

Continuum Robot Modeling with Action Conditioned Flow Matching

Jiong Lin^{1,*}, Jinchen Ruan^{1,*}, and Hod Lipson¹

¹Department of Mechanical Engineering, Columbia University, New York, NY, USA

*Equal contribution.

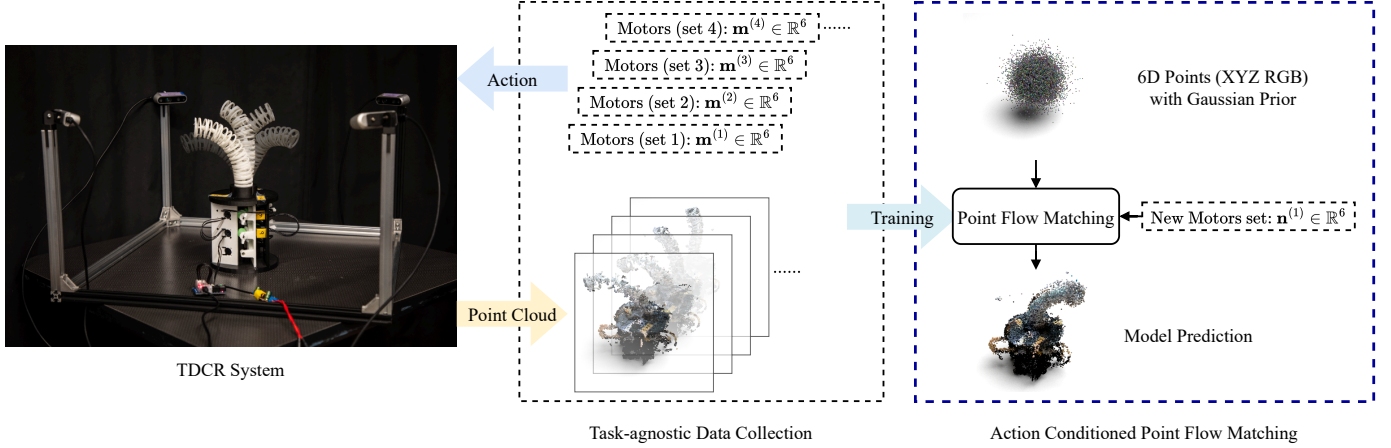


Fig. 1: Overview of our action conditioned point flow matching framework for a tendon driven continuum robot (TDCR) shape prediction. Task independent data samples (motor commands and point clouds) are collected from the real TDCR system, then used to train a flow matching model that predicts robot geometry for new motor inputs.

Abstract—Predicting the shape of tendon driven continuum robots (TDCRs) at steady state from actuation remains challenging due to continuous deformation, complex tendon routing, compliance, friction, and fabrication variability. In this paper, we address this problem as kinematic self modeling conditioned on action. We present a lightweight 3D printed TDCR hardware platform and an RGB-D data collection pipeline with multiple cameras, and we learn a point cloud flow matching model that maps motor actuation states to the robot’s settled 3D geometry. The model is trained from randomly sampled quasi static configurations and evaluated on test motor commands within the same TDCR design family and actuation range. We compare against prior 3D deformable object and robot self modeling approaches in both MuJoCo simulation and real hardware experiments. Experiments on simulated 2-, 3-, and 5-module TDCRs and real 2- and 3-module robots show improved shape prediction accuracy under CD and EMD metrics. We further show in simulation that the same conditional formulation generalizes to tip payload as a conditioning input, enabling payload conditioned steady-state shape prediction. These results demonstrate a data driven self modeling framework for quasi static TDCR geometry prediction. Project materials are available at: [TDCR Project Repo](#).

I. INTRODUCTION

Continuum robots, especially tendon driven continuum robots (TDCRs), offer high dexterity and inherent compliance, making them promising for operation in confined, cluttered, and unstructured environments [5, 53]. Unlike rigid link ma-

nipulators, TDCRs deform continuously along their bodies, and their observed shapes are affected by tendon routing, backbone compliance, friction, hysteresis, fabrication variability, and external loading. Classical kinematic representations such as constant curvature and piecewise constant curvature models provide compact and interpretable descriptions of continuum robot shape [55, 21]. More expressive mechanics based formulations, such as Cosserat rod models, can model distributed deformation, tendon routing, and external loads, but often require careful parameter identification and can be costly to calibrate for a specific physical platform [44, 40, 9]. These challenges motivate data driven models that learn the robot’s shape directly from observations.

In this paper, we study TDCR modeling as *action conditioned quasi static self modeling*. Given a motor or tendon actuation state, our goal is to predict the robot’s settled 3D geometry after the robot reaches a steady configuration. The learned mapping is induced by the robot’s physical deformation behavior, but our target is an observation level forward shape predictor rather than an explicit mechanics simulator. In particular, we do not attempt to recover material parameters, tendon forces, or transient deformation trajectories. Our primary setting is free space, quasi static TDCR shape prediction within a given robot design family and actuation range. We additionally evaluate a payload conditioned extension in sim-

ulation by augmenting the condition with a scalar tip payload magnitude, demonstrating that additional quasi static state variables can be incorporated when such data are available.

A key difficulty in TDCR modeling is that the robot body is high dimensional and continuously deformable. Predicting only a sparse set of backbone states or a small number of geometric parameters may miss shape details that are important for planning, collision checking, or visual servoing. We therefore represent each robot configuration as a dense point cloud reconstructed from RGB-D observations from multiple cameras. This representation is permutation invariant, directly compatible with real hardware depth sensing, and can capture full body bending and twisting without imposing a hand designed parametric shape model.

We propose an *action conditioned flow matching* formulation for learning this point cloud shape predictor. Rather than directly regressing a deterministic mapping from motor commands to point coordinates, the model learns a conditional velocity field in point cloud space that transports samples from a simple prior distribution to the distribution of robot shapes associated with a queried actuation state. At inference time, the model generates a dense 3D point cloud by integrating the learned flow from the prior to the predicted robot shape. The integration variable in this generative process is a transport time used by the flow model, not the physical time of robot deformation; the predicted output is the final quasi static shape.

To evaluate the approach, we build a lightweight, configurable 3D printed TDCR hardware platform and a synchronized RGB-D capture system with multiple cameras. The hardware is instantiated as 2- and 3-module real robots, while our MuJoCo simulation environments include 2-, 3-, and 5-module TDCR variants. Across simulated and real hardware settings, we compare against representative point based and view based deformable object and robot self modeling methods, including Visual Self Modeling (VSM) [7], conditional continuous normalizing flows (PointFlow) [56], NeRF inspired self simulation (FFKSM) [20], and articulated 3D Gaussian splatting [19]. Our experiments show that action conditioned flow matching achieves lower geometric error in steady state shape prediction, including more complex multi module settings and a simulated payload conditioned setting.

In summary, our main contributions are:

- A lightweight, configurable 3D printed TDCR platform and RGB-D capture pipeline with multiple cameras for collecting real hardware self modeling data.
- An action conditioned flow matching framework for quasi static TDCR self modeling, which predicts dense settled 3D robot geometry from motor actuation states.
- A systematic evaluation in MuJoCo simulation and on real TDCR hardware, including comparisons with point based and view based baselines, multi module TDCR variants, and a payload conditioned simulation extension.

II. RELATED WORK

This paper connects tendon driven continuum robot (TDCR) modeling, robot self modeling, and conditional 3D point

cloud generation. We review prior work with an emphasis on how existing methods represent continuum robot shape, what physical quantities they model, and how our work differs as an action conditioned quasi static shape predictor.

a) TDCR design and hardware diversity: TDCRs are a widely used class of continuum manipulators because tendon actuation enables slender robot bodies, remote actuation, and compliant interaction with confined environments [5, 52, 53, 40, 45]. However, TDCR designs vary substantially in structural backbone material, tendon routing, module construction, actuation layout, and manufacturing process. Prior systems have explored helical spring backbones and redundant tendon actuation [27], helical tendon routing for enlarged workspace and dexterity [50], extensible or follow the leader sections [34, 2, 33], fully actuated or programmable stiffness segments [15, 14], universal jointed and compressible backbone designs [47, 49], open source TDCR benchmarking platform [13], and monolithic 3D printed TDCR designs [25]. Beyond tendon driven mechanisms, octopus inspired soft arms and architected soft structures such as trimmed helicoids further illustrate how geometry and material topology can be used to tune bending, axial stiffness, workspace, and compliance [31, 17]. This diversity motivates data driven modeling pipelines that can be instantiated on a given robot platform, and cautions against claiming a single morphology independent model for all continuum robots. Our hardware is therefore used as a configurable TDCR platform for collecting self modeling data, rather than as a proposed universal TDCR mechanical design.

