GenParticles: Probabilistic Particle-Based Modeling for Object-Centric Motion

Arijit Dasgupta*, Eric Li*, Mathieu Huot, William T. Freeman, Vikash Mansinghka, Joshua B. Tenenbaum
MIT, Cambridge, MA, USA
{arijitdg, esli, mhuot, billf, vkm, jbt}@mit.edu

***Abstract*—Modeling and manipulating the physical world from visual input requires tracking entities and inferring their structure under uncertainty. We introduce GenParticles, a probabilistic, particle-based generative model that that supports Bayesian inference of persistent object-level structure from observed positions and motion cues over time. The model defines latent particles representing spatially localized matter via 3D Gaussians and imposes hierarchical motion constraints by clustering particles into groups with coherent dynamics. Approximate inference is performed via parallelized block Gibbs sampling, facilitating tracking and refinement of latent structure across naturalistic video sequences with dense per-frame observations. GenParticles maintains temporally consistent inference by updating particle structure, allowing it to track both rigid and deformable motion without requiring explicit point correspondences. Beyond video analysis, this method offers an online framework for identifying and mapping moving objects within a scene, with potential relevance for downstream applications in robotic manipulation.**

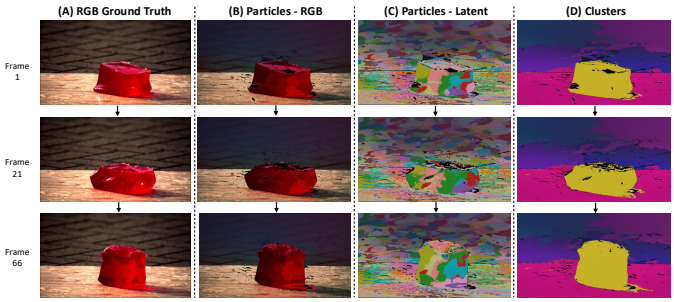

Fig. 1. **Tracking deformable matter with GenParticles**. A dropped block of Jello is tracked across time using a color variant of our model, described in Appendix B. The color of each pixel in each frame is given by **(a)** the color in the raw video data **(b)** the inferred color of its assigned particle **(c)** a false color randomly chosen for its assigned particle, for visual clarity **(d)**: a false color randomly chosen for its assigned cluster.

I. Introduction

Perception of object motion is central to structured world modeling, especially in robotic manipulation where agents interact with complex, disturbed and dynamic scenes [1, 2, 3, 4, 5]. Human perception studies suggest that motion-based grouping and rigidity cues enable us to infer object structure even without clear boundaries or texture [6]. Coherent motion reveals how deformable parts relate over time, while shared transformations help delineate rigid structures, offering a powerful signal for generalizing to novel objects beyond learned categories. Translating these insights to robotics motivates models that aim to capture persistent identity and flexible shape. While 3D models and dense tracking may struggle with deformation or require heavy supervision, end-to-end learning approaches often lack the compositional structure needed for planning and control.

We introduce GenParticles, a structured probabilistic generative model with a massively parallel approximate inference algorithm that jointly tracks 3D Gaussians ("particles") over time and infers the latent structures they belong to. Inspired by particle systems in computer graphics, the model represents scenes as collections of spatially localized 3D Gaussians, each corresponding to a region of visual matter undergoing approximately local translational motion. Particles are grouped into latent clusters that impose shared rigid transformations, enabling coherent motion modeling across parts. Cluster parameters are re-inferred at each frame, allowing the model to flexibly adapt to changing object geometry and motion. As qualitatively illustrated in Figure 1, this enables the recovery of coherent entities even under significant deformation.

We evaluate GenParticles on naturalistic RGB videos from a subset of the DAVIS dataset [7] with single-object segmentation masks. GenParticles produces dense, temporally consistent particle representations that track flexibly moving objects such as humans and animals. When clusters are initialized with object identity in the first frame, the model maintains accurate object coverage over time and outperforms state-of-the-art particle video baselines. The findings suggest that under the tested conditions, approximate inference in our model recovers object representations that reflect motion and structural persistence.

II. Related Works

a) Structured Generative Models of Scenes: Structured generative models have long offered a powerful framework for interpreting visual scenes via latent object representations and physically grounded dynamics. Early works like Attend-Infer-Repeat (AIR) [8] and its sequential extensions (SQAIR [9], SCALOR [10], SILOT [11]) pioneered unsupervised object discovery and tracking using variational inference in dynamic scenes. Slot Attention [12] and its video extension SAVi++[13] improved object-centric representation learning with iterative attention and spatial inductive biases. Generative structured models like C-SWM[14], G-SWM [15], and

*Equal contribution

STOVE [16] incorporated explicit object-level dynamics and physics-informed priors, enabling generalization and long-horizon prediction; capabilities that directly motivate the structured particle dynamics in `GenParticles`.

b) Structured World Models for Robotic Manipulation: Structured world models support object-centric representations that improve planning and control in robotics. Visual Foresight [17] used video prediction for model predictive control, while Transporter Networks [18] introduced spatial inductive biases for data-efficient manipulation. Recent methods like SWIM [19] transfer structured video models from human demonstrations to robots, showing the benefits of decomposable and structured representations for downstream control.

c) Particle-Based Tracking: The idea of persistently tracking semi-dense point sets was introduced by Particle Video [20] and later refined by Particle Video Revisited [21] with occlusion reasoning and trajectory consistency. Recent models like CoTracker3 [22], TAPIR [23], and SpaTracker [24] improve tracking using learned motion fields, temporal models, and rigidity constraints. DynOMo [25] performs online 3D tracking from monocular video with 3D Gaussian splatting, while PhysTwin [26] reconstructs and simulates deformable objects using physics-informed models. These methods track points but do not model hierarchical structure with global shared motion and locally approximate motion, which is important for interpretable, generalizable representations of both rigid and deformable motion.

III. Generative Particle Model

We introduce the **Generative Particle Model** (`GenParticles`), a two-level hierarchical generative model for structured motion in deformable scenes (Algorithm 1). *Clusters* represent rigid groups, each parameterized by a Gaussian over space and a per-frame rigid-body transformation. *Particles* are local Gaussians drawn from clusters and encode spatially localized variability. Data points are sampled from a mixture over particles. This hierarchy decouples global motion from local structure, allowing rigid objects of arbitrary shape to be represented as spatial mixtures of localized components. `GenParticles` is implemented in the `GenJAX` probabilistic programming framework [27, 28]

A. Cluster Model

Each cluster $k \in \{1, \ldots, K\}$ defines a spatial prior over its assigned particles via a 3D Gaussian with mean $\boldsymbol{\mu}_k^{\mathcal{H}} \sim \mathcal{N}(\boldsymbol{\mu}^{\mathcal{H}}, \sigma^2_{\mu^{\mathcal{H}}}\mathbf{I})$ and covariance $\boldsymbol{\Sigma}_k^{\mathcal{H}} \sim \mathcal{W}^{-1}(\boldsymbol{\Psi}^{\mathcal{H}}, \nu^{\mathcal{H}})$. Its rigid motion into the next frame is parameterized by a discrete translation $\mathbf{t}_k \sim \text{DiscreteNormal}(\mathbf{0}, s^2\mathbf{I})$ and rotation $\mathbf{R}_k \sim \text{DiscreteVMF}(\kappa^{\text{vmf}}, \theta_{\max})$, representing small frame-to-frame transformations.

B. Particle Model

Each particle $\ell \in \{1, \ldots, L\}$ is assigned to a cluster via a categorical latent $z_\ell^{\mathcal{H}} \sim \text{Cat}(\boldsymbol{\pi}^{\mathcal{H}})$, with cluster weights drawn from $\boldsymbol{\pi}^{\mathcal{H}} \sim \text{Dir}(\boldsymbol{\alpha})$. Given cluster $k = z_\ell^{\mathcal{H}}$, the particle's position is drawn from a Gaussian centered at the cluster mean: $\boldsymbol{\mu}_\ell^{\mathcal{B}} \sim \mathcal{N}(\boldsymbol{\mu}_k^{\mathcal{H}}, \boldsymbol{\Sigma}_k^{\mathcal{H}})$, with covariance $\boldsymbol{\Sigma}_\ell^{\mathcal{B}} \sim \mathcal{W}^{-1}(\boldsymbol{\Psi}^{\mathcal{B}}, \nu^{\mathcal{B}})$.

Particles inherit velocity from their parent cluster's rigid transformation, with expected velocity:

$$\bar{\mathbf{v}}_\ell = \mathbf{t}_k + (\mathbf{R}_k - \mathbf{I})(\boldsymbol{\mu}_\ell^{\mathcal{B}} - \boldsymbol{\mu}_k^{\mathcal{H}}),$$

where $(\mathbf{R}_k - \mathbf{I})$ approximates first-order rotation about the cluster center. The actual velocity is sampled as $\mathbf{v}_\ell \sim \mathcal{N}(\bar{\mathbf{v}}_\ell, \sigma_V^2\mathbf{I})$, with velocity noise covariance $\boldsymbol{\Sigma}_\ell^{\mathcal{V}} \sim \mathcal{W}^{-1}(\boldsymbol{\Psi}^{\mathcal{V}}, \nu^{\mathcal{V}})$.

