Scaling Up and Distilling Down: Language-Guided Robot Skill Acquisition

Anonymous Author(s)
Affiliation
Address
email

Abstract: We present a framework for exploring and learning visual-and-language-conditioned robot skills. In doing so, we address two core challenges in data-driven robot learning: 1) scaling up the generation of language-labeled visuomotor data, and 2) effectively distilling this data down into a robust policy. For (1), we use a large language model (LLM) to guide high-level planning, and sampling-based robot planners (*e.g.* motion or grasp samplers) for generating diverse action trajectories. To robustify this data-collection process, the LLM also infers a code-snippet for the success condition of each task, simultaneously enabling the data-collection process to detect failure and retry as well as the automatic labeling of trajectories with success/failure. For (2), we extend the diffusion-based policy to multi-task settings with language conditioning. Finally, we propose a new multi-task benchmark with 18 tasks across five domains to test long-horizon behavior, common-sense reasoning, tool-use, and intuitive physics. We find that our distilled policy successfully learned the robust retrying behavior in its data collection procedure, while improving absolute success rates by 28% on average across five domains. All code and data will be publicly available. The task and policy performance is best viewed on this anonymous website: scalingup-distillingdown.github.io.

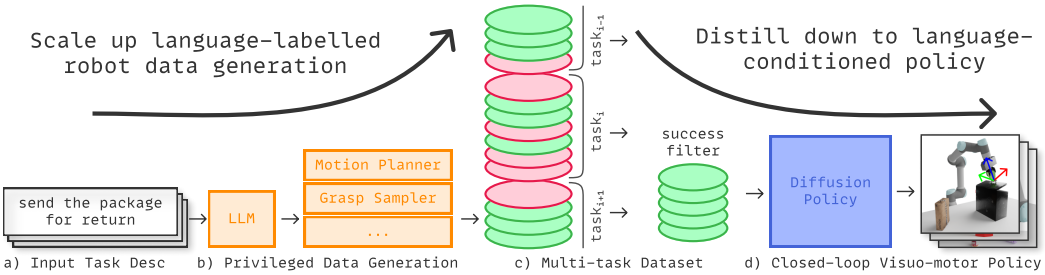

Figure 1: **Language-guided Skill Acquisition** enables scalable robot learning. In the data generation stage, a LLM takes as input task descriptions and uses sampling-based robotic planners and privileged simulation information to perform task-directed exploration. This enables the scaling up of language and task-success labeled dataset generation. In the second stage, the dataset is filtered for success and distilled down into a closed-loop language-conditioned visuomotor policy that uses only proprioception and RGB inputs.

1 Introduction

How can we scalably acquire robust and reusable manipulation skills? This question has been the driving force behind extensive research in robot learning. Attempts in the field have focused on two primary aspects: First, how to **scale up the data collection** for a diverse range of manipulation skills, which involves efforts such as improving the hardware [1, 2] and software [3] which support demonstration collection, utilization of non-robotics datasets [4], or trial-and-error explorations [5]. The second aspect of this question concerns **effective learning** from the collected data, which delves into exploring effective action representations [6] and policy formulations [7, 8] that can robustly model the training data and generalize to novel scenarios.

This paper proposes a new framework that provides a comprehensive solution for both aspects by leveraging language guidance. We contribute two key components with our framework:

Submitted to the 7th Conference on Robot Learning (CoRL 2023). Do not distribute.

- **Language-Guided Data Generation (Scaling Up):** Our data-collection policy is a large language model (LLM) which has access to a suite of 6DoF exploration primitives (*i.e.*, sampling-based robot planners and utilities). Given an input task description, this policy first **simplifies** the task by recursively decomposing it into subtasks, resulting in a hierarchical plan (*i.e.*, task tree). Next, this plan is **grounded** into a sequence of 6DoF exploration primitives, which generates diverse robot trajectories for the task. Finally, the data collection policy **verifies** the trajectories' success with an inferred success function and **retries** the task until it succeeds. This verify & retry step not only improves the data-collection policy's success, but also adds robot experience on how to recover from failure, an important trait for downstream policy distillation. This data generation approach is scalable, enabling significantly more efficient autonomous task-directed exploration than unguided alternatives (*i.e.*, reinforcement learning) while not being limited by the lack of low-level understanding of the LLM-only solution.

- **Language-Conditioned Visuomotor Policy (Distilling Down):** We distill these robot experiences into a visuo-linguo-motor policy that infers control sequences from visual observations and a natural language task description. To enable effective learning of high entropy, diverse robot trajectories, we extend the diffusion policy formulation [8] to handle language-based conditioning for multi-task learning. This allows the learned policy to be reused and recomposed through language-based planners. On most domains, we found that our distilled policy outperforms its data-collection policy by a significant margin, while inheriting robust retrying behavior from its training data.

