

A Cloth-Splatting Implementation

A.1 Action-Conditioned Dynamics Architecture and Training

The action-conditioned dynamics model builds upon the GNS architecture [4], which consists of three parts: encoder, processor, and decoder. The encoder consists of two MLPs, ϕ_p and ϕ_e , which map vertices and edge features into latent embeddings h_i and g_{jk} respectively. The processor comprises $L = 15$ Graph Network (GN) blocks with residual connections that propagate the information throughout the mesh. Each GN block includes an edge update MLP, a vertex update MLP, and a global update MLP. The decoder is an MLP ψ that outputs acceleration for each point: $\ddot{x}_i = \psi(h_i^L)$, which we use to update the position of each vertex of the cloth mesh via Euler integration.

The input vertex features consist of past $k = 3$ velocities and the vertex type. The vertex type is a binary flag used to distinguish grasped vertices from non-grasped vertices. The edge features include the distance vector $(v_j - v_k)$ and its norm $\|v_j - v_k\|$. To condition the model on the actions of the robot, we update the velocity of the pick point based on the robot’s action before giving the state of the cloth in input to the network. This facilitates the propagation of the actions throughout the GNS to predict future states.

We train the action-conditioned dynamics on towel objects, using the mean-squared error between predicted and simulator-obtained accelerations for 200 epochs using Adam [48].

A.2 Mesh-constrained Gaussian Splatting

For the mesh-constrained Gaussian Splatting, we build upon the original Gaussian Splatting procedure, with the main modification that we constrain the Gaussian positions on the surface of a pre-defined mesh as described in the 4.2. Details of Gaussian Splatting, such as the pruning, densification, and regular resetting of opacities, remain unchanged. Nevertheless, in order to keep the number of 3D Gaussians low, we increase the required opacity for Gaussians to not be pruned, since we can assume that there are no transparent parts on the reconstructed cloth. Therefore, a normal reconstruction of the appearance of cloth only requires about 4k Gaussians.

We observe that when the Gaussians are optimized over the whole range of training, the visual appearance and the tracking degrades. For example, the Gaussian position on the mesh starts to fit the deformed appearance instead of the residual dynamics model learning the proper offset. Therefore, the learning rates of the Gaussians’ attributes (color, position, scale, ...) are annealed over the first 6k iterations and afterward frozen so only the residual dynamics model is optimized.

A.3 Residual dynamics model

We implement the residual dynamics model as a 3-layer ReLU MLP with a width of 256. The input to the MLP is a scalar value in the range $0 - 1$, corresponding to the normalized time step, which is encoded with the sinusoidal frequency encoding also used in NeRF [49], using 6 frequencies. The output size is $3 \times N$, with N being the number of vertices in the mesh.

We randomly initialize weights and biases of the output layer with a zero-centered normal distribution with a covariance of 0.0001, to start with a residual close to zero.

A.4 Regularization

As discussed in Section 4.3, we learned the state updated by adding the following regularization losses: $\mathcal{L}_{\text{reg}} = \mathcal{L}_{\text{SSIM}} + \mathcal{L}_{\text{iso}} + \mathcal{L}_{\text{magn}}$, where $\mathcal{L}_{\text{SSIM}}$ is the SSIM loss [38], \mathcal{L}_{iso} ensures neighboring vertices in the cloth maintain a constant distance, and $\mathcal{L}_{\text{magn}}$ minimizes overall motion.

The isometric loss:

$$\mathcal{L}_{\text{iso}} = \sum_{t=0}^{T-1} \sum_{i=0}^{N-1} \sum_{\mathcal{N}(v_{t,i})} |d(v_{t,i}, v_{t,j}) - d(v_{t+1,i}, v_{t+1,j})| \quad (11)$$

ensures that the neighbouring vertices $\mathcal{N}(v_{t,i})$ of $v_{t,i}$ maintain a constant distance from time t to $t + 1$.

The structural similarity index measure loss (SSIM) [50] is estimated for windows of the images and goes beyond the purely per-pixel color loss in Eq. 10 and also considers the pixel neighbor. The loss between two windows w and v can be estimated with:

$$\mathcal{L}_{\text{SSIM}}(v, w) = \frac{(2\mu_v\mu_w + c_1)(2\sigma_{vw} + c_2)}{(\mu_v^2 + \mu_w^2 + c_1)(\sigma_v^2 + \sigma_w^2 + c_2)}, \quad (12)$$

where μ is the mean color of each window, σ^2 the color (co-)variances, and c_1 and c_2 are constants to stabilize the loss.

