PolyFold: A Generalizable Framework for Language-Conditioned Bimanual Cloth Folding

Haozhe Du, Kechun Xu, Rong Xiong, Yue Wang

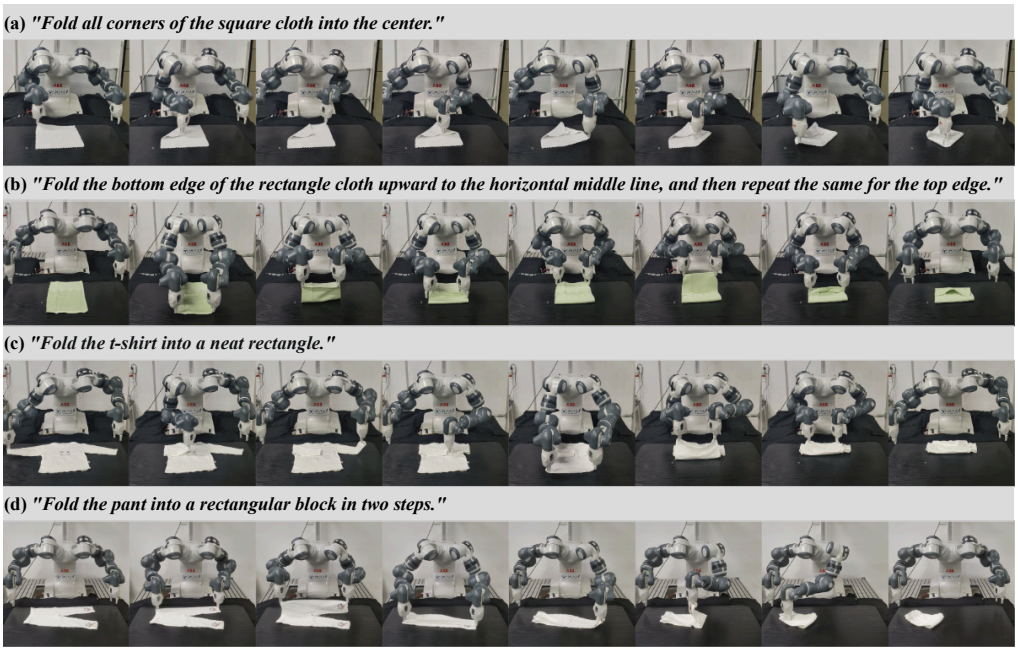


Fig. 1: Our proposed PolyFold framework facilitates autonomous cloth folding based on user language instructions, whether explicit or ambiguous. PolyFold demonstrates zero-shot generalization across various types and shapes of cloth, as well as different cloth folding tasks. Videos are available at our project webpage: https://sites.google.com/view/polyfold.

***Abstract*—Cloth folding stands as an intricate subject in robot manipulation, requiring robots to fold diverse fabrics into different configurations according to human intentions. Most previous approaches address this problem in a vision/language-goal-conditioned way. Relying on substantial expert demonstrations for training and precise subgoals for inferring, they lack inherent multi-step reasoning ability and struggle to generalize to novel cloth appearances and tasks. To tackle these problems, our key insight is incorporating the common sense reasoning ability of Large Language Models (LLMs) into cloth manipulation while addressing the limitations of LLMs in manipulating deformable objects, which involves an effective grounding module and rational planning hierarchy. To this end, we present PolyFold, a novel language-conditioned bimanual cloth folding framework that leverages the parameterized polygon model as an effective abstraction and grounding module for cloth representation. Moreover, PolyFold enables LLMs to infer an intermediate-level action—specifically, the symmetrical fold line, while delegating the pick-and-place calculations to a fold-line-guided downstream policy, which is learned through self-supervision using random data. Experiments on 70 cloth folding tasks and 4 cloth types show that PolyFold excels in zero-shot generalization and inherent multi-step reasoning capability, while also operating in a sample-efficient expert-demonstration-free manner, surpassing previous SOTA vision-conditioned and language-conditioned methods. Our method can also be directly deployed in real-world scenarios.**

***Note to Practitioners*—This paper is motivated by the need for a deformable object manipulation algorithm, particularly for cloth, with robust reasoning and generalization capabilities to handle diverse unseen objects and tasks. Such an algorithm holds significant promise for service robots in assisting with daily household organization. Existing robot cloth folding methods often rely on imitating expert demonstrations and predefined subgoals, limiting their adaptability to new tasks, new fabrics, or scenarios where only a final goal is provided. To address these limitations, we propose PolyFold, a novel framework that integrates Large Language Models (LLMs) into the cloth-folding process while overcoming their challenges in handling deformable objects in two aspects. Firstly PolyFold introduces a parameterized polygon model to abstract and represent cloth geometry and then it enables LLMs to identify intermediate-level actions—specifically symmetrical fold lines—while leaving detailed pick-and-place execution to a downstream policy trained via self-supervised learning. Experiments in simulation and the real world demonstrate PolyFold's strong generalization to unseen tasks and fabrics, its capacity for multi-step reasoning, and its independence from expert demonstrations.**

***Index Terms*—Deformable object manipulation, Large Language Models, parameterized polygon model, self-supervised learning**

All authors are with the State Key Laboratory of Industrial Control and Technology, Zhejiang University, Hangzhou, P.R. China. Rong Xiong and Yue Wang are the corresponding authors: rxiong@zju.edu.cn and wangyue@iipc.zju.edu.cn.

I. Introduction

CLOTH folding, as a representative task in deformable object manipulation, presents a formidable challenge in robotics. It focuses on the automation of the folding process for various types of fabrics, tailored to meet specific configurations [1]–[3]. Classical paradigms of previous works

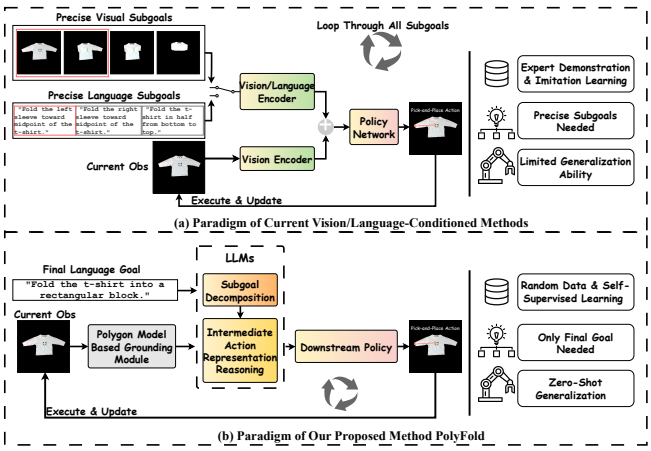


Fig. 2: Comparison of (a) current vision-conditioned/language-conditioned methods in cloth folding with (b) our proposed language-conditioned method PolyFold.

can be seen in Fig 2. One line of works named vision-conditioned methods [4]–[7] typically utilizes a sequence of precise subgoal images and compares current state with the corresponding subgoal to plan current cloth folding action. Another line of works [8], [9] leverage language instructions for deformable object manipulation, which offer a new perspective to distill task-related information while filtering out extraneous details in visual inputs. One representative work proposed by Deng et al. [9] introduces a Transformer-based architecture that processes language instructions, depth image, and a predicted cloth mesh in a unified way to generate pick-and-place actions. However, it is necessary for both categories of works to provide all subgoals in terms of vision or language to support the cloth folding task which involves multiple steps of pick-and-place actions. This is hard to achieve in an open environment. Meanwhile, they usually require a large amount of expert demonstration for imitation learning, demanding significant time and effort for data collection. The generalization capabilities of these methods are quite limited, making it challenging for them to adapt to previously unseen cloth shapes and unseen cloth folding tasks.

Recent development of Large Language Models (LLMs) and Vision Language Models (VLMs) [10] endows robot agents with exceptional common sense reasoning capability and generalization ability to perform long-horizon robot manipulation tasks [11]–[14], in a zero-shot or few-shot way and without huge reliance on expert data. However, they usually focus on rigid object tabletop manipulation, leaving deformable object manipulation relatively unexplored. Recently there are a few works leveraging LLMs/VLMs for subgoal-guided [15] or keypoint-based [16], [17] cloth folding, as well as cloth unfolding [15], [18] and learning reward models for cloth manipulation [19]. But no method has yet achieved expert-demonstration-free, zero-shot generalizable cloth manipulation based solely on the final language goal, which is the focus of our work.

Here we focus on incorporating LLMs into cloth manipulation tasks to enhance the method's reasoning and generalization capabilities while reducing dependence on carefully collected data. However, the challenges exist in both the input and output of the LLMs: (1) Perceptual grounding modules of object understanding for Large Language Models (LLMs), such as open-vocabulary detectors [20], [21] and segmentation tools [22], may fail for deformable cloth objects. (2) LLMs struggle to fully comprehend the deformable properties of cloth and the coordination relationship required for dual-arm manipulation, which hinders their ability to determine appropriate bimanual pick-and-place (PnP) actions directly.

This paper introduces PolyFold, a language-conditioned bimanual cloth folding framework, which performs exceptionally well in zero-shot generalization and inherent multi-step reasoning, without relying on expert demonstrations. To tackle the input challenge mentioned above, we propose to leverage geometric grounding by using parameterized polygonal models [23], [24] to represent cloth objects and inform LLMs' action generation. Polygon models serve as a simple yet effective structured abstraction for cloth that can extract the semantic meaningful keypoints and their connection relationships easily and accurately, meanwhile maintaining the internal structure of cloth and being updated continuously during the whole folding process. For the output challenge, we design a detailed planning hierarchy, utilizing LLMs to decompose user instruction into subtasks and deduce an intermediate action representation, specifically, the symmetrical fold line. Using the fold line as a strong prior and guidance, we can train a robust downstream bimanual action policy solely through random pick-and-place interaction in simulation, with a self-supervision signal derived from an ideal folded polygon model. Specifically, the policy comprises two parts: trainable spatial action map for folding result prediction and an optimization module for calculating grasp points with the highest affordance. These two modules work together, using the fold line and current RGB image as inputs and generating reliable dual-arm pick-and-place points to achieve folding result close to the ideal expected outcome.