Our framework combines these two components to get the best of both worlds – leverage LLM's common-sense reasoning abilities for efficient exploration while learning rich, robust, and reusable 6DoF skills. In summary, the key contribution of this paper is a new framework for **visuo-linguo-motor policy learning** that is enabled by three novel components:

- A new language-guided data collection framework that combines language-based task planner with 6DoF robot utilities (*e.g.* motion planning, grasp sampling).

- New formulation of diffusion-based policy that effectively learns multi-task language-conditioned policies.

- In addition to our algorithmic contributions, we also contribute a new multi-task benchmark that includes 18 tasks across five domains highlights tasks that require long-horizon (≈ 400 control cycles), common sense, tool-use, and intuitive physics – capabilities lacking in existing manipulation benchmarks.

2 Related Works

Scaling visuo-linguo-motor data. An increasing area of interest in robot learning has been in vision-and-language-conditioned motor ("visuo-linguo-motor") policies [9, 10, 11, 12, 13, 14, 15, 16], and one of the most important underlying questions is how to scale data collection. Perhaps the most prevalent paradigm is to use humans to annotate both actions and language. Using humans to provide action labels, i.e. teleoperation, has been prodigiously used for real-world skill learning [9, 10, 11, 12, 13, 14, 15, 16]. Within this category, humans can either provide episodic task-specific expert demonstrations [11, 12, 13, 16], or non-task-specific ("play") data [9, 10, 17, 14]. A primary limitation, however, is that data scalability is human-limited.

Other prior works have proposed strategies to enable more-autonomously-scalable data. To scale language annotation, prior works study using visual-language models [18], or procedurally post-hoc provided in simulation [17]. To scale the action labels, methods study how to use *autonomous sub-optimal policies* that are still viable for effective downstream learning, from random policies [5] to more costly options like learned policies using behavior cloning or from reinforcement learning (RL) [19]. As another option for scalability not depending on robots, human ego-vision data [20, 21, 22] has been shown to be relevant to robot learning [23, 4], but lacks "action" (motor) labels, and requires cross-embodiment transfer. While these approaches reduce human efforts, they are still limited in optimality, generality, and have not been demonstrated to provide all of visuo-linguo-motor data generation all in one scalable framework without human annotation involvement.

Another option for the autonomous data collection policy is to use a model-based policy, e.g. from task and motion planning (TAMP) [24]. Our method is in this category, but we improve it with an LLM - this greatly aids in the flexibility and task generality by leveraging LLM's common-sense

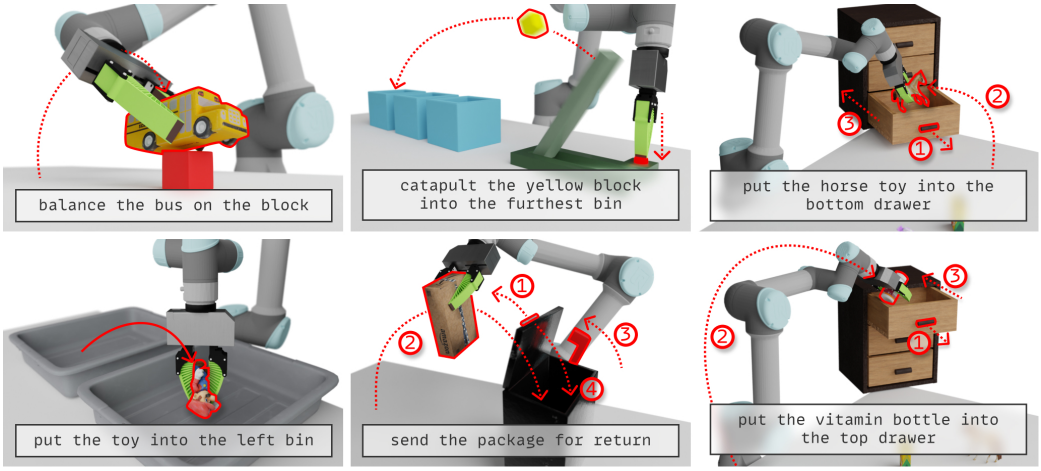

Figure 2: **Benchmark.** We validate our approach on a new multi-task benchmark addressing challenging long-horizon tasks (*i.e.*, 400 control cycles) requiring language understanding (e.g., put [object] to [top] drawer), common sense knowledge (e.g., send a package for return requires raising the mailbox flag), tool-use (e.g., catapult), and intuitive physics (e.g., balance the bus). The task is best viewed on this anonymous website.

knowledge. In contrast to recent works exploring the use of LLMs as robotic task planners (*i.e.*, as the *final* policy) [25, 26, 27, 28, 29, 30], we use the LLM-based planner as a suboptimal *data-collection* policy. We then distill this data from a privileged-information policy into an observable-information [31, 32, 33] policy in a way that allows us to learn a better policy than the LLM-based planner.

