A Task Frames

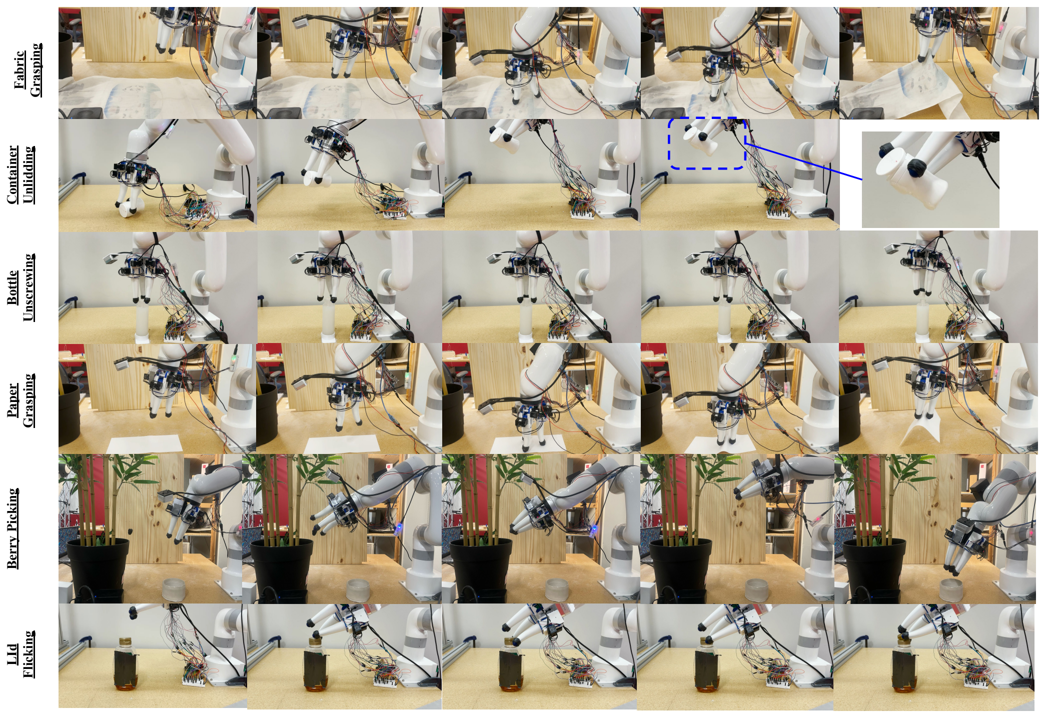

Figure 5: **Tasks**

B Shape Estimation and Tracking Evaluation

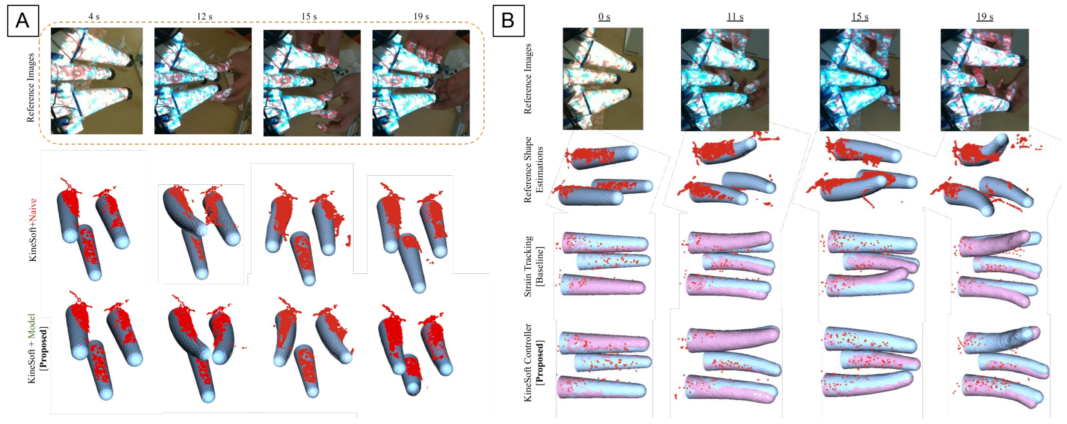
A
4 s
12 s
15 s
19 s
Reference Images
KineSoft+Naive
KineSoft + Model [Proposed]
B
0 s
11 s
15 s
19 s
Reference Images
Reference Shape Estimations
Strain Tracking [Baseline]
KineSoft Controller [Proposed]