In conclusion, the key contributions of our paper are:

(1) We introduce a parameterized polygon model based grounding framework, enabling LLMs to comprehend and reason about cloth states. Additionally, we propose to leverage LLMs to deduce intermediate action representation, specifically the symmetrical fold line, thereby enhancing their reasoning performance in cloth folding tasks.

(2) We propose a symmetrical fold line guided bimanual pick-and-place policy (FG-PnP policy) that combines spatial action map with affordance score optimization. For efficient policy training, we utilize a self-supervised learning approach with only randomly collected data.

(3) Extensive experiments in simulation and real-world scenes demonstrate that our proposed method PolyFold can zero-shot generalize to different cloth shapes and 70 unseen manipulation tasks, surpassing previous SOTA cloth folding algorithms by a significant margin.

II. Related Works

A. Cloth Manipulation

Early works on cloth manipulation [23]–[25] focus on analytical approaches utilizing geometric methods and executing scripted cloth folding sequences. While geometric representations like parameterized polygon model initially proposed

by [23] provide effective task-level abstraction of cloth, they lack autonomous reasoning ability and rely heavily on human-annotated scripted actions. Recent learning-based methods accomplish cloth manipulation by learning the policy from visual observations [4]–[6], [26], [27]. Policy-based methods usually leverage accurate visual subgoals [4], [5] and learn to imitate expert demonstrations [6], [26], [27]. However, this subgoal-conditioned pattern reveals that these methods lack multi-step reasoning abilities and can only infer single-step actions based on the nearest subgoal. Moreover, expert data collection is time-consuming and labor-intensive; the generalization ability to novel tasks, unseen cloth shapes, and real-world scenarios remains considerably limited.

Spatial action maps [28] are also widely employed in cloth manipulation [29]–[31], which efficiently recover per-pixel action values for robotic manipulation, often trained via self-supervision in simulation environments. Flingbot [29] employs a single spatial action map to learn bimanual fling actions for cloth unfolding, predicting the center position of dual-arm grasp points and using predefined rules to compute the complete fling action. It directly utilizes cloth coverage as a self-supervision signal for training the action map prediction network. However, in intricate and fine-grained bimanual cloth folding tasks, we need to consider more action variables while ensuring that dual arms' actions can meet the requirements of affordance and cooperate well. Moreover, identifying suitable self-supervision signals for folding tasks remains a challenge. To address these issues, we propose a method that combines spatial action map learning with affordance score optimization to obtain bimanual cloth folding actions. We utilize the ideal folded polygon model contour as a guiding signal for efficient self-supervision.

B. Language-conditioned Robot Manipulation

Language instructions offer a flexible and accessible means of providing task-related information, which has led recent research efforts [8], [32], [33] to concentrate on grounding language in robot manipulation tasks. In the domain of deformable object manipulation, Deng et al. [9] propose a language-conditioned approach that integrates language instructions, images, and graphs, learning from expert demonstrations to accomplish cloth folding.

The advent of Large Language Models (LLMs) and Vision Language Models (VLMs) [10] has highlighted their robust common sense reasoning ability. Recent works [11], [13], [14], [34], [35] leverage LLMs for planning feasible actions in various rigid object manipulation tasks. For deformable object manipulation, GPT-Fabric [15] utilizes VLMs to perform single-arm cloth folding tasks, but with reliance on visual subgoals. RL-VLM-F [19] employs Vision Foundation Model to provide efficient feedback for reinforcement learning in cloth folding. A concurrent work similar to ours is CLASP [16], which also focuses on developing suitable representations for cloth manipulation and uses LLMs for task planning. CLASP trains a masked autoencoder (MAE) to learn a keypoint-based representation for cloth and utilizes LLMs for high-level cloth folding action planning. In contrast, our PolyFold employs a parameterized polygon model

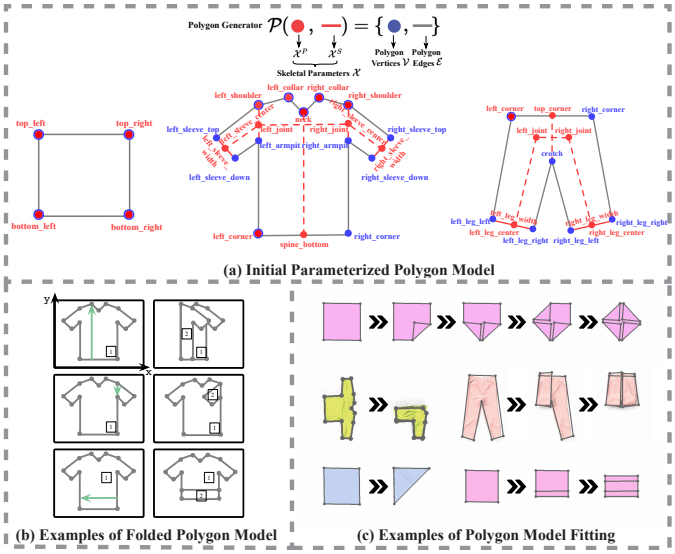


Fig. 3: Parameterized polygon model. (a) Initial parameterized polygon models of square/rectangle, t-shirt, and pant [23]. (b) Folded polygon model. (c) Fitted polygon model during the cloth folding process based on RGB image observation.

that maintains graph structure with multi-layered nodes and edges, offering richer geometric and semantic information than pure keypoints without requiring additional training. This grounding module enables a deeper understanding of deformable fabric. Furthermore, unlike CLASP's heuristic-based low-level pick-and-place planning, our method introduces a more flexible hierarchical planning framework. In our planning framework, LLMs are responsible only for inferring subgoals and intermediate action representation—fold lines, while self-supervised learning based downstream policy completes the computation of the pick-and-place actions, which is more generalizable and efficient.

III. Preliminary: Parameterized Polygon Model

The parameterized polygon model proposed by Miller et al. [23] is an effective graph-based method for representing cloth. It defines the polygon shape through a set of parameters and optimizes the shape to make it best aligned with visual observation, which also allows continuous updates to the structure during task execution. It comprises two parts: skeletal parameters $\mathcal{X}$ and polygon generator $\mathcal{P}$. Skeletal parameters are the minimal set of parameters necessary for polygon representation, which are composed of K two-dimensional skeletal or interior points $\mathcal{X}^P \in \mathbb{R}^{2\times K}$ and M one-dimensional scalar parameters $\mathcal{X}^S \in \mathbb{R}^M$. Therefore $\mathcal{X} = \left[vec(\mathcal{X}^P), \mathcal{X}^S\right]^T$ and the dimension is $2K+M$. The polygon generator $\mathcal{P}$ takes the skeletal parameters as input and produces the polygon model (vertices, edges), denoted as $\mathcal{P}(\mathcal{X}) = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V}$ is the set of polygon vertices and adjacent vertices are connected to form the edges $\mathcal{E}$ of the polygon model.

1) Initial parameterized polygon model: Fig 3(a) shows initial parameterized polygon models of square/rectangle cloth, t-shirt, and pant, which represent flattened cloth without folds. A set of skeletal parameters $\mathcal{X}_0$ of the initial polygon model is drawn in red color. The polygon generator $\mathcal{P}_0(\mathcal{X}_0)$ provides

the vertices $\mathcal{V}_0$ drawn in blue color and edges $\mathcal{E}_0$ drawn in gray color.

2) Folded polygon model: In the polygon model, every folding action is represented by a directional symmetrical fold line vector $\mathbf{F_t} = \overrightarrow{f_{t1}f_{t2}}$ for folding step t, where f_{t1} is the start point of the fold line vector and f_{t2} is the end point. As shown in Fig 3(b), the polygon generator $\mathcal{P}_t$ operates as follows: the section of the polygon situated on the left side of the fold line is lifted vertically, and placed in the mirrored position of its original position with the fold line as the axis of symmetry. For folded model, the parameters of fold line $\mathbf{F_t}$ are added to the skeletal parameters: $\mathcal{X}_{t+1} = [\mathcal{X}_t, vec([f_{t1}, f_{t2}])]^T$. The polygon generator also contains layer information of different segments, with the layer numbers increasing for upper segments [23]. For instance, in the second row of Fig 3(b), the layer number of the folded right sleeve is greater than the rest of the garment because it lies on top after folding.

3) Polygon model fitting: From polygon generator $\mathcal{P}_t(\mathcal{X}_t) = \{\mathcal{V}_t, \mathcal{E}_t\}$ we can extract the contour $\mathcal{C}_t$ of the polygon, as a function of $\mathcal{X}_t$. For folding step t, we optimize skeletal parameters $\mathcal{X}_t$ to make the polygon model contour $\mathcal{C}_t(\mathcal{X}_t)$ best aligned with the actual contour of the cloth $\tilde{\mathcal{C}}_t$ extracted from current RGB image. This alignment is achieved through energy optimization, as described by [23]. The energy function is defined as the Chamfer Distance between sampled points of $\mathcal{C}_t$ and $\tilde{\mathcal{C}}_t$.

$$
\begin{aligned}
E(\mathcal{X}_t) &= \frac{1}{2}\mathcal{D}(\Lambda(\mathcal{C}_t(\mathcal{X}_t)), \Lambda(\tilde{\mathcal{C}}_t)) + \frac{1}{2}\mathcal{D}(\Lambda(\tilde{\mathcal{C}}_t), \Lambda(\mathcal{C}_t(\mathcal{X}_t))) \\
&= \frac{1}{2}\mathcal{D}(\mathcal{C}_t^s, \tilde{\mathcal{C}}_t^s) + \frac{1}{2}\mathcal{D}(\tilde{\mathcal{C}}_t^s, \mathcal{C}_t^s) \\
\mathcal{D}(X, Y) &= \frac{1}{|X|}\sum_{x \in X} \min_{y \in Y} \|x - y\|
\end{aligned}
$$

where Λ represents the sampling process that selects a fixed number of points $\mathcal{C}_t^s$ and $\tilde{\mathcal{C}}_t^s$ uniformly along the polygon model contour $\mathcal{C}_t$ or the actual cloth contour $\tilde{\mathcal{C}}_t$ respectively.