b) Kinematics, mechanics, and quasi static TDCR modeling: Classical continuum robot kinematics often uses constant curvature or piecewise constant curvature (PCC) assumptions to obtain compact shape coordinates and tractable forward/inverse kinematics [55, 21]. These kinematic representations can provide useful geometric descriptions, but they do not by themselves model material stress, tendon force, friction, or external loading. Mechanics based approaches, including tendon driven manipulator mechanics, Cosserat rod models, geometrically exact steady state formulations, and loading/stiffness analyses, explicitly reason about distributed bending, shear, torsion, tendon routing, actuation, and external loads [6, 44, 42, 12, 36, 3]. TDCR specific comparisons and benchmarks further show that model accuracy depends strongly on assumptions about curvature, tendon routing, friction, and calibration quality [40, 9]. Recent learning based TDCR kinematic models also address uncertainty or hysteresis in forward kinematics [51, 10]. Our approach is complementary to these analytical and learned kinematic models. We learn an observation level map from actuation, and optionally payload, to the robot's settled 3D geometry. The learned map captures the kinematic deformation seen in the data, but it is not a physical based simulator and does not recover material parameters, tendon forces, or transient deformation trajectories.

c) Shape sensing and learning based state estimation: Accurate shape information is central to TDCR control, planning, and self modeling. Embedded sensing methods, such as

fiber Bragg grating (FBG) sensors, can provide high rate curvature, shape, or force estimates, but require sensor integration, calibration, and often robot specific modeling assumptions [43, 24, 46]. External sensing with cameras, stereo, or RGB-D sensors avoids embedding sensors into the continuum body, but the reconstructed shape can still be affected by depth accuracy, calibration error, segmentation quality, and view dependent sampling. Learning based visual methods have been used for soft continuum shape control and inverse kinematics, showing that data driven models can absorb difficult to model geometric and mechanical effects from observations [1]. In contrast to methods that estimate a low dimensional curvature state or image space target, we use RGB-D observations from multiple cameras to supervise dense 3D point clouds and learn a forward self model from actuation to full body geometry.

d) Robot self modeling and neural 3D reconstruction:

Robot self modeling aims to learn an internal representation of a robot's body from interaction or sensory observations, enabling adaptation when the robot morphology or environment is uncertain [4, 11]. Recent visual self modeling methods use modern perception and neural rendering to reconstruct robot geometry and appearance from images. Visual Self-Modeling (VSM) learns approximate robot morphology from camera observations and masks [7]. NeRF style implicit representations and explicit 3D Gaussian splatting have also been used for robot self simulation and detailed neural reconstruction [32, 23, 20, 19]. These view based methods are powerful for appearance modeling and novel view rendering, but their objectives are not always aligned with predicting the full 3D body geometry from a queried actuation state. Our method instead operates directly in point cloud space, which matches RGB-D supervision and provides a direct geometry representation for shape prediction, collision reasoning, and downstream planning.

e) Point cloud generative modeling and flow matching: Point clouds are unordered and irregular, motivating architectures that process point sets directly, such as PointNet/PointNet++, graph-based models, transformers, and point voxel hybrids [38, 39, 54, 57, 30]. For 3D generation, continuous normalizing flows such as PointFlow model point cloud distributions through invertible continuous time transformations [56], while diffusion and score based models have enabled high quality generation and completion of point, voxel, and implicit 3D representations [18, 48, 58, 35, 22]. Flow Matching trains continuous time generative models by regressing velocity fields along probability paths, providing a simulation free objective for learning transport maps [8, 16, 28, 29]. In our setting, the continuous time variable is a generative transport parameter, not physical deformation time. We use flow matching to learn a conditional transport from a simple prior point distribution to the steady state robot shape distribution associated with a motor command.

f) Positioning of our work: Classical TDCR models provide interpretable kinematic or mechanics based structure, but they require modeling choices and calibration that can be difficult for fabricated continuum hardware. Visual self mod-

eling methods reconstruct robot geometry from observations, but they often focus on image or rendering objectives rather than point cloud prediction from actuation. Our contribution is to combine a real TDCR data collection platform with an action conditioned flow matching model that predicts dense quasi static 3D robot geometry from motor actuation states. The method is evaluated on simulated and real TDCRs, and we further test a payload conditioned simulation extension by adding a scalar tip payload condition.

III. METHOD

A. Problem Definition and Notation

We formulate TDCR modeling as action conditioned quasi static self modeling. For each actuation command, the robot is allowed to settle before observation, and the learning target is the final 3D geometry rather than the transient deformation process. A training dataset contains K settled configurations,

$$\mathcal{D} = \{(X_1^i, c^i)\}_{i=1}^K, \quad X_1^i \in \mathbb{R}^{N \times d}, \quad (1)$$

where X_1^i is a permutation invariant point cloud with N points and $d \in \{3, 6\}$ channels (XYZ or XYZ+RGB). The condition vector c^i contains the normalized action variables used to specify the quasi static state. For the standard TDCR experiments, $c^i = \tilde{m}^i$ is the normalized motor/actuation vector. The raw motor vector m denotes tendon command/rest length states in simulation and servo absolute positions on hardware. For the payload conditioned simulation, $c^i = [\tilde{m}^i \parallel \tilde{p}^i]$, where $\tilde{p}^i$ is a scalar gravity aligned tip payload magnitude. For an S -module TDCR, the motor dimension is $D = 3S$; thus $D = 6, 9$, and 15 for the 2-, 3-, and 5-module robots, respectively. Novel motor commands refer to test commands within the same robot morphology and actuation limits.

All point clouds are represented in a fixed robot/world frame. During training, we use an origin anchored global scaling

$$\tilde{X} = X/s, \quad (2)$$

where s is a dataset level scale factor and no translation is applied, preserving the shared base coordinate system. Motor commands and optional payload values are normalized per dimension using dataset statistics, e.g.,

$$\tilde{m}_j = \frac{m_j - (m_{\min})_j}{(m_{\max})_j - (m_{\min})_j}, \quad j = 1, \dots, D, \quad (3)$$

and payload values are normalized analogously. At inference, metric predictions are recovered by multiplying predicted XYZ coordinates by s . Figure 2 summarizes the training and prediction pipeline.

B. TDCR Hardware Platform and RGB-D Observation

The learning algorithm only requires pairs of actuation states and observed point clouds, but the physical platform defines the real hardware action and observation spaces. We use a lightweight 3D printed TDCR platform instantiated as 2- and 3-module real robots. Each module is driven by three tendons arranged at 120° around a PLA structural backbone, so

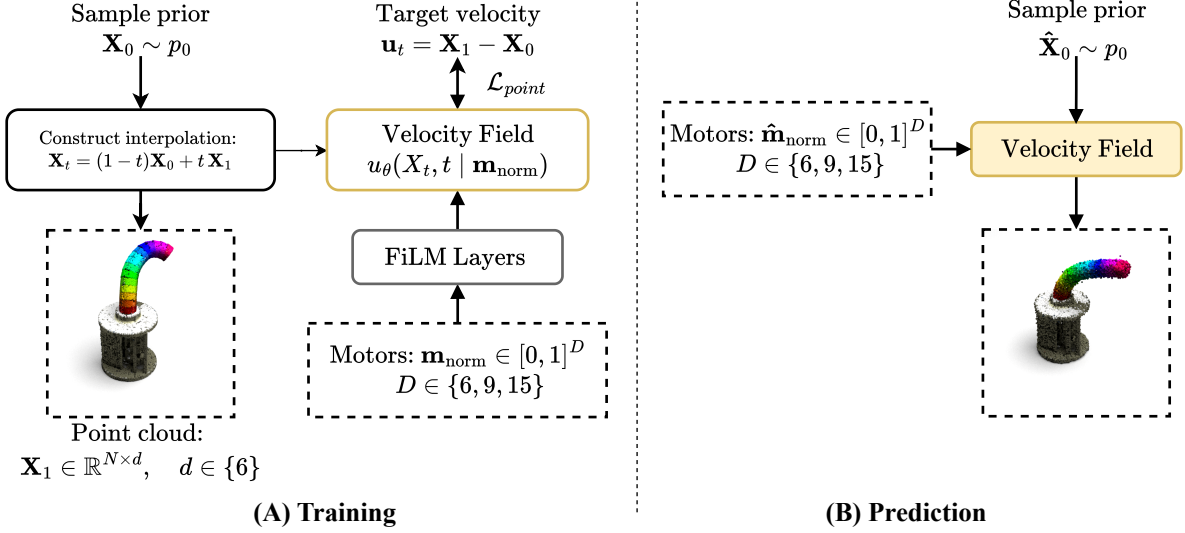


Fig. 2: Action conditioned flow matching framework. (A) **Training**: sample Gaussian prior X_0 , select a ground truth settled point cloud X_1 with condition c (motor command, optionally augmented with payload), construct the interpolation state X_t , and supervise the velocity field u_θ with the target transport direction. Conditions are injected via FiLM layers. (B) **Prediction**: integrate the learned ODE from noise to generate the predicted steady state point cloud $\hat{X}^*$ conditioned on a query $\hat{c}$. The integration variable is flow time, not physical deformation time.

an S -module robot uses $3S$ motors. The robot is configurable in module count at fabrication/assembly time, but a finished robot is not intended for rapid hot swapping of modules. The DYNAMIXEL implementation used in the real experiments has robot masses of 582.33 g (2 modules) and 767.96 g (3 modules). Detailed tendon routing, actuator options, BOM costs, spring thickness choices, and manufacturing notes are provided in Appendix VI-A.

