

Task Robustness via Re-Labeling Vision-Action Robot Data

†Artur Kuramshin^{1,2}, Özgür Aslan^{1,2}, Cyrus Neary^{1,2,3}, Glen Berseth^{1,2}

Abstract—The recent trend in scaling models for robot learning has resulted in impressive policies that can perform various manipulation tasks and generalize to novel scenarios. However, these policies continue to struggle with following instructions, likely due to the limited linguistic and action sequence diversity in existing robotics datasets. This paper introduces Task Robustness via RE-Labeling Vision-Action Robot Data (TREAD), a scalable framework that leverages large Vision-Language Models (VLMs) to augment existing robotics datasets without additional data collection, harnessing the transferable knowledge embedded in these models. Our approach leverages a pretrained VLM through three stages: generating semantic sub-tasks from original instruction labels and initial scenes, segmenting demonstration videos conditioned on these sub-tasks, and producing diverse instructions that incorporate object properties, effectively decomposing longer demonstrations into grounded language-action pairs. We further enhance robustness by augmenting the data with linguistically diverse versions of the text goals. Evaluations on LIBERO demonstrate that policies trained on our augmented datasets exhibit improved performance on novel, unseen tasks and goals. Our results show that TREAD enhances both planning generalization through trajectory decomposition and language-conditioned policy generalization through increased linguistic diversity. Project website: <https://akuramshin.github.io/tread>.

I. INTRODUCTION

Recent trends in robot learning research have demonstrated the importance of data scale and diversity for learning more robust and capable robotic manipulation policies [4, 18, 5]. While the robotics community has made progress in expanding dataset sizes through collaborative efforts, these datasets continue to exhibit limited diversity along certain text and trajectory modalities [26, 16, 30]. This limitation manifests as a weakness of the current models to reliably follow text-based instructions [28, 17, 22].

The dataset modality imbalance persists due to the challenges of collecting diverse, real-world expert demonstrations at scale, often requiring multiple robots and months of operation to obtain even modest datasets [7, 16, 11]. However, these vision-based manipulation demonstrations have grown in length and complexity, covering many potential sub-goals per trajectory. While it is possible to manually review the thousands of demonstration videos and solicit human annotators, this will not scale with the demands of modern robotic learning systems. This raises the question: How can we systematically augment existing robotics datasets with greater language-action diversity in a scalable way?

Recent advances in large-scale vision-language models (VLMs) [1, 2, 10] offer a promising new approach to address the language-action diversity challenge in robotics datasets. Using VLMs for robot data augmentation is appealing for two reasons. By leveraging internet-scale pretraining, VLMs are capable of zero-shot generation of contextually appropriate language labels that are also grounded in the physical scenes depicted in demonstration videos. Second, these models can effectively reason about temporal sequences in videos [1, 2, 36, 10], enabling segmentation of demonstrations into meaningful sub-tasks. This capability allows for decomposing existing demonstrations into more granular language-action segments, effectively increasing the diversity of our training data without requiring additional data collection.

Motivated by the strong capabilities of large-scale pretrained VLMs, we propose Task Robustness via RE-Labeling Vision-Action Robot Data (TREAD), a framework to augment robotics datasets. TREAD shown in Figure 1 operates in an iterative three-stage process, where the model’s outputs from previous steps are fed back as inputs to the next: First, a pretrained VLM analyzes the original instruction labels and the initial scenes to generate a sequence of semantic sub-task descriptions that collectively accomplish the original goal. Then the demonstration videos are processed by the the VLM to temporally segment the trajectories, identifying which portions of the demonstration correspond to each sub-task description. Lastly, the VLM generates diverse instructions for these sub-tasks incorporating object properties and spatial relationships conditioned on the initial scenes of the sub-tasks and the semantic sub-task descriptions. This temporal-semantic alignment creates multiple language-conditioned sub-demonstrations from each original demonstration from the offline dataset.

In this work, we make two key contributions: (1) a novel framework to utilize iterative querying of a VLM to augment and diversify the original dataset, and (2) the augmented dataset obtained by applying our framework, which will be released to facilitate future research. In our experiments with the LIBERO dataset [20] we show that the resulting augmented dataset improves zero-shot performance and text goal following of the vision-language-action policy Octo [27] and π_0 -FAST [28].

II. RELATED WORK

A. Robot learning datasets

Recently, the robotics community has invested substantial effort into collecting larger manipulation datasets. While

¹Mila — Quebec AI Institute, ²Université de Montréal, ³The University of British Columbia, [†]For correspondence and questions: artur.kuramshin@mila.quebec

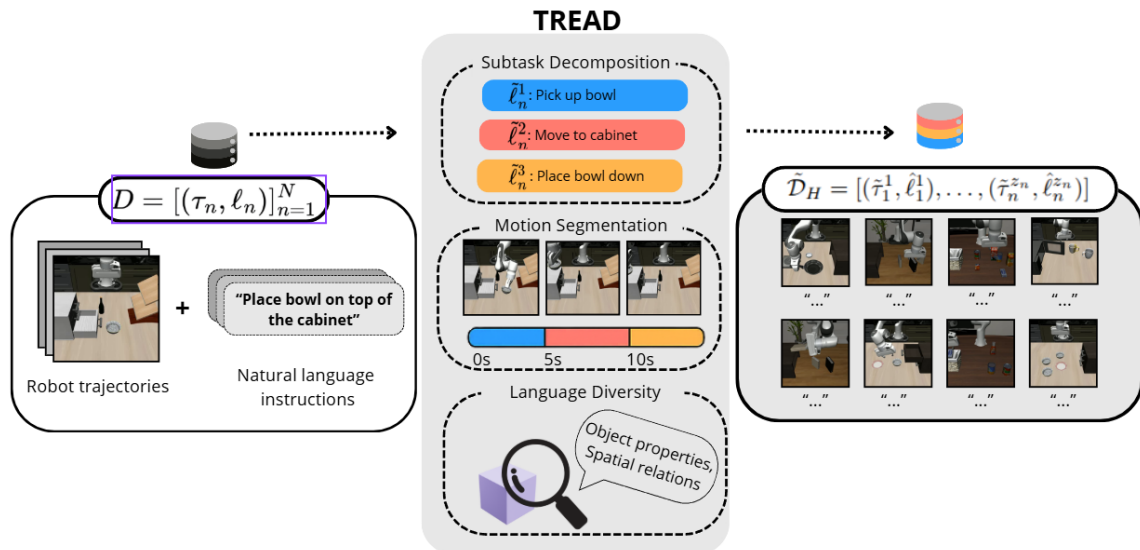


Fig. 1. TREAD uses a large-scale VLM to programmatically cut trajectories at sub-goals, label those sub-goals and add variations to the goal-text.