During the energy optimization, soft constraints are introduced to define a permissible parameter range that maintains specific cloth shapes. The structural penalty function $\Gamma(\mathcal{X}_t)$ imposes penalties on parameters that violate these soft constraints and otherwise yields zero. For more details of these, refer to [23] and our Appendix.

Hence, the final energy function to be minimized is expressed as:

$$
\min_{\mathcal{X}_t} E_{\text{final}}(\mathcal{X}_t) = \frac{E(\mathcal{X}_t)}{E_{\max}} + \Gamma(\mathcal{X}_t)
$$

where $E_{\max}$ serves as a normalization constant. Using black-box optimization to optimize $\min_{\mathcal{X}_t} E_{\text{final}}(\mathcal{X}_t)$, the optimal set of skeletal parameters $\mathcal{X}_t$ is obtained, and the polygon model can be generated using $\mathcal{P}_t(\mathcal{X}_t)$. Fig 3(c) shows different fitted polygon models during the cloth folding procedure.

IV. Methodology

A. System Overview

We consider that the cloth is initially laid flattened on a horizontal surface. Given a user language instruction $\mathcal{L}$, our method PolyFold learns to generate a sequence of bimanual pick-and-place actions $\{a_t = (a_t^{pick1}, a_t^{place1}, a_t^{pick2}, a_t^{place2})\}$ $(t = 0, 1, ..., T)$, where each t represents a complete folding step, in order to fulfill the requirements of the language instruction $\mathcal{L}$. To elaborate further, as depicted in Fig 4, PolyFold can be broken down into three main components: First, the language instruction $\mathcal{L}$ is fed into the subgoal decomposition LLM module, which decomposes it into a set of divided language subgoals $\{\mathcal{L}_t\}$. Subsequently, the process iterates from $t = 0$ to $t = T$. At each folding step t, PolyFold conducts parameterized polygon model fitting to optimize parameters $\mathcal{X}_t$ for aligning the polygon model contour $\mathcal{C}_t(\mathcal{X}_t)$ with actual cloth contour $\tilde{\mathcal{C}}_t$ and obtain current polygon model $\mathcal{P}_t(\mathcal{X}_t)$. Then symmetrical fold line generation LLM takes the current polygon representation $\mathcal{P}_t$ and the language subgoal $\mathcal{L}_t$ as input and outputs the required symmetrical fold line $\mathbf{F_t}$ to achieve this subgoal. Given the fold line, PolyFold generates bimanual pick-and-place actions a_t through a learned spatial action map and conditional pick point affordance score optimization. After execution, the parameters of fold line $\mathbf{F_t}$ are integrated into the parameterized polygon model parameters $\mathcal{X}_{t+1}$. This process iterates until the final folding step T is reached.

B. Subgoal Decomposition LLM

PolyFold utilizes dual-layered Large Language Models to facilitate the analysis of cloth folding tasks from high-level task comprehension to low-level action planning. The upper layer, denoted as the subgoal decomposition LLM ($\mathbf{LLM_{sd}}$), processes the user language instruction $\mathcal{L}$ and also the initial polygon model template $\mathcal{P}_0$ as input, yielding a sequence of decomposed language subgoals $\{\mathcal{L}_t\}$, each of which is intended to be executed in one step of bimanual pick-and-place action. The subgoal decomposition LLM ($\mathbf{LLM_{sd}}$) is designed to produce language subgoals in a structured format, as follows: *fold <which part of the cloth > <direction, optional> to <which part of the cloth><symmetrical axis, optional>*. Therefore it is denoted as:

$$
\{\mathcal{L}_t\} = \mathbf{LLM_{sd}}(\mathcal{L}, \mathcal{P}_0) \qquad (t = 0, 1, 2, ..., T)
$$

C. Symmetrical Fold Line Generation LLM

1) Parameterized polygon model fitting and grounding: For each step t, PolyFold starts with utilizing black box optimization to derive an optimal set of skeletal parameters $\mathcal{X}_t$ that aligns current polygon model contour $\mathcal{C}_t(\mathcal{X}_t)$ with current observed cloth contour $\tilde{\mathcal{C}}_t$. The obtained polygon model $\mathcal{P}_t(\mathcal{X}_t)$ is utilized as an effective representation for current cloth state. To help LLM better understand the polygon model structure, it is then transformed into a structured text-based representation, specifically in the form of a Python dictionary, as shown in Fig 4. This dictionary comprises two distinct entries for *vertex* and *edge*. The *vertex* dictionary uses the *vertex_name* as keys, with corresponding values being instances of the *Point* class that include coordinates, a detailed description of the vertex, and the layer information. The *edge* dictionary maintains the connectivity relationships for each layer of the polygon model.

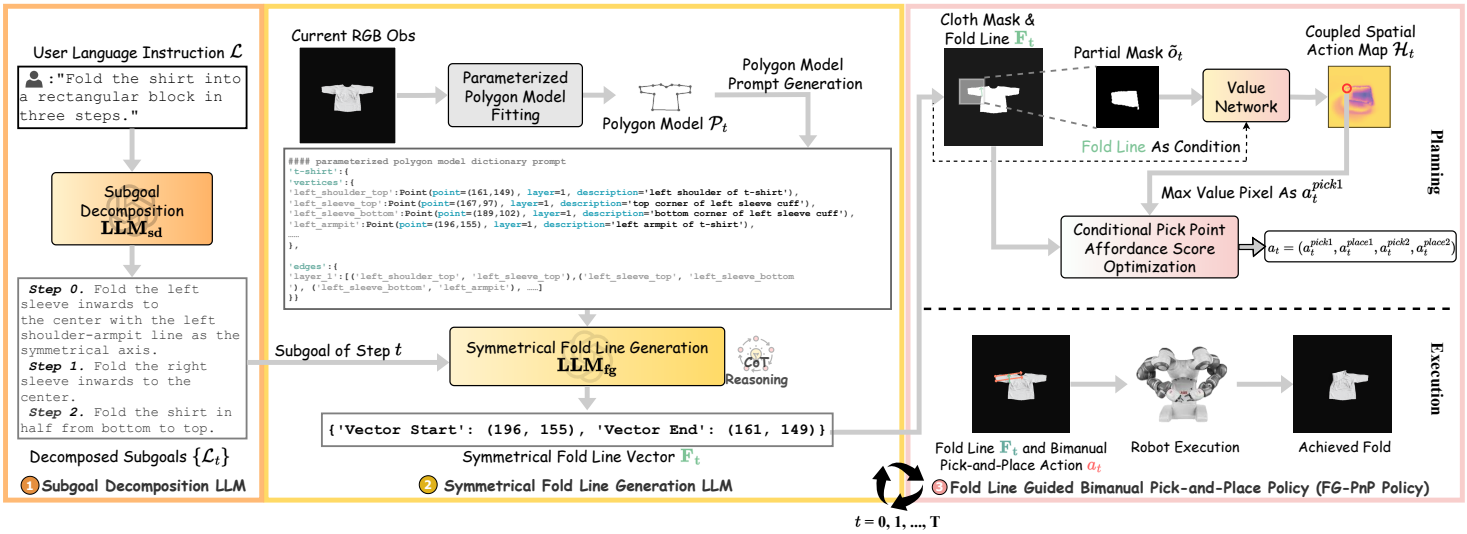


Fig. 4: PolyFold architecture overview. (1) Given the user's final goal instruction $\mathcal{L}$, we first decompose it into subgoals using Large Language Model $\mathbf{LLM_{sd}}$. (2) For each subgoal step from $t = 0$ to $t = T$, current RGB observation is used to perform parameterized polygon model fitting. The resulting polygon model $\mathcal{P}_t$ and current subgoal $\mathcal{L}_t$ are fed into the symmetrical fold line generation LLM $\mathbf{LLM_{fg}}$. (3) Using the inferred symmetrical fold line $\mathbf{F_t}$ as guidance, we learn a policy (FG-PnP policy) to generate a bimanual pick-and-place action. After robot execution, the current observation is updated, and the algorithm proceeds to the next subgoal.

The polygonal model representation aids the Large Language Model (LLM) in comprehending the current cloth structure, thereby facilitating the LLM's ability to reason about actions that can fulfill the language subgoal at the present time step.

2) Symmetrical fold line as an intuitive and efficient intermediate action representation: Given the challenges in LLM's abilities in comprehending the deformable nature and complex shapes of various fabrics, as well as the complexities of dual-arm coordination, allowing the LLM to directly deduce bimanual pick-and-place actions may result in undesirable cloth folding actions, as demonstrated in ablation experiments in Table V. As illustrated in Section III, the fold line is employed in parameterized polygon model to represent folding action via a symmetry relationship, where one segment of the polygon split by the fold line is folded on another. This symmetry-based representation is easier for LLMs to reason about than precise bimanual pick-and-place actions. Moreover, in the planning of bimanual pick-and-place action a_t, once the fold line is determined, the number of variables that need to be calculated can be reduced from 8 ((x, y) *pixels of* $a_t^{pick1}, a_t^{place1}, a_t^{pick2}, a_t^{place2}$) to 4 ($(x, y)$ *pixels of* a_t^{pick1}, a_t^{pick2}) as the place points are symmetrical to the pick points relative to the fold line. The fold line also constrains the search space for pick points to the segment of the image located on one side of the fold line due to its definition, further enhancing action planning efficiency. Therefore we adopt the symmetrical fold line as an efficient intermediate action representation for LLMs to reason about.

