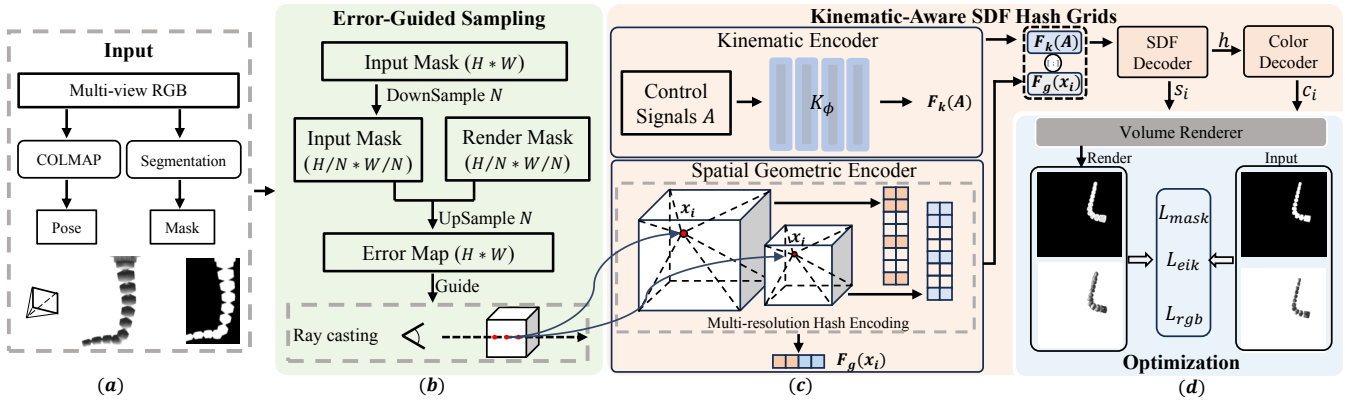

Fig. 2. Overall framework of SoftNeRF. For processing multi-view images, we utilize COLMAP [12] to estimate the camera's intrinsic parameters and pose and apply a segmentation algorithm to extract the robot mask. To enhance training efficiency, we compute an error map using the rendered and input masks. Pixels are then selectively sampled from the training set via an error-guided adaptive sampling strategy. By integrating the camera pose and intrinsic parameters, ray casting is executed, followed by sampling along the rays. The sampling points extract spatial features through a spatial geometry encoder, while the control signals associated with the sampling points extract kinematic features through a kinematic encoder. By concatenating $\mathbf{F}_g$ and $\mathbf{F}_k$, we predict the SDF and color values using the SDF and color decoders, respectively. The final step involves calculating the rendered RGB and mask through differentiable volume rendering. These outputs are compared with the input RGB and mask to construct the loss functions. Finally, we optimize the entire model by minimizing the global loss function.

We then concatenate the spatial geometry features $F_g(x)$ with the kinematic features $F_k(A)$. These concatenated features are subsequently fed into the SDF decoder D_τ to predict the SDF value s and generate a feature vector h:

$$(s, h) = D_\tau(F_g(x), F_k(A)) \tag{4}$$

Following [15], the feature vector h, the sampling point position p, and the viewing direction v are input into the color decoder D_γ to estimate the directional emitted radiance c at the sampling point. This process can be expressed as:

$$c = D_\gamma(p, v, h) \tag{5}$$

Here, $\Theta = \{\phi, \alpha, \tau, \gamma\}$ are learnable parameters.

C. Error-Guided Adaptive Sampling Strategy

In the segmentation mask, pixels representing the robot body and the background are severely imbalanced, which negatively impacts both training efficiency and the speed of model convergence. To tackle this challenge, we implement an error-guided adaptive sampling strategy to enhance sampling efficiency and accelerate network convergence. The procedure is illustrated in Fig. 2.(b).