Figure 6: **Shape estimation and trajectory tracking performance evaluation.** We provide each of the shape estimation models and controllers with kinesthetically deformed shape trajectories. $\boxed{\text{A}}$: Shape estimation model comparisons with real-world ground-truth data (red points). $\boxed{\text{B}}$: Shape tracking comparisons with the real-world ground-truth data (red points), references shapes (red), and achieved shapes (blue).

C Details on Sensor Model

We assume the embedded sensors are perfectly incompressible and isotropic, a common assumption in soft body mechanics for highly elastic rubber, particularly when infused with particle fillers [60].

These fillers, like those used in the off-the-shelf conductive rubbers embedded in MOE, enable the sensors to exhibit changes in resistivity when stretched. The sensors have a cylindrical shape, so we model the relationship between the cross-sectional area and the strain in the axial direction for sensor i at time $t \geq 0$ with the incompressibility assumption as:

$$L_{i,0} A_{i,0} = L_{i,t} A_{i,t}, \tag{3}$$

where $L_{i,0}$ and $A_{i,0}$ are the initial length and cross-sectional area; $L_{i,t}$ and $A_{i,t}$ are the corresponding values at time t.

For conductive materials, resistance generally has a linear relationship with strain. The observed resistance for the sensor indexed at i is given by:

$$R_{i,t} = \rho_i \frac{L_{i,t}}{A_{i,t}}, \tag{4}$$

where ρ_i is the conductivity factor, assumed to be constant for sensor i across time. Relating Equation 3 and Equation 4, we derive:

$$\sqrt{\frac{R_{i,t}}{R_{i,0}}} - 1 = \frac{L_{i,t} - L_{i,0}}{L_{i,0}}. \tag{5}$$

This relationship is independent of the material conductivity ρ_i, enabling a direct mapping from observed resistance to strain. However, in real-world applications, fabrication imperfections, such as connecting wires to the DAQ boards, can introduce errors into the initial length of the embedded sensors. These imperfections result in a deviation between the real sensor lengths ($L^R_{i,0}$, $L^R_{i,t}$) and simulated sensor lengths ($L^S_{i,0}$, $L^S_{i,t}$):

$$L^R_{i,0} = L^S_{i,0} + \epsilon_i, \quad L^R_{i,t} = L^S_{i,t} + \epsilon_i,$$

where ϵ_i is a constant error specific to each sensor i. This error propagates to the strain relationship as:

$$\frac{L^R_{i,t} - L^R_{i,0}}{L^R_{i,0}} = \frac{1}{1 + \frac{\epsilon_i}{L^S_{i,0}}} \cdot \frac{L^S_{i,t} - L^S_{i,0}}{L^S_{i,0}}. \tag{6}$$

The constant factor $\frac{1}{1+\frac{\epsilon_i}{L^S_{i,0}}}$ can be denoted as $\kappa_i \in \kappa$, representing the constant correction factor for sensor i. Substituting this into Equation 5, we obtain:

$$\sqrt{\frac{R_{i,t}}{R_{i,0}}} - 1 = \kappa_i \frac{L^S_{i,t} - L^S_{i,0}}{L^S_{i,0}}, \tag{7}$$

where the observed resistances $R_{i,t}, R_{i,0}$ are measured with the DAQ setup. For the n embedded sensors, aligning the simulated and observed distributions involves optimizing the constant correction parameters $\kappa_0, \kappa_1, \ldots, \kappa_{n-1}$.