Algorithm 1 Generative Particle Model

Input:
  K, L, N ▷ Number of clusters, particles, and observed data points
  Priors: $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$, $(\boldsymbol{\mu}^{\mathcal{H}}, \sigma^2_{\mu^{\mathcal{H}}}, \boldsymbol{\Psi}^{\mathcal{H}}, \nu^{\mathcal{H}})$, $(\boldsymbol{\Psi}^{\mathcal{B}}, \nu^{\mathcal{B}})$, σ_V^2, $(\boldsymbol{\Psi}^{\mathcal{V}}, \nu^{\mathcal{V}})$
Sample cluster weights: $\boldsymbol{\pi}^{\mathcal{H}} \sim \text{Dir}(\boldsymbol{\alpha})$
Sample particle weights: $\boldsymbol{\pi}^{\mathcal{B}} \sim \text{Dir}(\boldsymbol{\beta})$
for $k = 1$ to K **do**
  Sample cluster covariance: $\boldsymbol{\Sigma}_k^{\mathcal{H}} \sim \mathcal{W}^{-1}(\boldsymbol{\Psi}^{\mathcal{H}}, \nu^{\mathcal{H}})$
  Sample cluster mean: $\boldsymbol{\mu}_k^{\mathcal{H}} \sim \mathcal{N}(\boldsymbol{\mu}^{\mathcal{H}}, \sigma^2_{\mu^{\mathcal{H}}}\mathbf{I})$
  Sample cluster translation: $\mathbf{t}_k \sim \text{DiscreteNormal}(\mathbf{0}, s^2\mathbf{I})$
  Sample cluster rotation: $\mathbf{R}_k \sim \text{DiscreteVMF}(\kappa^{\text{vmf}}, \theta_{\max})$
end for
for $\ell = 1$ to L **do**
  Sample cluster assignment: $z_\ell^{\mathcal{H}} \sim \text{Cat}(\boldsymbol{\pi}^{\mathcal{H}})$
  Let $k = z_\ell^{\mathcal{H}}$
  Sample particle covariance: $\boldsymbol{\Sigma}_\ell^{\mathcal{B}} \sim \mathcal{W}^{-1}(\boldsymbol{\Psi}^{\mathcal{B}}, \nu^{\mathcal{B}})$
  Sample particle mean: $\boldsymbol{\mu}_\ell^{\mathcal{B}} \sim \mathcal{N}(\boldsymbol{\mu}_k^{\mathcal{H}}, \boldsymbol{\Sigma}_k^{\mathcal{H}})$
  Compute cluster-induced velocity: $\bar{\mathbf{v}}_\ell = \mathbf{t}_k + (\mathbf{R}_k - \mathbf{I})(\boldsymbol{\mu}_\ell^{\mathcal{B}} - \boldsymbol{\mu}_k^{\mathcal{H}})$
  Sample particle velocity mean: $\mathbf{v}_\ell \sim \mathcal{N}(\bar{\mathbf{v}}_\ell, \sigma_V^2\mathbf{I})$
  Sample particle velocity covariance: $\boldsymbol{\Sigma}_\ell^{\mathcal{V}} \sim \mathcal{W}^{-1}(\boldsymbol{\Psi}^{\mathcal{V}}, \nu^{\mathcal{V}})$
end for
for $n = 1$ to N **do**
  Sample particle assignment: $z_n^{\mathcal{B}} \sim \text{Cat}(\boldsymbol{\pi}^{\mathcal{B}})$
  Let $\ell = z_n^{\mathcal{B}}$
  Sample data point position: $\mathbf{x}_n \sim \mathcal{N}(\boldsymbol{\mu}_\ell^{\mathcal{B}}, \boldsymbol{\Sigma}_\ell^{\mathcal{B}})$
  Sample data point velocity: $\mathbf{v}_n \sim \mathcal{N}(\mathbf{v}_\ell, \boldsymbol{\Sigma}_\ell^{\mathcal{V}})$
end for

C. Observation Model

Each data point $n \in \{1, \ldots, N\}$ is drawn from a latent particle indexed by $z_n^{\mathcal{B}} \sim \text{Cat}(\boldsymbol{\pi}^{\mathcal{B}})$, where $\boldsymbol{\pi}^{\mathcal{B}} \sim \text{Dir}(\boldsymbol{\beta})$. Given assignment $\ell = z_n^{\mathcal{B}}$, the spatial position and velocity are independently drawn as:

$$\mathbf{x}_n \sim \mathcal{N}(\boldsymbol{\mu}_\ell^{\mathcal{B}}, \boldsymbol{\Sigma}_\ell^{\mathcal{B}}), \quad \mathbf{v}_n \sim \mathcal{N}(\mathbf{v}_\ell, \boldsymbol{\Sigma}_\ell^{\mathcal{V}}).$$

IV. Approximate Inference

We implement a blocked Gibbs sampler over the model's hierarchical latent variables—datapoints, particles, and clusters—using conjugate updates where available (e.g., Normal-Normal, Normal-Inverse-Wishart) [29, 30]. All computations are parallelized using `vmap` in `JAX` [31], with vectorized likelihoods and batched parameter updates across components. Assignment steps evaluate all datapoints against all components simultaneously, while cluster and particle updates are fused and executed in parallel. For non-conjugate terms like SE(3) transforms, we use exhaustive enumeration over discretized

candidates, restricting computations to each variable's Markov blanket [32, Ch. 4]. This setup enables inference on a single NVIDIA L4 GPU (24GB).

a) Latent Assignments: Each data point is assigned to a particle via a categorical posterior proportional to the prior mixture weight and the joint likelihood of observed position and velocity:

$$p(z_n^{\mathcal{B}} = \ell \mid \cdots) \propto \pi_\ell^{\mathcal{B}} \cdot \mathcal{N}(\mathbf{x}_n \mid \mu_\ell^{\mathcal{B}}, \Sigma_\ell^{\mathcal{B}}) \cdot \mathcal{N}(\mathbf{v}_n \mid \mathbf{v}_\ell, \Sigma_\ell^{\mathcal{V}}).$$

Particles are in turn assigned to clusters using a similar expression, incorporating rigid motion prediction:

$$p(z_\ell^{\mathcal{H}} = k \mid \cdots) \propto \pi_k^{\mathcal{H}} \cdot \mathcal{N}(\mu_\ell^{\mathcal{B}} \mid \mu_k^{\mathcal{H}}, \Sigma_k^{\mathcal{H}}) \cdot \mathcal{N}(\mathbf{v}_\ell \mid \bar{\mathbf{v}}_{\ell,k}, \sigma_V^2 I),$$

where $\bar{\mathbf{v}}_{\ell,k}$ is the velocity predicted by cluster k's rigid transformation. Mixture weights $\boldsymbol{\pi}^{\mathcal{B}}, \boldsymbol{\pi}^{\mathcal{H}}$ are resampled via conjugate Dirichlet updates.

b) Particle Parameter Updates: Each particle's spatial and velocity covariances $(\boldsymbol{\Sigma}_\ell^{\mathcal{B}}, \boldsymbol{\Sigma}_\ell^{\mathcal{V}})$ are sampled from Inverse-Wishart posteriors via Normal-Inverse-Wishart conjugacy. The velocity mean $\mathbf{v}_\ell$ is updated via a Gaussian posterior that combines a prior from the cluster's rigid motion with observed data point velocities. The spatial mean $\boldsymbol{\mu}_\ell^{\mathcal{B}}$ is sampled from a Gaussian posterior that integrates three sources: a prior from the cluster, a likelihood from the assigned data point positions, and a velocity-based constraint modeled as an affine likelihood. See Appendix A for details.

c) Cluster Parameter Updates: Cluster transformations $(\mathbf{R}_k, \mathbf{t}_k)$ are sampled by enumerating over discretized candidates, preferring transforms that align better with the assigned point motions. Spatial covariances $\boldsymbol{\Sigma}_k^{\mathcal{H}}$ are updated via Inverse-Wishart posteriors, based on the scatter of assigned particle means. The cluster mean $\boldsymbol{\mu}_k^{\mathcal{H}}$ is sampled from a Gaussian posterior combining the global prior, particle positions, and velocity residuals corrected by the current transform. See Appendix A for a full description.

d) Extending to Video: Although the generative model is defined over two frames, it naturally extends to sequential inference by propagating particle means forward using their inferred velocities: $\tilde{\boldsymbol{\mu}}_\ell^{\mathcal{B},t+1} = \boldsymbol{\mu}_\ell^{\mathcal{B},t} + \mathbf{v}_\ell^t$. All other latent variables are re-inferred at each frame via block Gibbs updates, conditioned on this propagated particle state. This yields a sequential MCMC filtering procedure that supports temporally consistent object tracking without requiring pointwise correspondences. This scheme is detailed in Appendix A-C.

V. Experiment: Object Persistence in Video

We evaluate whether `GenParticles` can maintain persistent particle representations of a moving object over time when given object information in the first frame of a video. The task is inspired by semi-supervised video object segmentation but differs in that we use ground truth segmentation only as a proxy to evaluate the quality of particle-based tracking.

As shown in Figure 2, `GenParticles` takes as input an RGB video and uses Video Depth Anything [33] and RAFT [34] to estimate monocular depth and optical flow, respectively.

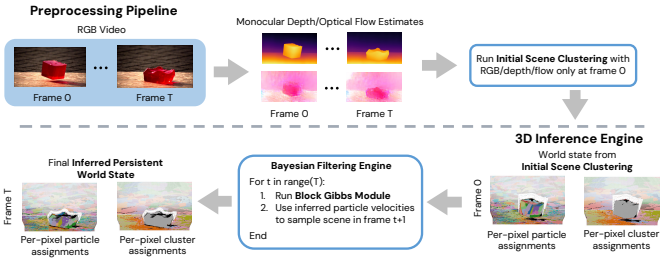

Fig. 2. Overview of the RGB video inference pipeline. The system takes an RGB video as input and initializes latent particles and clusters using monocular depth and optical flow estimates from frame 0. The `GenParticles` 3D inference engine takes the initial scene fit and runs Bayesian filtering to propagate the latent representation of the scene forward using inferred particle velocities obtained via block Gibbs sampling. The result is a temporally consistent particle-based representation of the scene, with per-pixel particle and cluster assignments across all frames.

particles are initialized in the first frame by performing k-means clustering. The ground truth segmentation mask from the initial frame is then used to group all object particles into a single cluster. We do not resample clusters assignments after the first frame, anchoring the model to the object throughout the sequence. This is followed by blocked Gibbs sampling to assign initialize to all latent variables.