Policy Representations and Multi-task Policy Distillation. One primary question in visuo-motor learning [34] has been how to represent the policy for effective learning, i.e. to enable high precision, multi-modal robot behavior [7, 35, 36, 8, 2]. Another related question has been how to best train multi-task policies [37, 38], including those conditioned on language [9, 11, 12, 13, 14, 16]. Our work presents the novel formulation of bringing diffusion-based [39, 40] policies [8] into the language-conditioned [41, 42] visuomotor domain. Additionally, prior works in multi-task language-conditioning typically focus on cloning policies from experts, meanwhile we study distilling data from a success-filtered suboptimal policy. Success-filtering [7, 43] can be viewed as the simplest form of offline RL [44].

3 Approach

We propose a new framework for robot learning that performs automatic data collection and policy learning from only a task description. Our design is grounded on four key observations:

- We recognize the importance of random exploration in reinforcement learning, but aim to not be constrained by its inefficiency for long-horizon, sparse reward tasks.
- We acknowledge the usefulness of LLM's common-sense and zero-shot capabilities, but believe language *is not by itself* the ideal representation for robust, rich, and precise robotic manipulation.
- We are inspired by the effectiveness of robotic planning methods, e.g. TAMP, but wish to be flexible to novel tasks and domains and non-reliant on ground truth state during policy inference.
- We aim to achieve the simplicity and effectiveness of behavior cloning in distilling collected robot experience into a policy for real-world deployment, while side-stepping the requirement for costly human demonstrations or play data collection.

Using no human demonstration or manually specified reward, our framework combines the strengths of these four areas, resulting in a unified framework for efficient task-directed exploration and multi-task visuo-linguo-motor policy learning from only task descriptions.

Method Overview. In the data generation phase, we use an LLM to recursively decompose (§3.1.1) tasks into a plan for exploration and ground the plan into sampling-based robot utilities and motion primitives (§3.1.2). Next, the LLM infers success-detection functions for each task (§ 3.1.3), providing success-labeling. This autonomous data generation process outputs a replay buffer of task-directed

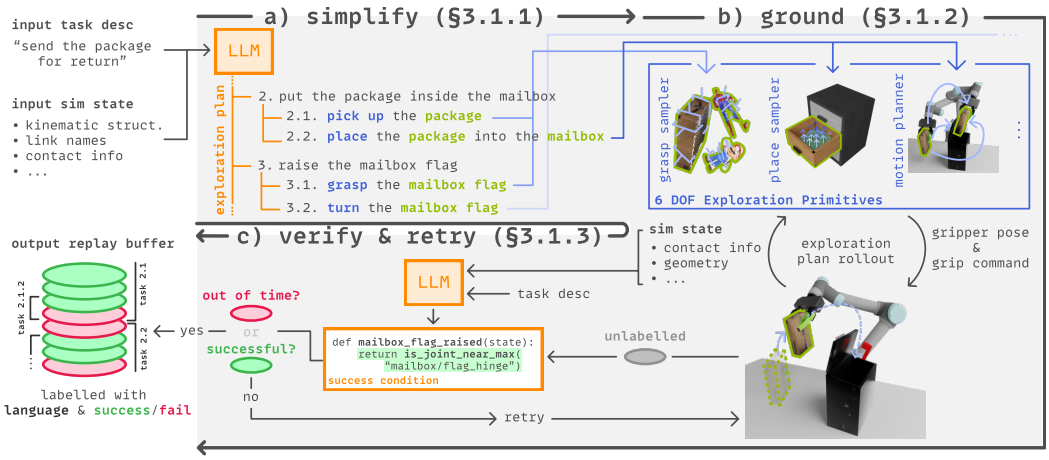

Figure 3: **Language-Driven Robot Data Generation** takes as input the task description and simulation state, and outputs a replay buffer, labelled with language descriptions and success. It starts by using an LLM to simplify tasks recursively (a) until the task involves only one object, resulting in a hierarchical exploration plan. Next, the plan is grounded (b) into a sequence of 6 DOF exploration primitives (*e.g.* grasp samplers, motion planners, etc.) and rolled out in simulation to give an unlabelled robot trajectory. Finally, an LLM infers a success function code-snippet, and uses it to verify (c) and label it with succeeded or failed. If the trajectory failed, the LLM retries the exploration plan with a different random seed (*e.g.* a different grasp pose from the grasp sampler). If the robot succeeds or run out of time, the labeled trajectory is returned.

exploration experience, labeled with language descriptions and success labels. In the training phase (§3.2), we distill this data into a multi-task vision-and-language-conditioned diffusion policy [8] on successful trajectories from the replay buffer.