earlier datasets such as RT-1 [7], MT-Opt [15], and BC-Z [13] boast significant numbers of trajectories, they lack diversity in tasks and scenes, which is more important for generalization [16, 19, 32]. Responding to this need, several works have focused on increasing dataset diversity by scaling data collection, including BridgeV2 [30], DROID [16], and RDT-1B [22]. These newer datasets not only collect data across more scenes but also demonstrate more complicated tasks consisting of multiple meaningful sub-goals. However, this advancement creates a new challenge: although the number of tasks has increased, the language diversity is diluted by the larger amount of image-action frames per instruction due to longer trajectories. Our work presents a generic framework for expanding the language-image-action distribution of any robotic manipulation dataset without requiring additional data collection, effectively reducing this dilution problem.

B. Vision-language-action models as scalable robot policies

Vision-Language Models (VLMs) have shown increasing capabilities in complex visual tasks such as state identification, reasoning, and visual question answering [21, 3, 1, 10]. Inspired by the success of VLMs, the robotics community adopted similar multi-modal architectures for imitation learning [7, 27, 14, 22]. More recent methods start from a pre-trained VLM backbone to leverage internet-scale pretraining [18, 5]. Followup works have explored architectural modifications [17, 22, 28] to improve performance in instruction generalization and following. In contrast, our work improves existing policy performance through data augmentation alone. Therefore, our method can be used alongside any policy architecture.

C. Data augmentation methods.

Data augmentation is a powerful technique for enhancing model robustness and generalization by generating additional synthetic training examples from existing data. In

robot learning, augmentation methods have developed along three main dimensions: visual augmentation [4, 33, 25, 9], language augmentation [35, 24, 31], and trajectory augmentation [35, 8, 29]. Our work bridges language and trajectory augmentation by using VLMs to both relabel trajectories and identify meaningful sub-task motions within longer demonstrations. DIAL [31], which is most similar to our approach in terms of language augmentation, relies on a predetermined set of labels and requires fine-tuning on related robotics data. In contrast, our method leverages zero-shot VLM capabilities. NILS [6] is another framework that incorporates multiple large models and heuristics to segment and relabel trajectories, whereas our method (TREAD) relies on a single iteratively queried VLM without heuristics, making it simpler to reproduce. Lastly, SPRINT [35], another closely related work, focuses on composing shorter skills into longer sequences. However, our approach does the opposite by decomposing longer-horizon demonstrations into meaningful sub-tasks. This makes our method particularly suitable for enhancing language-action diversity in open-source robotics datasets.

III. METHOD

Here we describe how we design TREAD to segment and stitch robotics data, in three stages: (1) *segmentation* using a VLM to process trajectories for semantic and motion-based properties to find key sub-goal transitions, (2) *labeling* of the new sub-trajectories with sub-goals and to augment the original dataset with language diversity, and (3) *train* a vision-language-action model on the resulting dataset to improve robustness. The pseudocode for TREAD is shown in Algorithm 1.

A. Problem Formulation

We assume access to an offline dataset \mathcal{D} of robot trajectories labelled with natural language task instructions. Formally, we define a dataset $\mathcal{D} = [(\tau_n, \ell_n)]_{n=1}^N$ consisting of N

labeled trajectories, where each trajectory $\tau_n = [(\mathbf{o}_t^n, \mathbf{a}_t^n)]_{t=1}^T$ contains observation-action pairs over T timesteps. Here, \mathbf{o}_t^n and \mathbf{a}_t^n denote the image observation and action at time t respectively, and ℓ_n denotes the natural language instruction describing the task demonstrated in trajectory τ_n . We then train a policy $\pi(\cdot | \mathbf{o}_t, \ell)$ via imitation learning to generate action distributions that mimic demonstrated behavior. Specifically, we look at models trained by minimizing the behavior cloning objective:

$$\hat{\pi}^* = \arg \min_{\pi} \sum_{(\tau, \ell) \in \mathcal{D}} \sum_{(\mathbf{o}_t, \mathbf{a}_t) \in \tau} \mathcal{L}_{BC}(\pi(\cdot | \mathbf{o}_t, \ell), \mathbf{a}_t)$$

where \mathcal{L}_{BC} denotes a supervised loss (e.g., negative log-likelihood) between the predicted action distribution and the demonstrated action.

We are interested in using data augmentation to improve the learned policy’s generalization performance. Specifically, we address the question: how can we augment \mathcal{D} with additional trajectory-instruction pairs (τ, ℓ) derived from existing demonstrations, without requiring any new data collection?

B. Decomposition of Trajectories

Although our framework does not depend on a specific VLM, in this work we use Gemini Pro 2.5 [10]. Gemini is a multi-modal model that can take as input images, videos, and text, and output text. Given a trajectory τ_n completing a long-horizon task ℓ_n , we first derive semantic sub-tasks and then identify which parts of the trajectory complete those sub-tasks. We found that just asking the VLM to segment the video in one-shot provided poor results; providing the model with subtasks resulted in qualitatively better video segmentation. We accomplish this by chaining two VLM inference calls shown in Figure 2. First, we prompt the VLM to create a plan of sub-task motion labels $[\tilde{\ell}_n^1, \tilde{\ell}_n^2, \dots, \tilde{\ell}_n^{z_n}]$, given the higher-level task ℓ_n , and the first frame of the trajectory \mathbf{o}_1^n for grounding and context. Here z_n denotes the number of sub-tasks that are required to complete task ℓ_n in the scene as determined by the VLM. For example, the task “place bowl in the top drawer” will require a different number of sub-task motions depending on whether the top drawer is currently closed or open and the algorithm should identify this and produce three sub-tasks instead of two.