3) Symmetrical fold line generation LLM: Given current subgoal language instruction $\mathcal{L}_t$ and current polygon model $\mathcal{P}_t(\mathcal{X}_t)$, the fold line vector $\mathbf{F_t}$ is generated using the symmetrical fold line generation LLM ($\mathbf{LLM_{fg}}$). This process employs chain-of-thought [36] reasoning with a few user-provided examples, instructing the LLM to determine, based on the characteristics and complexity of the current task and current polygon model, whether to directly infer the fold line from the geometric shape or to first reason about a suitable unimanual pick-and-place action and subsequently inferring the fold line as the perpendicular bisector of the line segment connecting the pick and place pixels. This process can be represented as follows:

$$\mathbf{F_t} = \mathbf{LLM_{fg}}(\mathcal{L}_t, \mathcal{P}_t(\mathcal{X}_t))$$

D. Fold Line Guided Bimanual Pick-and-Place Policy (FG-PnP Policy)

Given the symmetrical fold line $\mathbf{F_t}$ generated by $\mathbf{LLM_{fg}}$, we propose a fold line guided bimanual pick-and-place policy (FG-PnP policy). Due to the complex dynamics of deformable objects, we follow previous method [28] to learn a spatial action map for action value prediction. Specifically here we leverage the spatial action map for folding result prediction to determine suitable pick-and-place points. However, bimanual cloth folding poses additional complexities. Naively learning the actions of two grippers using two individual spatial action maps would lead to poor learning outcomes because it ignores garment manipulation affordance and the collaborative relationship between the two arms. Another approach is to acquire dual-arm actions in a carefully designed cascaded way. Previously Flingbot [29] first learns a spatial action map for the center position of dual-arm grasp positions and then uses predefined rules to calculate the entire fling action. Inspired by this, we adopt a more advanced method: a spatial action map predicts the folding result and a pick position a_t^{pick1} is obtained from it. Subsequently, we use conditional affordance score optimization to acquire the second gripper's pick point a_t^{pick2} conditioned on a_t^{pick1} and the fold line $\mathbf{F_t}$. This ensures the selection of bimanual operational points with the highest affordance and optimal folding outcomes.

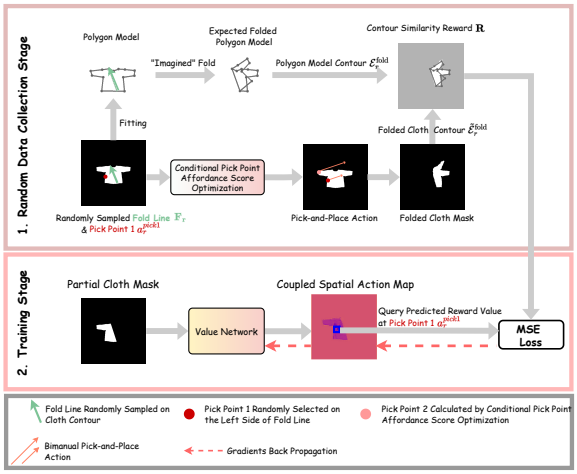


Fig. 5: Training fold line guided pick-and-place policy (FG-PnP policy). (1) Random data collection stage: In each collection step, fold line and robot actions are randomly on the cloth contour and we record the contour similarity between the expected folded model and the resulting folded cloth, which serves as the ground truth similarity reward $\mathbf{R}$. (2) Training stage: With the partial cloth mask as input, the output spatial action map is supervised by ground truth similarity reward through $\mathrm{MSELoss}$.

1) Conditional pick point affordance score optimization: Here we first introduce how to determine the pick position of the second gripper a_t^{pick2}, conditioned on the given fold line $\mathbf{F_t} = \overrightarrow{f_{t1}f_{t2}}$, the first gripper's pick position a_t^{pick1}, and current contour of the cloth $\tilde{\mathcal{C}}_t$. In light of the affordance in deformable cloth manipulation, it is crucial to prioritize the stability of bimanual folding actions to minimize internal deformation in the folded cloth. We utilize two simple yet effective key metrics to assess this: the quadrilateral area $\mathcal{S}$ and the perpendicular distance ℓ from the grasp point to the fold line. The quadrilateral area $\mathcal{S}(f_{t1}, f_{t2}, a_t^{pick1}, a_t^{pick2})$ denotes the area of the quadrilateral formed by the starting point f_{t1} and ending point f_{t2} of the fold line, and also two pick points a_t^{pick1}, a_t^{pick2}. The perpendicular distance $\ell(a_t^{pick2}, \mathbf{F_t})$ represents the distance between the second pick point and its projection onto the fold line. Maximizing the area and perpendicular distance can optimize the operational space of dual-arm manipulation, minimizing deformation of the fabric during manipulation and achieving the folding outcome that aligns with what the fold line desires. The conditional pick point affordance score optimization process can be detailed as follows and also shown in Fig 6:

$$\max_{a_t^{pick2}} \frac{\mathcal{S}(f_{t1}, f_{t2}, a_t^{pick1}, a_t^{pick2})}{\mathcal{S}_{\max}} + \gamma \cdot \frac{\ell(a_t^{pick2}, \mathbf{F_t})}{\sqrt{\mathcal{S}_{\max}}}$$
$$\text{s.t.} \quad a_t^{pick2} \in \tilde{\mathcal{C}}_t \quad \text{and} \quad \overrightarrow{f_{t1}a_t^{pick2}} \times \overrightarrow{f_{t1}f_{t2}} > 0$$

where $\mathcal{S}_{max}$ is the area of the bounding box encompassing the partial cloth mask $\tilde{o}_t$, which is utilized to normalize $\mathcal{S}$ and ℓ within the range of 0 to 1. Due to the definition of the fold line in Section III, a_t^{pick2} is only selected on the cloth contour located on the left side of the symmetrical fold line $\mathbf{F_t}$. Similar to polygon fitting in Section III, the process is also achieved through black box optimization.

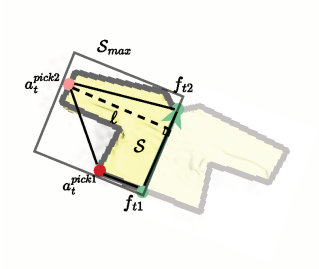


Fig. 6: Illustration of the objective function for conditional pick point affordance score optimization, which contains the quadrilateral area $\mathcal{S}(f_{t1}, f_{t2}, a_t^{pick1}, a_t^{pick2})$ and the perpendicular distance $\ell(a_t^{pick2}, \mathbf{F_t})$.

2) Coupled spatial action map: As shown in Fig 4, given current top-down RGB observation, we employ a value network to learn a spatial action map. Each pixel in this map represents the predicted folding outcome of a bimanual action, derived from the current pixel and conditional pick point affordance score optimization. Since a single action map predicts the outcome of a coupled bimanual action, where one arm's action depends on the other's, we term this map the *coupled spatial action map*. According to the definition of the fold line in Section III (2), only the part on the left side of the fold line is grasped and folded over to the right. To minimize the influence of irrelevant information, we crop the segmented mask, retaining only the portion on the left of the fold line, denoted as $\tilde{o}_t$. Using this partial mask as input enhances the generalization capacity of our value network. The value network is a 2D conditional UNet [37]. With the positional encoding [38] of the start point f_{t1} and end point f_{t2} as condition, the value network takes the partial mask $\tilde{o}_t$ as input and outputs a spatial action map $\mathcal{H}_t$. Then one pick point a_t^{pick1} is selected as: $a_t^{pick1} = \arg\max \ \mathcal{H}_t$.

3) Learning coupled spatial action map with random interaction data and self-supervision: In this learning process, we aim to completely eliminate expert demonstrations and human annotations, relying solely on randomly collected data and self-supervised signals to learn a coupled spatial action map with strong predictive capabilities. As shown in Fig 5, training data is collected through random interactions in simulation. In each collection step, a symmetrical fold line $\mathbf{F_r} = \overrightarrow{f_{r1}f_{r2}}$ is first arbitrarily chosen along the cloth's contour $\tilde{\mathcal{C}}_r$, and one pick point a_r^{pick1} is randomly selected on the left side of $\mathbf{F_r}$ on $\tilde{\mathcal{C}}_r$. Then the second pick point a_r^{pick2} is obtained by conditional pick point affordance score optimization. Following the execution of the pick-and-place actions, the resulting folded cloth contour is denoted as $\tilde{\mathcal{C}}_r^{\text{fold}}$. Here we reuse the polygon model as an approximate groundtruth to evaluate the folding result by contour similarity reward $\mathbf{R}$, which compares the resulting folded cloth contour $\tilde{\mathcal{C}}_r^{\text{fold}}$ with the expected folded polygon model contour $\mathcal{C}_r^{\text{fold}}$:

$$\mathbf{R} = \beta - \alpha \cdot [\frac{1}{2}\mathcal{D}(\Lambda(\tilde{\mathcal{C}}_r^{\text{fold}}), \Lambda(\mathcal{C}_r^{\text{fold}})) + \frac{1}{2}\mathcal{D}(\Lambda(\mathcal{C}_r^{\text{fold}}), \Lambda(\tilde{\mathcal{C}}_r^{\text{fold}}))]$$

where α and β are constants for scaling the contour similarity to a relatively reasonable range. A higher $\mathbf{R}$ reflects that the folding result is better. As shown in Fig 5 we train the value

TABLE I: Evaluated cloth folding tasks.

Cloth Type	Task Type	single-step task	multi-step task
Square	S-Corner-Folding	4	7
	S-Triangle-Folding	4	8
Rectangle	R-Edge-to-Middle-Folding	4	4
	R-Edge-to-Opposite-Folding	4	8
T-shirt	T-Sleeve-Folding	4	8
	T-Half-Folding	3	0
	T-Block-Folding	0	2
Pant	P-Half-Folding	4	2
	P-Block-Folding	0	4
Total		**27**	**43**

network to predict the contour similarity reward $\mathbf{R}$, utilizing MSE Loss with ground truth contour similarity reward as supervision, or more precisely, self-supervision, because the fitting and folding process of the polygon model do not require manual labeling and can be fully autonomously completed.