Initially, we perform several iterations to coarsely train SoftNeRF, which aids in rendering multi-view mask images of the robot. To improve rendering efficiency, we first downsample the original input masks M_{ori} at a resolution of $H \times W$ to produce masks M_{down} at a resolution of $H/N \times W/N$. Mask images M_{render} at this reduced resolution are then rendered using the coarsely trained SoftNeRF. We calculate a reconstruction error map E_{down} by comparing M_{render} with M_{down}, and then upsample E_{down} back to the original resolution of $H \times W$. Utilizing this error map, we compute the Probability Density Function (PDF) for each pixel, where pixels with higher residuals are more likely to be sampled.

D. Volume Rendering and Optimization

Volume rendering of SDF. Building on recent advancements in neural rendering [19], we optimize our model through differentiable volume rendering. We start by randomly selecting an image from the training set and apply the error-guided adaptive sampling strategy to sample a batch of pixels. To render a pixel's color, we cast a ray r from the camera's origin o through the pixel in its normalized viewing direction v. Using a hierarchical sampling method [20], we sample Q points along each ray. We then extract the spatial geometric features F_g and kinematic features F_k of these samples. The SDF values s_i and color values c_i for each point are predicted according to equations (4) and (5). We adopt the SDF-based volume rendering technique from [20] to convert SDF values s_i into opacity values α_i:

$$\alpha_i = \max(0, \frac{\Phi_s(s_i) - \Phi_s(s_{i+1})}{\Phi_s(s_i)}). \tag{6}$$

where Φ_s is the Sigmoid function. Then, the pixel color of each ray can be computed via (2) and (1).

Loss Functions. For training SoftNeRF, our goal is to minimize the difference between the rendered colors and the ground truth colors without relying on 3D supervision. Additionally, we use binary masks for further supervision. Specifically, for the color loss L_{color}, we follow the original NeRF [14] to construct the color reconstruction loss, as shown in (3). We also add the Eikonal term [21] to the sampling points to regularize the SDF values:

$$\mathcal{L}_{reg} = \frac{1}{QR} \sum_{k,i} \left(\|\nabla s_{k,i}\|_2 - 1 \right)^2. \tag{7}$$

where R is the batch size. Following [20], the mask loss L_{mask} is defined as:

$$L_{mask} = \text{BCE}(M_k, \hat{O}_k). \tag{8}$$

Here, $\hat{O}_k = \sum_{i=1}^{Q} T_{k,i}\alpha_{k,i}$ represents the cumulative weights along the camera ray, where BCE denotes the binary cross-entropy loss. The global loss function is formulated as:

$$L = \lambda_{color} L_{color} + \lambda_{reg} L_{reg} + \lambda_{mask} L_{mask}. \tag{9}$$

Here, $\{\lambda_{color}, \lambda_{reg}, \lambda_{mask}\}$ are the weighting coefficients.

TABLE I

DEFINITIONS OF 3D METRICS USED FOR EVALUATION OF SHAPE PREDICTION QUALITY.

3D Metric	Definition
Acc.	$\frac{1}{\lvert P\rvert}\sum_{p\in P}\min_{q\in Q}\lVert p-q\rVert$
Comp.	$\frac{1}{\lvert Q\rvert}\sum_{q\in Q}\min_{p\in P}\lVert p-q\rVert$
Comp. Ratio	$\frac{1}{\lvert Q\rvert}\sum_{q\in Q}\min_{p\in P}\lVert p-q\rVert < 2cm$

III. EXPERIMENTS

A. Experimental settings

Simulation Dataset. To collect the simulation dataset, we employ a modified version of the Trunk environment from SofaGym [22], which contains a cable-driven soft robot. We control the robot using four cables and move the camera around it, capturing multi-view images with a resolution of 1300x1300 pixels. After each round of image capture, we send a new drive signal to the robot to initiate its movement. In total, we collect 123 driving signals, each corresponding to 65 RGB images from various views. We select 102 driving signals and their associated RGB images as the training set, reserving the remaining 21 sets for testing. Using color segmentation, we obtain binary masks from the distinct grayscale values of the robot and its background. Finally, we estimate camera poses using COLMAP.