During tracking, inference is performed via blocked Gibbs sweeps with a structured order to ensure stable convergence. Each sweep begins by propagating particles forward using inferred velocities. Observed datapoints are then assigned to particles based on spatial proximity, providing a reliable initial alignment before incorporating velocity or rigid transform constraints. Conditioned on these assignments, we update particle level variables: positions, velocities, and covariances. Cluster level parameters, including rigid transforms and spatial statistics, are then inferred. This bottom-up ordering grounds higher level structure in current frame evidence. After this initialization pass, additional Gibbs sweeps refine the estimates, with inferred clusters serving as priors that encourage particles to stay within object consistent regions. To stabilize inference, we fix particle covariances, preserving the spatial extent of deformable visual matter.

We compare against two recent state-of-the-art particle video baselines, both initialized with a grid of points to produce a structured particle grid: **CoTracker3** [22] and **SpaTracker** [24]. When masked by the ground truth segmentation in the first frame, CoTracker3 and SpaTracker's tracked points implicitly define an empirical, particle-based representation of the object. Each point is assigned an object identity, and the collective trajectory of the points forms a distributed approximation to the object as it moves.

Experiments are conducted on a 33-video subset of DAVIS [7], each with a single object segmentation mask. We resample all videos to resolution (520, 960) and evaluate performance using object particle persistence (%): the percentage

TABLE I

OBJECT PARTICLE PERSISTENCE (%) PER VIDEO. WE REPORT THE MEAN PERCENTAGE OF PARTICLES THAT REMAIN WITHIN THE GROUND TRUTH SEGMENTATION MASK AT EACH FRAME OVER TIME, RELATIVE TO THOSE INITIALIZED INSIDE IT IN THE FIRST FRAME. PARTICLES ARE LABELED BY THE INITIAL FRAME MASK, TRACKED OVER TIME, AND ASSIGNED ACCURACY BASED ON EACH FRAME'S MASK. `GenParticles` REPORTS MEAN AND STANDARD DEVIATION OVER 5 RANDOM SEEDS. BEST PER VIDEO IS BOLDED AND ITALICIZED VIDEOS CONTAIN HEAVY OCCLUSION.

DAVIS Video	`GenParticles` (Ours)	CoTracker3 [22]	SpaTracker [24]	DAVIS Video	`GenParticles` (Ours)	CoTracker3 [22]	SpaTracker [24]
boat	**100.00 ± 0.00**	90.69	92.14	bus	**93.33 ± 0.52**	82.16	82.79
car-turn	**100.00 ± 0.00**	97.15	99.89	dance-jump	**93.09 ± 1.80**	88.46	85.21
drift-chicane	**100.00 ± 0.00**	74.84	51.28	dog	92.78 ± 2.45	96.34	**97.09**
car-roundabout	**99.79 ± 0.25**	96.50	97.78	dance-twirl	**92.64 ± 1.12**	83.87	90.83
flamingo	**99.53 ± 0.42**	80.83	90.98	mallard-water	92.56 ± 1.00	**97.51**	94.28
breakdance-flare	**99.51 ± 0.25**	82.19	98.74	goat	**92.26 ± 2.82**	88.73	88.61
camel	**99.40 ± 0.53**	93.45	96.34	koala	**91.46 ± 1.03**	63.95	57.65
cows	**99.28 ± 0.89**	94.25	93.02	lucia	87.73 ± 2.18	93.77	**97.72**
rallye	98.96 ± 0.66	**100.00**	80.00	dog-agility	**84.78 ± 1.50**	78.65	74.20
rollerblade	**97.96 ± 1.92**	95.78	94.64	libby	**83.35 ± 3.19**	77.78	79.23
rhino	**97.50 ± 0.39**	88.19	87.87	parkour	78.57 ± 1.72	**84.96**	76.00
blackswan	96.89 ± 0.69	99.91	**100.00**	mallard-fly	69.23 ± 3.03	71.01	**85.27**
bear	**96.72 ± 1.29**	95.87	94.28	drift-turn	64.75 ± 2.76	**96.46**	91.47
elephant	**96.71 ± 1.11**	89.82	90.14	drift-straight	63.75 ± 12.47	**94.17**	74.00
breakdance	**96.31 ± 1.21**	69.06	94.83	***varanus-cage***	51.93 ± 9.78	**67.19**	64.93
hike	94.36 ± 2.56	92.67	**94.46**	***soccerball***	11.85 ± 0.22	**88.94**	87.22
car-shadow	94.11 ± 0.63	96.08	**97.28**	**Median Accuracy**	**94.11**	89.82	90.98

of particles that remain inside the segmentation mask over time, relative to those initialized within it in the first frame. This metric is averaged across frames to measure the stability of object association. It is well suited to our unsupervised generative setting, where no ground truth trajectories or identity labels are available. Particle persistence captures how well any method maintains consistent object association over time based solely on its internal representations. To remove unreliable particles, we discard those with fewer than 100 assigned points after initialization, an order of magnitude less than the average 1000 assigned points per particle. As `GenParticles` is probabilistic, each video is run with 5 random seeds. For fair comparison, all baselines operate in offline mode with full video access and are constrained to a 25×25 particle grid, the largest DAVIS sequence supported by a 24GB GPU. As shown in Table I, `GenParticles` outperforms SpaTracker and CoTracker3 on **20 of 33** sequences.

While `GenParticles` performs well overall, it underperforms on sequences such as *soccerball*, *varanus-cage*, *drift-straight*, and *drift-turn*, due to challenges in maintaining object continuity during extended occlusion and out-of-frame motion. In these cases, the posterior over particle trajectories can collapse without sustained visual input, as the model lacks a dynamics prior to carry motion forward when the object is no longer observed. In contrast, CoTracker3 and SpaTracker incorporate components specifically designed to handle occlusions, giving them an advantage in such scenarios.

VI. POTENTIAL APPLICATIONS TO ROBOTIC MANIPULATION

The structured particle-based representation provided by `GenParticles`, along with potential extensions, may offer advantages for robotic manipulation tasks:

a) Adaptive Resolution: `GenParticles` supports variable-resolution by adjusting the particle count, balancing accuracy and computational cost. Finer models suit complex deformable objects, while coarser ones suffice for simpler rigid cases. Although the current implementation sets this manually, future work could infer resolution adaptively from data [35, 36].

b) Persistent Object Representation: By organizing particles into coherent clusters, `GenParticles` provides stable representations even as objects deform. This is valuable in manipulation tasks where robot actions induce shape changes, allowing identity and structure to temporally persist.

c) Interpretable Structure: `GenParticles` 's explicit particle representation yields interpretable states for planning and control. Cluster transforms capture object dynamics, while particle grouping infers shape and segmentation from motion alone, without relying on texture, labels, or correspondences, supporting robust manipulation of novel objects.

d) Inferring Internal Properties of Non-Rigid Objects: A future extension could infer physical properties such as stiffness or elasticity from observed deformations [37], enabling more physically grounded representations. For instance, modeling the stiffness of non-homogeneous ropes.

e) Uncertainty-Aware Inference: `GenParticles` maintains uncertainty over structure and motion, enabling risk-aware planning, improved robustness in unstructured settings, and safer behavior by deferring or adapting actions when confidence is low.

f) Physical Dynamics Priors: A potential future extension of `GenParticles` is to incorporate explicit physical dynamics priors, such as simulated motion models. This could improve coherence under occlusion and enable more robust tracking when direct observations are missing.

`GenParticles` has the potential to serve as a general purpose probabilistic tool for robotic manipulation by capturing object structure and motion in an interpretable form that supports downstream tasks from perception to planning

ACKNOWLEDGMENTS

This work was supported in part by CoCoSys, one of seven centers in JUMP 2.0, a Semiconductor Research Corporation (SRC) program sponsored by DARPA

References

[1] Ben Burgess-Limerick, Jesse Haviland, Chris Lehnert, and Peter Corke. Reactive base control for on-the-move mobile manipulation in dynamic environments. *IEEE Robotics and Automation Letters*, 9(3):2048–2055, 2024.

[2] Yulong Li and Deepak Pathak. Object-aware gaussian splatting for robotic manipulation. In *ICRA 2024 Workshop on 3D Visual Representations for Robot Manipulation*, 2024.

[3] Huixu Dong, Ehsan Asadi, Guangbin Sun, Dilip K Prasad, and I-Ming Chen. Real-time robotic manipulation of cylindrical objects in dynamic scenarios through elliptic shape primitives. *IEEE Transactions on Robotics*, 35(1):95–113, 2018.

[4] Philipp S Schmitt, Florian Wirnshofer, Kai M Wurm, Georg v Wichert, and Wolfram Burgard. Modeling and planning manipulation in dynamic environments. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 176–182. IEEE, 2019.

[5] Zhenjia Xu, Zhanpeng He, Jiajun Wu, and Shuran Song. Learning 3d dynamic scene representations for robot manipulation. *arXiv preprint arXiv:2011.01968*, 2020.

[6] Madeleine Y Stepper, Cathleen M Moore, Bettina Rolke, and Elisabeth Hein. The role of object history in establishing object correspondence. *Attention, Perception, & Psychophysics*, 82:1038–1050, 2020.

[7] Jordi Pont-Tuset, Federico Perazzi, Sergi Caelles, Pablo Arbeláez, Alex Sorkine-Hornung, and Luc Van Gool. The 2017 davis challenge on video object segmentation. *arXiv preprint arXiv:1704.00675*, 2017.

[8] SM Eslami, Nicolas Heess, Theophane Weber, Yuval Tassa, David Szepesvari, Geoffrey E Hinton, et al. Attend, infer, repeat: Fast scene understanding with generative models. *Advances in neural information processing systems*, 29, 2016.

