Point-based Correspondence Estimation for Cloth Alignment and Manipulation

Author Names Omitted for Anonymous Review. Paper-ID [add your ID here]

***Abstract*—Automating cloth folding is a challenging task with practical implications in various domains. Existing methods often struggle with unaligned configurations, limiting their applicability in real-world scenarios. In this research, we present FabricFlowAlignNet (FFAN), a novel approach that learns flow-based correspondences on point clouds between the current observed and goal cloth configurations. We use these learned 3D correspondences for both cloth alignment and manipulation: correspondences are used to align the observed cloth with the goal, and the flow-based correspondences are re-used as action proposals. Our experiments demonstrate that FFAN demonstrates superior performance compared to a state-of-the-art folding approach, particularly in scenarios where observed cloth is rotated or otherwise unaligned with the goal. This research highlights the multi-faceted effectiveness of learning 3D correspondences for deformable object manipulation.**

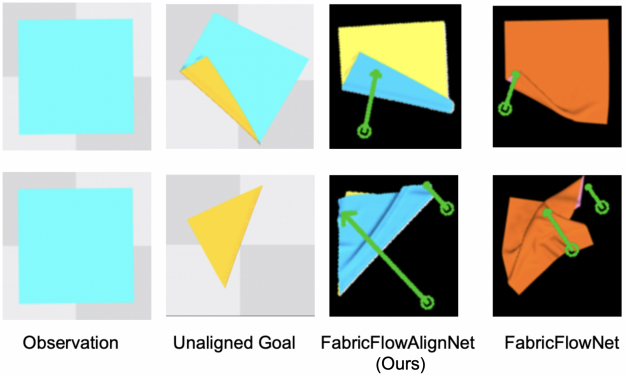

Fig. 1. Performance of FFAN (ours) vs FNN on unaligned goals.

I. Introduction

Cloth manipulation is a challenging task, with difficulties in both perception and control due to the deformability of cloth. Manual cloth manipulation techniques are time-consuming, labor-intensive, and prone to human error. As a result, there is a growing demand to automate cloth manipulation processes in various domains such as folding laundry, handling textiles in manufacturing processes, and assisting individuals with disabilities in dressing.

A fundamental aspect of successful cloth manipulation is establishing correspondences between the current observation and the goal configuration. These correspondences provide critical spatial associations necessary for planning and executing folding actions. However, while prior methods have proposed to learn correspondences for cloth [11, 4], they do not explicitly use such methods for reasoning about the *alignment* between the observed cloth and the desired configuration. Alignment is a crucial step in cloth manipulation, and prior correspondence-based policies do not handle cases where the cloth and goal are not already aligned [11], or rely on human demonstrations [4].

In this work, we propose **FabricFlowAlignNet** (**FFAN**), an approach that combines the use of correspondences and symmetry-handling techniques to learn a goal-conditioned cloth manipulation policy. Our method leverages correspondences to "virtually" align the observation and goal point clouds, enabling the policy to determine the appropriate actions to execute on the observation. By incorporating these correspondences and symmetry handling, our approach aims to acquire an understanding of cloth folding strategies and develop a manipulation policy capable of accurately and efficiently folding clothes. This is particularly beneficial in challenging scenarios where the observed cloth is rotated or unaligned with the desired goal configuration.

We evaluate the performance of our method against a state-of-the-art folding approach [11] on a folding task, where the goal and observation poses are not aligned. Our method, reasons about symmetries and employs correspondences to deal with unaligned goals, unlike the baseline approach. The results demonstrate the effectiveness and robustness of our approach in achieving successful cloth folding when the observation and goal configurations are unaligned.

II. Prior Work

FabricFlowNet (FFN) [11] performed bimanual cloth folding by estimating flow correspondences between the observed cloth image and goal cloth image. However, FFN relies on strict alignment between observation and goal cloth poses in the image. Our approach extends FFN by proposing an approach for aligning learned 3D correspondences to overcome these limitations. By establishing spatial relationships between points in observation and goal configurations, we enable precise alignment and achieve better folding performance for unaligned goals than FFN.

Fabric Descriptors [4] is a method for estimating correspondences in fabric manipulation tasks. However, it heavily relies on human demonstrations to learn the necessary correspondences. In contrast, our method can learn and estimate correspondences without any human intervention.