Metrics. We evaluate our method's performance by examining the quality of three-dimensional reconstructions. Following common practice [23], [24], we use **Accuracy**, **Completeness**, and **Completion Ratio** as evaluation metrics. Their detailed definitions are provided in Table I, where p and q represent point clouds sampled from the ground truth mesh and the reconstructed mesh, respectively.

Baselines. The work most closely related to ours is [5]. However, this approach requires five RGB-D cameras to collect training data and then computes point clouds and surface normals to supervise network training, which entails cumbersome data collection and processing [25], [26]. Subsequently, [19] extended [5] by using RGB images as the supervision signal, which is similar to our approach. Therefore, we use [19] as our baseline. Since the original code is not publicly available, we have faithfully reimplemented the method described in [19]. For simplicity, we refer to this reimplemented method as TRBST in our discussion.

Implementation details. The kinematic encoder and SDF decoder are both four-layer MLPs with 128 channels in each hidden layer. The color decoder is a two-layer MLP with 64 channels in each hidden layer. For the error-guided adaptive sampling strategy, we start by using the robot's mask information for supervision and conduct coarse training over 10,000 iterations. The downsampling and upsampling factors are set to $N = 20$. During each epoch, the sampled pixels are divided into two groups: 64 pixels sampled via the Probability Density Function (PDF) derived from the error map, and 192 pixels randomly sampled according to the original NeRF [14]. For each camera ray, we perform uniform sampling of 64 points and importance sampling of another 64 points, following [20]. Our hash encoding resolution ranges from 2^5 to 2^{11} across 16 levels, with each hash entry having a channel size of 8. Following [27], we progressively activate the hash grid from coarse to fine throughout the optimization process. The weighting coefficients for the global loss function are set as $\lambda_{\text{color}} = 1$, $\lambda_{\text{reg}} = 0.1$, and $\lambda_{\text{mask}} = 0.1$. We train the models for 100,000 iterations on a single NVIDIA Quadro RTX 6000 GPU using the Adam optimizer.

TABLE II

SHAPE PREDICTION RESULTS ON SIMULATION DATASET. ↑ INDICATES THAT HIGHER VALUES REPRESENT GREATER ACCURACY.

Methods	Acc. (cm)↓	Comp. (cm)↓	Comp.Ratio↑
TRBST [19]	0.314	0.915	90.112
w/o Error-Guide	**0.232**	0.803	93.732
Ours	0.241	**0.766**	**95.375**

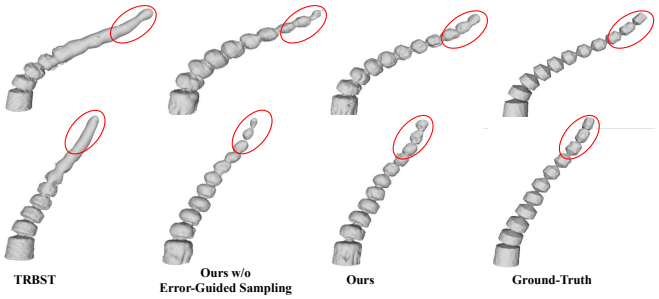

Fig. 3. Comparison of Soft Robot Shape Prediction Results. We present 3D visualization results comparing the shape predictions of TRBST [19] with two configurations of SoftNeRF (ours). Even in the absence of the error-guided sampling strategy, our method still surpasses TRBST [19]. By incorporating the error-guided sampling method, our approach achieves greater accuracy in predicting the end of the soft robot.

