1 Supplementary Materials

1.1 Baselines

- **Quasi-static**: [Pick & Place] and [Pick & Drag] are policies which uses a quasi-static "pick and place" (similar to Lee et al. [1], visualized in Fig. 2b) and "pick and drag" (similar to Seita et al., with no lift step compared to pick and place) primitive respectively. [Stretch & Drag] is identical to [Pick & Drag] with an extra stretch step (identical to FlingBot's stretch) after picking. These baselines are implemented with dual-arm set up, therefore provides the same physical reach range as our dual-arm fling system.
- **Dynamic manipulation**: [FlingBot] is our policy which predicts the optimal two-arm grasp locations for a fixed stretching and flinging routine described in Sec. 3.2.
- **Dynamic manipulation with fling speed prediction**: [FlingBot-S] is identical to [FlingBot] with an additional fling speed module, which predicts the fling speed within the range of $[0.1\mathrm{m\,s^{-1}}, 1.0\mathrm{m\,s^{-1}}]$ from visual input of the cloth after stretching and lifting. This fling speed module is trained using Deep Deterministic Policy Gradients (DDPG) [3] on the delta-coverage rewards to maximize single-step returns (discount factor γ is set to 0), where the grasps are predicted by a converged and frozen [FlingBot] value network. However, despite the cloth physical parameter variations in the training and testing dataset (Sec. 1.2), Tab. 1 and Fig. 5 (in main paper) suggest that there aren't significant performance gains for [FlingBot-S] over [FlingBot] within the cloth parameter ranges tested (Sec. 1.2). These results justify [FlingBot]'s usage of a single fling speed, even for varying cloth mass, stiffness, and size. Thus, we prefer the simpler [FlingBot] approach for comparisons with baselines.
- **Dynamic manipulation with fling parameter regression**: [Fling-Reg] is identical to [FlingBot], but directly predicts the action parameters $\langle C_x, C_y, \theta, w \rangle$ from visual inputs instead of exploiting the task's equivariances. Its policy is trained using DDPG on delta-coverage rewards to maximize single-step returns. However, from Tab. 1, [Fling-Reg] completely fails to perform the task, demonstrating the advantage of encoding inductive biases which leverage equivariances in the problem structure.

1.2 Task Dataset Generation

Each task is specified by a cloth mesh, mass, stiffness, and initial configuration. The cloth mesh is sampled from one of three types:

1. **Normal Rect**, which contains rectangular cloths with size within the reach range. Edge lengths are sampled from $[0.40\mathrm{m}, 0.65\mathrm{m}]$.
2. **Large Rect**, which contains rectangular cloths with at least one edge larger than the reach range $(0.70\mathrm{m})$. Otherwise, edge sizes are sampled from $[0.40\mathrm{m}, 0.75\mathrm{m}]$, which means the shorter edge could still be smaller than the reach range.
3. **Shirt**, which contains a subset of shirts sampled from CLOTH3D's [4] test split, all of which are resized to be within the reach range. These shirts include tank tops, crop tops, short and long sleeves.

The cloth mass is sampled from $[0.2\mathrm{kg}, 2.0\mathrm{kg}]$ and an internal stiffness from $[0.85\mathrm{kg/s^2}, 0.95\mathrm{kg/s^2}]$. Finally, the cloth's initial configuration is varied by holding a randomly grasped the cloth at a random height between $[0.5\mathrm{m}, 1.5\mathrm{m}]$ then dropping and allowing the cloth to settle (similar to Lee et al. and tier-3 in Seita et al.), resulting in a severely crumpled configuration.

To calculate the normalized coverage, we use the maximum possible coverage of the cloths in their flattened configurations. For rectangular cloths in simulation, the flattened configuration can be analytically calculated using the undeformed vertex positions of the grid mesh, which means the normalized coverage could still be higher if the cloth rests in a stretched position due to friction. For shirts in simulation, we opted to calculate its maximum possible coverage as its outer surface area divided by 2, since a qualitatively flattened T-shirt may not actually maximize coverage. While this

choice also makes it possible for the normalized coverage to be greater than 1, it will still preserve performance rankings. For real world experiments, the flattened configurations are manually set.