Subsequently, we query the model with the video of the trajectory and prompt it to retrieve the starting and ending seconds at which the previously identified sub-tasks occur. As a result, we now have a process by which we can decompose a labeled trajectory (τ, ℓ) into an ordered set of labeled sub-trajectories $[(\tilde{\tau}^1, \tilde{\ell}^1), (\tilde{\tau}^2, \tilde{\ell}^2), \dots, (\tilde{\tau}^z, \tilde{\ell}^z)]$, where $\tilde{\tau} \subset \tau$ denotes a sub-trajectory. Finally, we can now process \mathcal{D} to construct a new *decomposed* dataset:

$$\tilde{\mathcal{D}}_A = [(\tilde{\tau}_1^1, \tilde{\ell}_1^1), \dots, (\tilde{\tau}_1^{z_1}, \tilde{\ell}_1^{z_1}), \dots, (\tilde{\tau}_n^1, \tilde{\ell}_n^1), \dots, (\tilde{\tau}_n^{z_n}, \tilde{\ell}_n^{z_n})]$$

For details on the full prompts, see Appendix I. Practically, we decompose a dataset of 82 unique scene-instruction tasks into 146 subtasks (see Section IV-A for details on the dataset).

C. Instruction Diversity and Data Augmentation

Recent research has highlighted instruction following as a weakness of current VLA models [17, 28, 22], attributing the difficulty to the models paying more attention to visual inputs than the language goal. While the mentioned works address the problem algorithmically, we argue that additional textual diversity will help robustness to language. Many robotics datasets have considerably small linguistic diversity which potentially leads to policies being overly sensitive to instruction wording. We use the generated sub-trajectories to augment \mathcal{D} . We consider the following textual augmentation method for language diversity:

Grounded textual diversity. We can enhance linguistic diversity by leveraging VLMs to generate visually grounded paraphrases of instructions. For each sub-trajectory $(\tilde{\tau}_i, \tilde{\ell}_i)$, we prompt the VLM with both the original instruction $\tilde{\ell}_i$ and the first visual frame from $\tilde{\tau}_i$, requesting k language alternatives that preserve the task semantics. As shown in Figure 2, including the image in the context allows the model to incorporate spatial relationships and perceivable characteristics for more accurate goal diversity beyond just using synonyms. For instance, given an instruction “pick up the blue coffee mug,” our VLM might generate alternatives like “grasp the small coffee mug” (object attributes) or “retrieve the coffee mug next to the laptop,” (spatial relationships). We apply this language augmentation to our already decomposed dataset $\tilde{\mathcal{D}}_A$ and denote the resulting dataset $\tilde{\mathcal{D}}_H$:

$$\tilde{\mathcal{D}}_H = [(\tilde{\tau}_1^1, \tilde{\ell}_1^1), \dots, (\tilde{\tau}_1^{z_1}, \tilde{\ell}_1^{z_1}), \dots, (\tilde{\tau}_n^1, \tilde{\ell}_n^1), \dots, (\tilde{\tau}_n^{z_n}, \tilde{\ell}_n^{z_n})]$$

See Appendix I for the templated prompt we use for grounded textual diversity.

Algorithm 1 The TREAD algorithm

Input: Robotic dataset \mathcal{D} and VLM \mathcal{G}

```

1: function TREAD( $\mathcal{D}, \mathcal{G}$ )
2:    $\tilde{\mathcal{D}}_A \leftarrow \text{DECOMPOSE}(\mathcal{D}, \mathcal{G})$ 
3:    $\tilde{\mathcal{D}}_H \leftarrow \text{DIVERSIFY}(\tilde{\mathcal{D}}_A, \mathcal{G})$ 
4:   Train policy  $\pi_A$  on mixture of  $\mathcal{D}$  and  $\tilde{\mathcal{D}}_A$ 
5:   Train policy  $\pi_H$  on mixture of  $\mathcal{D}$  and  $\tilde{\mathcal{D}}_H$ 
6: function DECOMPOSE( $\mathcal{D}, \mathcal{G}$ ) ▷ Section III-B
7:    $\tilde{\mathcal{D}}_A \leftarrow \{\}$ 
8:   for trajectory-instruction pair  $(\tau_n, \ell_n)$  in  $\mathcal{D}$  do
9:      $\mathbf{o}_1 \leftarrow$  first observation in  $\tau_n$ 
10:     $\mathbf{v} \leftarrow [\mathbf{o}_1, \dots, \mathbf{o}_T]$  (video of trajectory  $\tau_n$ )
11:     $[\tilde{\ell}_n^1, \dots, \tilde{\ell}_n^{z_n}] \leftarrow \mathcal{G}(\mathbf{o}_1, \ell_n)$ 
12:     $[(\tilde{\tau}_n^1, \tilde{\ell}_n^1), \dots, (\tilde{\tau}_n^{z_n}, \tilde{\ell}_n^{z_n})] \leftarrow \mathcal{G}(\mathbf{v}, [\tilde{\ell}_n^1, \dots, \tilde{\ell}_n^{z_n}])$ 
13:     $\tilde{\mathcal{D}}_A \leftarrow \tilde{\mathcal{D}}_A \cup [(\tilde{\tau}_n^1, \tilde{\ell}_n^1), \dots, (\tilde{\tau}_n^{z_n}, \tilde{\ell}_n^{z_n})]$ 
14:   return  $\tilde{\mathcal{D}}_A$ 
15: function DIVERSIFY( $\tilde{\mathcal{D}}_A, \mathcal{G}$ ) ▷ Section III-C
16:    $\tilde{\mathcal{D}}_H \leftarrow \{\}$ 
17:   for sub-trajectory-instruction pair  $(\tilde{\tau}_n^i, \tilde{\ell}_n^i)$  in  $\tilde{\mathcal{D}}_A$  do
18:      $\mathbf{o}_1 \leftarrow$  first observation in  $\tilde{\tau}_n^i$ 
19:      $\tilde{\ell}_n^i \leftarrow$  choose one from  $\mathcal{G}(\mathbf{o}_1, \tilde{\ell}_n^i)$ 
20:      $\tilde{\mathcal{D}}_H \leftarrow \tilde{\mathcal{D}}_H \cup (\tilde{\tau}_n^i, \tilde{\ell}_n^i)$ 
21:   return  $\tilde{\mathcal{D}}_H$ 