B. Task1: Shape Prediction for Soft Robots

This task assesses the accuracy of SoftNeRF in predicting the shapes of soft robots. Table II presents the quantitative results. Our method outperforms the baseline method TRBST [19], regardless of whether the error-guided sampling strategy is used. This indicates that our visual self-model enables the prediction of more precise and comprehensive shapes of soft robots. Moreover, in the absence of the error-guided sampling strategy, there is a decline in both completeness and the completion ratio, underscoring the strategy's efficacy and significance. Fig. 3 compares the predicted meshes from various methods against the ground truth. The TRBST [19] struggles to capture the details between segments of the soft robot. Without the error-guided sampling strategy, our method shows deviations at the ends of the predicted 3D mesh. The full model ensures an accurate and comprehensive visualization of the robot's morphology.

We further examine the convergence rates of the different methods. As illustrated in Fig. 4, our method achieves a higher completion ratio within the same number of iterations, even without error-guided sampling. This indicates that our hash grid-based method captures the soft robot's morphology more efficiently and quickly. Incorporating the error-guided sampling strategy further enhances performance.

C. Task2: Damage Detection and Recovery

Detecting damage and self-updating are crucial for robots' lifelong learning. Humans can become aware of changes in

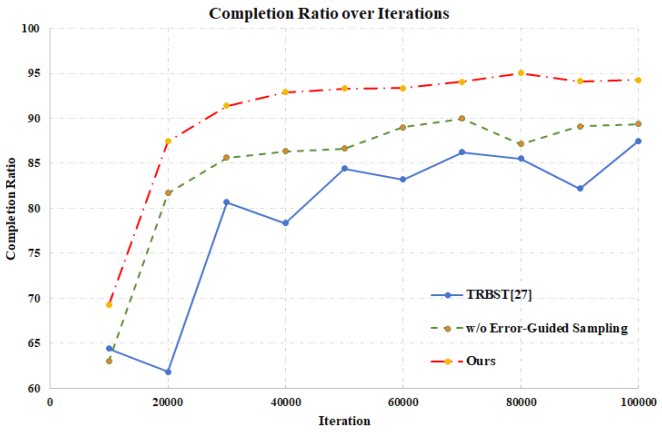

Fig. 4. Comparison of Convergence Speed of Different Methods/Configurations. We present the completion ratio for predicting the shapes of soft robots at different iterations. Our approach achieves faster convergence, and incorporating an error-guided sampling strategy further enhances the convergence speed of our method.

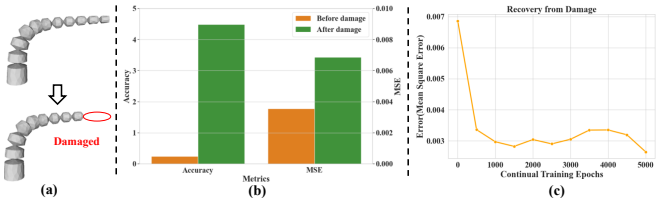

Fig. 5. Damage Detection and Recovery. (a) The soft robot is damaged, resulting in the loss of its two terminal segments. (b) We collect new observational data and use the current self-model to identify the damage by comparing the original and current prediction errors from both 2D and 3D perspectives. (c) We illustrate the update process of the visual self-model, ensuring it accurately reflects the soft robot's current morphology.

their self-model by looking in the mirror. When the state we imagine ourselves to be in, based on our current self-model, diverges from how we see ourselves in the mirror, we infer that our self-model has changed and update it accordingly. We leverage SoftNeRF to accomplish this task for soft robots.

As shown in Fig. 5.(a), the end of the soft robot is damaged. To detect this damage, as illustrated in Fig. 5.(b), we first use the current SoftNeRF to predict the robot's present 3D model and 2D images. The process then focuses on analyzing the 3D prediction error and the 2D rendering error. A significant increase in the current prediction error compared to the model's original prediction error suggests a change in the robot's body shape [5]. The next step is to quickly recover from the detected damage using new observation data. Following [5], we first collect RGB images that reflect the current state of the self-model through random motions, utilizing these images as training data to iteratively refine and update SoftNeRF. The feasibility of this method is due to the self-supervised strategy, which enables straightforward collection of self-model training data. Fig. 5(c) shows the performance of the intermediate model every 1000 epochs on the test set. It is evident that our visual self-model can be updated after 5000 iterations.