3.1 Language-Guided Robot Data Generation

Our data generation approach follows a common recipe of using a data-collection policy that uses privileged information [31, 32, 33] to generate robot experience, then distilling it into a visuo-linguo-motor policy for execution. The privileged data collection policy is typically either a reinforcement learning agent [31, 33] or a manually designed heuristic policy [32, 45, 12], and the distillation into observable inputs can enable Sim2Real transfer [33]. In our approach, we use a large language model (LLM) and 6 DoF robot sampling-based utilities in tandem, which can parse through and use the full simulation state's kinematic structure, link names, and object geometry.

3.1.1 Simplify: Task Planning and Decomposition

Given a task description, the first step is to generate a high-level task plan. To improve the flexibility to work with any tasks and 3D assets, we opted for an LLM-based planner to leverage their common-sense reasoning skills. Unlike classical TAMP planners, our framework does not require domain-specific engineering and transition function design to work with new tasks.

Concretely, our recursive LLM planner takes as input the task description, the simulation state, and outputs a plan in the form of a task tree (Fig. 3a)*. To do so, the LLM first checks whether the task description involves the robot interacting with multiple or only one object. For instance, "move the package into the mailbox" involves opening the mailbox before picking up the package and putting the mailbox in, and should be considered a multi-object task, while "with the mailbox opened, move the package into the mailbox" should be a single-object task. For the base case of single-object tasks, we prompt the LLM to which object part name to to interact. For the case of multi-object tasks, we prompt the LLM to decompose the task into subtasks, and recurse down each subtask.

3.1.2 Ground: Compiling a Plan into Robot Utilities

With the generated task tree §3.1.1, the next step is to ground the high-level plan into physical actions. Here the choice of robot *low-level skill representations* critically defines the robots' overall capability

*We provide all prompts in the supplementary material

and therefore becomes a key differentiating factor among different methods. In principle, there are three desired properties we want to see in the action space design:

- It should be flexible and precise. Importantly, it should not be limited to 2D planar actions [28].
- It should be scalable and not rely on expensive human demonstration to acquire [11, 26].
- Finally, It should be language-friendly: while a list of joints' values is flexible to encode any action, it is not a suitable interface for a language model.

To encompass these properties, we propose to ground the LLM's plan with API calls into a set of robot utility function, which include a sampling-based motion planner, a geometry-based grasp and placement sampler, and motion primitives for articulated manipulation. We refer to these utilities as 6 DOF Exploration Primitives (Fig 3b) because, by virtue of being *pseudo-random*, the sampling-based utilities generate *diverse* robot trajectories, enabling effective exploration for rich 6 DoF manipulation settings. For instance, the rapidly-exploring random trees [46] motion planner samples uniformly in joint space, while the grasp and placement samplers samples uniformly amongst all points in the object part's point cloud to find good grasps and placements poses, respectively.

For each leaf node in the inferred task tree (§ 3.1.1), the grounding process takes as input the node's task description (*e.g.* "open the mailbox"), its associated object part name (*e.g.* "mailbox lid"), and the simulation state, and outputs a sequence of robot utility calls. Using the object part name, we can parse the object's kinematic structure from the simulation state and handle rigid and articulated objects separately. For rigid objects, the LLM is prompted to choose the pick & place object names, used to sample grasp and placement pose candidates. For articulated objects (with either revolute or prismatic joints), the leaf node's associated object part name is used to sample a grasp candidate followed by a rotation or translation primitive conditioned on its joint parameters (i.e., joint type, axis, and origin). The full prompt and implementation details can be found in the supp. material.

Exploration Plan Rollout. Each node in the exploration plan is grounded only when it is being executed, where the order of execution follows a pre-order tree traversal. By keeping track of the subtask's state, we could provide additional language descriptions for a sub-segment of robot trajectory, thereby providing **dense text labels** for the trajectory. For instance, all actions taken during the inferred subtask "open the mailbox" can be labeled with both the subtask's description "open the mailbox" and the root task description "move the package into the mailbox".

Since grounding happens only when a task node is visited, each node's grounding process is independent of the other leaf nodes, depending only on the simulation state when it is evaluated. While this simplifies planning significantly, it also means that failed execution can occur. For instance, a grasp candidate may render all placement candidates infeasible.

3.1.3 Verify & Retry: Robustifying the Data Collection Policy

As discussed in the earlier section, the planning and grounding step can fail, especially when we consider exploration and long-horizon tasks. To address this, we propose a verify & retry (Fig. 3c) scheme, which uses environment feedback to detect failed execution.

Verify. For each task, the LLM infers **a success function code snippet** (Fig. 3c, highlighted green) given the task description, simulation state, and API functions to for query simulation state (e.g., checking contact or joint values, etc). This amounts to prompting the LLM to complete a task success function definition that outputs a boolean value, indicating task success.

Retry. When a trajectory is labeled failed, the robot can retry the same sequence of robot utilities with a different random seed (*i.e.*, for the sampling-based robotic utilities) without resetting the simulation state until the task succeeds. For instance, in the bus balance case, the robot could try a different grasp candidate and place candidate. In the tree traversal process § 3.1.2, nodes only yield execution to its parent task when the node's inferred success condition returns true. This design not only leads to higher success rates in data generation but also provides useful demonstrations on **how to recover from failure**. In the output replay buffer, the only failed trajectories are ones which timed-out or led to invalid states (*e.g.* object dropped on the floor).