The motion loss:

$$\mathcal{L}_{\text{magn}} = \sum_{t=0}^{T-1} \sum_{i=0}^{N-1} \|v_{t,i} - v_{t+1,i}\|_2^2 \quad (13)$$

encourages to learn a solution with the smallest possible motion per vertice, which we found necessary to prevent instabilities during training.

B Synthetic Data

The synthetic dataset consists of meshes representing three types of cloth objects: t-shirts, shorts, and towels. We procedurally generate meshes with random configurations, sizes, and overall shapes for each category based on the methods detailed in [46]. Post-generation, the meshes are deformed using NVIDIA Flex [43, 44] with random manipulation trajectories.

The manipulation trajectories are constructed using quadratic Bézier curves with three control points. Specifically, the pick and place locations represent the primary control points, which we randomly selected on the cloth particles. The third control point, positioned midway between the pick and place points, was set to a random height within the range $[0.05, 0.15]$ cm. Additionally, this control point was randomly tilted between $[-\pi/4, \pi/4]$ rad around the axis formed by the pick and place points to add variability in the manipulation trajectories. We finally discretized the manipulation trajectory into a series of small displacements depending on the gripper velocity, $\Delta x_1, \dots, \Delta x_T$, ensuring:

$$x_{\text{pick}} + \sum_{i=1}^T \Delta x_i = x_{\text{place}},$$

randomly sampling the gripper velocity in the interval $[0.5, 2]$ cm/s.

To bridge the simulation-to-reality gap, we rendered the complete manipulation trajectory using Blender [45].

C Real-world Set-up and Data Collection

The real-world set-up is shown in Fig. 6. We used 3 calibrated RealSense d435 cameras to collect RGB observations of the environment. We utilized one rectangular cloth for the experiments, also visualized in Fig. 6. The robot used for the experiments was a Franka-Emika Panda robot. We employed a Cartesian position controller to execute a folding trajectory, which was randomly generated using the same procedure as the simulated data. We assumed prior knowledge of the pick and place locations and that the cloth was already in a grasped configuration.

We recorded RGB observations from all three cameras throughout the manipulation process. Depth observations were additionally captured at $t = 0$ to initialize the cloth mesh for dynamics predictions. At each timestep, segmentation and video tracking modules pre-trained on Grounding-DINO [51] and Segment Anything (SAM) [52] were used to generate masks of the cloth and the gripper, respectively, using the prompts "cloth" and "robot gripper". These masks were subsequently tracked over time using the video tracker XMEN [53].

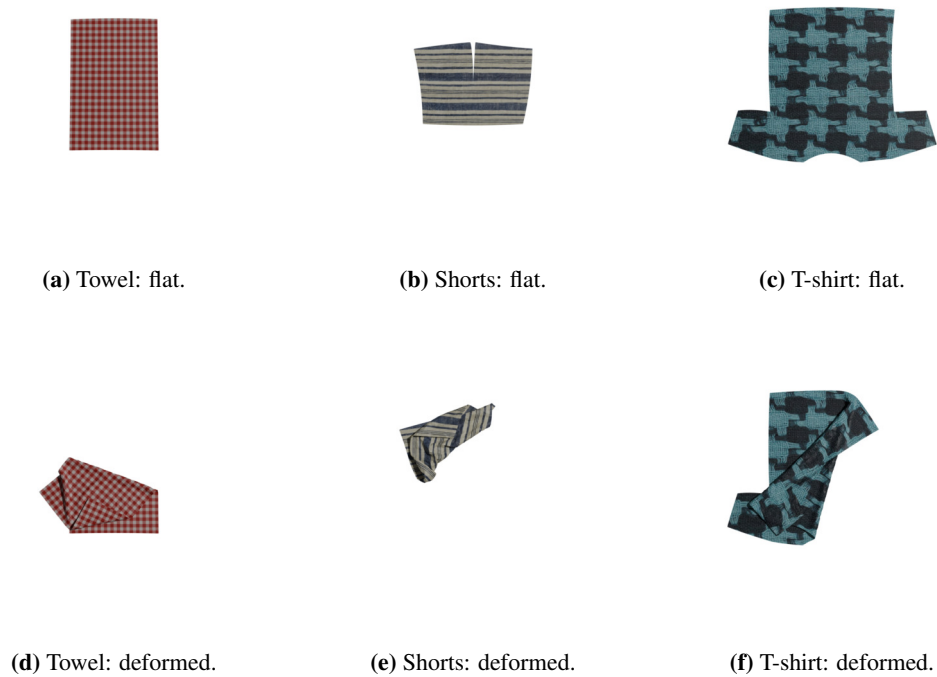


Figure 5: Example of synthetic images generated for the objects considered in our experiments (towel, shorts, t-shirt). For each object, we show the flat (upper row) and the deformed (lower row) states, rendered with Blender.