For real hardware self modeling data, four calibrated Intel RealSense D435 RGB-D cameras surround the robot. For each settled command, we back project synchronized RGB-D views using camera intrinsics, transform them into a shared world frame with calibrated camera to camera extrinsics, and fuse them into a colored point cloud. The resulting point clouds are voxel downsampled and randomly sampled or padded to a fixed point count for learning. This representation gives the model direct supervision on full body 3D geometry rather than on a sparse backbone state or a hand designed low dimensional shape parameterization.

C. Action Conditioned Flow Matching in Point Space

We learn a conditional velocity field that transports a simple point prior to the distribution of settled TDCR shapes for a queried condition [8, 28]. Let $X_0 \sim p_0 = \mathcal{N}(0, \sigma^2 I)$ be a Gaussian point cloud with the same shape as a data sample X_1 . For a randomly sampled flow time $t \in [0, 1]$, a straight interpolation path is

$$X_t = (1 - t)X_0 + tX_1, \quad (4)$$

with pairwise transport velocity $u_t = X_1 - X_0$. The learning target is the marginal conditional velocity field

$$u^*(X, t | c) = \mathbb{E}[X_1 - X_0 | X_t = X, t, c]. \quad (5)$$

The continuous variable t in Eqs. (4)–(5) is the generative flow matching transport time; it is not the physical time of TDCR deformation.

We parameterize the velocity field with a neural network $u_\theta(X_t, t | c)$ and optimize

$$\mathcal{L}_{\text{FM}} = \mathbb{E}_{t, X_0, X_1, c} \left[\|u_\theta(X_t, t | c) - (X_1 - X_0)\|_F^2 \right], \quad (6)$$

where $\|\cdot\|_F$ is the Frobenius norm over the full $N \times d$ point cloud residual matrix. We sample $t \sim \text{Beta}(\alpha, 1)$ with $\alpha > 1$ to emphasize later interpolation states near the data distribution. For colored point clouds, the loss is split into geometry and color terms,

$$\mathcal{L}_{\text{FM}} = \mathcal{L}_{\text{xyz}} + \lambda_{\text{rgb}} \mathcal{L}_{\text{rgb}}, \quad (7)$$

with $\lambda_{\text{rgb}} = 0.05$ in our experiments. After training, a prediction for a query condition $\hat{c}$ is generated by sampling $\hat{X}_0 \sim p_0$ and integrating

$$\dot{X}_t = u_\theta(X_t, t | \hat{c}), \quad X_0 = \hat{X}_0, \quad (8)$$

from $t = 0$ to $t = 1$. The endpoint $X_{t=1}$ of this generative ODE is the predicted settled point cloud.

D. Condition Injection and Velocity Network Architectures

We evaluate two interchangeable architectures for u_θ , both permutation equivariant with respect to point ordering. Throughout this subsection, “velocity network” refers to the neural architecture; the mechanical component of the robot is referred to as the structural backbone.

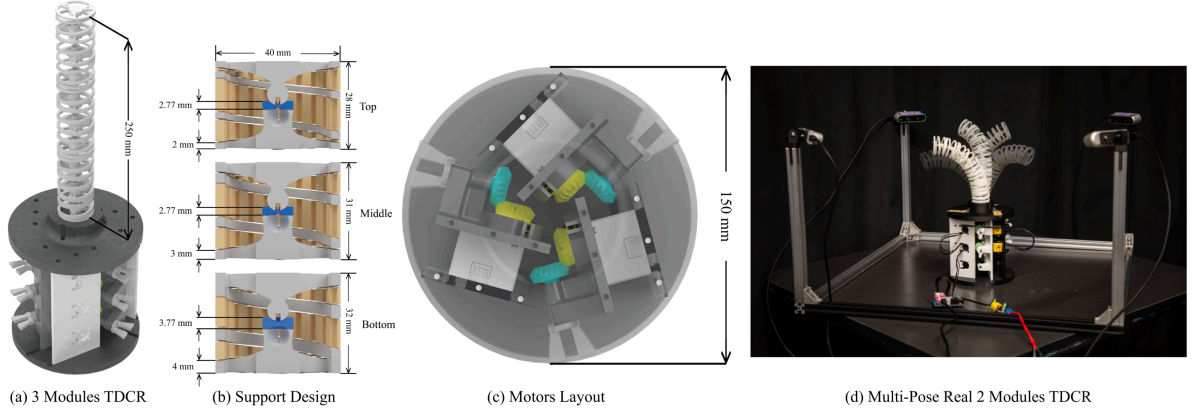


Fig. 3: TDCR hardware platform and real hardware data capture. (a) Three module CAD rendering. (b) Support design for single pass 3D printing. (c) Actuation base and motor layout. (d) Real two module TDCR during RGB-D data collection.

a) Time and action embeddings: We embed flow time t using sinusoidal features and project the condition vector c with a linear layer. The two embeddings are fused additively,

$$e(t, c) = \phi_t(t) + \phi_c(c), \quad (9)$$

and injected into intermediate features using FiLM modulation [37]:

$$\text{FiLM}(h; e) = (1 + \gamma(e)) \odot \text{LN}(h) + \beta(e), \quad (10)$$

where $\gamma(\cdot)$ and $\beta(\cdot)$ are learned affine heads, LN is Layer-Norm, and $\odot$ denotes element wise multiplication.

b) MLP velocity network: The lightweight MLP architecture processes each point independently with shared weights. For each point $x_i \in \mathbb{R}^d$, we concatenate its current coordinates/features with the global embedding and predict a per point velocity:

$$u_\theta(X_t, t | c) = \{f_\theta([x_i || e(t, c)])\}_{i=1}^N. \quad (11)$$

Residual MLP blocks with FiLM modulation provide an efficient $O(N)$ architecture for large point clouds.

c) Hybrid PVConv velocity network: To model local geometric context, the Hybrid architecture augments the per point head with a multi scale point voxel context module. A PVConv pyramid [30] extracts per point context features

$$C_{\text{pv}} = \text{ContextNet}(X_t, t, c) \in \mathbb{R}^{N \times d_c}, \quad (12)$$

which are concatenated with each point and the global embedding before the FiLM MLP head predicts velocities. Because the early part of the generative trajectory starts from unstructured Gaussian noise, we use a time gated blend between global and local context,

$$C(t) = \alpha(t)C_{\text{pv}} + (1 - \alpha(t))C_{\text{global}}, \quad \alpha(t) = \sigma(k(t - \tau)), \quad (13)$$

where $\sigma(\cdot)$ is the sigmoid and k, τ control the transition from global conditioning to local geometric context.

Algorithm 1 TDCR Training (Action-Conditioned Flow Matching)

Require: Dataset $\mathcal{D} = \{(X_1^i, c^i)\}_{i=1}^K$, where $c = \tilde{m}$ or $[\tilde{m} || \tilde{p}]$; prior $p_0 = \mathcal{N}(0, \sigma^2 I)$; velocity field u_θ ; optimizer OPT.

- 1: **for** step = 1 to T **do**
- 2: Sample minibatch $(X_1, c) \sim \mathcal{D}$
- 3: Sample $X_0 \sim p_0$ $\triangleright$ same $N \times d$ shape as X_1
- 4: Sample $t \sim \text{Beta}(\alpha, 1)$
- 5: Construct $X_t \leftarrow (1 - t)X_0 + tX_1$
- 6: Target velocity $u_t \leftarrow X_1 - X_0$
- 7: Predict $\hat{u} \leftarrow u_\theta(X_t, t | c)$ $\triangleright \hat{u} \in \mathbb{R}^{N \times d}$
- 8: $\mathcal{L}_{\text{xyz}} \leftarrow \|\hat{u}_{\text{xyz}} - u_{t, \text{xyz}}\|_F^2$
- 9: **if** $d = 6$ **then**
- 10: $\mathcal{L}_{\text{rgb}} \leftarrow \|\hat{u}_{\text{rgb}} - u_{t, \text{rgb}}\|_F^2$
- 11: $\mathcal{L} \leftarrow \mathcal{L}_{\text{xyz}} + \lambda_{\text{rgb}} \mathcal{L}_{\text{rgb}}$
- 12: **else**
- 13: $\mathcal{L} \leftarrow \mathcal{L}_{\text{xyz}}$
- 14: **end if**
- 15: Update $\theta \leftarrow \text{OPT.STEP}(\nabla_\theta \mathcal{L})$
- 16: Update EMA weights
- 17: **end for**

Algorithm 2 TDCR Shape Prediction (Heun RK2 Sampling)

Require: Trained velocity field u_θ with EMA weights; query condition $\hat{c}$; prior p_0 ; number of steps S ; step size $h \leftarrow 1/S$.