```

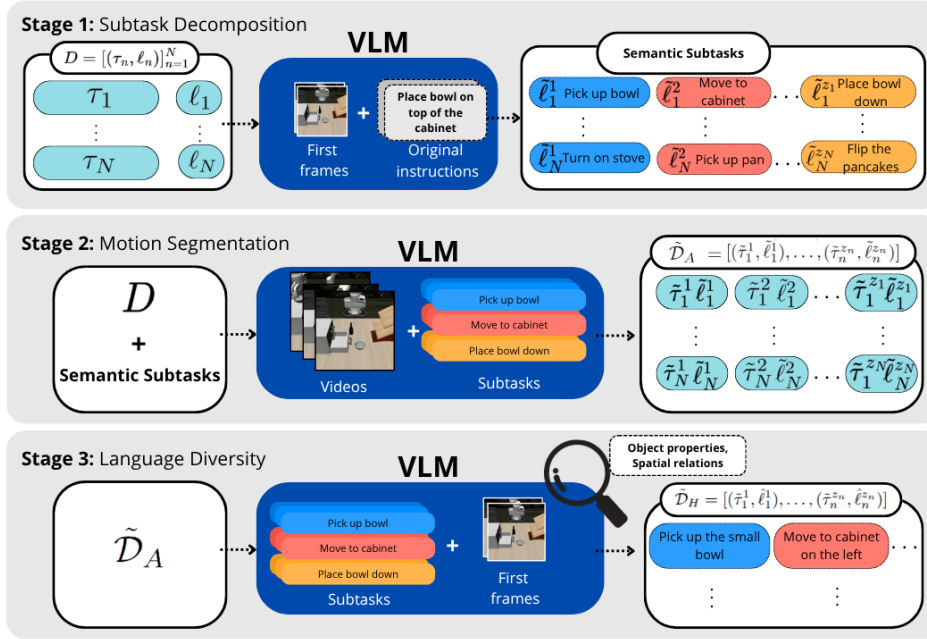


Fig. 2. **TREAD pipeline overview.** Given dataset \mathcal{D} of N labeled trajectories $[(\tau_n, \ell_n)]_{n=1}^N$, TREAD decomposes the dataset into semantically meaningful sub-trajectories through three stages: (1) **Subtask Decomposition:** Given the original task instruction ℓ and initial frame \mathbf{o}_1 , we prompt the VLM to generate a sequence of sub-task labels $[\tilde{\ell}^1, \tilde{\ell}^2, \dots, \tilde{\ell}^{z_n}]$. (2) **Motion Segmentation:** The VLM identifies temporal boundaries for each sub-task, resulting in a set of labeled sub-trajectories $[(\tilde{\tau}^1, \tilde{\ell}^1), \dots, (\tilde{\tau}^{z_n}, \tilde{\ell}^{z_n})]$. (3) **Language Diversity:** The VLM generates diverse paraphrases for each sub-task instruction by leveraging visual context to incorporate object properties (color, shape, material) and spatial relationships (relative positions, orientations).

D. Training a Vision-Language-Action Model

We study the quality of our dataset augmentation by training vision-language-action manipulation policies on the augmented data. Using the datasets generated by TREAD, we finetune two generalist robot policies: Octo (specifically Octo-Small 1.5) [27] and π_0 -FAST [28]. Octo is a diffusion-based policy trained on the Open-X dataset [26], while π_0 -FAST leverages a pre-trained VLM backbone, PaliGemma [3], trained on the cross-embodied robot data mixture from π_0 [5], with actions discretized using the FAST tokenization scheme.

To determine the dataset weighting ratio such that the finetuned model achieves the best performance across the original full trajectories (\mathcal{D}) and relabeled sub-trajectories ($\tilde{\mathcal{D}}_A/\tilde{\mathcal{D}}_H$), we conducted ablation studies across multiple mixing ratios. We evaluated ratios $(\mathcal{D}:\tilde{\mathcal{D}}_A/\tilde{\mathcal{D}}_H)$ of $1:2$, $1:1.5$ and $1:1.1$. The final ratio was selected using Re-Mix [12]. For all experiments reported in Section IV-A, we use the best-performing weighting ratio for each respective data mixture. Detailed performance results for each weighting ratio across the evaluation tasks are provided in Appendix II.

For all data mixtures, we employ consistent finetuning parameters across models. We finetune Octo for 50,000 steps, batch size of 256, using a linear warmup followed by cosine decay [23] with a peak learning rate of 3×10^{-4} . For π_0 -FAST, we perform full finetuning for 30,000 steps with a batch size of 32, employing the same cosine decay schedule with warmup but with a peak learning rate of 2.5×10^{-5} . Based on test performance, we use checkpoints at step 30,000 for Octo and step 15,000 for π_0 -FAST across all dataset

ablations (see Section IV).

IV. RESULTS

In our evaluation, we design experiments to investigate the effectiveness of training an agent on a dataset augmented by TREAD toward generalizing to unseen tasks. Specifically, our experiments answer the following research questions: (1) Does TREAD’s *trajectory* decomposition increase planning generalization? (2) Does *language* augmentation help policy generalization to text-based goals?

We focus on robot manipulation and evaluate our method on the image-based dataset LIBERO [20], a simulation benchmark designed for studying lifelong learning in robotic manipulation. The use of LIBERO to evaluate VLA models pretrained on real-world data is a common practice to show the capabilities of these models [28, 18, 17]. Due to resource constraints, we evaluate our method on a subset of LIBERO as described below.