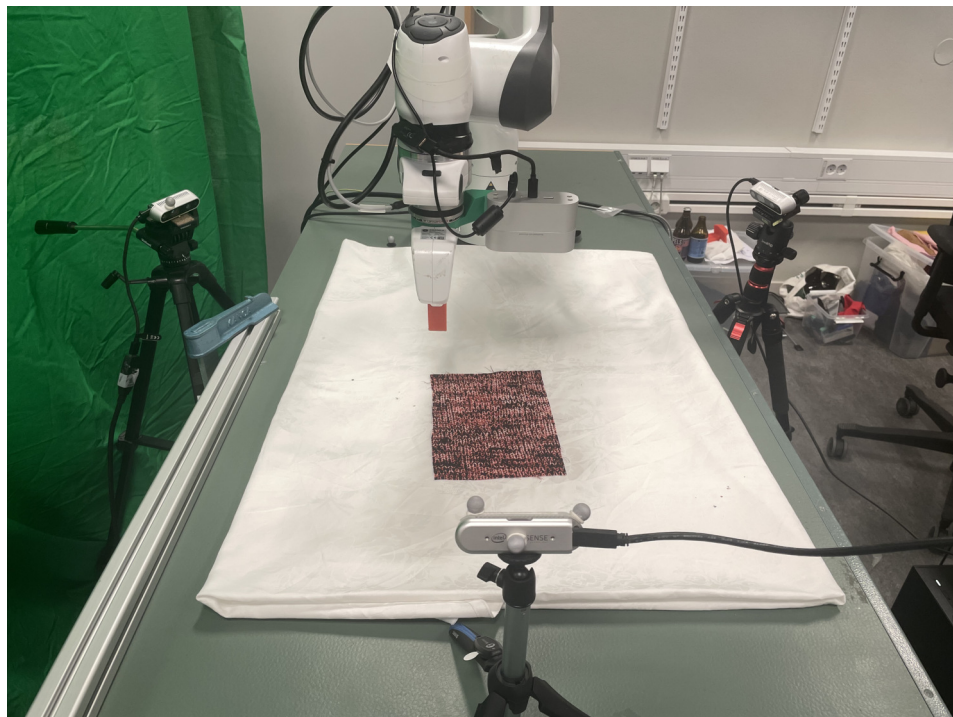


Figure 6: Overview of experimental set-up.