- 1: Sample $X \sim p_0$ $\triangleright X \in \mathbb{R}^{N \times d}$
- 2: **for** $k = 0$ to $S - 1$ **do**
- 3: $t_0 \leftarrow kh$
- 4: $v_1 \leftarrow u_\theta(X, t_0 | \hat{c})$
- 5: $\bar{X} \leftarrow X + h v_1$
- 6: $t_1 \leftarrow (k + 1)h$
- 7: $v_2 \leftarrow u_\theta(\bar{X}, t_1 | \hat{c})$
- 8: $X \leftarrow X + \frac{h}{2}(v_1 + v_2)$
- 9: **end for**
- 10: **return** $\hat{X}^* \leftarrow X$ $\triangleright$ multiply XYZ by s to recover metric scale

E. Training and Shape Prediction Algorithms

Algorithms 1 and 2 summarize the implementation used for training and inference. During training, we draw a settled point cloud and condition vector, sample a Gaussian point cloud, construct the interpolation state, and regress the conditional transport velocity. During prediction, we integrate Eq. (8) with Heun’s method (improved Euler/RK2) and use EMA model weights.

IV. EXPERIMENTS

We evaluate action conditioned flow matching on simulated and real TDCR self modeling tasks. Each experiment predicts the full settled 3D robot shape, represented as a point cloud, from a commanded actuation state. Unless otherwise specified, all methods are trained and evaluated per robot morphology and dataset split. The primary setting is unloaded/free space quasi static prediction; Sec. IV-F evaluates a payload conditioned simulation extension.

A. Datasets and Metrics

a) *Metrics:* Following prior point based self modeling work, we report the squared L2 Chamfer Distance (CD) and Earth Mover’s Distance (EMD) between the predicted point cloud $\hat{X}$ and the ground truth point cloud X . Let $P, Q \subset \mathbb{R}^3$ be two sets of points. The squared L2 Chamfer Distance is

$$\text{CD}(P, Q) = \frac{1}{|P|} \sum_{p \in P} \min_{q \in Q} \|p - q\|_2^2 + \frac{1}{|Q|} \sum_{q \in Q} \min_{p \in P} \|q - p\|_2^2, \quad (14)$$

and EMD is the minimum transport cost under a bijection π ,

$$\text{EMD}(P, Q) = \min_{\pi \in \Pi} \frac{1}{|P|} \sum_{i=1}^{|P|} \|p_i - q_{\pi(i)}\|_2, \quad (15)$$

where Π is the set of all permutations. For fair comparison, we compute both CD and EMD on point clouds of equal size by uniformly sampling $N_{\text{eval}}=10,000$ points from both P and Q .

b) *Normalization and evaluation scale:* All models are trained on globally normalized point clouds and normalized condition vectors (Sec. III). For reporting CD/EMD, we invert the point normalization and compute metrics in the original metric scale:

$$X_{\text{metric}} = \tilde{X} \cdot s, \quad (16)$$

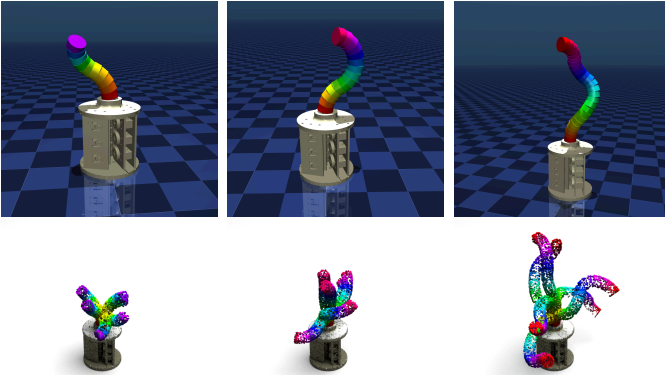


Fig. 4: MuJoCo renderings of our tendon driven continuum robot (TDCR) simulators for 2/3/5 modules (with base). Colors indicate the disc index along the backbone. Top row: single representative poses. Bottom row: multiple poses overlaid, illustrating the diversity of configurations in our dataset. The same simulated scene and camera setup is used to generate the RGB-D observations from multiple cameras.

where s is the scale factor for the entire dataset under origin anchored scaling without translation.

c) *Simulation datasets:* We build a family of tendon driven continuum robot simulators in MuJoCo. Each module is approximated as a chain of rigid disks connected by ball joints, while tendons are implemented using MuJoCo spatial tendons that pass through fixed sites on each disk. Actuators control tendon rest lengths, producing bending and twisting responses similar to physical TDCRs. This modeling choice is inspired by recent musculoskeletal simulation practices that rely on tendon like elements for complex deformation [26].

We created six standard simulation datasets: **2m**, **3m**, and **5m** (2/3/5 modules) $\times$ **without** base and **with** base. The *with base* variant includes the rigid base in the rendered observations, whereas the *no base* variant excludes it via XML visibility/group settings so that the point cloud contains only the deformable body. The no base setting is an analysis only ablation: with a fixed point budget, removing the rigid base increases sampling density on the deformable body. Because the compared methods use different point, implicit, and rendering objectives, the no base and with base settings can show different trends; we use this ablation to analyze sensitivity to rigid scene geometry rather than to argue that removing the base always improves prediction.

For each dataset, we uniformly sample motor commands within actuator limits and let the simulator relax to a static configuration before capture. We render RGB-D observations from a ring of fixed cameras around the robot, apply segmentation masks to keep foreground geometry, and back project depth to colored point clouds. Each sample is voxel downsampled and then randomly sampled or padded to $N_{\text{train}}=20,000$ points for training. Dataset sizes are 5,000 motor configurations for all 2m/3m settings and 10,000 for all 5m settings, with an 80%/10%/10% train/val/test split. Motor dimensions are $D = 6$ (2 modules), $D = 9$ (3 modules), and $D = 15$ (5 modules). Figure 4 shows example MuJoCo renderings of the three TDCR morphologies.

d) *Payload conditioned simulation datasets:* For the payload extension, we use the with base simulation scenes and augment the condition from $\tilde{m}$ to $[\tilde{m} \parallel \tilde{p}]$, where p is a scalar tip payload mass. For each sample, the simulator first settles under the sampled motor command, then applies a gravity aligned tip load $f_p = pg$ to the distal body, relaxes again, and captures the loaded steady state point cloud. Payload masses are sampled from $[0, 0.030]$ kg, with a 10% probability of exactly zero load. We collect 5,000 motor and payload configurations for each loaded 2m/3m/5m with base morphology and use the same 80%/10%/10% split. This setting evaluates payload conditioned quasi static shape prediction, not arbitrary contact, torque, or dynamic loading.

e) *Real hardware datasets:* We collect real hardware datasets with physical 2-module and 3-module TDCRs (Sec. III-B). Motor commands are generated via a joystick to motor mapping, executed to a settled configuration, and captured with four Intel RealSense D435 cameras. We fuse calibrated RGB-D frames from multiple cameras into a colored