3.2 Language-conditioned Policy Distillation

Our data generation is capable of generating a large and diverse set of robot data. However, this data was generated from sampling-based planners, which are highly **multi-modal** and is a challenge to fit with a vanilla policy formulation.

We chose to build our approach on top of diffusion policy [8], the state-of-the-art approach for behavior cloning from multi-modal demonstrations. We extend their formulation to the multi-task domain. This policy takes as input a task description, robot proprioception history, and visual observations, and outputs a sequence of end effector control commands. The task description feature is extracted using CLIP [47]'s text encoder. We use a wrist-mounted view and a workspace view for visual observations, and learn separate ResNet18-based [48] vision encoders for each view. We found that using only the latest visual observation along with the full observation horizon of proprioception maintains the policy's high performance while reducing training time. When used in conjunction with the DDIM [49] noise scheduler, we found that we could use a $5\times$ shorter diffusion process at inference (10 timesteps at inference, 50 timesteps at training) while retaining a comparable performance. Quantitatively, when using a 10 dimensional action space[†], our policy can be run at $\approx 45Hz$ on an NVIDIA A6000.

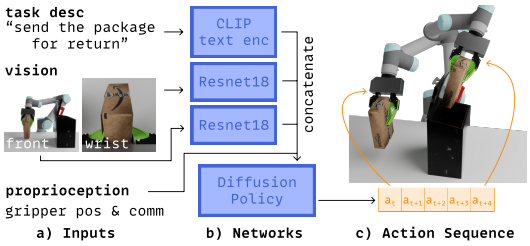

Figure 4: **Language-Conditioned Policy Distillation**. The policy takes as input a task description, two RGB camera views, and gripper proprioception data, and outputs a sequence of gripper poses and closing command.

4 Evaluation

Our experiments try to validate two questions: 1) Can our data generation approach efficiently perform task-directed exploration? 2) Can our policy learning approach effectively distill a multi-modal, multi-task dataset into a generalizable and robust visuo-linguo-motor policy?

Domain	Complex geometry	Articulation	Common sense	Tool use	Multi-task	Long horizon
Balance	✗	✗	✗	✗	✗	✗
Catapult	✗	✓	✓	✓	✓	✗
Transport	✓	✗	✗	✗	✗	✗
Mailbox	✗	✓	✓	✗	✗	✓
Drawer	✓	✓	✗	✗	✓	✓

Table 1: **Benchmark Suite.**

Benchmark contains 18 tasks across 5 domains (Fig. 2 Tab. 1) highlights the following properties:

- **6DoF & articulated manipulation.** Due to object's complex geometry and articulation, the robot needs to utilize 6DoF manipulation skills.
- **Geometry Generalization.** In our bin transport domain, the robot must generalize its bin transport skill to unseen object instances, with novel shapes, sizes, and colors.
- **Intuitive physics.** Robots should understand the physical properties of the world and use them to their advantage. Typically, this means to achieve a specific state, the robot needs to perform an informed precised manipulation. In the bus balance domain, the robot needs to figure out the precise grasping and placement pose to balance a large bus toy on a small block. In the catapult domain, where the robot places the block along a catapult arm determines how far the block will be launched, and, thus, which bin (if any) the block will land in.
- **Common-sense reasoning & Tool-use.** Natural language task description is user-friendly but often under-specifies the task. Common-sense can help to fill in the gaps. In the mailbox domain, given the task "send the package for return", the robot should understand that it not only needs put the package inside, but also raise the mailbox flag to indicate that the package is ready for pickup. In the catapult domain, the robot needs to understand that pressing the catapult's button will activate the catapult, and that the block needs to be placed on the catapult arm to be launched.
- **Multi-task conditioning.** Given the same visual observations but different task description, the robot should perform different and task-relevant actions. The catapult domain has 3 tasks for three target bins, and the drawer domain has 12 tasks.

[†]3 for position, 6 for rotation using the upper rows of the rotation matrix, and a gripper close command

- **Long horizon behaviour.** Our longest horizon domain, mailbox, takes at least 4 subtasks to complete (open the mailbox, put the package in the mailbox while its opened, close the mailbox, then raise the mailbox flag) which can require up to 400 control cycles. In the drawer domain, the robot needs to open the drawer, move the object into the drawer, then close it, which takes about 150 control cycles.

The benchmark is built on top of the Mujoco [3] simulator, using assets from the Google Scanned dataset [50, 51]. We use a table-top manipulation set-up with a 6DoF robot arm. The task success in evaluation is a manually designed function, instead of LLM generated function used for data collection.