[9] Adam Kosiorek, Hyunjik Kim, Yee Whye Teh, and Ingmar Posner. Sequential attend, infer, repeat: Generative modelling of moving objects. *Advances in Neural Information Processing Systems*, 31, 2018.

[10] Jindong Jiang, Sepehr Janghorbani, Gerard De Melo, and Sungjin Ahn. Scalor: Generative world models with scalable object representations. *arXiv preprint arXiv:1910.02384*, 2019.

[11] Eric Crawford and Joelle Pineau. Exploiting spatial invariance for scalable unsupervised object tracking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3684–3692, 2020.

[12] Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention. *Advances in neural information processing systems*, 33:11525–11538, 2020.

[13] Gamaleldin Elsayed, Aravindh Mahendran, Sjoerd Van Steenkiste, Klaus Greff, Michael C Mozer, and Thomas Kipf. Savi++: Towards end-to-end object-centric learning from real-world videos. *Advances in Neural Information Processing Systems*, 35:28940–28954, 2022.

[14] Thomas Kipf, Elise Van der Pol, and Max Welling. Contrastive learning of structured world models. *arXiv preprint arXiv:1911.12247*, 2019.

[15] Zhixuan Lin, Yi-Fu Wu, Skand Peri, Bofeng Fu, Jindong Jiang, and Sungjin Ahn. Improving generative imagination in object-centric world models. In *International conference on machine learning*, pages 6140–6149. PMLR, 2020.

[16] Jannik Kossen, Karl Stelzner, Marcel Hussing, Claas Voelcker, and Kristian Kersting. Structured object-aware physics prediction for video modeling and planning. *arXiv preprint arXiv:1910.02425*, 2019.

[17] Frederik Ebert, Chelsea Finn, Sudeep Dasari, Annie Xie, Alex Lee, and Sergey Levine. Visual foresight: Model-based deep reinforcement learning for vision-based robotic control. *arXiv preprint arXiv:1812.00568*, 2018.

[18] Andy Zeng, Pete Florence, Jonathan Tompson, Stefan Welker, Jonathan Chien, Maria Attarian, Travis Armstrong, Ivan Krasin, Dan Duong, Vikas Sindhwani, et al. Transporter networks: Rearranging the visual world for robotic manipulation. In *Conference on Robot Learning*, pages 726–747. PMLR, 2021.

[19] Russell Mendonca, Shikhar Bahl, and Deepak Pathak. Structured world models from human videos. *arXiv preprint arXiv:2308.10901*, 2023.

[20] Peter Sand and Seth Teller. Particle video: Long-range motion estimation using point trajectories. *International journal of computer vision*, 80:72–91, 2008.

[21] Adam W Harley, Zhaoyuan Fang, and Katerina Fragkiadaki. Particle video revisited: Tracking through occlusions using point trajectories. In *European Conference on Computer Vision*, pages 59–75. Springer, 2022.

[22] Nikita Karaev, Iurii Makarov, Jianyuan Wang, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker3: Simpler and better point tracking by pseudo-labelling real videos. *arXiv preprint arXiv:2410.11831*, 2024.

[23] Carl Doersch, Yi Yang, Mel Vecerik, Dilara Gokay, Ankush Gupta, Yusuf Aytar, Joao Carreira, and Andrew Zisserman. Tapir: Tracking any point with per-frame initialization and temporal refinement. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10061–10072, 2023.

[24] Yuxi Xiao, Qianqian Wang, Shangzhan Zhang, Nan Xue, Sida Peng, Yujun Shen, and Xiaowei Zhou. Spatialtracker: Tracking any 2d pixels in 3d space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20406–20417, 2024.

[25] Jenny Seidenschwarz, Qunjie Zhou, Bardienus Duisterhof, Deva Ramanan, and Laura Leal-Taixé. Dynomo: Online point tracking by dynamic online monocular gaussian reconstruction. *arXiv preprint arXiv:2409.02104*,

2024.

[26] Hanxiao Jiang, Hao-Yu Hsu, Kaifeng Zhang, Hsin-Ni Yu, Shenlong Wang, and Yunzhu Li. Phystwin: Physics-informed reconstruction and simulation of deformable objects from videos. *arXiv preprint arXiv:2503.17973*, 2025.

[27] McCoy Becker, Mathieu Huot, Sam Ritchie, and Colin Smith. GenJAX: Probabilistic Programming with Gen, built on top of JAX. URL https://github.com/ChiSym/genjax.

[28] McCoy R Becker, Alexander K Lew, Xiaoyan Wang, Matin Ghavami, Mathieu Huot, Martin C Rinard, and Vikash K Mansinghka. Probabilistic programming with programmable variational inference. *Proceedings of the ACM on Programming Languages*, 8(PLDI):2123–2147, 2024.

[29] Kevin P Murphy. Conjugate bayesian analysis of the gaussian distribution. *def*, 1(2σ2):16, 2007.

[30] Andrew Gelman, John B Carlin, Hal S Stern, and Donald B Rubin. *Bayesian data analysis*. Chapman and Hall/CRC, 1995.

[31] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/jax-ml/jax.

[32] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[33] Sili Chen, Hengkai Guo, Shengnan Zhu, Feihu Zhang, Zilong Huang, Jiashi Feng, and Bingyi Kang. Video depth anything: Consistent depth estimation for super-long videos. *arXiv preprint arXiv:2501.12375*, 2025.

[34] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 402–419. Springer, 2020.

[35] David JC MacKay. Bayesian interpolation. *Neural computation*, 4(3):415–447, 1992.

[36] Carl Rasmussen and Zoubin Ghahramani. Occam's razor. *Advances in neural information processing systems*, 13, 2000.

[37] Veronica E Arriola-Rios, Puren Guler, Fanny Ficuciello, Danica Kragic, Bruno Siciliano, and Jeremy L Wyatt. Modeling of deformable objects for robotic manipulation: A tutorial and review. *Frontiers in Robotics and AI*, 7:82, 2020.

[38] Vivekananda Roy. Convergence diagnostics for markov chain monte carlo. *Annual Review of Statistics and Its Application*, 7(1):387–412, 2020.

[39] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. Technical report, Stanford, 2006.

[40] Wolfgang Kabsch. A solution for the best rotation to relate two sets of vectors. *Foundations of Crystallography*, 32(5):922–923, 1976.

APPENDIX A
BLOCKED GIBBS SAMPLING

We describe the Gibbs sampling approach in greater detail than in the main text. We first independently describe each blocked Gibbs step in Appendix A-A. Then, we describe the procedure of these steps used for initialization in Appendix A-B and tracking in Appendix A-C.

A. Gibbs Update Steps

There are twelve variables of interest, separated at different hierarchical levels as shown:

1) Cluster-level variables: $\{\boldsymbol{\mu}_k^{\mathcal{H}},\ \boldsymbol{\Sigma}_k^{\mathcal{H}},\ \mathbf{R}_k,\ \mathbf{t}_k, \pi_k^{\mathcal{H}}\}_{k=1}^{K}$
2) Particle-level variables: $\{\boldsymbol{\mu}_\ell^{\mathcal{B}},\ \boldsymbol{\Sigma}_\ell^{\mathcal{B}},\ \mathbf{v}_\ell,\ \boldsymbol{\Sigma}_\ell^{\mathcal{V}},\ z_\ell^{\mathcal{H}}, \pi_\ell^{\mathcal{B}}\}_{\ell=1}^{L}$
3) Datapoint-level variables: $\{z_n^{\mathcal{B}}\}_{n=1}^{N}$

For each of these variables, we independently describe each of the Gibbs updates.

1) Datapoint-to-Particle Assignments ($z_{1:N}^{\mathcal{B}}$): We update each datapoint's particle assignment $z_n^{\mathcal{B}}$ for $n = 1, \ldots, N$, using the conditional:

$$p(z_n^{\mathcal{B}} = \ell \mid \mathbf{x}_n, \mathbf{v}_n, \text{rest}) \propto \pi^{\mathcal{B}}(\ell) \cdot \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_\ell^{\mathcal{B}}, \boldsymbol{\Sigma}_\ell^{\mathcal{B}}) \quad \cdot \mathcal{N}(\mathbf{v}_n \mid \mathbf{v}_\ell, \boldsymbol{\Sigma}_\ell^{\mathcal{V}})$$

The prior is given by categorical weights $\pi^{\mathcal{B}}$; the likelihood is a product of two Gaussians over position $\mathbf{x}_n$ and velocity $\mathbf{v}_n$. We compute unnormalized log-probabilities $\tilde{p}_{n,\ell}$ for each particle:

$$\tilde{p}_{n,\ell} = \log \pi^{\mathcal{B}}(\ell) + \log \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_\ell^{\mathcal{B}}, \boldsymbol{\Sigma}_\ell^{\mathcal{B}}) + \log \mathcal{N}(\mathbf{v}_n \mid \mathbf{v}_\ell, \boldsymbol{\Sigma}_\ell^{\mathcal{V}})$$

and normalize to obtain the categorical:

$$p(z_n^{\mathcal{B}} = \ell) = \frac{\exp(\tilde{p}_{n,\ell})}{\sum_{\ell'=1}^{L} \exp(\tilde{p}_{n,\ell'})}$$

from which we sample:

$$z_n^{\mathcal{B}} \sim \text{Categorical}(p(z_n^{\mathcal{B}} = 1), \ldots, p(z_n^{\mathcal{B}} = L))$$

All datapoints are jointly reassigned in a blocked manner, each selecting the particle that best explains its position and motion, weighted by the prior over particles.