4) Place point calculation and single-arm action switching: Given a_t^{pick1}, a_t^{pick2}, the place points a_t^{place1}, a_t^{place2} are calculated as the mirrored points with respect to the fold line $\mathbf{F_t}$. If the two pick points are too close to each other, the bimanual action is switched to single-arm action: $a_t^{pick} = \frac{a_t^{pick1}+a_t^{pick2}}{2}, a_t^{place} = \frac{a_t^{place1}+a_t^{place2}}{2}$.

5) Implementation details: The training data contains 10000 random folding actions and it is only collected on 100 t-shirts and results in Section V prove that it can generalize to different cloth types. The dimension of RGB image is 400×400. Hyperparameters are selected as: $\alpha = 1500$, $\beta = 1$, $\gamma = 0.04$. The threshold for single-arm action switching in pixel coordinates is 8 pixels. The network is trained on a 2×RTX 2080 Ti machine, which takes 39 hours.

V. Experimental Results

A. Simulation Experiment Setup

All simulation experiments are conducted in the SoftGym simulator [39], using cloth meshes from CLOTH3D [40], with unsuitable or unrealistic objects excluded. The evaluation includes 100 square cloths, 100 rectangular cloths, 100 t-shirts, and 100 pants. Importantly, the t-shirts used for evaluation differ from those in Section IV-D for policy training, ensuring that all evaluation cloths are unseen, presenting significant challenges for generalization.

1) Evaluated Tasks: We define 9 types of folding tasks for 4 cloth types, named in the format *cloth_type-folding_type*, resulting in a total of 70 evaluated tasks, each comprising multiple sub-tasks. For square cloth, tasks include *S-Corner-Folding* and *S-Triangle-Folding*. For rectangle cloth, we define *R-Edge-to-Middle-Folding* and *R-Edge-to-Opposite-Folding*. T-shirt tasks include *T-Sleeve-Folding*, *T-Half-Folding*, and *T-Block-Folding*, while pant tasks consist of *P-Half-Folding* and *P-Block-Folding*. Details of the generated tasks are presented in TABLE I. Tasks are categorized as single- or multi-step and are evaluated using 5 variations of language instructions expressing the same meaning differently. Further text and image descriptions are provided in the Appendix.

2) Baselines: We compare the performance of our proposed PolyFold to the following baselines:

- *Foldsformer* [4] is current state-of-the-art vision-conditioned cloth folding method. It uses a sequence of visual subgoals to determine pick-and-place actions. For comparison, we also implement a dual-arm variant of Foldsformer, referred to as *Foldsformer-2Arm*.
- *Language-Deformable* [9] (abbreviated as *LangDef*) is the state-of-the-art language-conditioned cloth manipulation approach. It takes in different user-provided language instructions for different folding sub-steps and utilizes multi-modal embeddings for pick-and-place actions generation. We also implement a dual-arm version LangDef for comparison, annotated as *LangDef-2Arm*.

3) Evaluation metrics: Following previous works [4], [9], we use the *Mean Particle Position Error (MPPE)* metric to measure the similarity between the executed fold and the ground truth from the oracle demonstrator, as cloth is modeled as particles in the simulation. Additionally, the *Mean Intersection over Union (mIoU)* metric is employed to evaluate the overlap between the folded cloth mask and the ideal mask from the oracle. Finally, the *success rate* is calculated based on certain thresholds of MPPE and mIoU metrics, which are explained in detail in the Appendix.

B. Simulation Experiment Results

1) Generalization ability: Generalization involves both object-level and task-level evaluation. For object-level generalization, we assess the performance of our method and baselines on clothing shapes unseen during training. For task-level generalization, we evaluate performance on entirely new tasks. As PolyFold does not use task-specific data for training, all tasks are considered unseen. Moreover, all evaluated cloth objects are also unseen for PolyFold. In contrast, for baselines, sub-tasks within each task type are evenly divided into seen and unseen categories, with expert demonstrations of only the seen tasks used for training. Two experiments are designed to evaluate generalization ability: *unseen cloth + seen task* and *unseen cloth + unseen task*, with the latter posing greater difficulty (see TABLE II). PolyFold is evaluated exclusively in the *unseen cloth + unseen task* scenario, while baselines are tested in both. To mitigate the complexity for baselines, we provide accurate visual or language subgoals based on their original settings [4], [9]. In contrast, PolyFold relies solely on the language description of the final goal, making its evaluation more challenging.

From TABLE II and Fig 8, it is evident that in the *unseen cloth + unseen task* evaluation setting, PolyFold significantly outperforms all baselines across four types of cloth folding tasks in terms of mean particle position error, mIoU, and success rate metrics. In contrast, baselines perform poorly in this challenging scenario, highlighting their inability to generalize when faced with clothing shapes and folding tasks not seen during training. Visualization of representative task evaluations in Fig. 7 further illustrates that under the *unseen cloth + unseen task* setting, PolyFold produces folding results that align closely with the given language instructions. In contrast, baselines either replicate folds from their training data or infer only partial actions, failing to generalize effectively despite being provided with accurate per-step subgoals. Moreover, as our method is trained solely on t-shirt data, with no exposure to square, rectangle, or pant clothes during training, PolyFold demonstrates exceptional zero-shot generalization capability.

TABLE II: Simulation evaluation of our proposed PolyFold and baselines. Each method is tested on four types of cloth (square, rectangle, t-shirt, and pant) and evaluated using three metrics MPPE (mm), mIoU, success rate (%).

Method	Square			Rectangle			T-shirt			Pant		
	MPPE ↓	mIoU ↑	success% ↑	MPPE ↓	mIoU ↑	success% ↑	MPPE ↓	mIoU ↑	success% ↑	MPPE ↓	mIoU ↑	success% ↑
Unseen Cloth + Seen Task †												
Foldsformer [4]	12.59	0.915	95.3	72.67	0.555	35.1	29.60	0.839	65.5	59.43	0.693	43.6
Foldsformer-2Arm	**10.03**	**0.923**	**98.0**	**15.28**	**0.911**	**98.3**	**26.11**	0.835	**71.3**	**51.60**	**0.812**	**68.7**
LangDef [9]	12.81	0.913	94.9	71.03	0.563	34.7	28.42	**0.845**	67.8	59.59	0.700	46.3
LangDef-2Arm	18.21	0.886	91.1	19.72	0.869	90.2	52.95	0.717	32.0	75.52	0.730	36.1
Unseen Cloth + Unseen Task												
Foldsformer [4]	92.39	0.596	25.3	150.77	0.420	0.0	54.95	0.735	54.2	236.82	0.340	14.6
Foldsformer-2Arm	118.23	0.486	3.7	113.29	0.523	6.7	50.35	0.718	40.0	111.79	0.612	15.6
LangDef [9]	125.34	0.515	13.5	147.15	0.425	0.0	52.59	0.750	58.2	235.02	0.334	17.3
LangDef-2Arm	148.25	0.447	5.5	163.90	0.375	0.8	89.77	0.584	14.7	276.52	0.231	4.0
PolyFold (Ours)	**19.58**	**0.872**	**94.0**	**23.30**	**0.862**	**90.8**	**19.74**	**0.894**	**88.3**	**49.66**	**0.830**	**79.6**

† In this unseen cloth + seen task setting, only four baselines are evaluated to compare their generalization ability to unseen cloth, as well as a cross-comparison with the performances under the unseen cloth + unseen task setting. Since all evaluated cloth and tasks are unseen for our method PolyFold, our method does not appear in this setting and only appears in the unseen cloth + unseen task setting.

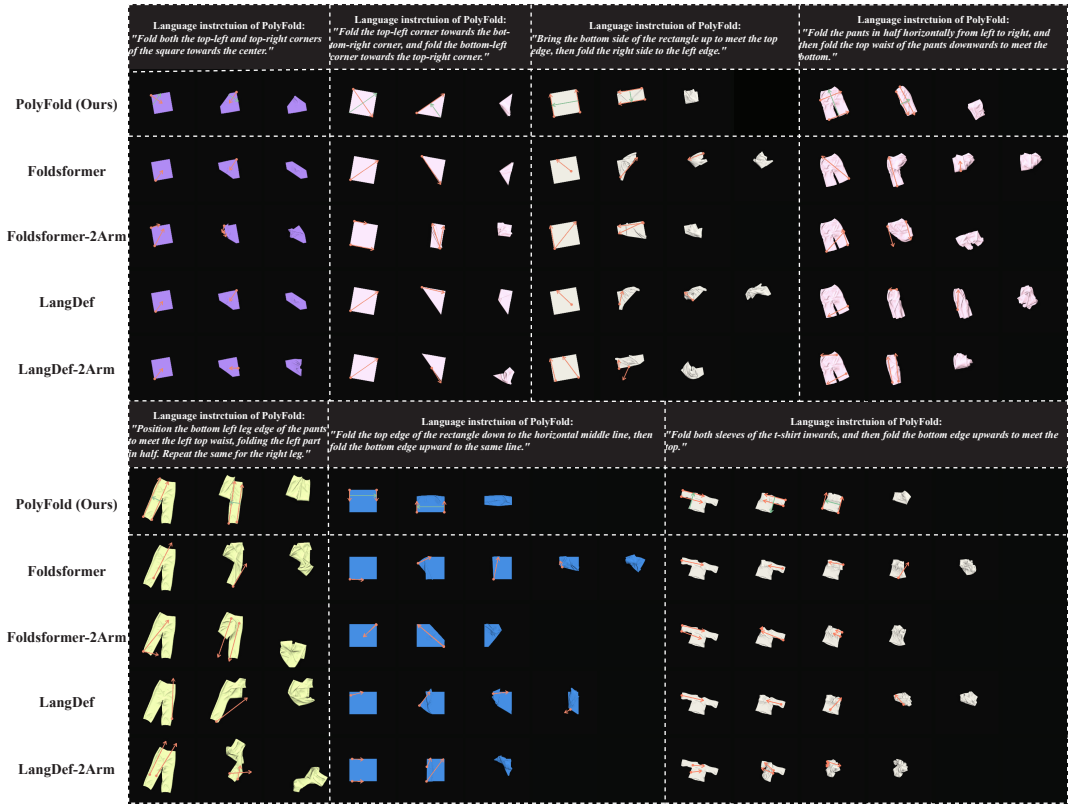


Fig. 7: Visualization of task execution in the SoftGym simulation compares our method, PolyFold, with four baseline models under the *unseen cloth + unseen task* setting. The orange arrow indicates the pick-and-place action for one robot arm, while the green arrow represents the symmetrical fold line deduced by PolyFold's LLM. For brevity, only PolyFold's language instructions are displayed; all baselines are provided with precise visual or language subgoals, detailed in the Appendix.