Metrics. We report the success rate averaged over 50 seeds per task. If a domain has multiple tasks then we report the average performance of all tasks. We report the quantitative results in Table 2, an task completion efficiency plot in Fig. 5, and qualitative results in Fig. 6.

Baselines. Code-as-Policy [28] is a state-of-the-art approach for using an LLM directly as a robot policy by making state (*e.g.* query present objects) and action primitive API calls to a robot. Given an LLM-inferred code string, they execute the snippet in an open-loop fashion. Crucially, in their table top manipulation setting, they assume access to planar action primitives. Thus, we introduce the following baselines, which build on top of Code-as-Policy and each other as follows:

- **LLM-as-Policy (2D)**: Similar to code-as-policy using planar pick-and-place, but with ground truth segmentation.
- **(+) 6 DOF robot utils**: Builds on top of the previous baseline by adding access to 6 DOF robot utilities for grasping, placement, motion planning, and articulated manipulation.
- **(+) Verify & Retry**: Adding to the previous baselines, this baseline uses the LLM's predicted success condition to label trajectories and retry failed ones. Since the robot utilities involve pseudo-random samplers (*e.g.* RRT, grasp sampling), retrying the task means running these samplers again using the pseudo-random state and environment state from where failed trajectory left it. As the most general and robust baseline, we use this approach as our data generation approach.

4.1 Results on Data Collection Policies

This section provides an evaluation and analysis of our data collection policy and its main design decisions.

Approach	Planar			6DoF	
	Balance	Catapult	Transport	Mailbox	Drawer
LLM-as-Policy (2D)	36%	33%	18%	0%	0%
(+) 6DoF Robot Utils	4%	2%	42%	0%	10%
(+) Verify & Retry	24%	2%	**72%**	6%	**39%**
(+) Distill (Ours)	**72%**	**80%**	50%	**48%**	33%

Table 2: **Quantitative Results.**

6DOF exploration is critical. First, we verify different approach's ability to perform and explore in 6DoF, which is crucial for general manipulation. When 6DoF exploration is introduced, we first observe a drop in the average success rate for simple tasks that could be accomplished with planar actions. However, this ability is critical for successfully exploring complex tasks and provides data to improve upon in the later distilling stage. In particular, we observed that 6DoF actions are important for grasping diverse objects with complex geometry (Transport, Tab. 2), and manipulating articulated objects (Drawer, Mailbox, Tab. 2).

Moreover, 6DOF exploration also helps in **diversifying** the data collection strategy, which provides the **possibility to improve upon** in the later distilling stage. For example in the catapult, LLM-as-Policy (2D) is only able to solve one of three possible goals (shooting in the middle) using a deterministic strategy, getting 33% success rate. However, it provides no useful data for learning the other two goals, making it a poor data-collection policy. In contrast, incorporating 6 DOF robot utilities achieves lower but non-zero average success rates in all bins (5.3%, 0.8%, and 0.7%, full table in supp.), which provide much better exploration data for downstream policy distillation.

Verify & Retry always helps. Through the verify & retry step, the LLM can retry a task after a failed attempt until its inferred success condition passes. We observe that this simple addition results in performance improvements across all domains, achieving $1.7\times$, $6\times$, and $6.5\times$ success rates for the transport, balance, and drawer domain respectively. In fact, without this crucial step, we observe 0.0% success rate in the mailbox domain, underscoring the challenging long-horizon 6 DOF manipulation requirements of the domain, the difficulty of flawlessly executing a long sequence of 6 DOF actions, and the importance of recovery after a failure.

4.2 Distilled Policy's Emergent Robustness

We observe that the distilled policy can robustly recover from failures (e.g., failed grasps or placements) and continuously retries a task until it succeeds. For instance, in the balance domain, the policy continuously readjusts its grasp and placement of the bus after each failed balancing attempt until it succeeds, which results in higher success rates as the policy is given more time (Fig. 5). Further, our distilled policy achieved a $+48\%$ higher success rate than its data collection policy, indicating that it has not only learned robust behaviors from the data collection policy but also improved its performance. This emergent robustness is the result of distilling high entropy, diverse robot trajectories that are verified and filtered with the LLM's inferred success condition into a visuo-motor policy capable of learning and generating high entropy actions (Fig. 6). In contrast, without the LLM's inferred success condition, the baseline data-collection policy's success rate plateaus at 4%, despite having access to the same 6 DOF robot utilities. This highlights the importance of the verify & retry step in our approach for both collecting and filtering high-quality robot data.

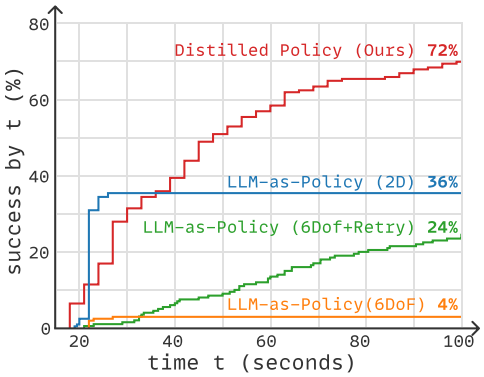

Figure 5: **Distilled Robustness**. Our distilled policy achieves higher success, while inheriting robust recovery from failure and retrying behavior.