D. Task3: 3D-Aware Motion Planning

We apply SoftNeRF to the task of 3D shape servoing for soft robots, which requires aligning the robot's initial body shape with a target shape.

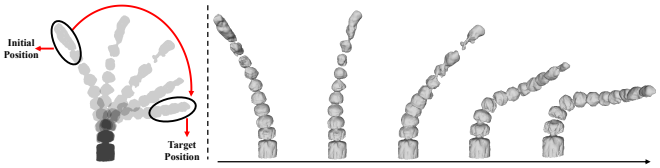

Fig. 6. 3D Self-Aware Motion Planning Results. Our objective is to enable a soft robot to transition from its initial shape to the target shape utilizing visual self-modeling.

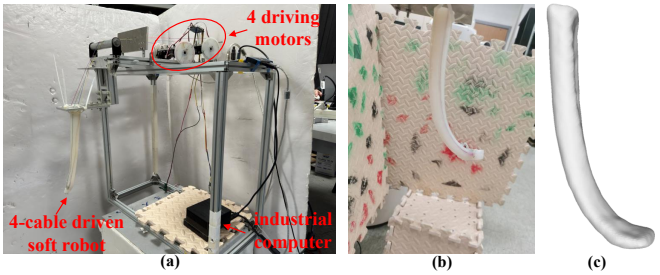

Fig. 7. (a) Overview of the soft robot system deployed in real-world experiments. (b) An example real robot picture for the experiments. (c) 3D mesh reconstruction of the soft robot using our proposed method.

According to the definition of the Signed Distance Function, the SDF value of a 3D point located on the object's surface medial axis is minimal (negative). Therefore, our optimization goal is to identify a set of cable lengths that minimizes the cumulative SDF output for all sampled target points along the target shape's centerline. Initially, we randomly generate 1000 sets of cable lengths. These sets, along with the sampled points, serve as inputs for SoftNeRF, which then calculates the sum of SDF values for the sampled points. By comparing these sums, we identify the set of cable lengths with the minimum sum of SDF values. Subsequently, we introduce minor variations around this optimal cable length to generate new sets of cable lengths. Through this iterative process, we find the set of cable lengths with the minimal sum of SDF values, continuing until the sum falls below a specified threshold, thus determining the optimal cable lengths for our goal. Fig. 6 illustrates the change in the soft robot's shape during this process.

SoftNeRF demonstrates significant generalization capabilities for this task, eliminating the need for task-specific training. It also does not require the construction of a kinematic model for the robot or the use of explicit geometric models, providing a more straightforward solution.

E. Real Soft Robot Experiment

We further validate our method on a real soft robot. Fig. 7.(a) illustrates our testing platform. The silicone-made soft robot is driven by four embedded cables that extend through its body from the proximal to the distal end. The lengths of the four cables are controlled by four independent motors, which are driven by PWM signals. The robot's state is adjusted by rotating the winches connected to each cable. Given the lengths of the four cables, when the robot reaches a stable position, we use an HONOR 30 Pro smartphone to record video footage around the soft robot. We collect 21 sets of signals along with their corresponding videos.

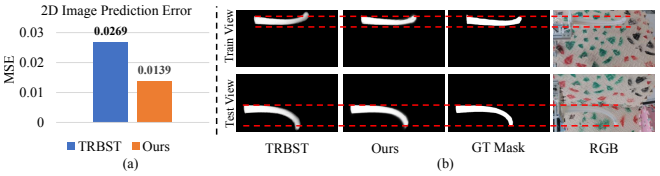

Fig. 8. (a) Quantitative comparison of morphology prediction for soft robot. (b) Qualitative comparison of the soft robot morphology predicted by TRBST [13], our method and the ground truth.