In the simpler *unseen cloth + seen task* setting, baselines perform exceptionally well on square folding tasks due to the standardized nature of square shapes, which only vary in size. However, their performance declines as the complexity of the clothing shapes increases in rectangle, t-shirt, and pant folding tasks. Among the four baselines, Foldsformer-2Arm achieves the best overall performance. Notably, our proposed PolyFold in the more challenging *unseen cloth + unseen task* setting surpasses the baselines' performance on t-shirt and pant folding tasks, even when the baselines operate in the simpler *unseen cloth + seen task* scenario. This highlights PolyFold's superior generalization capability.

2) Multi-step reasoning ability: In this experiment, each baseline is provided only with the final goal, either as a goal image or a language instruction, to verify whether the methods can infer the entire sequence of actions based solely on the

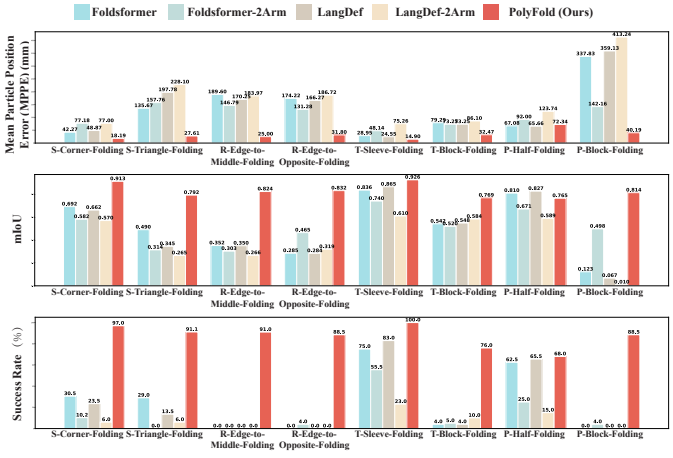


Fig. 8: Evaluation results of our proposed PolyFold and baselines on multi-step tasks conducted in the *unseen cloth + unseen task setting.*

final goal, rather than a series of precise subgoals. All methods are evaluated on the 21 multi-step tasks in seen tasks, which are seen in the training process for four baselines. The results in TABLE III show that when provided only with a final goal, success rates of these four baseline methods are very low, indicating that four baselines lack sufficient inherent multi-step reasoning abilities when given only the final goal. This limits the application scenarios of these methods, as precise subgoals for each step must be provided to complete a successful cloth folding task. In contrast, our method demonstrates strong multi-step reasoning and planning capability, with the aid of Large Language Models and our carefully designed hierarchical architecture.

3) Sample efficiency and scalability with random interaction data: Both our method and baselines rely on supervised-learning-based policies; however, our method is trained solely with random interaction data, while the baselines are trained with expert demonstration data. Here we evaluate how the performance changes while the amount of training data changes. We train FG-PnP policy of PolyFold with 2500, 5000, 10000 sets of training data and 10000 is the full dataset size as described in Section IV-D. We train four baselines with 2500, 5000, 10000, 15000 sets of expert demonstration data, while 15000 is the full dataset size for baselines. From Fig 9 we find that as the amount of training data increases, the metrics of all methods become better. Remarkably, our method trained with only 2500 random interaction data already surpasses most of the results from the baselines trained with expert demonstrations. This demonstrates the high sample efficiency of our approach, which is due to the utilization of LLMs, allowing the policy to focus exclusively on downstream planning tasks, specifically the calculation of pick-and-place points, by leveraging fold lines as useful prior knowledge rather than requiring the model to understand the entire task from scratch.

TABLE III: Experiment results of multi-step reasoning. Each method is provided with only the final goal instead of precise subgoals.

Method	MPPE ↓	mIoU ↑	success% ↑
Foldsformer [4]	120.52	0.526	18.5
Foldsformer-2Arm	63.38	0.702	50.3
LangDef [9]	118.46	0.536	19.5
LangDef-2Arm	148.17	0.420	9.2
PolyFold (Ours)	**29.65**	**0.847**	**88.4**

4) Inference time: The inference speeds are presented in Table IV. Due to the LLM's API usage, our inference time is significantly slower than baselines purely using local neural networks for inference. This presents the bottleneck and limitation of our algorithm. In the actual execution process, we can perform reasoning in parallel with the reset of robot arms, making full use of time and reducing the noticeable delay.

C. Ablation Study

A series of ablation methods are conducted to evaluate the rationality and effectiveness of each part in PolyFold and to answer: (1) whether the parameterized polygon model is an appropriate abstraction of cloth states for grounding LLMs in garment folding tasks; (2) whether the common sense reasoning ability of LLMs is sufficient to directly generate bimanual pick-and-place actions in cloth folding, and if it is necessary to introduce the concept of a fold line as an intermediate representation; (3) whether our proposed FG-PnP policy combining coupled spatial action map learning with conditional pick-point affordance score optimization outperforms pure learning-based methods; and (4) how the performance of different LLMs varies.

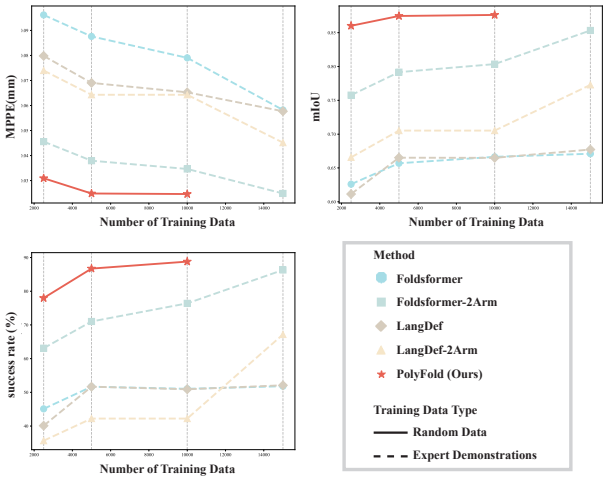


Fig. 9: Variation in method performance with varying training data sizes and different training data types. These methods are evaluated on the challenging multi-step tasks.

To address (1), we compare PolyFold with four ablation methods representing different grounding approaches in fabric manipulation, denoted as: *Keypoints*, *Image*, *Keypoints + Image*, and *Polygon Model + Image*. *Keypoints* utilizes method in ClothFunnels [30] to detect keypoints in the outermost contour of the cloth, employing these anonymous keypoints as the grounding module (which lack exact semantic meaning and topological structure of the cloth) for LLMs. *Image* uses raw RGB image of cloth as the input, with the Large Language Models upgraded into Vision Language Models (VLMs) to process visual input. *Keypoints + Image* combines both detected keypoints and current image as inputs, while *Polygon Model + Image* uses parameterized polygon model and the image as joint multi-modal inputs. Table V shows that *Image* directly using raw images performs poorly in almost all tasks. In simpler and easily understandable task settings like square and rectangle folding, performance differences among methods other than *Image* are minimal. However, for complex shapes like t-shirts and pants, most tasks of *Keypoints* fail at a rate higher than 50% due to its lack of semantic and structural grounding. While combining keypoints and images in VLMs improves performance, a significant gap remains compared to our polygon model. Furthermore, the result of *Polygon Model + Image* shows a slight improvement compared to pure *Polygon Model* due to image assistance; however, introducing images in VLMs causes considerable additional token/money costs. Therefore, considering the trade-off, we opt for the polygon model alone, as the performance of parameterized polygon model alone is already good enough.

To validate (2), we perform two experiments: removing the fold line guided pick-and-place policy and replacing the fold line generation LLM with an LLM that directly infers pick-and-place actions. In *Direct-1Arm*, the LLM directly infers

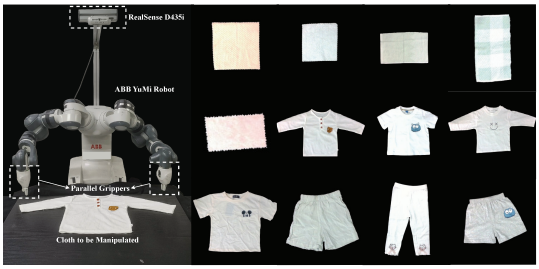
TABLE IV: Inference time (s) for each step of cloth folding.

Algorithm	Baselines				PolyFold (Ours)			
	Foldsformer [4]	Foldsformer-2Arm	LangDef [9]	LangDef-2Arm	Polygon Model Fitting	LLM Inference	FG-PnP Policy	All
Time	0.0100 ± 0.0023	0.0109 ± 0.0025	0.0382 ± 0.0196	0.0417 ± 0.0207	0.2911 ± 0.1574	2.4130 ± 0.1398	0.0071 ± 0.0005	2.7112 ± 0.2111

All results reported are tested on the same machine with an Intel i5-12600KF CPU and an NVIDIA RTX 3070Ti GPU.

TABLE V: Quantitative results of ablation studies on the choice grounding module, usage of fold line as an intermediate representation, our proposed FG-PnP policy, and different types of LLMs.