Diffusion Policy Capacity. We observe that the distilled policy can achieve 48% (+$>42\%$ higher than other approaches) and 33% success rates on the mailbox and drawer domain, which involves hundreds of control cycles and a dozen of different tasks. This demonstrates our approach's ability to effectively distill a large amount of multi-modal robot data into a high-performance visuo-linguo-motor policy.

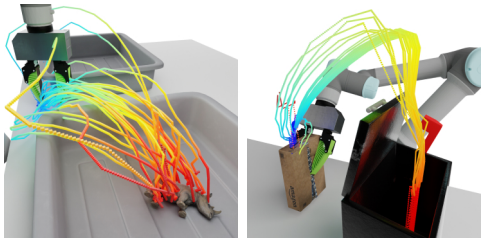

Figure 6: **High Entropy yet Precise Policies.** Running the diffusion process through multiple seeds given the same policy inputs yields a high entropy, multi-modal actions when possible (left, grasps of *unseen* object in bin transport) and precise actions when necessary (right, narrow mailbox with large package).

4.3 Limitations

Our data generation method relies on existing 3D assets and environments, which can be time-consuming to create. This framework can be further scaled up with assets from 3D generative models or procedural generation. Further, while our approach's dataset contains text labels and success labels for all subtasks, we have only evaluated its effectiveness in learning the root task. Learning from all subtasks and growing a robot's set of learned, reusable sub-skills over time to enable compositional generalization is left for future work.

5 Conclusion

We proposed "Scaling Up and Distilling Down", a framework that combines the strengths of LLMs, sampling-based planners, and policy learning into a single system that automatically generates, labels, and distills diverse robot exploration experience into a multi-task visuo-linguo-motor policy. The distilled policy inherits long-horizon behaviour, rich low-level manipulation skills, and robustness from its data collection policy while improving upon performance beyond its training distribution. We believe that this integrated approach is a step towards putting robotics on the same scaling trend as that of LLM development while not compromising on the rich low-level control.

References

[1] S. Song, A. Zeng, J. Lee, and T. Funkhouser. Grasping in the wild: Learning 6dof closed-loop grasping from low-cost demonstrations. *IEEE Robotics and Automation Letters*, 5(3):4978–4985, 2020.

[2] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

[3] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012. doi:10.1109/IROS.2012.6386109.

[4] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.

[5] J. Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

[6] 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.

[7] P. Florence, C. Lynch, A. Zeng, O. Ramirez, A. Wahid, L. Downs, A. Wong, J. Lee, I. Mordatch, and J. Tompson. Implicit behavioral cloning. *Conference on Robot Learning (CoRL)*, November 2021.

[8] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[9] C. Lynch and P. Sermanet. Language conditioned imitation learning over unstructured data. *arXiv preprint arXiv:2005.07648*, 2020.

[10] S. Stepputtis, J. Campbell, M. Phielipp, S. Lee, C. Baral, and H. Ben Amor. Language-conditioned imitation learning for robot manipulation tasks. *Advances in Neural Information Processing Systems*, 33:13139–13150, 2020.

[11] E. Jang, A. Irpan, M. Khansari, D. Kappler, F. Ebert, C. Lynch, S. Levine, and C. Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In A. Faust, D. Hsu, and G. Neumann, editors, *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 991–1002. PMLR, 08–11 Nov 2022. URL https://proceedings.mlr.press/v164/jang22a.html.

[12] 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.

[13] M. Shridhar, L. Manuelli, and D. Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Proceedings of the 6th Conference on Robot Learning (CoRL)*, 2022.

[14] C. Lynch, A. Wahid, J. Tompson, T. Ding, J. Betker, R. Baruch, T. Armstrong, and P. Florence. Interactive language: Talking to robots in real time. *arXiv preprint arXiv:2210.06407*, 2022.

[15] O. Mees, L. Hermann, and W. Burgard. What matters in language conditioned robotic imitation learning over unstructured data. *IEEE Robotics and Automation Letters*, 7(4):11205–11212, 2022.

[16] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[17] O. Mees, L. Hermann, E. Rosete-Beas, and W. Burgard. Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters*, 7(3):7327–7334, 2022.

[18] T. Xiao, H. Chan, P. Sermanet, A. Wahid, A. Brohan, K. Hausman, S. Levine, and J. Tompson. Robotic skill acquisition via instruction augmentation with vision-language models. *arXiv preprint arXiv:2211.11736*, 2022.