From each video, 22 evenly distributed images are extracted, each with a resolution of 1024×576. Fig. 7.(b) displays one such image. EfficientSAM [13] segments the soft robot within these images to create a binary mask. COLMAP [12] estimates the camera poses for these images. We designate 20 sets of images for training and reserve one set for testing. Fig. 7.(c) shows our reconstruction results.

As the real soft robot lacks ground truth 3D morphology, we use 2D predicted images for the quantitative evaluation of our method. The trained model predicts the mask images in the test set, and the mean squared error (MSE) quantifies the difference between these predictions and the ground truth. As shown in Fig. 8.(a), our method demonstrates the lowest error compared to the baseline [19]. Fig. 8.(b) showcases the visual reconstruction and prediction mask images, illustrating that our method predicts the shape of the real soft robot more accurately compared to the baseline method [19].

IV. CONCLUSIONS

In this paper, we introduce a novel visual self-modeling framework for soft robots. We utilize a hybrid representation method based on the Signed Distance Function (SDF) to capture both the geometric shape and kinematics of soft robots. Moreover, the training process of SoftNeRF is entirely self-supervised, thanks to differentiable rendering, which avoids reliance on training data with heavy annotations and facilitates the collection of generalizable and scalable data samples for lifelong learning. To improve the model's convergence speed and detail in modeling, we introduce an error-guided adaptive sampling method. We validate the applicability of SoftNeRF across three tasks within a simulation dataset and assess SoftNeRF's effectiveness on a real-world soft robot. Overall, SoftNeRF is a fully differentiable, compact, and kinematics-aware visual self-model that can be widely applied to a range of tasks, independent of training data. We plan to combine other easily accessible sensor data to further improve the accuracy of our model and explore more downstream tasks such as grasping and object interaction.

REFERENCES

[1] G. G. Gallup, "Self recognition in primates: A comparative approach to the bidirectional properties of consciousness." *American psychologist*, vol. 32, no. 5, p. 329, 1977.

[2] G. G. Gallup Jr, "Self-awareness and the emergence of mind in primates," *AJP*, vol. 2, no. 3, pp. 237–248, 1982.

[3] T. Metzinger, "Empirical perspectives from the self-model theory of subjectivity: a brief summary with examples," *PBR*, vol. 168, pp. 215–278, 2007.

[4] M. Levin, "Life, death, and self: Fundamental questions of primitive cognition viewed through the lens of body plasticity and synthetic organisms," *BBRC*, vol. 564, pp. 114–133, 2021.

[5] B. Chen, R. Kwiatkowski, C. Vondrick, and H. Lipson, "Fully body visual self-modeling of robot morphologies," *Sci. Robot.*, vol. 7, no. 68, p. eabn1944, 2022.

[6] J. Bongard, V. Zykov, and H. Lipson, "Resilient machines through continuous self-modeling," *Science*, vol. 314, no. 5802, pp. 1118–1121, 2006.

[7] R. Kwiatkowski and H. Lipson, "Task-agnostic self-modeling machines," *Sci. Robot.*, vol. 4, no. 26, p. eaau9354, 2019.

[8] K. Friston, B. Sengupta, and G. Auletta, "Cognitive dynamics: From attractors to active inference," *Proc. IEEE*, vol. 102, no. 4, pp. 427–445, 2014.

[9] G. Pezzulo, F. Rigoli, and K. Friston, "Active inference, homeostatic regulation and adaptive behavioural control," *Prog. Neurobiol.*, vol. 134, pp. 17–35, 2015.

[10] B. Shih, D. Shah, J. Li, T. G. Thuruthel, Y.-L. Park, F. Iida, Z. Bao, R. Kramer-Bottiglio, and M. T. Tolley, "Electronic skins and machine learning for intelligent soft robots," *Sci. Robot.*, vol. 5, no. 41, p. eaaz9239, 2020.