A. Experimental Setup

In our experiments, we evaluate the effects of both trajectory decomposition and instruction diversity on policy performance through ablation studies using the following data mixtures:

- 1) **Finetuned Octo/ π_0 -FAST with no augmentations (Original Fine-tuned):** We finetune the Octo/ π_0 -FAST on the unaltered subset of LIBERO-100 trajectories \mathcal{D} . This serves as our primary comparison point, expected to perform adequately on in-distribution task-instruction pairs but struggle with language-following in novel scenes or instruction text variations.

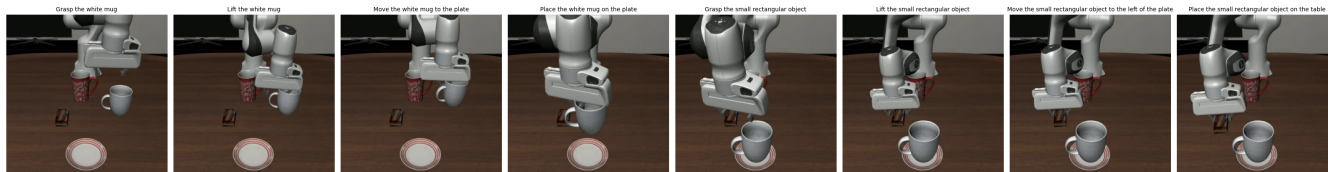


Fig. 3. Example visualization of the sub-task keypoints produced by TREAD for the trajectory “Put the white mug on the plate and put the chocolate pudding to the left of the plate”. Each frame (from left to right) corresponds to the first frame of the sub-task (labeled above). The sub-task labels in order are: (1) “Grasp the white mug” (2) “Lift the white mug” (3) “Move the white mug to the plate” (4) “Place the white mug on the plate” (5) “Grasp the small rectangular object” (6) “Lift the small rectangular object” (7) “Move the small rectangular object to the left of the plate” (8) “Place the small rectangular object on the table.”

2) Finetuned Octo/ π_0 -FAST with Trajectory Decomposition (TREAD w/o diverse labels):

We finetune Octo/ π_0 -FAST on a mixture of the unaltered dataset \mathcal{D} and its decomposition $\hat{\mathcal{D}}_A$. This ablation isolates the effect of trajectory decomposition, which should help the policy perform tasks in new environments by leveraging subtask compositions seen during training, even when the complete task-environment pairing is novel.

3) Finetuned Octo/ π_0 -FAST with Trajectory Decomposition + Label Diversity (TREAD):

We finetune Octo/ π_0 -FAST on a mixture of the unaltered dataset \mathcal{D} and its linguistically enriched decomposition $\hat{\mathcal{D}}_H$. This full implementation of our method is expected to handle both novel environments and textual variations by bringing diverse text instructions in distribution.

LIBERO. The LIBERO benchmark [20] consists of four task suites that are designed to examine distribution shifts in the object types, the spatial arrangement of objects, the task goals, or the mixture of the previous three. For our experiments, we focus on the LIBERO-100 task suite, which contains 100 tasks involving diverse object interactions and versatile skills with 50 human-teleoperated demonstrations each. In LIBERO-100, tasks are defined by the combination of scene and instruction, meaning that identical instructions performed in different scenes are treated as distinct tasks. Importantly, we utilize the full LIBERO-100 suite rather than the smaller subsets employed by recent works [18, 28]. This larger benchmark presents a more challenging evaluation setting that reduces the risk of overfitting and provides a more rigorous test of our method’s ability to improve motion and language generalization (see Section IV-B).

Due to resource constraints, we labeled five demonstrations per task and omitted any tasks performed in STUDY_SCENES for a total of 570 trajectories. An example of a labeled trajectory is shown in Figure 3. To evaluate our research questions, we defined two sets of novel tasks within LIBERO:

B. Evaluation Tasks

Motion Generalization (MG): For research question (1), we created new instruction-scene pairs by taking existing

language instructions and pairing them with *compatible* but previously unpaired scenes. For example, the instruction “open the top drawer of the cabinet” is paired with scene KITCHEN_SCENE4 containing a cabinet, with this specific combination not appearing in the training data. These tasks will evaluate whether trajectory decomposition helps the agent transfer learned skills to novel environmental contexts. Importantly, these new instruction-scene pairs also evaluate the model’s language-following capability, as the model must avoid defaulting to previously trained tasks from those scenes and instead execute the specified instruction. We create 7 such novel scene-instruction environments within the LIBERO framework.

Language Generalization (LG): For research question (2), we created linguistically modified versions of existing LIBERO-100 tasks. These modifications included variations such as reordering clauses around conjunctions (changing “turn on the stove and put the moka pot on it” to “put the moka pot on the stove and turn it on”) or removing unnecessary object specifiers when context makes them redundant (changing “close the bottom drawer of the cabinet” to “close the drawer”). These variations assess whether our language augmentation strategies improve policy robustness to variations in instruction wording. We found 14 existing tasks within LIBERO-100 that satisfied our criteria for instruction modification.

We evaluate all dataset ablations across both test cases. For each task, we conduct 30 policy rollouts and report the average success rate. To provide a more nuanced understanding of policy capabilities, we separately analyze performance on single-goal tasks (e.g. “close the drawer”) and two-goal tasks (e.g. “put the bowl in the drawer of the cabinet and close it”). For two-goal tasks, we report both partial success (completing only one goal) and complete success (achieving both goals).

C. Increased Zero-Shot Performance

The Motion Generalization results in Table I demonstrate TREAD’s effectiveness when policies encounter familiar instructions in novel environments. Both TREAD and TREAD w/o diverse labels outperform Original Fine-tuned across Single Goal tasks, with TREAD achieving 15% and 6%

TABLE I

LIBERO TASK PERFORMANCE RESULTS. SUCCESS RATES (SR) WITH STANDARD ERROR ACROSS THE CUSTOM EVALUATION TASK SUITES (SEE SECTION IV-B) AND LIBERO-10 WITHIN THE LIBERO FRAMEWORK [20]. RESULTS INCLUDE POLICIES FINE-TUNED ON DIFFERENT DATA MIXTURES: ORIGINAL TRAJECTORY DATA (ORIGINAL FINE-TUNED), MIXTURE OF THE UNALTERED DATASET AND ITS DECOMPOSITION (TREAD W/O DIV.), AND MIXTURE OF THE UNALTERED DATASET AND ITS LINGUISTICALLY ENRICHED DECOMPOSITION (TREAD). BOLD AND UNDERLINED VALUES INDICATE BEST AND SECOND-BEST PERFORMANCE.