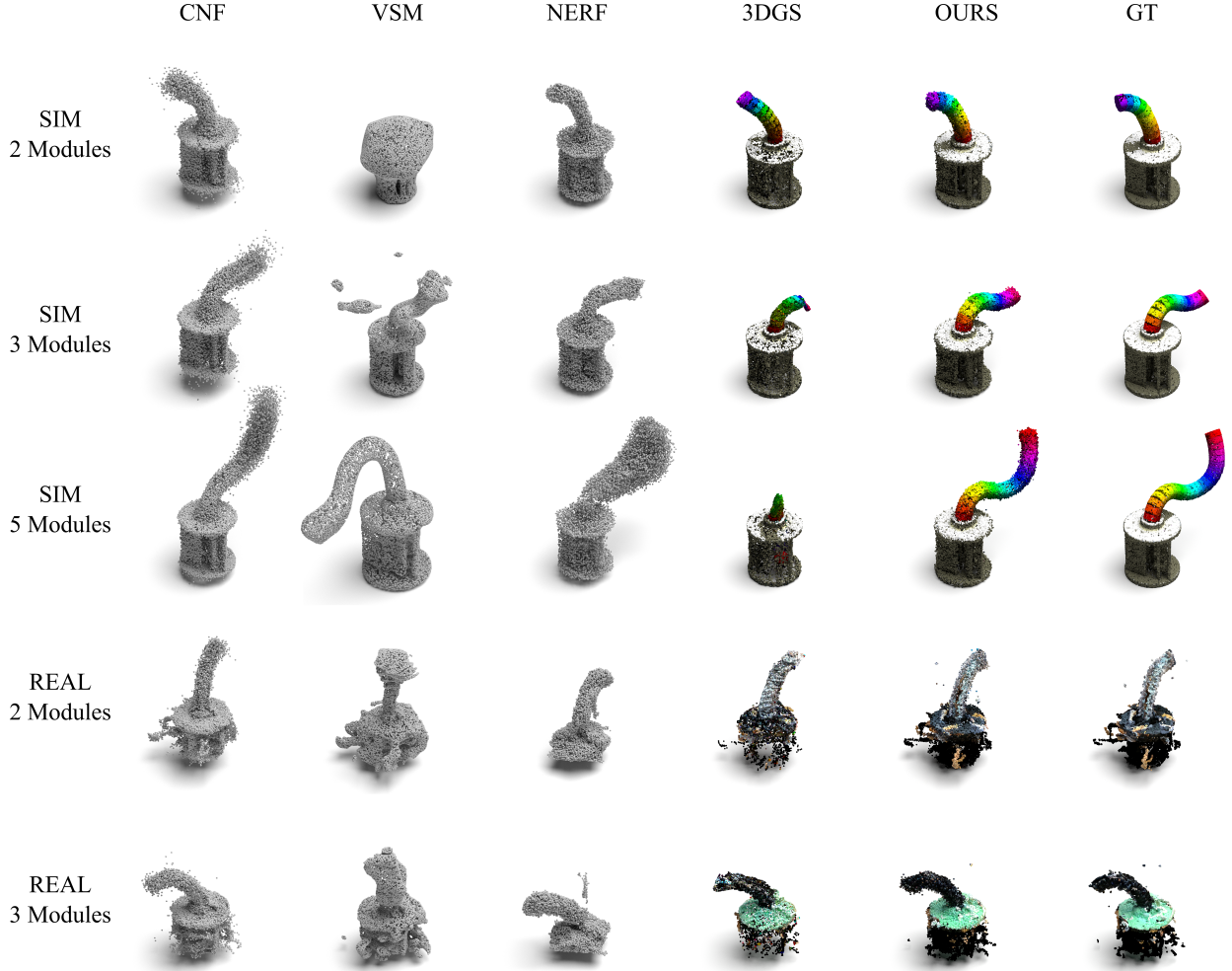


Fig. 5: Qualitative comparison on simulated and real TDCR datasets. Rows show simulated 2-/3-/5-module with base settings and real 2-/3-module hardware. Columns show CNF (PointFlow), VSM, NeRF (FFKSM), 3DGS, our method, and the ground truth (GT). For our method, we visualize the best performing velocity network architecture (MLP or Hybrid) selected by validation CD.

point cloud in a shared world frame, then apply the same voxel downsampling, $N_{\text{train}}=20,000$ point sampling, and normalization as in simulation.

B. Baselines

We compare against representative robot self modeling and 3D generative baselines, spanning point based and view based formulations. All baselines are trained on the same train split as ours and evaluated on the same test split using the point cloud metrics in Sec. IV-A.

a) VSM: Visual Self-Modeling (VSM) [7] learns a robot self model from $xyzn$ observations. We follow the official data interface, estimate stable normals for each TDCR point cloud, and convert VSM outputs back to point clouds for CD/EMD evaluation.

b) CNF / PointFlow: We include a conditional continuous normalizing flow baseline based on PointFlow [56]. The

flow is conditioned on the normalized motor vector and trained on the same splits. We use XYZ only point clouds and keep the number of generated points identical to our evaluation protocol.

c) NeRF self simulation (FFKSM): We adapt the self simulation method in *Teaching Robots to Build Simulations of Themselves* [20], which introduces a Free Form Kinematic Self Model (FFKSM) inspired by NeRF style reasoning. FFKSM is a query based implicit model that predicts occupancy/visibility of 3D query points conditioned on robot configuration and is trained from binary silhouettes. We train it with 100×100 masks following the original protocol and extract point clouds by densely querying the workspace and thresholding predicted occupancy.

d) 3D Gaussian Splatting (3DGS): We include the articulated 3D Gaussian splatting self model from *Learning*

TABLE I: Simulation steady state shape prediction on TDCR datasets. Lower is better. We report $CD \times 10^4$ and $EMD \times 10^3$ (metric scale, $N_{\text{eval}}=10,000$ points). NB/WB denote no base/with base observations.

Method	$CD \times 10^4 \downarrow$						$EMD \times 10^3 \downarrow$					
	2m-NB	2m-WB	3m-NB	3m-WB	5m-NB	5m-WB	2m-NB	2m-WB	3m-NB	3m-WB	5m-NB	5m-WB
VSM [7]	14.773	10.065	92.509	32.494	105.017	12.827	5.795	4.881	19.990	7.386	19.470	7.112
PointFlow [56]	0.359	0.461	2.172	1.013	19.790	6.788	0.598	1.041	1.362	1.203	5.453	3.227
NeRF [20]	0.341	0.953	0.650	0.974	6.552	4.309	1.009	2.263	2.958	2.035	9.753	15.854
3DGS [19]	0.447	0.373	1.053	1.205	19.201	63.781	0.461	3.779	0.847	4.837	4.849	14.652
Ours (MLP)	0.100	0.147	0.158	0.194	0.260	0.260	0.230	0.587	0.349	0.672	0.870	1.110
Ours (Hybrid)	0.088	0.135	0.136	0.164	0.253	0.239	0.204	0.516	0.329	0.578	0.832	1.059

TABLE II: Real hardware steady state shape prediction. Lower is better.

Method	$CD \times 10^4 \downarrow$		$EMD \times 10^3 \downarrow$	
	Real-2m	Real-3m	Real-2m	Real-3m
VSM [7]	2.287	1.525	4.426	5.363
PointFlow [56]	0.340	0.266	0.828	0.672
NeRF [20]	9.569	10.377	11.111	9.251
3DGS [19]	1.030	0.809	12.592	14.952
Ours (MLP)	0.226	0.215	0.522	0.507
Ours (Hybrid)	0.234	0.212	0.575	0.548

High Fidelity Robot Self Model with Articulated 3D Gaussian Splating [19]. This method reconstructs a robot with 3D Gaussians and learns a neural bone/LBS style deformation model conditioned on robot configurations. We train 3DGS with RGB and mask supervision from multiple camera views at 1280×720 resolution and evaluate by fusing rendered depth from calibrated viewpoints into an XYZ point cloud.

e) Mask generation for view based baselines: Both NeRF (FFKSM) and 3DGS require foreground masks. For simulation, MuJoCo provides near perfect instance masks. For real hardware captures, masks derived from depth thresholding are unreliable; we therefore use SAM 2 [41] to generate masks by manually annotating a small number of example masks per view and propagating to the remaining frames.

C. Implementation Details

a) Ours: We train our action conditioned flow matching models for 500 epochs. We sample flow time with $t \sim \text{Beta}(\alpha, 1)$ using $\alpha=3.0$ and sample the point prior as $X_0 \sim \mathcal{N}(0, \sigma^2 I)$ with $\sigma=0.5$. For RGB point clouds, we use the color loss weight $\lambda_{\text{rgb}}=0.05$. We use Heun RK2 sampling with $S=100$ steps for both the MLP and Hybrid velocity networks. For payload conditioned models, we use the same training procedure but set the condition mode to motor plus payload, i.e., $c = [\tilde{m} \parallel \tilde{p}]$. Unless stated otherwise, we report results using EMA weights selected by validation CD.

b) Training and evaluation protocol: All methods use the same train/val/test split within each dataset. We train with $N_{\text{train}}=20,000$ points per sample from the normalized H5 datasets. For evaluation, we denormalize predictions and ground truth to metric scale and uniformly sample $N_{\text{eval}}=10,000$ points from each before computing CD and EMD.

TABLE III: Payload conditioned simulation extension. The condition is augmented from $\tilde{m}$ to $[\tilde{m} \parallel \tilde{p}]$, where p is a scalar tip payload mass. Lower is better.

Method	$CD \times 10^4 \downarrow$			$EMD \times 10^3 \downarrow$		
	2m	3m	5m	2m	3m	5m
Ours (MLP)	0.393	0.386	0.741	0.992	0.826	1.272
Ours (Hybrid)	0.340	0.378	1.780	0.709	0.701	1.290

D. Quantitative Results

Table I reports standard simulation results, and Table II reports real hardware results. For readability, we report $CD \times 10^4$ and $EMD \times 10^3$, computed on $N_{\text{eval}}=10,000$ points in metric scale.