We emphasize the difficulty of our unfolding tasks, where cloths in Normal Rect, Large Rect, and Shirt have average initial coverages of 28.8%, 27.1%, and 46.4% in simulation respectively, and 26.1%, 33.6%, and 24.0% in the real world respectively. In contrast, simulated cloths from Seita et al. start at 77.2%, 57.6%, and 42.0% for easy, medium, and hard tasks, while real world cloths from Ganapathi et al. starts at 71.4% and 68.4% on two different real-world trials. Yet, the challenging cloth configurations found in our tasks (Fig. 4) are realistic and prevalent in typical households.

In simulation, the policy is trained on 2000 *rectangular* cloths sampled evenly between Normal Rect and Large Rect, and evaluated on 600 novel tasks split evenly between Normal Rect, Large Rect, and Shirt cloths. In real, the simulation policy is deployed to collect real world experience on 150 Normal Rect episodes (257 steps), optimized on both simulation and real world data, then evaluated on 10 novel tasks in each cloth type.

1.3 Extra Qualitative Results

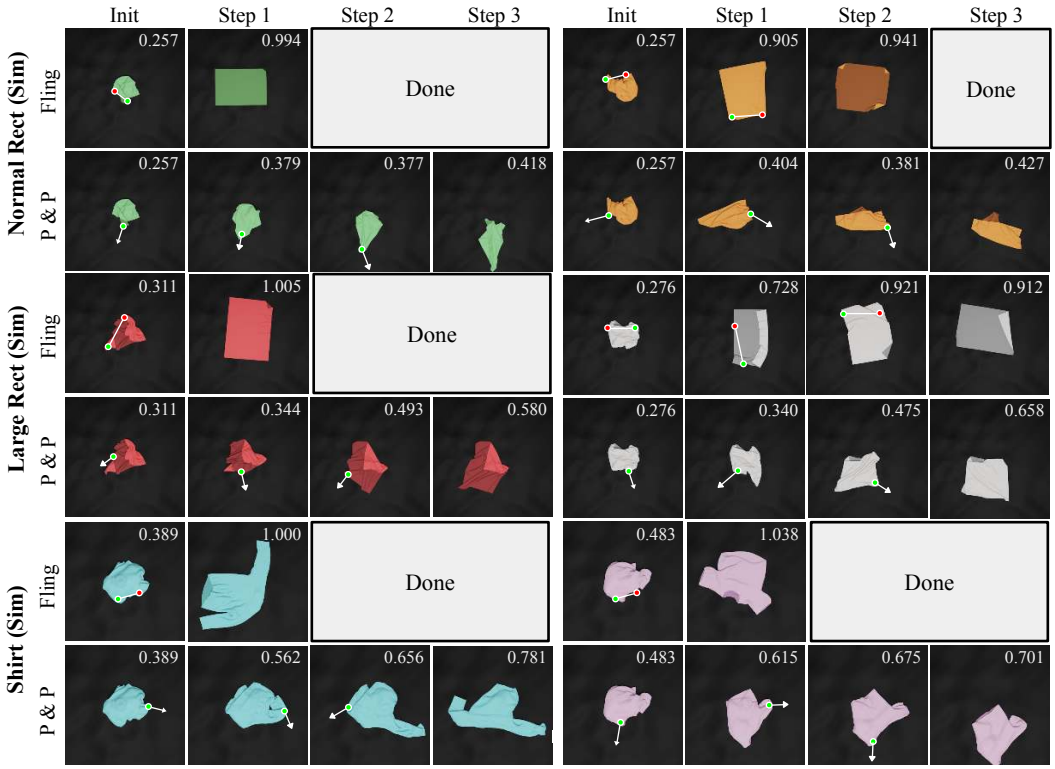

Figure 1: **Qualitative Results in Simulation Experiments.**

1.4 Failure cases

For normal rectangular cloths, the most common failure case is when a dual-arm corner grasped is slightly misaligned and becomes a single-arm grasp and fling instead, resulting in a low coverage configuration. For large rectangular cloths, cloths could fold in half almost perfectly, thus appearing completely unfolded, causing the policy to terminate the episode. For shirts, the self-discovered dual-arm corner and edge grasp for flinging which is effective for rectangular cloths fail on shirts in two main ways. First, if the sleeves of the shirt get stuck in the shirt's collar, FlingBot will be unable to pull the sleeves out. This failure case motivates future work on combining quasi-static and dynamic actions for cloth manipulation. Second, dual arm grasps where one grasp is on the outer surface and another grasp is on the inner surface of the shirt usually flings to low coverages.