SpeedFolding [1], another cloth manipulation method, is trained exclusively in real-world settings. In comparison, our approach, similar to FFN, undergoes training in a simulation environment before being transferred to real-world scenarios. While SpeedFolding employs self-supervised learning

techniques to handle unaligned observations and goals, our approach leverages learned correspondences for alignment purposes.

Regarding alignment methods, Cloth Funnels [2] is a notable technique that combines cloth folding and alignment. Their alignment procedure utilizes the Procrustes' algorithm, primarily designed for aligning rigid objects. However, due to its local alignment nature, it may not be as suitable for deformable objects. In contrast, our approach adopts the random sample consensus (RANSAC) algorithm, allowing for probabilistic achievement of a globally optimal alignment.

III. Point Cloud Correspondence Estimation for Cloth Alignment and Manipulation

In this section, we describe FabricFlowAlignNet (FFAN), our approach for estimating observation-goal correspondences to align and manipulate cloth.

A. Learning Correspondences for Point Clouds

We propose a 3D, flow-based correspondence estimator called "3DFlowNet", a component of our overall pipeline. 3DFlowNet takes the observation and goal point clouds c_o and c_g as input, and outputs 3D flow $\hat{f}$. 3DFlowNet is a non-trivial extension of the FlowNet from FabricFlowNet [11], which was limited to image input and 2D flow output. A schematic overview of 3DFlowNet can be found in Fig. 2.

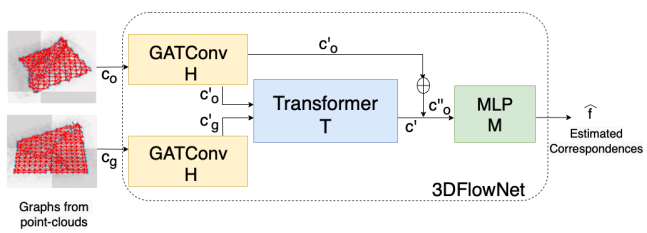

Fig. 2. 3DFlowNet Architecture

We first transform the point clouds into a graph, where nodes represent cloth particles and are connected to their neighboring particles on the cloth mesh. This step requires privileged state information from the simulator of the cloth mesh edges, which would not be available in the real world; estimating these edges is an area of future work and could leverage prior methods like VCD [6]. We embed the input graphs by employing a graph neural network H, which outputs embeddings for each node in the graph: $c'_o, c'_g \in \mathbb{R}^{N\times F}$.

We then use a Transformer network [8] denoted as T to perform cross-attention between observation and goal features. Our approach is inspired by prior Transformer-based per-point networks like DCP [10] and TAX-Pose [7]. T takes c'_o and c'_g as input and outputs transformed embeddings $c' \in \mathbb{R}^{N\times F}$. c' The resulting transformer embeddings, c', are then summed with the original observation embeddings c'_o to produce c''_o.

To estimate correspondences between the two configurations, we pass c''_o through MLP layers to produce estimated correspondences $\hat{f} \in \mathbb{R}^{Nx3}$. These correspondences represent how each cloth particle in N transports to achieve the goal configuration.

To train 3DFlowNet, we use a weighted L2 loss between the estimated and ground truth correspondences. The ground truth correspondences are computed as the difference between the point clouds c_g and c_o. The weighted L2 loss function is defined as:

$$\mathcal{L}_2(\hat{f}, f) = \sum_{i=1}^{N} w_i (f_i - \hat{f}_i)^2 \tag{1}$$

where $\hat{f}$ represents the estimated correspondences, f represents the ground truth correspondences, and N is the total number of points in the point cloud. The weights w_i are higher for ground truth pick points.

B. Iterative Correspondence Estimation

The trained 3DFlowNet model struggles to generate large correspondences, leading to suboptimal results. To address this limitation, we introduce an iterative approach to enhance the accuracy of correspondence estimation. Our iterative process involves updating the estimated correspondences and the intermediate point cloud to gradually refine the correspondence estimation.

In each iteration, we utilize the trained 3DFlowNet model to estimate the correspondence between the intermediate point cloud $\hat{c}_o$ and the target configuration c_g. By integrating the estimated correspondence into the observation, we simulate the application of the flow to progressively approach the target configuration. The algorithm for iterative correspondence estimation is summarized in Alg. 1.