Method	Square			Rectangle			T-shirt			Pant		
	MPPE ↓	mIoU ↑	success% ↑	MPPE ↓	mIoU ↑	success% ↑	MPPE ↓	mIoU ↑	success% ↑	MPPE ↓	mIoU ↑	success% ↑
Ablation on Grounding Module												
Keypoints	14.11	0.895	93.4	23.67	0.871	89.6	98.99	0.581	36.5	77.17	0.730	57.0
Image	29.96	0.822	78.8	70.20	0.705	55.9	51.90	0.742	48.2	175.57	0.503	18.5
Keypoints + Image	9.55	**0.925**	**99.2**	28.61	0.854	89.9	38.06	0.810	62.4	83.86	0.716	60.4
Polygon Model + Image	**9.29**	0.919	95.5	**21.23**	**0.883**	**94.9**	23.62	0.862	83.8	**45.98**	**0.821**	**84.7**
Polygon Model	16.27	0.892	95.3	22.60	0.872	90.0	**22.78**	**0.883**	**86.6**	50.04	0.820	79.0
Ablation on Usage of Fold Line												
Direct-1Arm	10.48	**0.909**	**97.6**	38.36	0.822	59.6	32.34	0.831	71.8	109.83	0.662	24.6
Direct-2Arm	**10.38**	0.909	97.2	48.35	0.815	62.8	25.37	0.852	76.9	81.65	0.720	46.0
Foldline	16.27	0.892	95.3	**22.60**	**0.872**	**90.0**	**22.78**	**0.883**	**86.6**	**50.04**	**0.820**	**79.0**
Ablation on FG-PnP Policy												
2sam	16.02	0.891	92.3	46.45	0.785	55.2	18.96	0.893	87.4	67.25	0.763	51.4
1sam	26.08	0.844	86.4	68.57	0.699	21.3	29.31	0.845	76.6	87.36	0.704	40.7
csam+caso	**10.47**	**0.918**	**98.6**	**16.05**	**0.913**	**98.0**	**16.26**	**0.906**	**93.3**	**31.25**	**0.871**	**92.8**
Ablation on Different LLMs												
gpt-3.5	9.71	0.922	96.5	95.99	0.626	51.1	70.81	0.667	49.8	113.85	0.629	48.4
claude-3-opus	**9.27**	**0.925**	96.9	**17.10**	**0.911**	**96.8**	26.70	0.857	86.2	80.74	0.703	57.3
gemini-1.5-pro	9.32	0.925	**97.1**	20.16	0.904	94.8	25.57	0.867	85.5	85.75	0.691	53.6
gpt-4o	16.27	0.892	95.3	22.60	0.872	90.0	**22.78**	**0.883**	**86.6**	**50.04**	**0.820**	**79.0**



(a) Real World Experiment Setup (b) Cloth Instances Used in Experiments

Fig. 10: (a) Real world experiment setup. (b) Cloth instances used in real-world experiments.

single-arm pick-and-place actions, while in *Direct-2Arm*, it directly infers bimanual pick-and-place actions. Results in Table V show that direct inference of pick-and-place actions performs relatively well, even surpassing PolyFold (which uses the fold line as an intermediate representation) in simple square folding tasks. However, performance significantly deteriorates in more complex scenarios, such as folding t-shirts and pants, where cloth shapes are more intricate. Across 70 tasks, PolyFold, which infers the intermediate representation (fold line) and relies on the downstream FG-PnP policy for bimanual pick-and-place action calculations, demonstrates the most stable and superior performance. These findings indicate that the reasoning capabilities of LLMs are insufficient for directly generating pick-and-place actions in complex cloth folding tasks, supporting the choice to utilize the fold line as an intermediate representation.

For question (3), we compare our downstream FG-PnP policy, which combines the **c**oupled **s**patial **a**ction **m**ap (*csam*) with **c**onditional pick-point **a**ffordance **s**core **o**ptimization (*caso*), denoted as *csam+caso*, against two alternative pure learning-based approaches: learning two spatial action maps (*2sam*) for bimanual actions and learning one spatial action map (*1sam*) for single-arm actions. To isolate the impact of incorrect fold lines inferred by the LLMs, we use fold lines annotated by the oracle demonstrator. Learning two individual spatial action maps results in poor outcomes due to the intricate coupling of bimanual actions, while only obtaining single-arm actions from one spatial action map is inadequate for complex deformable object manipulation. Our proposed *csam + caso* architecture in FG-PnP policy is best suited to the complex nature of bimanual deformable object manipulation.

To address point (4), we evaluate our proposed method using different Large Language Models (gpt-3.5-turbo, claude-opus, gemini-1.5-pro, gpt-4o). Results reveal that gpt-3.5-turbo model performs really well in the square tasks but performs poorly in other tasks. The performance of claude and gemini model is better than gpt-3.5-turbo overall but still worse than gpt-4o. Therefore gpt-4o is chosen as the default model.

TABLE VI: Real world experiment results of PolyFold.

Metrics\Cloth Type	Square	Rectangle	T-shirt	Pant	Average
mIoU	0.793	0.795	0.916	0.816	0.827
success%	82.6	82.5	94.1	85.0	85.7

D. Real World Experiments

We utilize the ABB Yumi robot as our real-world experimental platform, with an Intel RealSense D435i camera positioned overhead to provide a top-down view, as shown in Fig 10. In real-world experiments, we utilize 12 different cloth instances that vary in shapes, textures, sizes, and materials, categorized as square, rectangle, t-shirt, and pant. Those cloth instances are evaluated on the 70 tasks in Table I. Similar to simulation experiments, we calculate the *mIoU* and *success rate* metrics compared with the oracle demonstrations.

As shown in Fig 1 and Table VI, experiments on four different cloth types across different tasks perform well, with most results approaching those of the oracle demonstrations. This demonstrates that our proposed PolyFold framework can be seamlessly deployed in real-world experiments without any need for fine-tuning or adaptation, despite the variations in cloth shapes, textures, sizes, and materials. Videos of real-world experiments are available on website https://sites.google.com/view/polyfold.

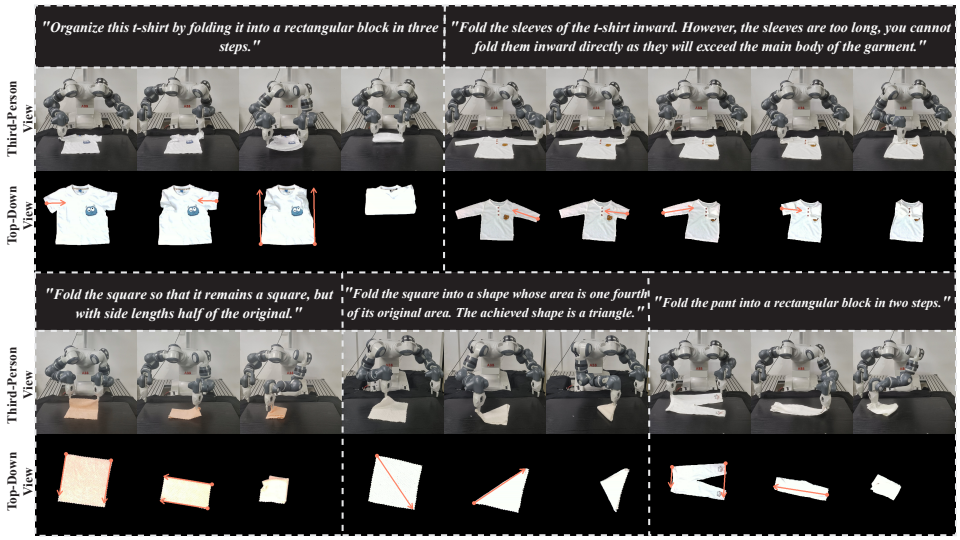


Fig. 11: Real-world task evaluation with only ambiguous user instructions as input. A third-person front view and a top-down view of the ABB YuMi robot execution process are shown and the orange arrow represents the pick-and-place action.

We also perform an interesting experiment using ambiguous user language instructions to guide cloth folding tasks, as illustrated in Fig 11. Instead of explicitly directing the robot on specific actions, we provide instructions based on geometric features, task objectives, and general task characteristics. Leveraging the common sense reasoning capabilities of LLMs, our framework effectively completes these tasks on a real-world robot. For example, when given the instruction to *organize the t-shirt by folding it into a rectangular block in three steps*, the robot infers that it should first fold the left and right sleeves inward, followed by folding the bottom edge upward. Using the downstream FG-PnP policy, the robot determines a sequence of appropriate pick-and-place actions. After execution, the results align with user expectations, showcasing the framework's ability to interpret ambiguous instructions and translate them into precise actions.

VI. CONCLUSIONS

In this paper we introduce PolyFold, a novel language-conditioned bimanual cloth folding framework that features strong zero-shot generalization, inherent multi-step reasoning capability, and expert-demonstration-free policy learning. The essence of PolyFold lies in the introduction of parameterized polygon model as an efficient grounding module for LLM, the introduction of fold line as an appropriate intermediate action for LLM reasoning and also sample efficient self-supervised pick-and-place policy purely trained with random data. We believe PolyFold points out an efficient way for generalizable deformable object manipulation.

There are limitations in our approach that require further investigation. As shown in Fig 12, our method lacks robust error recovery mechanisms, resulting in error propagation from earlier folding actions to subsequent steps. Furthermore, the approach does not include fine-grained operations for manipulating specific cloth layers or performing highly precise actions within the garment. Future work will focus on incorporating an explicit error detection module and developing more effective error recovery strategies. Additionally, integrating advanced sensors to capture precise 3D depth information could enable finer and more accurate operations.