Test Case	Metric (%)	π_0 -FAST [28]			Octo [27]		
		Original Fine-tuned	TREAD w/o div.	TREAD	Original Fine-tuned	TREAD w/o div.	TREAD
Language Generalization	Single Goal SR	47±20	77±7	<u>67±14</u>	82±2	76±17	91±6
	1 of 2 SR	<u>63±11</u>	62±10	67±9	<u>76±6</u>	70±6	77±4
	2 of 2 SR	<u>36±8</u>	35±7	39±10	<u>30±7</u>	28±5	31±5
Motion Generalization	Single Goal SR	28±18	31±20	34±16	7±4	27±10	<u>22±15</u>
	1 of 2 SR	73±3	82±11	82±9	13±4	50±10	<u>43±3</u>
	2 of 2 SR	7±3	0±0	0±0	0±0	0±0	0±0
LIBERO-10	1 of 2 SR	83±9	72±12	<u>74±8</u>	76±5	69±6	<u>72±4</u>
	2 of 2 SR	57±10	43±12	57±10	<u>40±7</u>	41±4	<u>38±4</u>
Average	SR	49±10	50±11	53±10	41±12	45±10	47±11

higher success rates on Octo and π_0 -FAST respectively. This shows that the automatic segmentation of TREAD is able to provide more trajectory and goal diversity which drives additional generalization across more tasks.

The benefits of trajectory decomposition become more pronounced for complex, multi-step tasks. In two-goal scenarios, TREAD achieves substantially higher first-goal completion rates - 30% improvement for Octo and 9% for π_0 -FAST - compared to baseline fine-tuning. The wider performance gap for multi-goal tasks likely stems from the increased opportunities for sub-trajectory overlap between training and test scenarios. Notably, the performance gains are more substantial for Octo, which lacks the pretrained vision-language understanding of π_0 -FAST, suggesting that trajectory decomposition provides particularly valuable inductive biases for models without built-in multimodal reasoning capabilities.

D. Better Text Goal Following

The Language Generalization results in Table I reveal TREAD’s ability to handle linguistic variations of familiar task-instructions in known environments. In single-goal tasks, TREAD demonstrates substantial improvements over Original Fine-tuned baselines, achieving 20% higher success rates with Octo and 9% with π_0 -FAST. This improvement provides strong evidence that our grounded textual diversity approach effectively prepares policies for the natural linguistic variations.

Interestingly, the performance benefits for Language Generalization are most apparent in shorter, single-goal tasks rather than complex multi-step scenarios. This pattern likely reflects the mismatch between the relatively concise language instructions in our decomposed sub-trajectories and the longer, more complex instructions typical of multi-goal tasks. Additionally, longer-horizon tasks provide more opportunities for compounding errors when policies encounter out-of-distribution language conditioning, making it inherently

more challenging to maintain robust language following throughout extended task execution.

Importantly, we verify that our augmentation approach does not compromise performance on in-distribution tasks. When evaluated on the in-distribution long-horizon tasks in LIBERO-10, TREAD maintains comparable performance to Original Fine-tuned for two-goal success rates as seen in the last row in Table I.

V. CONCLUSION

In summary, we presented TREAD, a framework that uses off-the-shelf VLMs to perform a type of trajectory segmentation and then stitching via training a large-scale BC model Octo [27]. We find that fine-tuning Octo on our new dataset improves the performance, especially on more difficult longer horizon tasks and novel tasks not seen during training. Despite these promising results, our work has limitations that point to interesting directions for future research. Our work uses a large closed-source VLM, which limits reproducibility. An important next step would be to evaluate TREAD with open-source alternatives [21, 2, 34] and potentially fine-tune these models specifically on robotics data to improve their embodied understanding. Future work can also explore applying TREAD to larger datasets and larger models. However, our work shows that even with small amounts of properly segmented and augmented data there are obvious performance gains for current VLA models.

APPENDIX I VLM PROMPTS

We provide the templated prompts we use for querying Gemini 2.5 [10] for semantic sub-task breakdown and motion identification in Figure 4, Figure 5, and the template prompt for grounded paraphrasing in Figure 6. The three example tasks used for few-shot prompting motion identification are taken from the LIBERO-90 dataset.

```

<FRAME 1>
You are a robot arm with a simple gripper. You are given the task <TASK INSTRUCTION>. Given the task and the image showing the layout of the scene, create a plan of the possible sub-task motions you will need to perform to complete the task.

Follow these guidelines:

1. Start off by pointing to no more than 8 items in the image . The answer should follow the json format: [{"point": <point>, "label": <label1>, ...}]. The points are in [y, x] format normalized to 0-1000. One element a line.
2. Think about where the object and its relational objects are located in the scene on a global scale. Think step by step and list the locations and relations of all objects, noting any object that might interfere with the task.
3. Finally, output the sub-tasks that will need to be completed to <TASK INSTRUCTION>. in this specific scene. Output in the json format: [{"sub_task": <sub-task>, "description": <description>}, ...]

Example sub-tasks: "Grasp the bowl to the right of the white mug", "Move the mug towards the red plate", "Rotate the knob", "Pull the top drawer open", "Push the bottom drawer closed", "Place the can on top of the cabinet", "Lift the middle bowl"