2) Particle Mixture Weights $\boldsymbol{\pi}^{\mathcal{B}}$: We update the particle mixture weights $\boldsymbol{\pi}^{\mathcal{B}}$ conditioned on datapoint-to-particle assignments $\{z_n^{\mathcal{B}}\}$. By Dirichlet–Categorical conjugacy, the conditional distribution becomes:

$$\boldsymbol{\pi}^{\mathcal{B}} \mid \{z_n^{\mathcal{B}}\} \sim \text{Dir}(\beta_1 + M_1, \ldots, \beta_L + M_L)$$

where $M_\ell = \#\{n : z_n^{\mathcal{B}} = \ell\}$ counts how many datapoints are currently assigned to each particle ℓ. This step reweights the prior particle proportions according to updated datapoint assignments.

3) Particle Spatial Means $\boldsymbol{\mu}_\ell^{\mathcal{B}}$: We update each particle center $\boldsymbol{\mu}_\ell^{\mathcal{B}}$ from its Gaussian conditional, combining: (1) a spatial prior from its assigned cluster, (2) position likelihoods from assigned datapoints, and (3) a velocity constraint derived from rigid motion.

Let $\mathbf{A}_\ell = \mathbf{R}_{z_\ell^{\mathcal{H}}} - \mathbf{I}$ and $\mathbf{b}_\ell = \mathbf{t}_{z_\ell^{\mathcal{H}}} - \mathbf{A}_\ell \boldsymbol{\mu}_{z_\ell^{\mathcal{H}}}^{\mathcal{H}}$. Then:

$$\mathbf{v}_\ell \sim \mathcal{N}(\mathbf{A}_\ell \boldsymbol{\mu}_\ell^{\mathcal{B}} + \mathbf{b}_\ell, \sigma_V^2 \mathbf{I})$$

The conditional distribution is a Gaussian-Gaussian conjugate of the form:

$$\boldsymbol{\mu}_\ell^{\mathcal{B}} \mid \boldsymbol{\mu}_{z_\ell^{\mathcal{H}}}^{\mathcal{H}}, \boldsymbol{\Sigma}_{z_\ell^{\mathcal{H}}}^{\mathcal{H}}, \mathbf{v}_\ell, \mathbf{t}_{z_\ell^{\mathcal{H}}}, \mathbf{R}_{z_\ell^{\mathcal{H}}}, \sigma_V^2, \{\mathbf{x}_n : z_n^{\mathcal{B}} = \ell\}, \boldsymbol{\Sigma}_\ell^{\mathcal{B}} \sim \mathcal{N}(\mathbf{P}_\ell^{-1}\mathbf{m}_\ell, \mathbf{P}_\ell^{-1})$$

with precision and mean:

$$\mathbf{P}_\ell = (\boldsymbol{\Sigma}_{z_\ell^{\mathcal{H}}}^{\mathcal{H}})^{-1} + N_\ell (\boldsymbol{\Sigma}_\ell^{\mathcal{B}})^{-1} + \frac{1}{\sigma_V^2} \mathbf{A}_\ell^\top \mathbf{A}_\ell$$

$$\mathbf{m}_\ell = (\boldsymbol{\Sigma}_{z_\ell^{\mathcal{H}}}^{\mathcal{H}})^{-1} \boldsymbol{\mu}_{z_\ell^{\mathcal{H}}}^{\mathcal{H}} + (\boldsymbol{\Sigma}_\ell^{\mathcal{B}})^{-1} \mathbf{S}_\ell + \frac{1}{\sigma_V^2} \mathbf{A}_\ell^\top (\mathbf{v}_\ell - \mathbf{b}_\ell)$$

where N_ℓ is the number of datapoints assigned to particle ℓ, and $\mathbf{S}_\ell = \sum_{n : z_n^{\mathcal{B}} = \ell} \mathbf{x}_n$ is the sum of their positions.

4) Particle Spatial Covariances $\boldsymbol{\Sigma}_\ell^{\mathcal{B}}$: We update each particle's spatial covariance matrix $\boldsymbol{\Sigma}_\ell^{\mathcal{B}}$ using Normal–Inverse-Wishart conjugacy. Let $N_\ell = \#\{n : z_n^{\mathcal{B}} = \ell\}$ be the number of datapoints assigned to particle ℓ, and define the scatter matrix:

$$\mathbf{S}_\ell = \sum_{n : z_n^{\mathcal{B}} = \ell} (\mathbf{x}_n - \boldsymbol{\mu}_\ell^{\mathcal{B}})(\mathbf{x}_n - \boldsymbol{\mu}_\ell^{\mathcal{B}})^\top$$

Given an Inverse-Wishart prior $\mathcal{W}^{-1}(\boldsymbol{\Psi}^{\mathcal{B}}, \nu^{\mathcal{B}})$, the conditional distribution is:

$$\boldsymbol{\Sigma}_\ell^{\mathcal{B}} \mid \boldsymbol{\mu}_\ell^{\mathcal{B}}, \{\mathbf{x}_n : z_n^{\mathcal{B}} = \ell\} \sim \mathcal{W}^{-1}(\boldsymbol{\Psi}'_\ell = \boldsymbol{\Psi}^{\mathcal{B}} + \mathbf{S}_\ell, \nu^{\mathcal{B}} + N_\ell)$$

This update adjusts each particle's spatial uncertainty based on the observed spread of its assigned datapoints.

5) Particle Velocity Means $\mathbf{v}_\ell$: We update each particle velocity anchor $\mathbf{v}_\ell$ via a Gaussian conditional distribution combining: (1) a rigid motion prior from its assigned cluster, and (2) velocity observations from assigned datapoints. Let $\bar{\mathbf{v}}_\ell = \mathbf{t}_{z_\ell^{\mathcal{H}}} + (\mathbf{R}_{z_\ell^{\mathcal{H}}} - \mathbf{I})(\boldsymbol{\mu}_\ell^{\mathcal{B}} - \boldsymbol{\mu}_{z_\ell^{\mathcal{H}}}^{\mathcal{H}})$ be the prior mean.

Given the set $\{\mathbf{v}_n : z_n^{\mathcal{B}} = \ell\}$ and count $N_\ell = \#\{n : z_n^{\mathcal{B}} = \ell\}$, the conditional is a Gaussian-Gaussian conjugate update:

$$\mathbf{v}_\ell \mid \bar{\mathbf{v}}_\ell, \sigma_V^2, \boldsymbol{\Sigma}_\ell^{\mathcal{V}}, \{\mathbf{v}_n : z_n^{\mathcal{B}} = \ell\} \sim \mathcal{N}(\boldsymbol{\mu}_\ell^v, \boldsymbol{\Sigma}_\ell^v)$$

with:

$$(\boldsymbol{\Sigma}_\ell^v)^{-1} = \frac{1}{\sigma_V^2}\mathbf{I} + N_\ell (\boldsymbol{\Sigma}_\ell^{\mathcal{V}})^{-1}$$

$$\boldsymbol{\mu}_\ell^v = \boldsymbol{\Sigma}_\ell^v \left(\frac{1}{\sigma_V^2} \bar{\mathbf{v}}_\ell + (\boldsymbol{\Sigma}_\ell^{\mathcal{V}})^{-1} \sum_{n : z_n^{\mathcal{B}} = \ell} \mathbf{v}_n \right)$$

This update accounts for the velocity prediction from the cluster's rigid transform along with the empirical datapoint velocities, with each contribution weighted by its respective uncertainty.

6) Particle Velocity Covariances $\mathbf{\Sigma}_\ell^{\mathcal{V}}$: Each particle's velocity covariance $\mathbf{\Sigma}_\ell^{\mathcal{V}}$ is inferred using Normal–Inverse-Wishart conjugacy. Let $N_\ell = \#\{n : z_n^{\mathcal{B}} = \ell\}$ be the number of datapoints assigned to particle ℓ, and define the velocity scatter:

$$\mathbf{T}_\ell = \sum_{n:z_n^{\mathcal{B}}=\ell} (\mathbf{v}_n - \mathbf{v}_\ell)(\mathbf{v}_n - \mathbf{v}_\ell)^\top$$

Given a prior $\mathcal{W}^{-1}(\mathbf{\Psi}^{\mathcal{V}}, \nu^{\mathcal{V}})$, the conditional distribution is:

$$\mathbf{\Sigma}_\ell^{\mathcal{V}} \mid \mathbf{v}_\ell, \{\mathbf{v}_n : z_n^{\mathcal{B}} = \ell\} \sim \mathcal{W}^{-1}(\mathbf{\Psi}'_\ell = \mathbf{\Psi}^{\mathcal{V}} + \mathbf{T}_\ell,\ \nu^{\mathcal{V}} + N_\ell)$$

This update reflects the velocity noise structure within each particle, accounting for spread in assigned datapoint velocities.