Algorithm 1 Iterative Correspondence Estimation

Input: Trained 3DFlowNet, Point Clouds c_o, c_g
Initialize all zeros $\bar{f} \in \mathbb{R}^{N\times 3}$
$\hat{c}_o := c_o$
for t = 1…T **do**
    $\hat{f}$ = 3DFlowNet($\hat{c}_o$, c_g)
    $\bar{f}$ += $\hat{f}$
    $\hat{c}_o$ += $\hat{f}$
end for
return $\bar{f}$

C. RANSAC Alignment for Unaligned Goals

To address the alignment issue between observation and goal, we propose utilizing estimated correspondences to perform alignment using the RANSAC algorithm [3]. The forward pass through 3DFlowNet provides the estimated correspondences. The steps involved in this process are as follows:

1) Sample two indices (i, j) on the cloth.
2) Compute the transformation matrix R using the correspondences (f_i, f_j), i.e. (p_i, p_j) and $(p_i + f_i), (p_j + f_j)$.
3) Compute inliers by comparing the distance between transformed points $||Rp_i - (p_i + f_i)||$ and a threshold epsilon ϵ.

4) Choose the transformation matrix R with the maximum number of inliers.

D. Estimating the Pick Location for an Action

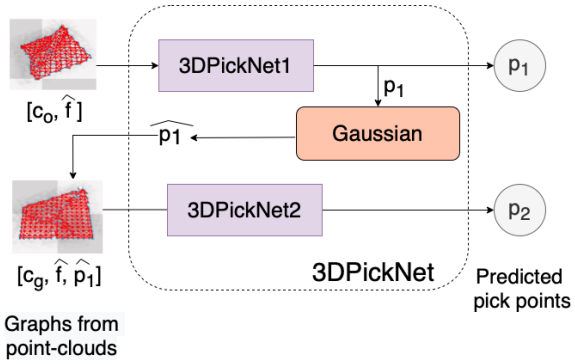

Fig. 3. 3DPickNet Architecture

To predict the pick points necessary for cloth manipulation, we introduce a neural network called 3DPickNet. This network is specifically designed to estimate the pick points p_1 and p_2. The inputs to 3DPickNet are the current observation c_o and the estimated correspondences $\hat{f}$ between c_o and the goal configuration c_g.

To enable the prediction of the second pick point conditioned on the first pick point, we utilize two separate networks: 3DPickNet1 and 3DPickNet2. In 3DPickNet1, we concatenate c_o and $\hat{f}$ and create a graph representation of the point cloud. Each node in the graph is represented as $[x, y, z, \hat{f}]$. 3DPickNet1 generates a probability value for each node to be selected as the first pick point p_1. The node with the highest probability is identified as p_1.

3DPickNet2 is responsible for predicting the second pick point p_2, taking into account the information about p_1. In this network, we introduce an additional channel called $\hat{p_1}$, which represents a 3D Gaussian distribution centered around p_1. This channel assigns lower values to nodes near p_1 and higher values to nodes farther away, preventing the selection of p_2 in close proximity to p_1.

The architecture of 3DPickNet is depicted in Figure 3.

For training 3DPickNet, we use a weighted binary cross-entropy loss. The loss function compares the predicted probabilities of nodes being pick points with the ground truth labels.

The binary cross-entropy loss function is defined as:

$$L(p, y) = \sum_{i=1}^{N} -w_i(y_i \log p_i + (1 - y_i) \log(1 - p_i)) \quad (2)$$

Here, p represents the predicted probabilities, y is the ground truth labels, and N is the total number of nodes. The weights w_i are higher for ground truth pick points.

Once the pick points p_1 and p_2 are predicted using the estimated correspondences $\hat{f}$, the corresponding manipulation actions can be executed to achieve the desired goal configuration.

E. Implementation Details

1) Dataset: We employ a dataset constructed using SoftGym [5], a deformable object simulator, to train and evaluate our approach. This dataset is the same as the one used in the FabricFlowNet method [11], but we extract point clouds from SoftGym to represent the cloth instead of using depth images.

The dataset is generated by sampling random actions biased towards grasping the corners of a square towel, following the approach of FabricFlowNet. Each instance in the training, validation, and test sets consists of a tuple (c_o, c_g, a), where c_o and c_g are point clouds representing the current observation and the desired goal configuration, respectively. The ground truth action a corresponds to the action that achieves the goal configuration.

In our approach, the action space is defined as $a = (p_1, p_2, q_1, q_2)$, where p and q represent the pick and place points, respectively, paired according to the subscripts. These pick and place points are selected from a set of indices representing the points in the point cloud.