References

- [1] J. Zhu, A. Cherubini, C. Dune, D. Navarro-Alarcon, F. Alambeigi, D. Berenson, F. Ficuciello, K. Harada, J. Kober, X. Li, et al. Challenges and outlook in robotic manipulation of deformable objects. *Robotics & Automation Magazine*, 29(3):67–77, 2022.
- [2] C. Wen, X. Lin, J. So, K. Chen, Q. Dou, Y. Gao, and P. Abbeel. Any-point trajectory modeling for policy learning. *arXiv preprint arXiv:2401.00025*, 2023.
- [3] Z. Huang, X. Lin, and D. Held. Self-supervised cloth reconstruction via action-conditioned cloth tracking. *arXiv preprint arXiv:2302.09502*, 2023.
- [4] A. Sanchez-Gonzalez, J. Godwin, T. Pfaff, R. Ying, J. Leskovec, and P. Battaglia. Learning to simulate complex physics with graph networks. In *International conference on machine learning*, pages 8459–8468. PMLR, 2020.
- [5] A. Longhini, M. Moletta, A. Reichlin, M. C. Welle, D. Held, Z. Erickson, and D. Kragic. Edonet: Learning elastic properties of deformable objects from graph dynamics. In *IEEE ICRA*, pages 3875–3881. IEEE, 2023.
- [6] P. Sundaresan, R. Antonova, and J. Bohgl. Diffcloud: Real-to-sim from point clouds with differentiable simulation and rendering of deformable objects. In *2022 IEEE/RSJ IROS*, pages 10828–10835. IEEE, 2022.
- [7] A. Longhini, M. C. Welle, Z. Erickson, and D. Kragic. Adafold: Adapting folding trajectories of cloths via feedback-loop manipulation. *arXiv preprint arXiv:2403.06210*, 2024.
- [8] X. Lin, Y. Wang, Z. Huang, and D. Held. Learning visible connectivity dynamics for cloth smoothing. In *CoRL*, pages 256–266. PMLR, 2022.
- [9] Z. Huang, X. Lin, and D. Held. Mesh-based dynamics with occlusion reasoning for cloth manipulation. *arXiv preprint arXiv:2206.02881*, 2022.
- [10] X. Ma, D. Hsu, and W. S. Lee. Learning latent graph dynamics for deformable object manipulation. *arXiv preprint arXiv:2104.12149*, 2, 2021.
- [11] D. Blanco-Mulero, O. Barbany, G. Alcan, A. Colomé, C. Torras, and V. Kyrki. Benchmarking the sim-to-real gap in cloth manipulation. *IEEE Robotics and Automation Letters*, 2024.
- [12] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM ToG*, 42(4), July 2023.
- [13] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.
- [14] S. Saito, J. Yang, Q. Ma, and M. J. Black. Scanimate: Weakly supervised learning of skinned clothed avatar networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2886–2897, 2021.
- [15] Z. Su, T. Yu, Y. Wang, and Y. Liu. Deepcloth: Neural garment representation for shape and style editing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(2):1581–1593, 2022.
- [16] R. Daněřek, E. Dibra, C. Öztireli, R. Ziegler, and M. Gross. Deepgarment: 3d garment shape estimation from a single image. In *Computer Graphics Forum*, volume 36, pages 269–280. Wiley Online Library, 2017.
- [17] B. Jiang, J. Zhang, Y. Hong, J. Luo, L. Liu, and H. Bao. Bcnet: Learning body and cloth shape from a single image. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XX 16*, pages 18–35. Springer, 2020.

- [18] C. Chi and S. Song. Garmentnets: Category-level pose estimation for garments via canonical space shape completion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3324–3333, 2021.
- [19] Y. Li, Y. Wang, M. Case, S.-F. Chang, and P. K. Allen. Real-time pose estimation of deformable objects using a volumetric approach. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1046–1052. IEEE, 2014.
- [20] N. Karaev, I. Rocco, B. Graham, N. Neverova, A. Vedaldi, and C. Rupprecht. Cotracker: It is better to track together, 2023.
- [21] X. Shi, Z. Huang, W. Bian, D. Li, M. Zhang, K. C. Cheung, S. See, H. Qin, J. Dai, and H. Li. Videoflow: Exploiting temporal cues for multi-frame optical flow estimation, 2023.
- [22] C. Doersch, A. Gupta, L. Markeeva, A. Recasens, L. Smaira, Y. Aytar, J. Carreira, A. Zisserman, and Y. Yang. Tap-vid: A benchmark for tracking any point in a video. *Advances in Neural Information Processing Systems*, 35:13610–13626, 2022.
- [23] Q. Wang, Y.-Y. Chang, R. Cai, Z. Li, B. Hariharan, A. Holynski, and N. Snavely. Tracking everything everywhere all at once. In *International Conference on Computer Vision*, 2023.
- [24] Z. Teed and J. Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *ECCV*, pages 402–419. Springer, 2020.
- [25] A. Pumarola, E. Corona, G. Pons-Moll, and F. Moreno-Noguer. D-NeRF: Neural Radiance Fields for Dynamic Scenes. In *CVPR*, pages 10313–10322, 2021. doi:10.1109/CVPR46437.2021.01018.
- [26] K. Park, U. Sinha, J. T. Barron, S. Bouaziz, D. B. Goldman, S. M. Seitz, and R. Martin-Brualla. Nerfies: Deformable Neural Radiance Fields. In *ICCV*, pages 5845–5854, 2021. doi:10.1109/ICCV48922.2021.00581.
- [27] Z. Li, S. Niklaus, N. Snavely, and O. Wang. Neural Scene Flow Fields for Space-Time View Synthesis of Dynamic Scenes. In *CVPR*, pages 6494–6504, 2021. doi:10.1109/CVPR46437.2021.00643.
- [28] Z. Li, Q. Wang, F. Cole, R. Tucker, and N. Snavely. DynIBaR: Neural Dynamic Image-Based Rendering. In *CVPR*, pages 4273–4284, 2023.
- [29] J. Luiten, G. Kopanas, B. Leibe, and D. Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. *arXiv preprint arXiv:2308.09713*, 2023.
- [30] G. Wu, T. Yi, J. Fang, L. Xie, X. Zhang, W. Wei, W. Liu, Q. Tian, and X. Wang. 4d gaussian splatting for real-time dynamic scene rendering. *arXiv preprint arXiv:2310.08528*, 2023.
- [31] B. P. Duisterhof, Z. Mandi, Y. Yao, J.-W. Liu, M. Z. Shou, S. Song, and J. Ichnowski. Md-splatting: Learning metric deformation from 4d gaussians in highly deformable scenes, 2023.
- [32] J. Waczyńska, P. Borycki, S. Tadeja, J. Tabor, and P. Spurek. Games: Mesh-based adapting and modification of gaussian splatting. *arXiv preprint arXiv:2402.01459*, 2024.
- [33] L. Gao, J. Yang, B.-T. Zhang, J.-M. Sun, Y.-J. Yuan, H. Fu, and Y.-K. Lai. Mesh-based gaussian splatting for real-time large-scale deformation. *arXiv preprint arXiv:2402.04796*, 2024.
- [34] Y. Jiang, C. Yu, T. Xie, X. Li, Y. Feng, H. Wang, M. Li, H. Lau, F. Gao, Y. Yang, and C. Jiang. Vr-gs: A physical dynamics-aware interactive gaussian splatting system in virtual reality. *arXiv preprint arXiv:2401.16663*, 2024.
- [35] T. Xie, Z. Zong, Y. Qiu, X. Li, Y. Feng, Y. Yang, and C. Jiang. Physgaussian: Physics-integrated 3d gaussians for generative dynamics. *arXiv preprint arXiv:2311.12198*, 2023.