7) Particle-to-Cluster Assignments ($z_{1:L}^{\mathcal{H}}$): We update each particle's cluster assignment $z_\ell^{\mathcal{H}}$ for $\ell = 1, \ldots, L$, using the conditional:

$$p(z_\ell^{\mathcal{H}} = k \mid \boldsymbol{\mu}_\ell^{\mathcal{B}}, \mathbf{v}_\ell, \text{rest}) \propto \pi^{\mathcal{H}}(k) \cdot \mathcal{N}(\boldsymbol{\mu}_\ell^{\mathcal{B}} \mid \boldsymbol{\mu}_k^{\mathcal{H}}, \mathbf{\Sigma}_k^{\mathcal{H}}) \cdot \mathcal{N}\big(\mathbf{v}_\ell \mid \mathbf{t}_k + \mathbf{R}_k(\boldsymbol{\mu}_\ell^{\mathcal{B}} - \boldsymbol{\mu}_k^{\mathcal{H}}), \sigma_V^2 \mathbf{I}\big)$$

The prior is given by categorical weights $\pi^{\mathcal{H}}$; the likelihood combines a spatial Gaussian over the particle's position $\boldsymbol{\mu}_\ell^{\mathcal{B}}$ and a velocity Gaussian that accounts for rigid-body motion induced by the cluster's rotation $\mathbf{R}_k$ and translation $\mathbf{t}_k$. We compute unnormalized log-probabilities $\tilde{p}_{\ell,k}$ for each cluster:

$$\tilde{p}_{\ell,k} = \log \pi^{\mathcal{H}}(k) + \log \mathcal{N}(\boldsymbol{\mu}_\ell^{\mathcal{B}} \mid \boldsymbol{\mu}_k^{\mathcal{H}}, \mathbf{\Sigma}_k^{\mathcal{H}}) + \log \mathcal{N}(\mathbf{v}_\ell \mid \mathbf{t}_k + (\mathbf{R}_k - \mathbf{I})(\boldsymbol{\mu}_\ell^{\mathcal{B}} - \boldsymbol{\mu}_k^{\mathcal{H}}),\ \sigma_V^2 \mathbf{I})$$

and normalize to obtain the categorical:

$$p(z_\ell^{\mathcal{H}} = k) = \frac{\exp(\tilde{p}_{\ell,k})}{\sum_{k'=1}^K \exp(\tilde{p}_{\ell,k'})}$$

from which we sample:

$$z_\ell^{\mathcal{H}} \sim \text{Categorical}(p(z_\ell^{\mathcal{H}} = 1), \ldots, p(z_\ell^{\mathcal{H}} = K))$$

This constitutes a blocked Gibbs step, where all particle-to-cluster assignments are jointly updated. Each particle selects the cluster whose spatial and rigid motion parameters best explain its position and velocity.

8) Cluster Mixture Weights $\boldsymbol{\pi}^{\mathcal{H}}$: We update the cluster mixture weights $\boldsymbol{\pi}^{\mathcal{H}}$ given particle-to-cluster assignments $\{z_\ell^{\mathcal{H}}\}$. Using Dirichlet–Categorical conjugacy, the conditional is:

$$\boldsymbol{\pi}^{\mathcal{H}} \mid \{z_\ell^{\mathcal{H}}\} \sim \text{Dir}(\alpha_1 + N_1, \ldots, \alpha_K + N_K)$$

where $N_k = \#\{\ell : z_\ell^{\mathcal{H}} = k\}$ is the number of particles assigned to cluster k. This step updates the prior cluster proportions based on current assignment counts.

9) Cluster Spatial Means $\boldsymbol{\mu}_k^{\mathcal{H}}$: We update each cluster center $\boldsymbol{\mu}_k^{\mathcal{H}}$ via a Gaussian conditional that integrates: (1) a Gaussian prior centered at $\mu^{\mathcal{H}}$, (2) assigned particle centers $\boldsymbol{\mu}_\ell^{\mathcal{B}}$, and (3) observed particle velocities corrected by the cluster's affine transform.

Let $\mathbf{A}_k = \mathbf{I} - \mathbf{R}_k$ and $\mathbf{b}_\ell = \mathbf{t}_k - \mathbf{A}_k \boldsymbol{\mu}_\ell^{\mathcal{B}}$. Then the velocity residual is:

$$\mathbf{r}_\ell = \mathbf{v}_\ell - \mathbf{b}_\ell$$

Given the sum of assigned particle means $\mathbf{S}_k = \sum_{\ell:z_\ell^{\mathcal{H}}=k} \boldsymbol{\mu}_\ell^{\mathcal{B}}$, the velocity residual sum $\mathbf{R}_k = \sum_{\ell:z_\ell^{\mathcal{H}}=k} \mathbf{r}_\ell$, and the count $N_k = \#\{\ell : z_\ell^{\mathcal{H}} = k\}$ of particles assigned to cluster k, the conditional is:

$$\boldsymbol{\mu}_k^{\mathcal{H}} \mid \mu^{\mathcal{H}},\ \sigma_H^2,\ \mathbf{\Sigma}_k^{\mathcal{H}},\ \mathbf{t}_k,\ \mathbf{R}_k,\ \sigma_V^2,\ \{\boldsymbol{\mu}_\ell^{\mathcal{B}},\ \mathbf{v}_\ell : z_\ell^{\mathcal{H}} = k\} \sim \mathcal{N}(P_k^{-1}\mathbf{m}_k,\ P_k^{-1})$$

with:

$$P_k = \frac{1}{\sigma_H^2}\mathbf{I} + N_k \left(\mathbf{\Sigma}_k^{\mathcal{H}\,-1} + \frac{1}{\sigma_V^2}\mathbf{A}_k^\top \mathbf{A}_k\right)$$

$$\mathbf{m}_k = \frac{1}{\sigma_H^2}\mu^{\mathcal{H}} + \mathbf{\Sigma}_k^{\mathcal{H}\,-1}\mathbf{S}_k + \frac{1}{\sigma_V^2}\mathbf{A}_k^\top \mathbf{R}_k$$

This update integrates global priors, spatial evidence from assigned particles, and velocity-based constraints under rigid motion. We parallelize this step by batching cluster-level quantities over K and particle-level inputs over L, with per-cluster residual aggregation. The final blocked multivariate normal update samples new cluster means in parallel from their respective posteriors.

10) Cluster Spatial Covariances $\mathbf{\Sigma}_k^{\mathcal{H}}$: We infer each cluster's spatial covariance $\mathbf{\Sigma}_k^{\mathcal{H}}$ using a Normal–Inverse-Wishart update conditioned on its assigned particles. Let $L_k = \#\{\ell : z_\ell^{\mathcal{H}} = k\}$ be the number of particles assigned to cluster k, and define the cluster-centered scatter:

$$\mathbf{S}_k = \sum_{\ell:z_\ell^{\mathcal{H}}=k} (\boldsymbol{\mu}_\ell^{\mathcal{B}} - \boldsymbol{\mu}_k^{\mathcal{H}})(\boldsymbol{\mu}_\ell^{\mathcal{B}} - \boldsymbol{\mu}_k^{\mathcal{H}})^\top$$

Given the Inverse-Wishart prior $\mathcal{W}^{-1}(\mathbf{\Psi}^{\mathcal{H}}, \nu^{\mathcal{H}})$, the conditional becomes:

$$\mathbf{\Sigma}_k^{\mathcal{H}} \mid \boldsymbol{\mu}_k^{\mathcal{H}}, \{\boldsymbol{\mu}_\ell^{\mathcal{B}} : z_\ell^{\mathcal{H}} = k\} \sim \mathcal{W}^{-1}(\mathbf{\Psi}'_k = \mathbf{\Psi}^{\mathcal{H}} + \mathbf{S}_k,\ \nu^{\mathcal{H}} + L_k)$$

This posterior captures the spatial extent of each cluster based on the spread of its assigned particle centers.

11) Cluster Rotation $\mathbf{R}_k$: We update each cluster's rotation matrix $\mathbf{R}_k$ by evaluating a discrete set of candidate rotations $\{\mathbf{R}^{(j)}\}_{j=1}^{M_r}$ drawn from a spherical cap (e.g., von Mises–Fisher). For each candidate, we compute a probability based on how well the induced rigid motion explains observed particle velocities. Let $\bar{\mathbf{v}}_\ell^{(j)} = \mathbf{t}_k + (\mathbf{R}^{(j)} - \mathbf{I})(\boldsymbol{\mu}_\ell^{\mathcal{B}} - \boldsymbol{\mu}_k^{\mathcal{H}})$ be the expected velocity for particle ℓ under candidate j. Then:

$$\log \tilde{q}_j = \sum_{\ell:z_\ell^{\mathcal{H}}=k} \log \mathcal{N}(\mathbf{v}_\ell \mid \bar{\mathbf{v}}_\ell^{(j)}, \sigma_V^2 \mathbf{I})$$

Adding the prior log-probabilities $\log p(\mathbf{R}^{(j)})$, we normalize the log-scores to obtain:

$$q_j = \frac{\exp(\log \tilde{q}_j + \log p(\mathbf{R}^{(j)}))}{\sum_{j'=1}^{M_r} \exp(\log \tilde{q}_{j'} + \log p(\mathbf{R}^{(j')}))}$$

from which we sample:

$$\mathbf{R}_k \sim \text{Categorical}(\{q_j\}_{j=1}^{M_r})$$

This update selects the rotation that best aligns relative particle positions with their observed velocities, conditioned on the current cluster translation $\mathbf{t}_k$, velocity noise σ_V^2,cluster means ($\boldsymbol{\mu}_k^{\mathcal{H}}$) and assigned particle means ($\{\boldsymbol{\mu}_\ell^{\mathcal{B}} : z_\ell^{\mathcal{H}} = k\}$).

12) Cluster Translation Velocities $\mathbf{t}_k$*:* We update each cluster's translation velocity $\mathbf{t}_k$ by evaluating a discrete set of candidate translations $\{\mathbf{t}^{(m)}\}_{m=1}^{M_t}$ sampled from an isotropic Gaussian prior $\mathcal{N}(\mathbf{0}, s^2\mathbf{I})$. Each candidate is scored based on how well it explains the observed particle velocities under the current rotation $\mathbf{R}_k$. Let $\bar{\mathbf{v}}_\ell^{(m)} = \mathbf{t}^{(m)} + (\mathbf{R}_k - \mathbf{I})(\boldsymbol{\mu}_\ell^{\mathcal{B}} - \boldsymbol{\mu}_k^{\mathcal{H}})$ be the expected velocity for particle ℓ under candidate m. Then:

$$\log \tilde{p}_m = \sum_{\ell : z_\ell^{\mathcal{H}} = k} \log \mathcal{N}(\mathbf{v}_\ell \mid \bar{\mathbf{v}}_\ell^{(m)}, \sigma_V^2 \mathbf{I})$$

We add prior log-probabilities and normalize to form a categorical:

$$p_m = \frac{\exp(\log \tilde{p}_m + \log p(\mathbf{t}^{(m)}))}{\sum_{m'=1}^{M_t} \exp(\log \tilde{p}_{m'} + \log p(\mathbf{t}^{(m')}))}$$

from which we sample:

$$\mathbf{t}_k \sim \text{Categorical}(\{p_m\}_{m=1}^{M_t})$$

This update selects the translation that best explains the observed particle velocities, conditioned on current cluster rotation $\mathbf{R}_k$, velocity noise σ_V^2, cluster center $\boldsymbol{\mu}_k^{\mathcal{H}}$, and assigned particle means $\{\boldsymbol{\mu}_\ell^{\mathcal{B}} : z_\ell^{\mathcal{H}} = k\}$.