D Baselines

For shape estimation, we compare with analytical and learning-based baselines:

Constant curvature model [51, 43]. Constant curvature model is a common representation for the continuum deformation behavior of soft robot that parametrizes the shape with a single curvature curve [61]. Typically, the independent parameters of the state of the robot are defined by r_{curve} and θ_{curve}. Assuming a constant length, L_{curve} of the robot, we get the constraint:

$$L_{\text{curve}} = r_{\text{curve}} \theta_{\text{curve}}.$$

In typical applications, additional term ϕ_{curve} is introduced to represent the plane of bending [48]. We implemented this simplified representation for soft robot shape using the proposed strain model as outlined in Section 4.3 and fitting r_{curve} and θ_{curve} to the observed strains in each side of the

Algorithm 1 Domain Alignment Optimization

```
Input: Shape predictor $f_\theta$, transformations $\{T\}$, observations $\mathcal{P}_{\text{obs}}$, initial resistances $R_0$
  Initialize $\theta = \{\kappa \in \mathbb{R}^{24}, \phi \in \mathbb{R}^3\} \leftarrow \mathbf{0}$
  Initialize CMA-ES optimizer $\mathcal{O}(\theta)$
  while not converged do
      Sample candidates $\{\theta\} \sim \mathcal{O}$
      for each $\theta$ do
          $\Delta R \leftarrow \sqrt{R/R_0} - 1$                                  ▷ Strain model for resistances
          Apply correction: $S \leftarrow \Delta R \cdot \kappa_{\mathbb{I}[\Delta R<0]}$
          $\mathbf{V} \leftarrow f_\theta(S)$                                        ▷ Predict vertices
          $\mathcal{P}_{\text{pred}} \leftarrow \bigcup \mathbf{V} \cdot T \cdot [R_y(\phi)|\mathbf{0}]$                ▷ Combine
          $\ell \leftarrow \mathcal{L}_{\text{UCD}}(\mathcal{P}_{\text{obs}}, \mathcal{P}_{\text{pred}})$                      ▷ Compute error
      end for
      Update $\mathcal{O}$ with candidates and losses
  end while
  return $\theta^*$ with minimum loss
```

curve. We transformed the cross-section boundary to the curve during the evaluation and measured the chamfer distance to the reference.

DeepSoRo [52]. DeepSoRo architecture deploys a FoldingNet [50] decoder conditioned on visual observations to predict the current shape of a deformable body. Crucially, it is trained with chamfer distance and originally trained on partial real-world shape observations, resulting in partial point cloud reconstruction outputs without frame-to-frame correspondences. Additionally, the model directly outputs the point cloud positions in contrast to KineSoft, which learns a deformation field and produces vertex displacement with frame-to-frame correspondences. We augment DeepSoRo for evaluation by training the model on KineSoft's simulated training data and using the proposed domain alignment process.

Shape-tracking Baselines. For shape tracking and task performance evaluation we provide the results against the following: **Strain Policy**: Strain policy, based on prior works that directly use sensor readings without intermediate representations for learning manipulation policies [53], uses raw sensor measurements instead of reconstructed shapes. For shape tracking evaluation, we modified the low-level controller from Section 4.4 to track reference sensor readings directly through proportional tendon actuation. For task performance evaluation, we trained a diffusion policy using the same 50 demonstrations we use for KineSoft, but with raw strain signals and wrist-mounted camera observations as input states.

E Data Generation and Shape Estimation Model Training

To train the model, we generate a large dataset of deformed meshes using SOFA (Simulation Open Framework Architecture) [62]. We simulate a tetrahedral finite-element mesh of the MOE finger with a Neo-Hookean hyperelastic material model parameterized by elastic material properties that are randomized at runtime.

We model the tendon actuation with massless, inextensible cables running through a series of fixed points within the finger body. We discretize each tendon path to segments defined by 3D attachment points embedded in the tetrahedral mesh. The cable constraint applies forces to these points to maintain constant length while allowing sliding, effectively simulating the mechanical behavior of Bowden cable transmission. The soft body scene is solved with an implicit Euler time integration scheme and uses a conjugate gradient solver for the system matrices. We generate training data by randomly sampling tendon actuation commands within the feasible range and recording the resulting deformed vertex positions and embedded sensor strains. To simulate rich deformation behaviors including contact-like effects, we apply random external forces to the finger surface. These forces are randomly applied over time with sufficiently large radii to ensure smooth deformations that mimic natural contact interactions, without requiring explicit and difficult-to-model contacts in the scene.