While this failure case is expected to the differences between rectangular cloths and shirts (presence of holes, inner/outer surfaces, etc.), FlingBot's performance on shirts still suggested generalizable cloth manipulation abilities.

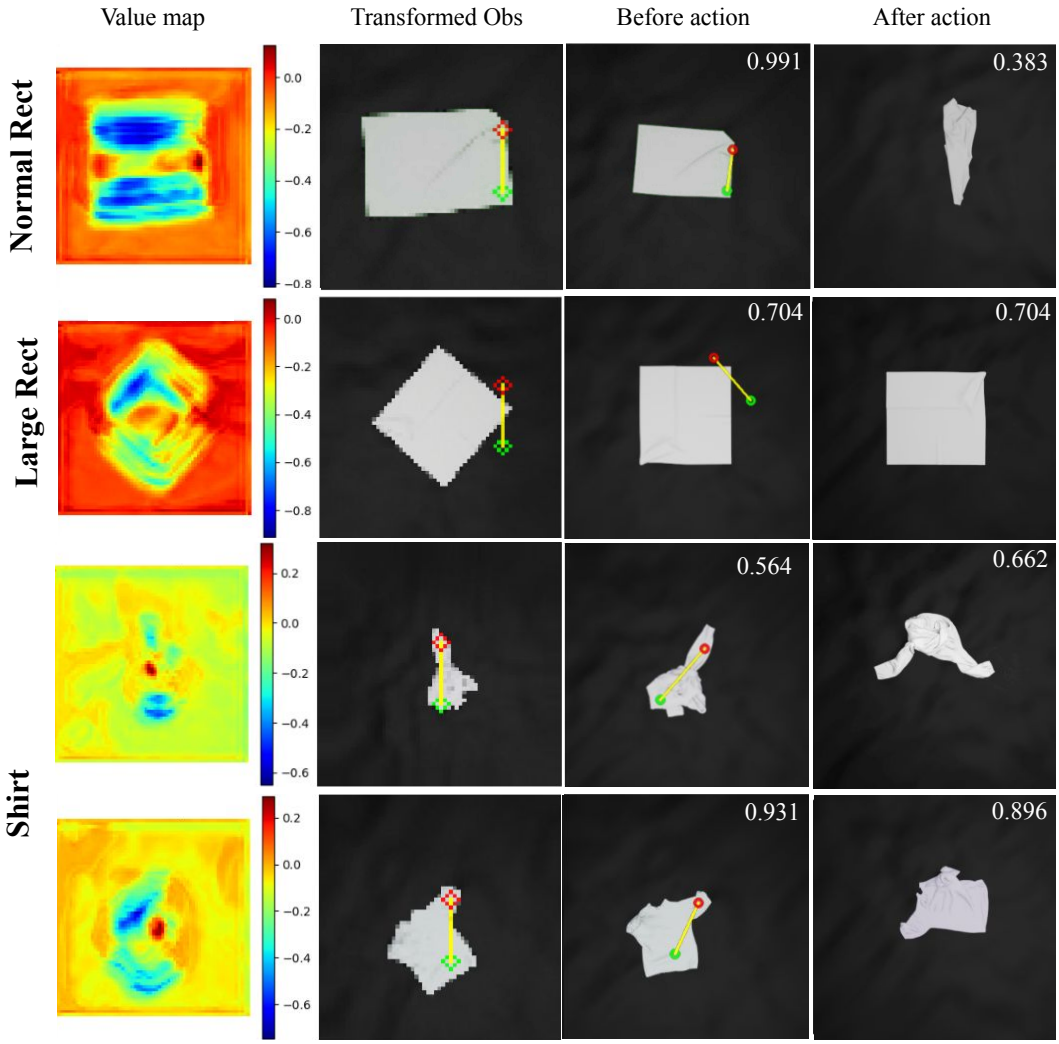

Figure 2: **Failure Cases in Simulation Experiments.**

1.5 Real world fling parameter robustness

Another large sim2real problem was poor collision handling in simulation. There were cloth configurations that the real world system experienced but was not observed in simulation, such as cloth twisting. While Nvidia Flex is decent at preventing self penetration, it does so at the cost of unrealistic collision handling. Qualitatively, this unrealistic collision handling means cloths untwist themselves when twisted (or are in any other state with high self-collision). Another case where this self-unfolding behavior was observed was for shirts due to the two layers of the shirts colliding with each other. We hypothesize that poor collision handling is the main reason why performance for the simulated shirt benchmark is higher across the board for all approaches, despite being unseen cloth types.

1.6 Real world Failures

Grasping failures, where the policy specified a grasp point on the cloth but the grippers failed to successful pinch grasp, constituted most of our real-world pipeline failure cases. Our real world

In designing our motion primitive, we optimized fling parameters (waypoints, velocities, acceleration) to maximize coverage assuming a good grasp (e.g.: a dual arm grasp on a normal rectangular cloth in a stretched state). We observed that the real world flinging setup system could robustly unfold the cloth using a wide range of fling parameters (i.e.: fling heights and speeds) if manually given a good grasp (Fig 3), while the simulated system was highly sensitive to such parameters. This sim2real gap was bridged in our work by tuning the simulated fling parameters such that flings from good grasps in simulation would also lead to high coverages like in the real system, which results in a variable fling height and a different fling speed in simulation compared to a fixed fling height and speed in real. Crucially, this gap underscores the importance of real world results for cloth manipulation as well as motivates future work on fast and accurate cloth simulation engines.