Across the six standard simulation settings, our Hybrid model achieves the lowest CD and EMD. Relative to the strongest non ours baseline in each setting, it reduces CD by approximately 64–96% and EMD by approximately 50–83%. On real data, the best of our two velocity networks reduces CD relative to PointFlow by 33.5% on Real-2m and 20.3% on Real-3m, and reduces EMD by 36.9% and 24.6%, respectively. These results indicate that direct point cloud flow matching is effective for predicting full body TDCR geometry from test actuation states.

E. Qualitative Results

Figure 5 compares representative predictions on simulated and real datasets. PointFlow captures global pose but can produce diffuse samples near the distal body. View based baselines are sensitive to mask quality and thin body geometry: NeRF (FFKSM) degrades in complex 5-module scenes, and 3DGS may reconstruct the rigid base more reliably than the thin continuum body. Our method maintains cleaner tip and body geometry across morphologies.

F. Payload Conditioned Simulation Extension

Table III and Fig. 6 evaluate the payload conditioned extension. The model is trained on loaded steady state point clouds with condition $c = [\tilde{m} \parallel \tilde{p}]$, where p is the scalar payload mass and the applied force is aligned with gravity. Both velocity network variants remain accurate under payload conditioning for 2-/3-/5-module TDCRs, and the overlays

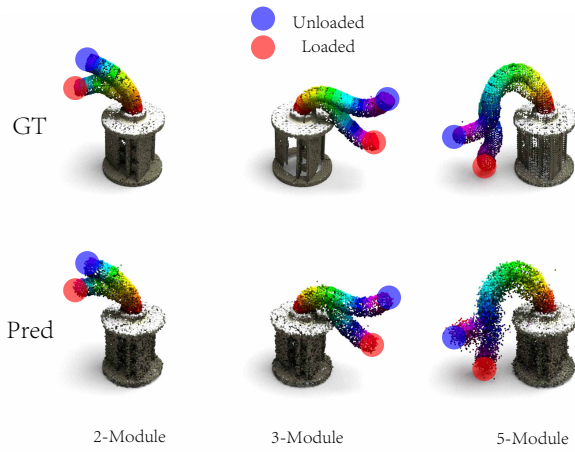


Fig. 6: Payload conditioned prediction in simulation. Blue/red markers show unloaded and maximum load tip states; rows show ground truth and prediction.

show load induced tip deflection. Hybrid is not uniformly better in this extension: it improves 2m/3m EMD but has a larger 5m CD than MLP. We treat this as evidence that additional quasi static state variables can be incorporated when such data are available, not as a claim of general load dependent mechanics.

G. Discussion

a) Takeaways: The standard simulation and real hardware results show that action conditioned flow matching can learn accurate quasi static TDCR self models for test commands within a fixed morphology and actuation range. The dense point cloud objective directly matches the fused RGB-D observation space, while the payload extension shows that the same conditioning mechanism can include a scalar tip load variable in simulation. Among baselines, CNF/PointFlow is the strongest point based competitor on real data but produces diffuse distal samples; NeRF/FFKSM and 3DGS are more sensitive to visibility, masks, and rigid base geometry. CNF also requires substantially more wall clock training time on simulated with base datasets: 21.5/18.2/60.5 hours for 2m/3m/5m, versus 8.8/10.9/17.4 hours for our Hybrid model.

b) Scope and limitations: Our experiments target observation level quasi static shape prediction, not transient dynamics; the flow matching ODE is a generative transport process rather than physical deformation time. The real hardware experiments use 2- and 3-module TDCRs, whereas the 5-module and payload conditioned results are simulation only. The payload experiment uses a scalar gravity aligned tip load and does not cover arbitrary contact, off axis torque, distributed load, unknown object geometry, or zero shot transfer across unseen continuum robot designs and materials. Extending the framework to richer contact, distributed loads, and unseen TDCR designs is left for future work, and we view the current results as a foundation for these directions.

V. CONCLUSION

We presented an action conditioned flow matching framework for quasi static self modeling of tendon driven continuum robots. Given a normalized actuation state, and optionally a scalar tip payload condition in simulation, the model predicts the robot’s settled 3D geometry as a dense point cloud. Across MuJoCo 2-, 3-, and 5-module TDCR simulations and real 2- and 3-module hardware experiments, our method achieves lower Chamfer Distance and Earth Mover’s Distance than representative point based and view based self modeling baselines. The payload conditioned simulation extension further shows that the same conditional formulation can incorporate a gravity aligned tip payload magnitude when corresponding training data are available. Future work will study real payload experiments, history- and contact aware conditioning, morphology conditioned transfer, and integration into closed loop planning and control.

REFERENCES

- [1] Elijah Almanzor, Fan Ye, Jialei Shi, Thomas George Thuruthel, Helge A. Würdemann, and Fumiya Iida. Static shape control of soft continuum robots using deep visual inverse kinematic models. *IEEE Transactions on Robotics*, 39(4):2973–2988, 2023. doi: 10.1109/TRO.2023.3275375.
- [2] Ernar Amanov, Thien-Dang Nguyen, and Jessica Burgner-Kahrs. Tendon-driven continuum robots with extensible sections—a model-based evaluation of motion capabilities. *The International Journal of Robotics Research*, 40(1):7–29, 2021. doi: 10.1177/0278364919886047.
- [3] Costanza Armanini, Frédéric Boyer, Anup Teejo Mathew, Christian Duriez, and Federico Renda. Soft robots modeling: A structured overview. *IEEE Transactions on Robotics*, 39(3):1728–1748, 2023. doi: 10.1109/TRO.2022.3231360.
- [4] Josh Bongard, Victor Zykov, and Hod Lipson. Resilient machines through continuous self-modeling. *Science*, 314(5802):1118–1121, 2006. doi: 10.1126/science.1133687.
- [5] Jessica Burgner-Kahrs, D. Caleb Rucker, and Howie Choset. Continuum robots for medical applications: A survey. *IEEE Transactions on Robotics*, 31(6):1261–1280, 2015. doi: 10.1109/TRO.2015.2489500.
- [6] David B. Camarillo, Christopher F. Milne, Christopher R. Carlson, Michael R. Zinn, and J. Kenneth Salisbury. Mechanics modeling of tendon-driven continuum manipulators. *IEEE Transactions on Robotics*, 24(6):1262–1273, 2008. doi: 10.1109/TRO.2008.2002311.
- [7] Boyuan Chen, Robert Kwiatkowski, Carl Vondrick, and Hod Lipson. Fully body visual self-modeling of robot morphologies. *Science Robotics*, 7(68):eabn1944, July 2022. doi: 10.1126/scirobotics.abn1944. URL <https://doi.org/10.1126/scirobotics.abn1944>.
- [8] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David K. Duvenaud. Neural ordinary differential