```

Fig. 4. The templated prompt we use for generating semantic sub-tasks. The placeholders <FRAME 1> and <TASK INSTRUCTION> are replaced with the first image of the trajectory and the trajectory instruction, respectively.

```

<VIDEO>
The video is showing a robot arm performing the task "Put the white mug on the left plate." The task can be broken down into these possible sub-task motions:

1. Grasp the white mug: Approach the white mug. Position the gripper around the body or handle of the mug and close the gripper to grasp it securely.
2. Lift the white mug: Move the gripper vertically upwards, lifting the mug clear of the table surface and the other mugs.
3. Move the white mug to the right plate: Move the gripper horizontally, carrying the mug across the table until it is positioned directly above the plate on the right.
4. Place the white mug on the right plate: Move the gripper vertically downwards to gently place the white mug onto the surface of the right plate. Open the gripper to release the mug.

Carefully look at the video and describe the sub-task motions you see being performed by the robot and the corresponding timecodes.

Okay, here's the breakdown of the sub-task motions for placing the white mug on the left plate:

1. 00:00 - 00:03 Grasp the white mug: The robot moves its gripper downwards and positions it around the white mug then closes the gripper to grasp it.
2. 00:03 - 00:04 Lift the white mug The robot lifts the grasped mug vertically upwards, clear of the table.
3. 00:04 - 00:06 Move the white mug to the right plate: The robot moves the mug horizontally towards the right, positioning it above the right plate.
4. 00:06 - 00:08 Place the white mug on the right plate: The robot lowers the mug down onto the surface of the right plate and then opens its gripper to release the mug

<VIDEO>
You are provided with a video showing a robot arm performing the task <TASK INSTRUCTION>. The task can be broken down into these possible sub-task motions:

<SUB-TASK 1>
<SUB-TASK 2>
...
<SUB-TASK Z>

You need to carefully look at the video and describe the sub-task motions you see being performed by the robot and the corresponding timecodes. Follow these guidelines:

1. Start off by pointing to no more than 8 items in the image . The answer should follow the json format: [{"point": <point>, "label": <label1>, ...}]. The points are in [y, x] format normalized to 0-1000. One element a line.
2. Determine the object movement and the resulting object relations. Explain the object movements.
3. Combine motions for approaching and grasping objects. Also combine motions for placing and releasing objects.
4. Finally, output the sub-tasks that you observe and the corresponding timecodes in the json format: [{"sub_task": <sub-task>, "time_range": <timecodes>}, ...].

```

Fig. 5. The templated prompt we use for recognizing semantic sub-tasks in a trajectory video. We use few-shot prompting by providing the model with 3 example user-model exchanges before the final query prompt (we include only one example for brevity). The placeholders <VIDEO> and <TASK INSTRUCTION> are replaced with the trajectory video and the trajectory instruction, respectively.

```

<FRAME 1>
You are a robot arm with a simple gripper. You were given the task <TASK INSTRUCTION>. Here is a break down of the sub-task motions you need to perform to complete this task.

<SUB-TASK 1>
<SUB-TASK 2>
...
<SUB-TASK Z>

Help yourself understand the goal by writing 5 alternate descriptions. For the overall task, and each of the sub-task motions write 5 other ways to instruct the same motion.

Follow these guidelines:

1. Start off by pointing to no more than 8 items in the image . The answer should follow the json format: [{"point": <point>, "label": <label1>, ...}]. The points are in [y, x] format normalized to 0-1000. One element a line.
2. Then describe each object both in terms of their properties and their spatial relation to each other and in the global scene (e.g. "the smaller object", "the black object", "the middle object").
3. For the final step rewrite the original task and subtasks using your object descriptions in the json format: [{"task": [<paraphrase-task>], "subtasks": [{"paraphrase-sub-task"}, <paraphrase-sub-task>], ...}].

```

Fig. 6. The templated prompt we use for grounded textual diversity (see Section III-C). The placeholders <FRAME 1> and <TASK INSTRUCTION> are replaced with the first image of the trajectory and the trajectory instruction, respectively.

APPENDIX II
DATASET WEIGHTING RATIOS

To determine the optimal balance between original demonstrations and augmented sub-trajectories, we conducted ablations across different dataset mixing ratios. Table II presents results for Octo, showing success rate performance across three task sets as described in Section IV-B.

TABLE II

SUCCESS RATE PERFORMANCE OF OCTO [27] FINETUNED WITH DIFFERENT DATASET WEIGHTING RATIOS FOR (1) TREAD W/O DIVERSE LABELS ($\mathcal{D}:\tilde{\mathcal{D}}_A$) AND (2) TREAD WITH DIVERSE LABELS ($\mathcal{D}:\tilde{\mathcal{D}}_H$).

Test Case	Metric (%)	1:2		1:1.5		Re-Mix	
		TREAD w/o div.	TREAD	TREAD w/o div.	TREAD	TREAD w/o div.	TREAD
Language Generalization	Single Goal SR	73	89	51	69	76	73
	1 of 2 SR	74	79	72	81	72	81
	2 of 2 SR	30	30	32	33	30	36
Motion Generalization	Single Goal SR	22	20	27	23	25	17
	1 of 2 SR	42	40	38	40	49	31
	2 of 2 SR	0	0	0	0	0	0
LIBERO-10	1 of 2 SR	69	77	68	71	74	70
	2 of 2 SR	37	41	37	37	41	37

ACKNOWLEDGMENTS

We acknowledge the use of Claude Sonnet 4 (Anthropic) for assistance with editing and refining the written presentation of this work. All scientific content, methodology, and conclusions remain the original contributions of the authors.

We want to acknowledge funding support from Natural Sciences and Engineering Research Council of Canada, Fonds de recherche du Québec and The Canadian Institute for Advanced Research (CIFAR) and compute support from Digital Research Alliance of Canada, Mila IDT, and NVidia. We would also like to thank John Edwards Leadership Fund (CFI) for funding the purchase of hardware for this project.

REFERENCES

[1] Josh Achiam et al. “Gpt-4 technical report”. In: *arXiv preprint arXiv:2303.08774* (2023).

[2] Shuai Bai et al. *Qwen2.5-VL Technical Report*. 2025. arXiv: 2502.13923 [cs.CV].

[3] Lucas Beyer et al. “Paligemma: A versatile 3b vlm for transfer”. In: *arXiv preprint arXiv:2407.07726* (2024).

[4] Homanga Bharadhwaj et al. “Roboagent: Generalization and efficiency in robot manipulation via semantic augmentations and action chunking”. In: *IEEE International Conference on Robotics and Automation*. 2024, pp. 4788–4795.