- [36] Y. Wang, D. Held, and Z. Erickson. Visual haptic reasoning: Estimating contact forces by observing deformable object interactions. *RA-L*, 7(4):11426–11433, 2022.
- [37] R. Brégier. Deep regression on manifolds: A 3d rotation case study. In *2021 International Conference on 3D Vision (3DV)*, pages 166–174, 2021. doi:10.1109/3DV53792.2021.00027.
- [38] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *TIP*, 2004.
- [39] B. P. Duisterhof, Z. Mandi, Y. Yao, J.-W. Liu, M. Z. Shou, S. Song, and J. Ichnowski. Md-splatting: Learning metric deformation from 4d gaussians in highly deformable scenes. *arXiv preprint arXiv:2312.00583*, 2023.
- [40] Z. Teed and J. Deng. RAFT: Recurrent All-Pairs Field Transforms for Optical Flow. In *ECCV*, pages 402–419, 2020. doi:10.1007/978-3-030-58536-5_24.
- [41] B. Delaunay et al. Sur la sphere vide. *Izv. Akad. Nauk SSSR, Otdelenie Matematicheskii i Estestvennyka Nauk*, 7(793-800):1–2, 1934.
- [42] S. Wang, V. Leroy, Y. Cabon, B. Chidlovskii, and J. Revaud. Dust3r: Geometric 3d vision made easy. *arXiv preprint arXiv:2312.14132*, 2023.
- [43] M. Macklin, M. Müller, N. Chentanez, and T.-Y. Kim. Unified particle physics for real-time applications. *ACM Transactions on Graphics (TOG)*, 33(4):1–12, 2014.
- [44] X. Lin, Y. Wang, J. Olkin, and D. Held. Softgym: Benchmarking deep reinforcement learning for deformable object manipulation. In *CoRL*, pages 432–448. PMLR, 2021.
- [45] B. O. Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. URL <http://www.blender.org>.
- [46] T. Lips, V.-L. De Gusseme, et al. Learning keypoints for robotic cloth manipulation using synthetic data. *arXiv preprint arXiv:2401.01734*, 2024.
- [47] Y. Zheng, A. W. Harley, B. Shen, G. Wetzstein, and L. J. Guibas. Pointodyssey: A large-scale synthetic dataset for long-term point tracking. In *CVPR*, pages 19855–19865, 2023.
- [48] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [49] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *ECCV*, pages 405–421, 2020. doi:10.1007/978-3-030-58452-8_24.
- [50] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli. Image quality assessment: from error visibility to structural similarity. *TIP*, 2004.
- [51] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, C. Li, J. Yang, H. Su, J. Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023.
- [52] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, et al. Segment anything. *arXiv preprint arXiv:2304.02643*, 2023.
- [53] H. K. Cheng and A. G. Schwing. Xmem: Long-term video object segmentation with an atkinson-shiffrin memory model. In *European Conference on Computer Vision*, pages 640–658. Springer, 2022.