- equations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. URL <https://arxiv.org/abs/1806.07366>.
- [9] Mohamed Taha Chikhaoui, Stefan Lilge, Simon Kleinschmidt, and Jessica Burgner-Kahrs. Comparison of modeling approaches for a tendon actuated continuum robot with three extensible segments. *IEEE Robotics and Automation Letters*, 4(2):989–996, 2019. doi: 10.1109/LRA.2019.2893610.
- [10] Brian Y. Cho, Daniel S. Esser, Jordan Thompson, Bao Thach, Robert J. III Webster, and Alan Kuntz. Accounting for hysteresis in the forward kinematics of nonlinearly-routed tendon-driven continuum robots via a learned deep decoder network. *IEEE Robotics and Automation Letters*, 9(11):9263–9270, 2024. doi: 10.1109/LRA.2024.3455797.
- [11] Antoine Cully, Jeff Clune, Sumeet Tarapore, and Jean-Baptiste Mouret. Robots that can adapt like animals. *Nature*, 521(7553):503–507, 2015. doi: 10.1038/nature14422.
- [12] Mohsen Moradi Dalvand, Saeid Nahavandi, and Robert D. Howe. An analytical loading model for n -tendon continuum robots. *IEEE Transactions on Robotics*, 34(5):1215–1225, 2018. doi: 10.1109/TRO.2018.2838548.
- [13] Bastian Deutschmann, Jens Reinecke, and Alexander Dietrich. Open source tendon-driven continuum mechanism: A platform for research in soft robotics. In *Proceedings of the IEEE International Conference on Soft Robotics (RoboSoft)*, pages 54–61, 2022. doi: 10.1109/RoboSoft54090.2022.9762144.
- [14] Puspita Triana Dewi, Priyanka Rao, and Jessica Burgner-Kahrs. A lightweight modular segment design for tendon-driven continuum robots with pre-programmable stiffness. In *Proceedings of the IEEE International Conference on Soft Robotics (RoboSoft)*, 2024. doi: 10.1109/RoboSoft60065.2024.10522016.
- [15] Reinhard M. Grassmann, Priyanka Rao, Quentin Peyron, and Jessica Burgner-Kahrs. FAS—a fully actuated segment for tendon-driven continuum robots. *Frontiers in Robotics and AI*, 9:873446, 2022. doi: 10.3389/frobt.2022.873446.
- [16] Will Grathwohl, Ricky T. Q. Chen, Jesse Bettencourt, David Duvenaud, and Ilya Sutskever. FFIORD: Free-form continuous dynamics for scalable reversible generative models. In *International Conference on Learning Representations (ICLR)*, 2019.
- [17] Qinghua Guan, Francesco Stella, Cosimo Della Santina, Jinsong Leng, and Josie Hughes. Trimmed helicoids: An architected soft structure yielding soft robots with high precision, large workspace, and compliant interactions. *npj Robotics*, 1:4, 2023. doi: 10.1038/s44182-023-00004-7.
- [18] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.
- [19] Kejun Hu, Peng Yu, and Ning Tan. Learning high-fidelity robot self-model with articulated 3D gaussian splatting. *The International Journal of Robotics Research*, 2025. doi: 10.1177/02783649251396980. URL <https://doi.org/10.1177/02783649251396980>. OnlineFirst.
- [20] Yuhang Hu, Jiong Lin, and Hod Lipson. Teaching robots to build simulations of themselves. *Nature Machine Intelligence*, 7:484–494, February 2025. doi: 10.1038/s42256-025-01006-w. URL <https://doi.org/10.1038/s42256-025-01006-w>.
- [21] Bryan A. Jones and Ian D. Walker. Kinematics for multisection continuum robots. *IEEE Transactions on Robotics*, 22(1):43–55, 2006. doi: 10.1109/TRO.2005.861458.
- [22] Heewoo Jun and Alex Nichol. Shap-E: Generating conditional 3D implicit functions. *arXiv preprint arXiv:2305.02463*, 2023. doi: 10.48550/arXiv.2305.02463. URL <https://arxiv.org/abs/2305.02463>.
- [23] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4):139:1–139:14, 2023. doi: 10.1145/3592433.
- [24] Fouzia Khan, Roy J. Roesthuis, and Sarthak Misra. Force sensing in continuum manipulators using fiber bragg grating sensors. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2531–2536, 2017. doi: 10.1109/IROS.2017.8206073.
- [25] Clara G. Kierbel, Nicola Perugini, and Matteo Russo. A comparison of monolithic 3D-printed tendon-driven continuum robot designs. *Robotica*, 43(11):3888–3901, 2025. doi: 10.1017/S0263574725102191.
- [26] Vittorio La Barbera, Fabio Pardo, Yuval Tassa, Monica Daley, Christopher Richards, Petar Kormushev, and John Hutchinson. OstrichRL: A musculoskeletal ostrich simulation to study bio-mechanical locomotion. In *NeurIPS 2021 Deep Reinforcement Learning Workshop, 35th Conference on Neural Information Processing Systems (NeurIPS 2021)*, December 2021. URL https://kormushev.com/papers/Pardo_NeurIPS-2021.pdf.
- [27] Minhan Li, Rongjie Kang, Shineng Geng, and Emanuele Guglielmino. Design and control of a tendon-driven continuum robot. *Transactions of the Institute of Measurement and Control*, 40(11):3263–3272, 2018. doi: 10.1177/0142331216685607.
- [28] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*, 2023.
- [29] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations (ICLR)*, 2023.
- [30] Zhijian Liu, Haotian Tang, Yujun Lin, and Song Han. Point-voxel CNN for efficient 3d deep learning. In *Advances in Neural Information Processing Systems*

- (*NeurIPS*), 2019.
- [31] Barbara Mazzolai, Laura Margheri, Matteo Cianchetti, Paolo Dario, and Cecilia Laschi. Soft-robotic arm inspired by the octopus: II. from artificial requirements to innovative technological solutions. *Bioinspiration & Biomimetics*, 7(2):025005, 2012. doi: 10.1088/1748-3182/7/2/025005.
 - [32] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.
 - [33] Maria Neumann and Jessica Burgner-Kahrs. Considerations for follow-the-leader motion of extensible tendon-driven continuum robots. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 917–923, 2016. doi: 10.1109/ICRA.2016.7487223.
 - [34] Thien-Dang Nguyen and Jessica Burgner-Kahrs. An extensible continuum robot for intracavitary surgery. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015. doi: 10.1109/IROS.2015.7353661.
 - [35] Alex Nichol, Heewoo Jun, Prafulla Dhariwal, Pamela Mishkin, and Mark Chen. Point-E: A system for generating 3D point clouds from complex prompts. *arXiv preprint arXiv:2212.08751*, 2022. doi: 10.48550/arXiv.2212.08751. URL <https://arxiv.org/abs/2212.08751>.
 - [36] Kaitlin Oliver-Butler, John Till, and D. Caleb Rucker. Continuum robot stiffness under external loads and prescribed tendon displacements. *IEEE Transactions on Robotics*, 35(2):403–419, 2019. doi: 10.1109/TRO.2018.2885923.
 - [37] Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18)*, pages 3942–3951, 2018. URL <https://cdn.aaai.org/ojs/11671/11671-13-15199-1-2-20201228.pdf>.
 - [38] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 652–660, 2017. doi: 10.1109/CVPR.2017.16.
 - [39] Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.
 - [40] Priyanka Rao, Quentin Peyron, Sven Lilge, and Jessica Burgner-Kahrs. How to model tendon-driven continuum robots and benchmark modeling performance. *Frontiers in Robotics and AI*, 7, 02 2021. doi: 10.3389/frobt.2020.630245.
 - [41] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. SAM 2: Segment anything in images and videos. In *International Conference on Learning Representations (ICLR)*, 2025. URL <https://openreview.net/forum?id=Ha6RTeWMd0>.
 - [42] Federico Renda, Matteo Cianchetti, Michele Giorelli, Andrea Arienti, and Cecilia Laschi. A 3D steady-state model of a tendon-driven continuum soft manipulator inspired by the octopus arm. *Bioinspiration & Biomimetics*, 7(2):025006, 2012. doi: 10.1088/1748-3182/7/2/025006.
 - [43] Roy J. Roesthuis and Sarthak Misra. Steering of multi-segment continuum manipulators using rigid-link modeling and FBG-based shape sensing. *IEEE Transactions on Robotics*, 32(2):372–382, 2016. doi: 10.1109/TRO.2016.2527047.
 - [44] D. Caleb Rucker and Robert J. III Webster. Statics and dynamics of continuum robots with general tendon routing and external loading. *IEEE Transactions on Robotics*, 27(6):1033–1044, 2011. doi: 10.1109/TRO.2011.2160469.
 - [45] Matteo Antonio Russo, S. M. Hadi Sadati, Xin Dong, Abdelkhalick Mohammad, Ian D. Walker, Christos Bergeles, Kai Xu, and Dragos A. Axinte. Continuum robots: An overview. *Advanced Intelligent Systems*, 5(5):2200367, 2023. doi: 10.1002/aisy.202200367.
 - [46] Sung-Chul Ryu and Pierre E. Dupont. FBG-based shape sensing tubes for continuum robots. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3531–3536, 2014. doi: 10.1109/ICRA.2014.6907368.
 - [47] Chengnan Shentu and Jessica Burgner-Kahrs. Universal-jointed tendon-driven continuum robot: Design, kinematic modeling, and locomotion in narrow tubes. *arXiv preprint arXiv:2409.13165*, 2024. doi: 10.48550/arXiv.2409.13165.
 - [48] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations (ICLR)*, 2021.
 - [49] Manu Srivastava and Ian D. Walker. On tendon driven continuum robots with compressible backbones. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 669–675, 2023. doi: 10.1109/ICRA48891.2023.10161208.
 - [50] Julia Starke, Ernar Amanov, Mohamed Taha Chikhaoui, and Jessica Burgner-Kahrs. On the merits of helical tendon routing in continuum robots. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6470–6476, 2017. doi: 10.1109/IROS.2017.8206554.
 - [51] Jordan Thompson, Brian Y. Cho, Daniel S. Brown, and Alan Kuntz. Modeling kinematic uncertainty of tendon-driven continuum robots via mixture density networks. In *Proceedings of the International Symposium on Medical*

Robotics (ISMR), 2024. doi: 10.1109/ISMR63436.2024.10586133.