[5] Kevin Black et al. π_0 : A Vision-Language-Action Flow Model for General Robot Control. 2025. arXiv: 2410.24164 [cs.LG].

[6] Nils Blank et al. “Scaling Robot Policy Learning via Zero-Shot Labeling with Foundation Models”. In: *Proceedings of Conference on Robot Learning*. 2024.

[7] Anthony Brohan et al. “Rt-1: Robotics transformer for real-world control at scale”. In: *Proceedings of Robotics: Science and Systems*. 2022.

[8] Jingjing Chen et al. *Towards Effective Utilization of Mixed-Quality Demonstrations in Robotic Manipulation via Segment-Level Selection and Optimization*. 2025. arXiv: 2409.19917 [cs.RO].

[9] Qiuyu Chen et al. “GenAug: Retargeting behaviors to unseen situations via Generative Augmentation”. In: *Proceedings of Robotics: Science and Systems*. 2023. DOI: 10.15607/RSS.2023.XIX.010.

[10] Gheorghe Comanici et al. *Gemini 2.5: Pushing the Frontier with Advanced Reasoning, Multimodality, Long Context, and Next Generation Agentic Capabilities*. 2025. arXiv: 2507.06261 [cs.CL].

[11] Hao-Shu Fang et al. “Rh20t: A comprehensive robotic dataset for learning diverse skills in one-shot”. In: *IEEE International Conference on Robotics and Automation*. 2024, pp. 653–660.

[12] Joey Hejna et al. “ReMix: Optimizing Data Mixtures for Large Scale Imitation Learning”. In: *Proceedings of Conference on Robot Learning*. Vol. 270. Proceedings of Machine Learning Research. PMLR, 2025, pp. 145–164.

[13] Eric Jang et al. “Bc-z: Zero-shot task generalization with robotic imitation learning”. In: *Conference on Robot Learning*. PMLR. 2022, pp. 991–1002.

[14] Yunfan Jiang et al. “VIMA: Robot Manipulation with Multimodal Prompts”. In: *Proceedings of Conference on Machine Learning*. Vol. 202. Proceedings of Machine Learning Research. PMLR, 2023, pp. 14975–15022.

[15] Dmitry Kalashnikov et al. “Scaling Up Multi-Task Robotic Reinforcement Learning”. In: *Proceedings of Conference on Robot Learning*. Vol. 164. Proceedings of Machine Learning Research. PMLR, 2022, pp. 557–575.

[16] Alexander Khazatsky et al. “DROID: A Large-Scale In-The-Wild Robot Manipulation Dataset”. In: *Robotics: Science and Systems*. 2024.

[17] Moo Jin Kim, Chelsea Finn, and Percy Liang. “Fine-Tuning Vision-Language-Action Models: Optimizing Speed and Success”. In: *arXiv preprint arXiv:2502.19645* (2025).

[18] Moo Jin Kim et al. “Openvla: An open-source vision-language-action model”. In: *arXiv preprint arXiv:2406.09246* (2024).

[19] Fanqi Lin et al. “Data scaling laws in imitation learning for robotic manipulation”. In: *arXiv preprint arXiv:2410.18647* (2024).

[20] Bo Liu et al. “Liber0: Benchmarking knowledge transfer for lifelong robot learning”. In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 44776–44791.

[21] Haotian Liu et al. “Visual Instruction Tuning”. In: *NeurIPS*. 2023.

[22] Songming Liu et al. “Rdt-1b: a diffusion foundation model for bi-manual manipulation”. In: *arXiv preprint arXiv:2410.07864* (2024).

[23] Ilya Loshchilov and Frank Hutter. “SGDR: Stochastic Gradient Descent with Warm Restarts”. In: *5th International Conference on Learning Representations*. 2017.

[24] Corey Lynch et al. “Interactive Language: Talking to Robots in Real Time”. In: *IEEE Robotics and Automation Letters* (2023), pp. 1–8. DOI: 10.1109/LRA.2023.3295255.

[25] Zhao Mandi et al. “Cacti: A framework for scalable multi-task multi-scene visual imitation learning”. In: *arXiv preprint arXiv:2212.05711* (2022).

[26] Abby O’Neill et al. “Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0”. In: *IEEE International Conference on Robotics and Automation*. 2024, pp. 6892–6903.

[27] Octo Model Team et al. “Octo: An Open-Source Generalist Robot Policy”. In: *Proceedings of Robotics: Science and Systems*. 2024.

[28] Karl Pertsch et al. *FAST: Efficient Action Tokenization for Vision-Language-Action Models*. 2025. arXiv: 2501.09747 [cs.RO].

[29] Divyanshu Raj et al. “Learning Temporally Composable Task Segmentations with Language”. In: *IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2024, pp. 5195–5202.

[30] Homer Rich Walke et al. “Bridgedata v2: A dataset for robot learning at scale”. In: *Proceedings of Conference on Robot Learning*. Vol. 229. PMLR. 2023, pp. 1723–1736.

[31] Ted Xiao et al. “Robotic Skill Acquisition via Instruction Augmentation with Vision-Language Models”. In: *Proceedings of Robotics: Science and Systems*. 2023.

[32] Annie Xie et al. “Decomposing the generalization gap in imitation learning for visual robotic manipulation”. In: *IEEE International Conference on Robotics and Automation*. 2024, pp. 3153–3160.

[33] Tianhe Yu et al. “Scaling Robot Learning with Semantically Imagined Experience”. In: *Proceedings of Robotics: Science and Systems*. 2023. DOI: 10.15607/RSS.2023.XIX.027.

[34] Boqiang Zhang et al. *VideoLLaMA 3: Frontier Multimodal Foundation Models for Image and Video Understanding*. 2025. arXiv: 2501.13106 [cs.CV].

[35] Jesse Zhang et al. “Sprint: Scalable policy pre-training via language instruction relabeling”. In: *IEEE International Conference on Robotics and Automation*. 2024, pp. 9168–9175.

[36] Yuanhan Zhang et al. *Video Instruction Tuning With Synthetic Data*. 2024. arXiv: 2410.02713 [cs.CV].