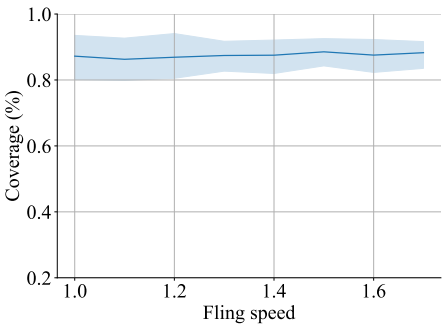

Figure 3: **Real world fling speed robustness.** By flinging at speeds in the range $[1.0\mathrm{m\,s^{-1}}, 1.7\mathrm{m\,s^{-1}}]$ at $0.1\mathrm{m\,s^{-1}}$ intervals, we observed that our fling primitive robustly achieves above 80% coverage if given two good grasps.

system uses its frontal RGBD view to detect grasp failures after the dual arm lifts the cloths up and automatically discards these episodes. The average grasp success rate is 78.0%, 45.0%, and 75.8% for normal rectangular, large rectangular, and shirts respectively. We used a bath towel for our large rectangular cloths, which is significantly thicker and stiffer than the tea towel and T-shirt we used for normal rect and shirts. Therefore, pinch grasps with large cloths failed significantly more. Additionally, even on successful grasps, the gripper may hold the cloth tight in its crumpled state, rendering flings ineffective. However, we do not discard of these episodes.

Cloth grasping failure is a common problem when working with real world cloth manipulation. For instance, Ganapathi et al. also observed that "the most frequent failure mode is an unsuccessful grasp of the fabric which is compounded for tasks that require more actions". However, we believe this issue can be mitigated by using specialized gripper hardware (like in Ganapathi et al., Seita et al.) or incorporating grasp success estimation.

In addition to grasping failures discussed in the main paper, we also observed occasional cloth stuck errors, where cloths get stuck to the gripper after the gripper opens in an attempt to release the cloth. To enable the system to automatically recover from this issue, after opening the gripper, the arms move to a predefined height above the workspace. Then, we use the frontal RGB-D view (used to implement the stretching primitive) to check whether the cloth is detected above a manually set height threshold above the workspace.