B. Initialization Procedure

It is well known that MCMC chains are sensitive to the initialization and should be initialized at a high density region [38]. Hence, we use K-Means clustering and a data-driven approach to initialize the MCMC chain for the initial frame ($T = 0$). This is denoted by the **Initial Scene Clustering** components of Figure 2.

1) K-Means and Data-driven Initialization at $T = 0$*:* Given the number of particles (L), we use K-means via a K-Means++ [39] initialization to initialize the particle spatial positions ($\boldsymbol{\mu}_\ell^{\mathcal{B}}$). Note that for the RGB video experiment on the DAVIS subset in section V, we assign $\lceil \gamma \cdot L \rceil$ particles to the pixels corresponding to the segmentation mask provided at $T = 0$, where $\gamma = \frac{|\mathcal{M}|}{|\mathcal{I}|}$ denotes the fraction of pixels within the segmentation mask $\mathcal{M}$ relative to the total image $\mathcal{I}$. We then use an additional K-means step to initialize the cluster spatial positions ($\boldsymbol{\mu}_k^{\mathcal{H}}$) by treating the particle spatial positions as datapoints to cluster. Note that because we have the segmentation mask of the object-of-interest in the experiment from section V, we assign all those particles to be the same cluster.

This K-means initialization provides initial values for assignments at both layers ($z_n^{\mathcal{B}}$, $z_\ell^{\mathcal{H}}$). We then use these assignments to initialize the mixture weights at both layers ($\pi^{\mathcal{B}}$, $\pi^{\mathcal{H}}$) by computing the empirical frequencies of each cluster and normalizing: $\pi_\ell^{\mathcal{B}} = \frac{M_\ell}{N}$ and $\pi_k^{\mathcal{H}} = \frac{N_k}{L}$, where M_ℓ is the number of datapoints assigned to particle ℓ and N_k is the number of particles assigned to cluster k. We initialize the velocity mean of each particle $\mathbf{v}_\ell$ by averaging the observed velocities of the datapoints assigned to it:

$$\mathbf{v}_\ell = \frac{1}{M_\ell} \sum_{n : z_n^{\mathcal{B}} = \ell} \mathbf{v}_n.$$

To initialize the covariance matrices, we compute the sample covariance of the relevant residuals for each component:

1) **Particle Spatial Covariance:**

$$\boldsymbol{\Sigma}_\ell^{\mathcal{B}} = \frac{1}{M_\ell - 1} \sum_{n : z_n^{\mathcal{B}} = \ell} (\mathbf{x}_n - \boldsymbol{\mu}_\ell^{\mathcal{B}})(\mathbf{x}_n - \boldsymbol{\mu}_\ell^{\mathcal{B}})^\top.$$

2) **Particle Velocity Covariance:**

$$\boldsymbol{\Sigma}_\ell^{\mathcal{V}} = \frac{1}{M_\ell - 1} \sum_{n : z_n^{\mathcal{B}} = \ell} (\mathbf{v}_n - \mathbf{v}_\ell)(\mathbf{v}_n - \mathbf{v}_\ell)^\top.$$

3) **Cluster Spatial Covariance:**

$$\boldsymbol{\Sigma}_k^{\mathcal{H}} = \frac{1}{N_k - 1} \sum_{\ell : z_\ell^{\mathcal{H}} = k} (\boldsymbol{\mu}_\ell^{\mathcal{B}} - \boldsymbol{\mu}_k^{\mathcal{H}})(\boldsymbol{\mu}_\ell^{\mathcal{B}} - \boldsymbol{\mu}_k^{\mathcal{H}})^\top.$$

To initialize each cluster's rigid transform $(\mathbf{R}_k, \mathbf{t}_k)$, we apply the Kabsch algorithm [40] to align assigned particle positions with their next-frame displacements. For cluster k, we collect all datapoints $\mathbf{x}_n$ assigned to particles ℓ with $z_\ell^{\mathcal{H}} = k$ and define their estimated displacements $\mathbf{x}'_n = \mathbf{x}_n + \mathbf{v}_n$. Let $\mathcal{X}_k = \{\mathbf{x}_n\}$ and $\mathcal{X}'_k = \{\mathbf{x}'_n\}$ be the source and target sets.

We compute centroids $\bar{\mathbf{x}}_k = \frac{1}{|\mathcal{X}_k|} \sum \mathbf{x}_n$, $\bar{\mathbf{x}}'_k = \frac{1}{|\mathcal{X}'_k|} \sum \mathbf{x}'_n$, and form centered sets $\tilde{\mathbf{x}}_n = \mathbf{x}_n - \bar{\mathbf{x}}_k$, $\tilde{\mathbf{x}}'_n = \mathbf{x}'_n - \bar{\mathbf{x}}'_k$. The cross-covariance matrix is:

$$\mathbf{H}_k = \sum_n \tilde{\mathbf{x}}_n \tilde{\mathbf{x}}_n'^\top$$

We compute the singular value decomposition $\mathbf{H}_k = \mathbf{U}_k \boldsymbol{\Sigma}_k \mathbf{V}_k^\top$, and define the optimal rotation as:

$$\mathbf{R}_k = \mathbf{V}_k \mathbf{D}_k \mathbf{U}_k^\top$$

where

$$\mathbf{D}_k = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \det(\mathbf{V}_k \mathbf{U}_k^\top) \end{bmatrix}$$

The corresponding translation is:

$$\mathbf{t}_k = \bar{\mathbf{x}}'_k - \mathbf{R}_k \bar{\mathbf{x}}_k$$

This provides an initialization of cluster motion consistent with the observed displacements of assigned particles. The update is applied independently for each cluster $k = 1, \ldots, K$.

2) Data-Dependent Hyperparameters: We initialize model hyperparameters directly from empirical statistics computed on the initial frame ($T = 0$). The global cluster location prior $\mu^{\mathcal{H}}$ is set to the median datapoint position, while the prior spatial scale $\boldsymbol{\Psi}^{\mathcal{B}}, \boldsymbol{\Psi}^{\mathcal{H}}, \boldsymbol{\Psi}^{\mathcal{V}}$ are initialized using the median initialized particle and cluster covariances length scales.

The degrees of freedom $\nu^{\mathcal{B}}, \nu^{\mathcal{H}}, \nu^{\mathcal{V}}$ are initialized proportionally to the number of datapoints assigned, weighted by particle or cluster weights:

$$\nu^{\mathcal{B}} = \left\lfloor \text{median}(w_\ell^{\mathcal{B}} \cdot N) \right\rfloor, \quad \nu^{\mathcal{H}} = \left\lfloor \text{median}(w_k^{\mathcal{H}} \cdot N) \right\rfloor,$$
$$\nu^{\mathcal{V}} = \left\lfloor \text{median}(w_\ell^{\mathcal{B}} \cdot N) \right\rfloor$$

where $w_\ell^{\mathcal{B}}$ and $w_k^{\mathcal{H}}$ are the normalized empirical weights of each particle and cluster.

Translation and rotation priors are defined using discretized supports. We set $\kappa^{\text{vmf}} = 100, \theta_{\max} = 25^\circ, s = 0.2$. All hyperparameters are kept constant across all videos in each experiment.

3) Initialization Gibbs: To infer the initial representation at the first frame, we perform an initial set of block Gibbs sweeps. We run 15 sweeps over datapoint and particle-level variables (including particle spatial covariance) only, keeping cluster-level variables fixed. This choice reflects the fact that cluster assignments are anchored by the object mask in the first frame, and early sweeps are more effectively used to resolve fine-grained datapoint-to-particle assignments.

C. Tracking Gibbs Procedure

To perform inference over video sequences, we extend our generative particle model into the sequential filtering regime using a structured Markov Chain Monte Carlo (MCMC) procedure. Specifically, we implement a blocked Gibbs sampler that leverages the causal ordering of the variables from the previous frame to initialize each frame and performs bottom-up inference to refine all datapoint-, particle-, and cluster-level variables. Our approach maintains a tractable posterior approximation at each timestep by propagating forward a subset of latent variables and resampling the remaining ones conditioned on new observations. This sequential per-frame MCMC design supports inference in dynamic scenes where data associations must be re-inferred at every timestep.