[11] P. Polygerinos, N. Correll, S. A. Morin, B. Mosadegh, C. D. Onal, K. Petersen, M. Cianchetti, M. T. Tolley, and R. F. Shepherd, "Soft robotics: Review of fluid-driven intrinsically soft devices; manufacturing, sensing, control, and applications in human-robot interaction," *Adv. Mater.*, vol. 19, no. 12, p. 1700016, 2017.

[12] J. L. Schonberger and J.-M. Frahm, "Structure-from-motion revisited," in *CVPR*, 2016, pp. 4104–4113.

[13] Y. Xiong, B. Varadarajan, L. Wu, X. Xiang, F. Xiao, C. Zhu, X. Dai, D. Wang, F. Sun, F. Iandola *et al.*, "Efficientsam: Leveraged masked image pretraining for efficient segment anything," in *CVPR*, 2024, pp. 16 111–16 121.

[14] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," *CACM*, vol. 65, no. 1, pp. 99–106, 2021.

[15] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *ACM Trans. Graph.*, vol. 41, no. 4, pp. 1–15, 2022.

[16] S. Fridovich-Keil, A. Yu, M. Tancik, Q. Chen, B. Recht, and A. Kanazawa, "Plenoxels: Radiance fields without neural networks," in *CVPR*, 2022, pp. 5501–5510.

[17] C. Sun, M. Sun, and H.-T. Chen, "Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction," in *CVPR*, 2022, pp. 5459–5469.

[18] S. Fridovich-Keil, G. Meanti, F. R. Warburg, B. Recht, and A. Kanazawa, "K-planes: Explicit radiance fields in space, time, and appearance," in *CVPR*, 2023, pp. 12 479–12 488.

[19] Y. Hu, J. Lin, and H. Lipson, "Teaching robots to build simulations of themselves," *arXiv preprint arXiv:2311.12151*, 2023.

[20] P. Wang, L. Liu, Y. Liu, C. Theobalt, T. Komura, and W. Wang, "Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction," *NIPS*, vol. 34, pp. 27 171–27 183, 2021.

[21] A. Gropp, L. Yariv, N. Haim, M. Atzmon, and Y. Lipman, "Implicit geometric regularization for learning shapes," in *ICML*. PMLR, 2020, pp. 3789–3799.

[22] P. Schegg, E. Ménager, E. Khairallah, D. Marchal, J. Dequidt, P. Preux, and C. Duriez, "Sofagym: An open platform for reinforcement learning based on soft robot simulations," *Soft Robot.*, vol. 10, no. 2, pp. 410–430, 2023.

[23] E. Sucar, S. Liu, J. Ortiz, and A. J. Davison, "imap: Implicit mapping and positioning in real-time," in *ICCV*, 2021, pp. 6229–6238.

[24] T. Deng, G. Shen, T. Qin, J. Wang, W. Zhao, J. Wang, D. Wang, and W. Chen, "Plgslam: Progressive neural scene represenation with local to global bundle adjustment," in *CVPR*, June 2024, pp. 19 657–19 666.

[25] J. Liu, G. Wang, C. Jiang, Z. Liu, and H. Wang, "Translo: A window-based masked point transformer framework for large-scale lidar odometry," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 2, 2023, pp. 1683–1691.

[26] J. Liu, G. Wang, W. Ye, C. Jiang, J. Han, Z. Liu, G. Zhang, D. Du, and H. Wang, "Difflow3d: Toward robust uncertainty-aware scene flow estimation with iterative diffusion-based refinement," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 15 109–15 119.

[27] Z. Li, T. Müller, A. Evans, R. H. Taylor, M. Unberath, M.-Y. Liu, and C.-H. Lin, "Neuralangelo: High-fidelity neural surface reconstruction," in *CVPR*, 2023, pp. 8456–8465.