Here, we summarize techniques for implementing a real-world cloth manipulation pipeline:

1. **High friction gripper finger**: In our dual-arm system, we used an OnRobot RG2 and Schunk WGS50 gripper, where the former had a rubber tip, and the latter had a metal tip. After observing a significantly lower pinch grasping success with the WSG50, we added a rubber fingertip to the WSG50, which improved its grasp successes significantly. Alternatively, picking a gripper that can apply lots of pressure using its fingers, like the da Vinci Research Kit surgical robot in Seita et al., should also help.
2. **Soft Workspace**: A successful pinch grasp should apply the right amount of pressure between the cloth and the workspace. Too little pressure and the cloth will not get grasped, while too much pressure could easily damage the cloth, gripper, and robot arm. Due to noise in-depth sensing, the arms may be asked to grasp points slightly below the surface of the workspace. Similar to prior works [6, 5], we found that using a firm and thin rubber mattress was sufficient to address this problem.
3. **Accurate depth sensing**: Building on the previous point, hardware improvements on the sensing side could help significantly. In our real world experiments, the Azure Kinect

(a) Crossover constraint

(b) Grasp width constraint

Figure 4: To minimize collisions, arms should grasp points closer to their side (a) and be a reasonable distance away from each other (b).

v3 has significantly less noisy and more accurate depth images than the Intel Realsense D415. While the real world numbers we reported in this paper are only with Realsense cameras, the codebase for our Kinect/Realsense real world setup is publicly accessible at https://github.com/columbia-ai-robotics/flingbot.

4. **4 DOF Grasp success prediction**: A grasp which is parallel to a small crease on the cloth is more likely to result in a successful pinch grasp. Therefore, we hypothesize that additionally considering gripper z-rotation, on top of the positional 3 DOFs we have in our system, and learning the optimal pinch grasp z-rotation using a grasp success predictor may result in higher overall grasp success. Future work could explore weighing task rewards with such grasp success predictions.

1.7 Designing dynamic motion primitives

To achieve the highest speed at the end effector while respecting the torque limit of each joint, the primitive must move upper joints (i.e.: wrist) more than the lower joints (i.e., base). We also found that adding a blending radius between each target joint configuration gave a much smoother flinging trajectory and cloth swinging, as opposed to a jerky cloth motion without blending radius. In designing our motion primitive, we optimized fling dynamics parameters (waypoints, velocities, acceleration) to maximize coverage assuming a dual arm grasp on a normal rectangular cloth in a stretched state. Automatically discovering dynamic motion primitives, such as flinging, and simultaneously learning their parameters is an important and interesting direction for future work.

1.8 Why is learning fling speeds unhelpful?

We hypothesize that light and thin cloths, whose air resistive forces to momentum during flinging ratio is significantly higher than the cloths tested, would require a higher fling speed. Therefore, fling speed learning may be helpful when cloths in the task dataset contain a wider variance in density and thickness. We also hypothesize that a successful fling speed prediction approach may require extra information about the cloths' physical parameters that could not be obtained through visual observation alone, which would be out of scope for this work.

References

[1] R. Lee, D. Ward, A. Cosgun, V. Dasagi, P. Corke, and J. Leitner. Learning arbitrary-goal fabric folding with one hour of real robot experience, 2020.

[2] D. Seita, P. Florence, J. Tompson, E. Coumans, V. Sindhwani, K. Goldberg, and A. Zeng. Learning to Rearrange Deformable Cables, Fabrics, and Bags with Goal-Conditioned Transporter Networks. In *ICAR*, 2021.

[3] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[4] H. Bertiche, M. Madadi, and S. Escalera. Cloth3d: Clothed 3d humans. In *European Conference on Computer Vision*, pages 344–359. Springer, 2020.

[5] D. Seita, A. Ganapathi, R. Hoque, M. Hwang, E. Cen, A. K. Tanwani, A. Balakrishna, B. Thananjeyan, J. Ichnowski, N. Jamali, K. Yamane, S. Iba, J. F. Canny, and K. Goldberg. Deep imitation learning of sequential fabric smoothing policies. *CoRR*, abs/1910.04854, 2019. URL http://arxiv.org/abs/1910.04854.

[6] A. Ganapathi, P. Sundaresan, B. Thananjeyan, A. Balakrishna, D. Seita, J. Grannen, M. Hwang, R. Hoque, J. E. Gonzalez, N. Jamali, K. Yamane, S. Iba, and K. Goldberg. Learning dense visual correspondences in simulation to smooth and fold real fabrics, 2020.