We train the model using a mean squared error (MSE) loss on vertex displacements:

$$\mathcal{L} = \frac{1}{3N} \sum_{j=1}^{3} \sum_{i=1}^{N} \|\Delta \mathbf{v}_{j,i} - \Delta \mathbf{v}_{j,i}^{*}\|^2,$$

where $\Delta \mathbf{v}_{j,i}^{*}$ represents the ground-truth displacement for vertex i of mesh finger j. This choice of loss function provides strong supervision by enforcing explicit vertex-wise correspondence between predicted and ground-truth meshes. Because we leverage simulated data to train the model, we can exploit the vertex-level correspondences in the meshes. We contrast this to prior works that relied on chamfer distance loss over real-world partial observations [52], MSE loss ensures that each vertex learns to track its specific local deformation patterns, enabling precise reconstruction of the full finger shape.

F Experiment Setup

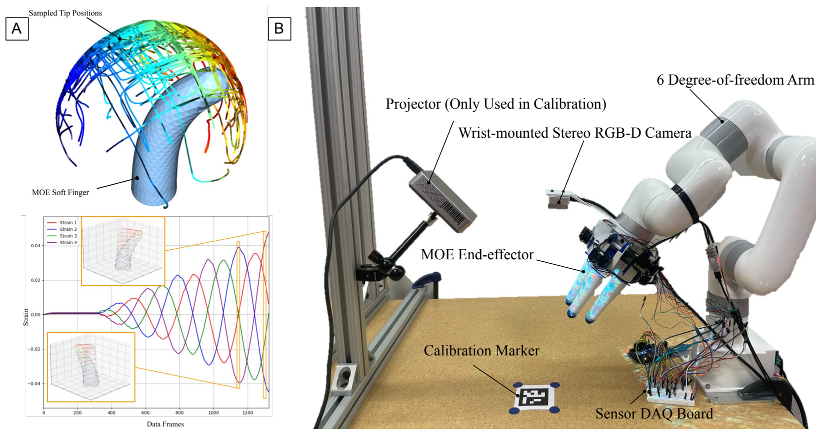

Figure 7: **Simulation and real-world setup**. $\boxed{\text{A}}$: Simulated robot workspace and sample of simulated strain signals. $\boxed{\text{B}}$: Real-world robot setup with the projected patterns to improve ground-truth shape observations for evaluation and calibration.

G Failiure Cases

We conducted stress testing through 1,000 continuous servo cycles over 14 hours, observing maximum signal drift of 8.3% in a sensor (Fig. **??**A) that resulted in 0.062 ± 0.0056 mm mean vertex error. After rest, the signals returned to baseline levels and our original manipulation experiments were conducted over several weeks without recalibration. We appreciate the reviewers highlighting this important concern and will update the manuscript with these discussions on long-term sensor reliability.

KineSoft failures occurred mainly when fingers lost or made unintended contact during manipulation transitions—a limitation of kinematic imitation without tactile sensing, which we will highlight in the limitations section. Baseline strain policies failed significantly more, as demonstration strain patterns often cannot be directly reproduced through tendon actuation. The results support KineSoft's approach of using shape as an intermediate representation between demonstration and execution modes.

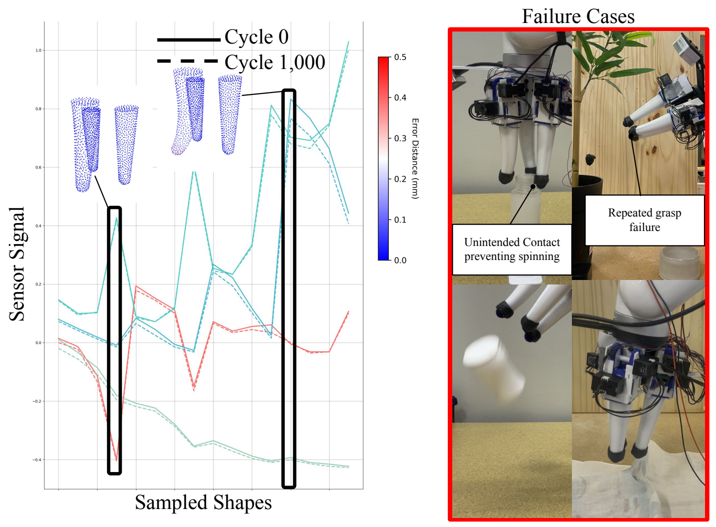

Figure 8: **Left:** Sensor signal degradation after 1000 cycles. **Right:** Failiure cases for manipulation tasks.