At each timestep t, we target the posterior over latent structure given the observed datapoint positions $\mathbf{x}^t_{1:N}$ and velocities $\mathbf{v}^t_{1:N}$:

$$p(\boldsymbol{\mu}^t_{\mathcal{H}},\, \boldsymbol{\Sigma}^t_{\mathcal{H}},\, \mathbf{R}^t_{\mathcal{H}},\, \mathbf{t}^t_{\mathcal{H}},\, \boldsymbol{\mu}^t_{\mathcal{B}},\, \mathbf{v}^t_{\mathcal{B}},\, \boldsymbol{\Sigma}^t_{\mathcal{V}},\\ z^t_{1:N},\, z^t_{1:L},\, \boldsymbol{\pi}^t_{\mathcal{B}},\, \boldsymbol{\pi}^t_{\mathcal{H}} \mid \mathbf{x}^t_{1:N},\, \mathbf{v}^t_{1:N})$$

where $\boldsymbol{\Sigma}_{\mathcal{B}}$ (particle spatial covariances) are held fixed throughout tracking to preserve the spatial extent of the deformable visual matter represented by each particle, and particle-to-cluster assignments $z^t_{1:L}$ are held fixed to keep consistency with the initial object segmentation mask.

a) Particle Propagation and Initialization: Each frame begins by propagating the inferred particle means using their previously inferred velocity vectors:

$$\tilde{\boldsymbol{\mu}}^{\mathcal{B},t}_{\ell} = \boldsymbol{\mu}^{\mathcal{B},t-1}_{\ell} + \mathbf{v}^{t-1}_{\ell}$$

This serves as an initialization for the particle positions in the next frame.

b) First Assignment: Spatial Anchoring: Datapoints are first assigned to particles based on spatial likelihoods alone:

$$p(z^{\mathcal{B},t}_n = \ell \mid \mathbf{x}^t_n) \propto \pi^{\mathcal{B}}_{\ell} \cdot \mathcal{N}(\mathbf{x}^t_n \mid \tilde{\boldsymbol{\mu}}^{\mathcal{B},t}_{\ell}, \boldsymbol{\Sigma}^{\mathcal{B}}_{\ell})$$

This step is crucial because, in the absence of known correspondences across frames, we cannot assume that datapoint n at time $t-1$ is the same as datapoint n at time t, that is, $\mathbf{x}^{t-1}_n \neq \mathbf{x}^t_n$ in general. Instead, we reinterpret each new frame as an unordered set of observations and rely on spatial proximity to propagated particle means to re-establish associations. By using position alone and excluding any top-down beliefs from velocity or cluster structure, this step provides a stable initialization for the rest of the Gibbs updates. Note that this is a partial version of the full assignment step described in Appendix A-A1, used here to anchor the initial framewise alignment. After assignments, we update the mixture weights $\boldsymbol{\pi}^{\mathcal{B}}$ by sampling from their conjugate Dirichlet distribution (Appendix A-A2).

c) Particle Mean Update: After datapoints have been assigned to particles based on spatial proximity, we update each particle's spatial mean to better reflect this assignment. Specifically, we sample the particle mean from its posterior conditioned on the assigned datapoints and the expected motion induced by its cluster assignment, as detailed in Appendix A-A3. Since the assignments in the previous Gibbs step compensate for the absence of pointwise correspondences, this update typically results in small adjustments to the propagated means, ensuring that particles remain anchored to observed data while maintaining temporal coherence with the previous frame.

d) Second Assignment and Particle Refinement: A second datapoint-to-particle assignment uses both spatial and velocity likelihoods as described in Appendix A-A1:

$$p(z^{\mathcal{B},t}_n = \ell \mid \mathbf{x}^t_n, \mathbf{v}^t_n) \propto \pi^{\mathcal{B}}_{\ell} \cdot \mathcal{N}(\mathbf{x}^t_n \mid \boldsymbol{\mu}^{\mathcal{B}}_{\ell}, \boldsymbol{\Sigma}^{\mathcal{B}}_{\ell}) \cdot \mathcal{N}(\mathbf{v}^t_n \mid \mathbf{v}_{\ell}, \boldsymbol{\Sigma}^{\mathcal{V}}_{\ell})$$

This step helps resolve ambiguous associations by combining spatial proximity with motion information. The mixture weights $\boldsymbol{\pi}^{\mathcal{B}}$ are updated again based on the refined assignments (Appendix A-A2).

Each particle's velocity mean $\mathbf{v}_{\ell}$ is updated from its posterior as described in Appendix A-A5, and the velocity covariance $\boldsymbol{\Sigma}^{\mathcal{V}}_{\ell}$ is resampled as shown in Appendix A-A6. These updates reflect the motion structure inferred from grouped datapoint velocities.

e) Cluster-level Updates: Each particle is assigned to a cluster using a joint spatial and velocity likelihood as described in Appendix A-A7, and the cluster mixture weights $\boldsymbol{\pi}^{\mathcal{H}}$ are resampled using the equation in Appendix A-A8. Conditioned on these assignments, the cluster mean $\boldsymbol{\mu}^{\mathcal{H}}_k$ and spatial covariance $\boldsymbol{\Sigma}^{\mathcal{H}}_k$ are updated from their conditional distributions (Appendix A-A9 and A-A10), and the rigid transform $(\mathbf{R}_k, \mathbf{t}_k)$ is inferred by categorical sampling over candidate rotations and translations (Appendix A-A11 and A-A12).

Particle-to-cluster assignments $z^t_{1:L}$ are held fixed throughout tracking to preserve consistency with the initial segmentation mask, which provides a reliable prior over object structure. However, cluster parameters including spatial statistics and rigid transforms are still inferred at each frame to update the spatial localization of the structure given in the original segmentation.

f) Sweep Schedule and Rationale: Cluster assignments are held fixed to preserve consistency with the initial segmentation, though cluster parameters are still inferred. In both cases, particle covariances remain fixed to preserve the spatial extent of deformable matter.

This bottom-up structure ensures that higher-level cluster inference remains grounded in updated particle and datapoint evidence. In the absence of pointwise correspondences between frames, low-level observations such as datapoint positions and motions serve as the most reliable source of information about current scene structure. By first resampling datapoint-to-particle assignments and then updating particle parameters, we allow the particle layer to accurately reflect the current frame's geometry and motion.

Only after particles are aligned to the new observations do we infer the cluster structure that explains their collective motion. The cluster layer thus becomes a statistic that reflects coherent groupings of particles based on their updated positions and velocities. Grounding cluster inference in current-frame particle evidence stabilizes inference by ensuring that clusters remain spatially localized and relevant to the current scene geometry. Once inferred, clusters act as structure-preserving priors that regularize their constituent particles.

APPENDIX B
COLOR-AUGMENTED VARIANT

A. Model Modification and Initialization

In the color-augmented variant of our model, used in the visualization shown in Figure 1, we assume that each particle is associated with a fixed RGB color, $c_\ell = (c_\ell^r, c_\ell^g, c_\ell^b)$. The sampling process of the datapoint color $c_n = (c_n^r, c_n^g, c_n^b)$ from the particle color is defined as a Gaussian with variance σ_C^2:

$$c_n \sim \mathcal{N}(c_\ell, \sigma_C^2 \mathbf{I})$$

Although this fixed-color assumption does not strictly hold in settings with dynamic lighting or changing object appearance, it remains a reasonable approximation for the controlled conditions in the demo of Figure 1. The goal of this demonstration is to highlight that the color-augmented version of our model can successfully track and maintain visual consistency of highly deformable structures, like Jello, over time.

We only fit our per-particle color parameter during initialization. We perform the steps described in Appendix A-B1, followed by computing the initial color of each particle c_ℓ as the average color of its assigned datapoints:

$$c_\ell^r = \frac{1}{M_\ell} \sum_{n: z_n^{\mathcal{B}} = \ell} c_n^r, \quad c_\ell^g = \frac{1}{M_\ell} \sum_{n: z_n^{\mathcal{B}} = \ell} c_n^g,$$
$$c_\ell^b = \frac{1}{M_\ell} \sum_{n: z_n^{\mathcal{B}} = \ell} c_n^b$$

where $M_\ell = \#\{n : z_n^{\mathcal{B}} = \ell\}$ is the number of datapoints assigned to particle ℓ. This RGB mean serves as the representative color for each particle throughout inference.

B. Datapoint-to-Particle Assignments ($z_{1:N}^{\mathcal{B}}$) with Color Likelihood

The main modification to the Gibbs sampler involves the datapoint-to-particle assignment step, which now also incorporates color similarity. We update each datapoint's particle assignment $z_n^{\mathcal{B}}$ for $n = 1, \ldots, N$, using the conditional distribution:

$$\begin{aligned} p(z_n^{\mathcal{B}} = \ell \mid \mathbf{x}_n, \mathbf{v}_n, c_n, \text{rest}) \propto \pi^{\mathcal{B}}(\ell) \cdot \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_\ell^{\mathcal{B}}, \boldsymbol{\Sigma}_\ell^{\mathcal{B}}) \\ \cdot \mathcal{N}(\mathbf{v}_n \mid \mathbf{v}_\ell, \boldsymbol{\Sigma}_\ell^{\mathcal{V}}) \\ \cdot \mathcal{N}(c_n \mid c_\ell, \sigma_C^2 \mathbf{I}) \end{aligned}$$

The prior is given by categorical weights $\pi^{\mathcal{B}}$, and the likelihood now consists of three independent Gaussian terms: one for position $\mathbf{x}_n$, one for velocity $\mathbf{v}_n$, and one for color c_n. The color likelihood uses a fixed spherical covariance $\sigma_C^2 \mathbf{I}$.

We compute unnormalized log-probabilities $\tilde{p}_{n,\ell}$ for each particle:

$$\begin{aligned} \tilde{p}_{n,\ell} = \log \pi^{\mathcal{B}}(\ell) + \log \mathcal{N}(\mathbf{x}_n \mid \boldsymbol{\mu}_\ell^{\mathcal{B}}, \boldsymbol{\Sigma}_\ell^{\mathcal{B}}) \\ + \log \mathcal{N}(\mathbf{v}_n \mid \mathbf{v}_\ell, \boldsymbol{\Sigma}_\ell^{\mathcal{V}}) \\ + \log \mathcal{N}(c_n \mid c_\ell, \sigma_C^2 \mathbf{I}) \end{aligned}$$

and normalize to obtain the categorical conditional distribution:

$$p(z_n^{\mathcal{B}} = \ell) = \frac{\exp(\tilde{p}_{n,\ell})}{\sum_{\ell'=1}^{L} \exp(\tilde{p}_{n,\ell'})}$$

from which we sample:

$$z_n^{\mathcal{B}} \sim \text{Categorical}(p(z_n^{\mathcal{B}} = 1), \ldots, p(z_n^{\mathcal{B}} = L))$$

This update is also a blocked update, executed in a computational manner similar to Appendix A-A1.