2) 3DFlowNet: The graph neural network H consists of two Graph Attention layers (GATConv) [9]. The MLP network architecture M consists of two fully-connected layers.

3) 3DPickNet: The 3DPickNet architecture consists of three Graph Attention Network layers and two Multi-Layer Perceptron (MLP) layers for both 3DPickNet1 and 3DPickNet2. At the end of each network, a Sigmoid layer computes the probability of each node being a pick point. 3DPickNet1 represents each node with a six-dimensional feature, while 3DPickNet2 utilizes a seven-dimensional feature to accommodate the additional information provided by $\hat{p_1}$.

IV. Experiments

Our experiments investigate the following questions: (1) How does FFAN compare with FabricFlowNet (FFN) [11] on aligned goals? (2) How does FFAN compare with FFN on unaligned goals? We evaluate the methods in simulation, using the average L2 distance between cloth points in the achieved vs. desired point clouds as our error metric.

A. Performance on Aligned Goals

We use the same test set as FFN [11] to evaluate performance on aligned goals. This test set consists of 40 single-step goals, where both the observation and the goal positioned at the center of the workspace with the same orientation. For this experiment, we do not use alignment estimation with FFAN to directly compare pre-aligned folding performance.

Table I presents the performance comparison between our method and FFN. The results demonstrate that our method performs comparably to FFN on aligned goals, with only a marginal difference in average particle distance.

TABLE I
Folding Performance on 40 Aligned Goals

Method	Average Particle Distance (mm) ↓
FFN [11]	4.26
FFAN (Ours)	5.54

B. Performance on Unaligned Goals

We also evaluate the performance on unaligned goals, where the goal cloth configuration is randomly rotated and therefore not aligned with the initial observed configuration. We conducted experiments on three test sets: Easy, Medium, and Hard, where each test set corresponds to a different range of rotations. Easy encompasses angles between -5 and 5 degrees, Medium ranges from -45 to 45 degrees, and Hard covers a complete rotation from 0 to 360 degrees.

We evaluated the performance of FFAN in two scenarios: utilizing ground truth correspondences for RANSAC alignment and utilizing estimated correspondences for RANSAC alignment. Figure 4 presents a comparison of the two methods against FFN across all four test sets: aligned, easy, medium, and hard.

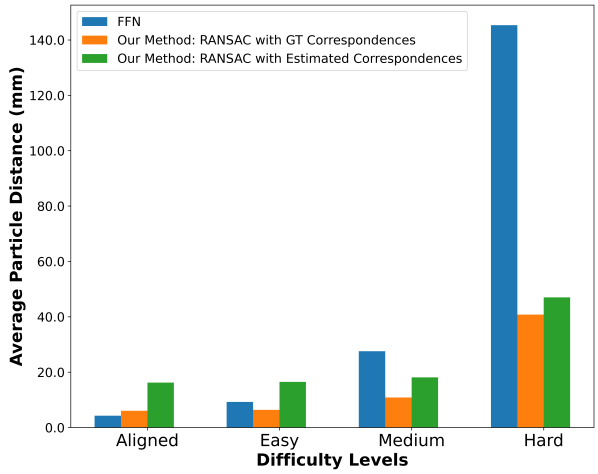

Fig. 4. Comparison of Folding Performance on Different Test Sets

From the results, we observe that our method with estimated correspondences outperforms FFN on the Medium and Hard tasks. However, using ground truth correspondences for RANSAC alignment yields even better results across all tasks, surpassing the performance of FFN. This demonstrates the potential for further improvement by improving the correspondence estimation. Qualitative results comparing our method and FFN on the Medium case can be found in Fig. 1.

C. Ablations

1) No Iterative Correspondence: In this section, we ablate our approach by removing iterative correspondence estimation (Sec. III-B). Table II shows that average particle distance error is higher when iterative correspondence estimation is removed.

TABLE II
ABLATION OF ITERATIVE CORRESPONDENCE ESTIMATION

Method	Average Particle Distance (mm) ↓
FFAN w/o Iter. Corresp.	10.591
FFAN w/ Iter. Corresp.	**5.54**

2) Number of Iterations for Iterative Correspondence: To determine the number of iterations to run for iterative correspondence estimation, we measured performance while increasing the number of iterations on a validation set. We used flow prediction error, an unweighted version of Eq. 1, as our performance metric. We evaluated number of iterations ranging from $k = 1$ (run 3DFlowNet once) to 4. Note that we did not retrain 3DFlowNet in an iterative manner.