H Sensor Signals

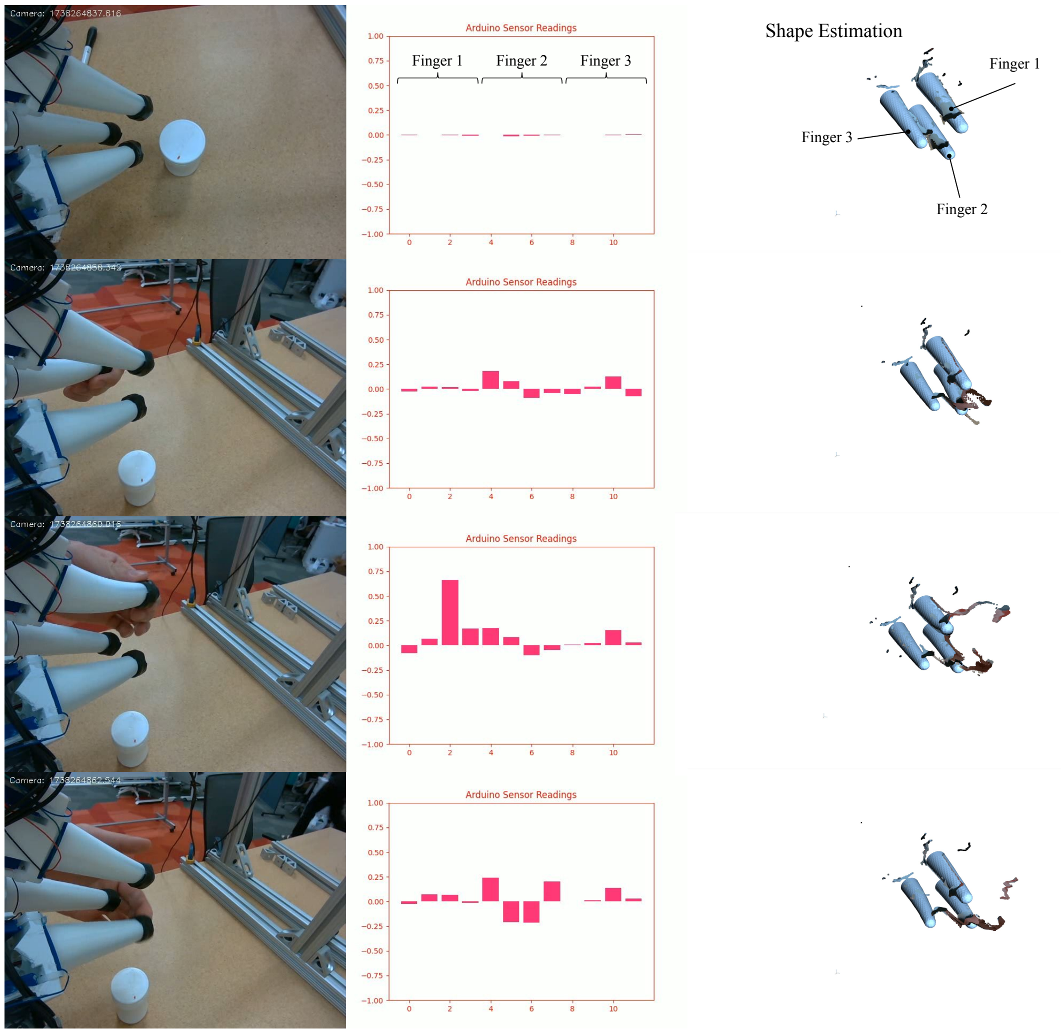

Figure 9: **Sensor signals and corresponding shape estimation**