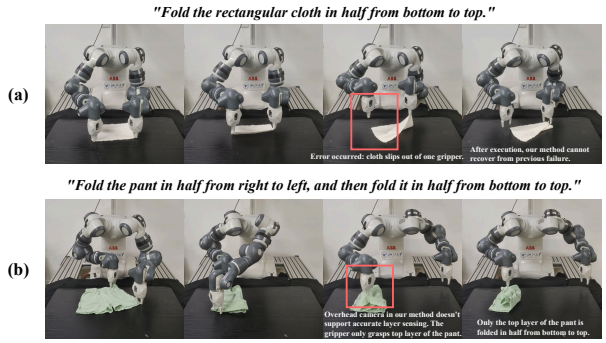


Fig. 12: Demonstration of failure cases and limitations of our proposed method.

REFERENCES

[1] H. Yin, A. Varava, and D. Kragic, "Modeling, learning, perception, and control methods for deformable object manipulation," *Science Robotics*, vol. 6, no. 54, p. eabd8803, 2021.

[2] 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," *IEEE Robotics & Automation Magazine*, vol. 29, no. 3, pp. 67–77, 2022.

[3] J. Sanchez, J.-A. Corrales, B.-C. Bouzgarrou, and Y. Mezouar, "Robotic manipulation and sensing of deformable objects in domestic and industrial applications: a survey," *The International Journal of Robotics Research*, vol. 37, no. 7, pp. 688–716, 2018.

[4] K. Mo, C. Xia, X. Wang, Y. Deng, X. Gao, and B. Liang, "Foldsformer: Learning sequential multi-step cloth manipulation with space-time attention," *IEEE Robotics and Automation Letters*, vol. 8, no. 2, pp. 760–767, 2023.

[5] T. Weng, S. M. Bajracharya, Y. Wang, K. Agrawal, and D. Held, "Fabricflownet: Bimanual cloth manipulation with a flow-based policy," in *Conference on Robot Learning*. PMLR, 2022, pp. 192–202.

[6] D. Seita, A. Ganapathi, R. Hoque, M. Hwang, E. Cen, A. K. Tanwani, A. Balakrishna, B. Thananjeyan, J. Ichnowski, N. Jamali *et al.*, "Deep imitation learning of sequential fabric smoothing from an algorithmic supervisor," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 9651–9658.

[7] R. Hoque, D. Seita, A. Balakrishna, A. Ganapathi, A. Tanwani, N. Jamali, K. Yamane, S. Iba, and K. Goldberg, "VisuoSpatial Foresight for Physical Sequential Fabric Manipulation," *arXiv preprint arXiv:2102.09754*, 2021.

[8] M. Shridhar, L. Manuelli, and D. Fox, "Cliport: What and where pathways for robotic manipulation," in *Proceedings of the 5th Conference on Robot Learning (CoRL)*, 2021.

[9] K. Mo, Y. Deng, C. Xia, and X. Wang, "Learning language-conditioned deformable object manipulation with graph dynamics," *arXiv preprint arXiv:2303.01310*, 2023.

[10] OpenAI, "Gpt-4 technical report," 2023.

[11] J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng, "Code as policies: Language model programs for embodied control," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 9493–9500.

[12] A. Brohan, Y. Chebotar, C. Finn, K. Hausman, A. Herzog, D. Ho, J. Ibarz, A. Irpan, E. Jang, R. Julian *et al.*, "Do as i can, not as i say: Grounding language in robotic affordances," in *Conference on Robot Learning*. PMLR, 2023, pp. 287–318.

[13] W. Huang, C. Wang, R. Zhang, Y. Li, J. Wu, and L. Fei-Fei, "Voxposer: Composable 3d value maps for robotic manipulation with language models," *arXiv preprint arXiv:2307.05973*, 2023.

[14] H. Ha, P. Florence, and S. Song, "Scaling up and distilling down: Language-guided robot skill acquisition," in *Conference on Robot Learning*. PMLR, 2023, pp. 3766–3777.

[15] V. Raval, E. Zhao, H. Zhang, S. Nikolaidis, and D. Seita, "Gpt-fabric: Folding and smoothing fabric by leveraging pre-trained foundation models," *arXiv preprint arXiv:2406.09640*, 2024.

[16] Y. Deng and D. Hsu, "General-purpose clothes manipulation with semantic keypoints," *arXiv preprint arXiv:2408.08160*, 2024.

[17] W. Huang, C. Wang, Y. Li, R. Zhang, and L. Fei-Fei, "Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation," *arXiv preprint arXiv:2409.01652*, 2024.

[18] T. Fu, C. Li, J. Liu, F. Li, C. Wang, and R. Song, "Flingflow: Llm-driven dynamic strategies for efficient cloth flattening," *IEEE Robotics and Automation Letters*, 2024.

[19] Y. Wang, Z. Sun, J. Zhang, Z. Xian, E. Biyik, D. Held, and Z. Erickson, "Rl-vlm-f: Reinforcement learning from vision language foundation model feedback," *arXiv preprint arXiv:2402.03681*, 2024.

[20] X. Gu, T.-Y. Lin, W. Kuo, and Y. Cui, "Open-vocabulary object detection via vision and language knowledge distillation," *arXiv preprint arXiv:2104.13921*, 2021.

[21] M. Minderer, A. Gritsenko, A. Stone, M. Neumann, D. Weissenborn, A. Dosovitskiy, A. Mahendran, A. Arnab, M. Dehghani, Z. Shen *et al.*, "Simple open-vocabulary object detection with vision transformers. arxiv 2022," *arXiv preprint arXiv:2205.06230*.

[22] 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.

[23] S. Miller, J. Van Den Berg, M. Fritz, T. Darrell, K. Goldberg, and P. Abbeel, "A geometric approach to robotic laundry folding," *The International Journal of Robotics Research*, vol. 31, no. 2, pp. 249–267, 2012.

[24] A. Doumanoglou, J. Stria, G. Peleka, I. Mariolis, V. Petrik, A. Kargakos, L. Wagner, V. Hlaváč, T.-K. Kim, and S. Malassiotis, "Folding clothes autonomously: A complete pipeline," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1461–1478, 2016.

[25] C. Bersch, B. Pitzer, and S. Kammel, "Bimanual robotic cloth manipulation for laundry folding," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2011, pp. 1413–1419.

[26] G. Salhotra, I.-C. A. Liu, M. Dominguez-Kuhne, and G. S. Sukhatme, "Learning deformable object manipulation from expert demonstrations," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 8775–8782, 2022.

[27] H. Xue, Y. Li, W. Xu, D. Zheng, and C. Lu, "Unifolding: Towards sample-efficient, scalable, and generalizable robotic garment folding," in *7th Annual Conference on Robot Learning*, 2023.

[28] J. Wu, X. Sun, A. Zeng, S. Song, J. Lee, S. Rusinkiewicz, and T. Funkhouser, "Spatial action maps for mobile manipulation," *arXiv preprint arXiv:2004.09141*, 2020.

[29] H. Ha and S. Song, "Flingbot: The unreasonable effectiveness of dynamic manipulation for cloth unfolding," in *Conference on Robotic Learning (CoRL)*, 2021.

[30] A. Canberk, C. Chi, H. Ha, B. Burchfiel, E. Cousineau, S. Feng, and S. Song, "Cloth funnels: Canonicalized-alignment for multi-purpose garment manipulation," in *International Conference of Robotics and Automation (ICRA)*, 2022.

[31] 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 *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 4568–4575.

[32] K. Xu, S. Zhao, Z. Zhou, Z. Li, H. Pi, Y. Zhu, Y. Wang, and R. Xiong, "A joint modeling of vision-language-action for target-oriented grasping in clutter," *arXiv preprint arXiv:2302.12610*, 2023.

[33] E. Stengel-Eskin, A. Hundt, Z. He, A. Murali, N. Gopalan, M. Gombolay, and G. Hager, "Guiding multi-step rearrangement tasks with natural language instructions," in *Conference on Robot Learning*. PMLR, 2022, pp. 1486–1501.

[34] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman *et al.*, "Do as i can, not as i say: Grounding language in robotic affordances," *arXiv preprint arXiv:2204.01691*, 2022.

[35] W. Huang, F. Xia, D. Shah, D. Driess, A. Zeng, Y. Lu, P. Florence, I. Mordatch, S. Levine, K. Hausman *et al.*, "Grounded decoding: Guiding text generation with grounded models for robot control," *arXiv preprint arXiv:2303.00855*, 2023.

[36] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24 824–24 837, 2022.

[37] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," 2021.

[38] 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," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[39] X. Lin, Y. Wang, J. Olkin, and D. Held, "Softgym: Benchmarking deep reinforcement learning for deformable object manipulation," in *Conference on Robot Learning*. PMLR, 2021, pp. 432–448.

[40] H. Bertiche, M. Madadi, and S. Escalera, "Cloth3d: clothed 3d humans," in *European Conference on Computer Vision*. Springer, 2020, pp. 344–359.

Haozhe Du received his B.Eng. degree in Control Science and Engineering and a minor in Mechatronical Engineering from Zhejiang University in 2022, where he is currently pursuing the M.S. degree. His latest research interests include embodied artificial intelligence and deformable object manipulation.

Kechun Xu received her B.Eng. in Control Science and Engineering from Zhejiang University, Hangzhou, China, in 2021.
She is currently working toward Ph.D. degree at the State Key Laboratory of Industrial Control Technology and Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou, China. Her research interests include manipulation and robot learning.

Rong Xiong received her PhD in Control Science and Engineering from the Department of Control Science and Engineering, Zhejiang University, Hangzhou, P.R. China in 2009. She is currently a Professor in the Department of Control Science and Engineering, Zhejiang University, Hangzhou, P.R. China. Her latest research interests include motion planning and SLAM.

Yue Wang received his Ph.D. degree in Control Science and Engineering from Department of Control Science and Engineering, Zhejiang University, Hangzhou, P.R. China in 2016. He is currently a Professor in the Department of Control Science and Engineering, Zhejiang University, Hangzhou, P.R. China. His latest research interests include mobile robotics and robot perception.