Figure 5 shows the flow prediction error as a function of the number of iterations (k) in the iterative flow process. As we increase k from 1 to 3, there is a notable decrease in the flow prediction error, indicating improved accuracy on correspondences. However, beyond $k = 3$, we observed a slight increase in the error. Based on these observations, we empirically determined that the number of iterations for iterative flow correspondence estimation is $k = 3$.

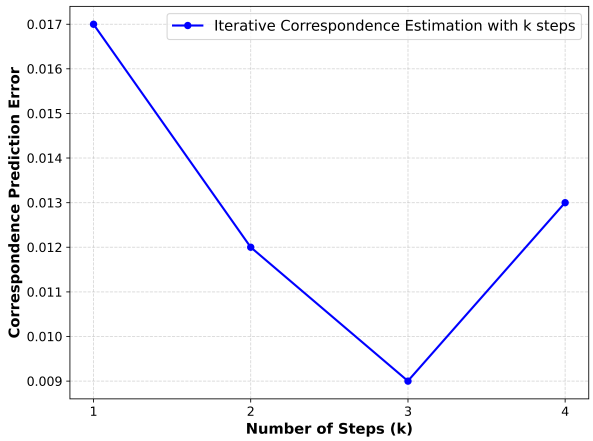

Fig. 5. Correspondence Prediction Error vs. Number of Iterative Correspondence Estimation Steps

V. CONCLUSION

Key Insights: Our method performs on par with FFN for aligned goals but excels in handling rotations between observation and goal. Robust correspondences and accurate estimation are crucial for achieving successful alignment and folding. Our approach offers valuable advancements in goal-driven manipulation tasks, providing a reliable and effective solution.

Limitations and Challenges: Like FFN, our method also relies on predefined sub-goals, which can be restrictive and may not generalize well to unseen fabrics and configurations. Complex and longer-horizon tasks pose challenges beyond our method's capabilities. Exploring alternative approaches that eliminate explicit sub-goals and address these challenges is important for broader applicability.

REFERENCES

[1] Yahav Avigal, Lars Berscheid, Tamim Asfour, Torsten Kröger, and Ken Goldberg. Speedfolding: Learning efficient bimanual folding of garments. In *2022 IEEE/RSJ*

International Conference on Intelligent Robots and Systems (IROS), pages 1–8. IEEE, 2022.

[2] Alper Canberk, Cheng Chi, Huy Ha, Benjamin Burchfiel, Eric Cousineau, Siyuan Feng, and Shuran Song. Cloth funnels: Canonicalized-alignment for multi-purpose garment manipulation. *arXiv preprint arXiv:2210.09347*, 2022.

[3] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[4] Aditya Ganapathi, Priya Sundaresan, Brijen Thananjeyan, Ashwin Balakrishna, Daniel Seita, Jennifer Grannen, Minho Hwang, Ryan Hoque, Joseph E Gonzalez, Nawid Jamali, et al. Learning dense visual correspondences in simulation to smooth and fold real fabrics. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11515–11522. IEEE, 2021.

[5] Xingyu Lin, Yufei Wang, Jake Olkin, and David Held. Softgym: Benchmarking deep reinforcement learning for deformable object manipulation. In *Conference on Robot Learning*, pages 432–448. PMLR, 2021.

[6] Xingyu Lin, Yufei Wang, Zixuan Huang, and David Held. Learning visible connectivity dynamics for cloth smoothing. In *Conference on Robot Learning*, pages 256–266. PMLR, 2022.

[7] Chuer Pan, Brian Okorn, Harry Zhang, Ben Eisner, and David Held. Tax-pose: Task-specific cross-pose estimation for robot manipulation. In *Conference on Robot Learning*, pages 1783–1792. PMLR, 2023.

[8] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[9] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

[10] Yue Wang and Justin M Solomon. Deep closest point: Learning representations for point cloud registration. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 3523–3532, 2019.

[11] Thomas Weng, Sujay Man Bajracharya, Yufei Wang, Khush Agrawal, and David Held. Fabricflownet: Bimanual cloth manipulation with a flow-based policy. In *Conference on Robot Learning*, pages 192–202. PMLR, 2022.