- [52] Deepak Trivedi, Christopher D. Rahn, William M. Kier, and Ian D. Walker. Soft robotics: Biological inspiration, state of the art, and future research. *Applied Bionics and Biomechanics*, 5(3):99–117, September 2008. doi: 10.1080/11762320802557865.
- [53] Ian D. Walker. Continuous backbone “continuum” robot manipulators. *ISRN Robotics*, 2013:1–19, 2013. doi: 10.5402/2013/726506.
- [54] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E. Sarma, Michael M. Bronstein, and Justin M. Solomon. Dynamic graph CNN for learning on point clouds. *ACM Transactions on Graphics*, 38(5), 2019. doi: 10.1145/3326362.
- [55] Robert J. III Webster and Bryan A. Jones. Design and kinematic modeling of constant curvature continuum robots: A review. *The International Journal of Robotics Research*, 29(13):1661–1683, 2010. doi: 10.1177/0278364910368147.
- [56] Guandao Yang, Xun Huang, Zekun Hao, Ming-Yu Liu, Serge J. Belongie, and Bharath Hariharan. PointFlow: 3D point cloud generation with continuous normalizing flows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4540–4549, 2019. doi: 10.1109/ICCV.2019.00464. URL <https://doi.org/10.1109/ICCV.2019.00464>.
- [57] Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip H. S. Torr, and Vladlen Koltun. Point transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.
- [58] Linfeng Zhou, Yipan Du, and Jiajun Wu. 3d shape generation and completion through point-voxel diffusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

Continuum Robot Modeling with Action Conditioned Flow Matching

Supplementary Material

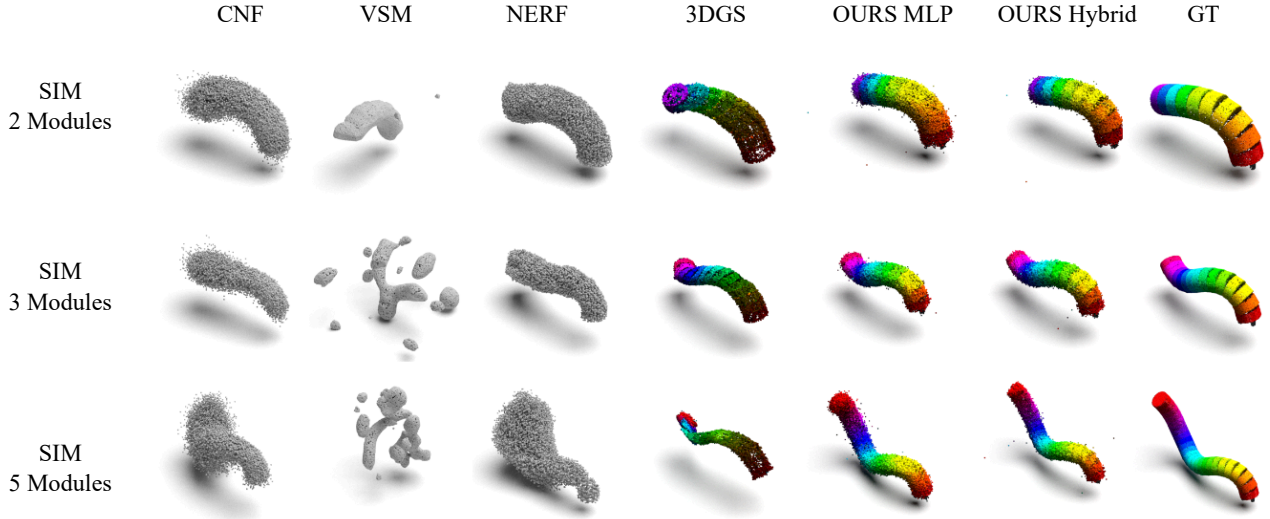


Fig. 7: Qualitative comparison on simulated *no base* datasets (2/3/5 modules). Columns show CNF (PointFlow), VSM, NeRF (FFKSM), 3DGS, our method, and the ground truth (GT). For our method, we visualize both MLP and Hybrid velocity networks.

VI. QUALITATIVE COMPARISON ON *no base* DATASETS

As shown in Fig. 7, our method yields the most accurate and visually stable predictions across all *no base* settings (2/3/5 modules). The Hybrid velocity network produces cleaner reconstructions near the robot tip, with reduced jitter and noise. VSM nearly fails to produce meaningful predictions in this regime, while NeRF self simulation (FFKSM) and 3D Gaussian Splatting (3DGS) degrade as the number of modules increases. Among the baselines, CNF (PointFlow) is the strongest point based competitor.

A. Manufacturing and Sensing System Details

This section provides the hardware details moved out of the main paper to satisfy the main text page limit. The real experiments use the compact DYNAMIXEL implementation. Table IV summarizes robot mass and approximate BOM costs; costs exclude cameras, power supply, and labor.

TABLE IV: TDCR hardware variants and approximate BOM costs.

Variant	Modules	Motors	Robot mass	BOM
DYNAMIXEL Real-2m	2	6	582.33 g	\$696.93
DYNAMIXEL Real-3m	3	9	767.96 g	\$1006.93

a) Print in place structure: During printing, dedicated sacrificial support structures support the spring geometry and ball joint components. The ball joints are printed in a disengaged state, with the ball and socket separated. By adjusting the designed initial clearance between the ball and socket

before assembly, we can tune the resulting module compliance once the joints are engaged. All support components are designed to be removable after printing. The structural backbone is fabricated as a single piece PLA print with repeating spring elements connected by ball joint interfaces. Each module contains four spring units, and neighboring spring units use a 90° helical phase shift to reduce nominal sideways bending at zero tension. For the 3-module robot, the structural backbone spring thickness decreases from base to tip as 4/3/2 mm, selected empirically across prototypes to reduce lower segment loading and nominal bending bias.

b) Tendon routing and module to motor assignment:

We use nylon coated stainless steel bead stringing wire as tendons. Tendons are routed through guide holes in the structural backbone, with three tendons distributed at 120° intervals around each module. For the 2-module TDCR, the lower motor layer controls the lower structural backbone module and the upper motor layer controls the upper module. For the 3-module TDCR, the three motor layers control the lower, middle, and upper modules, respectively. Each tendon is anchored at the top of its target module, passes through the lower guide channel, and attaches to the corresponding motor spool for winding/unwinding. A bevel gear interface on each spool enables manual tendon pretensioning at the home pose. Figure 8 visualizes the layer wise tendon assignment.

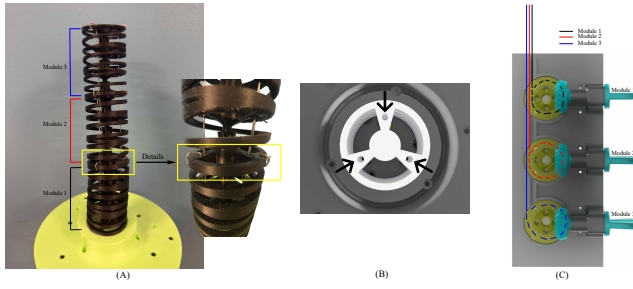


Fig. 8: Tendon routing overview. (A) Assembled 3-module robot and close-up. (B) Each module shares three tendon guide holes at 120° . (C) A tendon is anchored at the top of its assigned module, passes through the lower guide channel, and winds onto the spool in the corresponding motor layer.

c) Real hardware sensing setup: For real hardware data capture, we surround the TDCR with four Intel RealSense D435 RGB-D cameras. We estimate camera extrinsics via a camera to camera calibration procedure and use the factory depth-to-RGB calibration within each device to fuse synchronized depth and RGB images into colored point clouds. We back project each RGB-D view using camera intrinsics, transform all views into a shared coordinate frame using calibrated extrinsics, and merge point clouds from multiple cameras to obtain the observation used for learning.

B. Simulated Workspace Comparison

Figure 9 compares the reachable workspace of the simulated 2-, 3-, and 5-module TDCRs. We show the YZ-plane projection of the reachable tip positions, together with the outer boundary of the sampled workspace. As expected, increasing the number of modules expands both the lateral reach and the vertical span of the robot. This figure helps explain why the higher-module settings are more challenging: the robot explores a larger shape space and exhibits larger overall deformation.

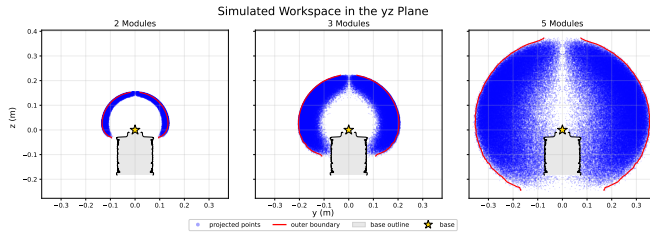


Fig. 9: YZ-plane workspace comparison for the simulated 2-, 3-, and 5-module TDCRs. Blue dots show sampled reachable tip positions, and the red curve outlines the outer workspace boundary.

C. Open Source Release

For reproducibility, we have publicly released the full project at our GitHub repository, including: (i) all code for training and inference, (ii) trained checkpoints for all experiments, (iii) the simulation models/assets used in our simulated datasets, and (iv) the SolidWorks CAD model of the real hardware TDCR.