[19] S. Nair, E. Mitchell, K. Chen, S. Savarese, C. Finn, et al. Learning language-conditioned robot behavior from offline data and crowd-sourced annotation. In *Conference on Robot Learning*, pages 1303–1315. PMLR, 2022.

[20] R. Goyal, S. Ebrahimi Kahou, V. Michalski, J. Materzynska, S. Westphal, H. Kim, V. Haenel, I. Fruend, P. Yianilos, M. Mueller-Freitag, et al. The" something something" video database for learning and evaluating visual common sense. In *Proceedings of the IEEE international conference on computer vision*, pages 5842–5850, 2017.

[21] K. Grauman, A. Westbury, E. Byrne, Z. Chavis, A. Furnari, R. Girdhar, J. Hamburger, H. Jiang, M. Liu, X. Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18995–19012, 2022.

[22] D. Damen, H. Doughty, G. M. Farinella, S. Fidler, A. Furnari, E. Kazakos, D. Moltisanti, J. Munro, T. Perrett, W. Price, et al. Scaling egocentric vision: The epic-kitchens dataset. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 720–736, 2018.

[23] A. S. Chen, S. Nair, and C. Finn. Learning generalizable robotic reward functions from" in-the-wild" human videos. *arXiv preprint arXiv:2103.16817*, 2021.

[24] C. R. Garrett, R. Chitnis, R. Holladay, B. Kim, T. Silver, L. P. Kaelbling, and T. Lozano-Pérez. Integrated task and motion planning. *Annual review of control, robotics, and autonomous systems*, 4: 265–293, 2021.

[25] W. Huang, P. Abbeel, D. Pathak, and I. Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International Conference on Machine Learning*, pages 9118–9147. PMLR, 2022.

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

[27] W. Huang, F. Xia, T. Xiao, H. Chan, J. Liang, P. Florence, A. Zeng, J. Tompson, I. Mordatch, Y. Chebotar, et al. Inner monologue: Embodied reasoning through planning with language models. In *6th Annual Conference on Robot Learning*.

[28] 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 *arXiv preprint arXiv:2209.07753*, 2022.

[29] D. Driess, F. Xia, M. S. Sajjadi, C. Lynch, A. Chowdhery, B. Ichter, A. Wahid, J. Tompson, Q. Vuong, T. Yu, et al. Palm-e: An embodied multimodal language model. *ICML*, 2023.

[30] K. Lin, C. Agia, T. Migimatsu, M. Pavone, and J. Bohg. Text2motion: From natural language instructions to feasible plans. *arXiv preprint arXiv:2303.12153*, 2023.

[31] A. Agarwal, A. Kumar, J. Malik, and D. Pathak. Legged locomotion in challenging terrains using egocentric vision, 2022.

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

[33] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter. Learning robust perceptive locomotion for quadrupedal robots in the wild. *Science Robotics*, 7(62):eabk2822, 2022.

[34] S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016.

[35] K. Hausman, Y. Chebotar, S. Schaal, G. Sukhatme, and J. J. Lim. Multi-modal imitation learning from unstructured demonstrations using generative adversarial nets. *Advances in neural information processing systems*, 30, 2017.

[36] N. M. M. Shafiullah, Z. J. Cui, A. Altanzaya, and L. Pinto. Behavior transformers: Cloning k modes with one stone, 2022.

[37] T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pages 1094–1100. PMLR, 2020.

[38] D. Kalashnikov, J. Varley, Y. Chebotar, B. Swanson, R. Jonschkowski, C. Finn, S. Levine, and K. Hausman. Mt-opt: Continuous multi-task robotic reinforcement learning at scale. *arXiv preprint arXiv:2104.08212*, 2021.

[39] J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015.

[40] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[41] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. L. Denton, K. Ghasemipour, R. Gontijo Lopes, B. Karagol Ayan, T. Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems*, 35:36479–36494, 2022.

[42] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

[43] L. Chen, K. Lu, A. Rajeswaran, K. Lee, A. Grover, M. Laskin, P. Abbeel, A. Srinivas, and I. Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

[44] S. Levine, A. Kumar, G. Tucker, and J. Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

[45] A. Zeng, P. Florence, J. Tompson, S. Welker, J. Chien, M. Attarian, T. Armstrong, I. Krasin, D. Duong, V. Sindhwani, et al. Transporter networks: Rearranging the visual world for robotic manipulation. In *Conference on Robot Learning*, pages 726–747. PMLR, 2021.

[46] S. M. LaValle et al. Rapidly-exploring random trees: A new tool for path planning.

[47] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[48] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[49] J. Song, C. Meng, and S. Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*.

[50] L. Downs, A. Francis, N. Koenig, B. Kinman, R. Hickman, K. Reymann, T. B. McHugh, and V. Vanhoucke. Google scanned objects: A high-quality dataset of 3d scanned household items, 2022. URL https://arxiv.org/abs/2204.11918.

[51] K. Zakka. Scanned Objects MuJoCo Models, 7 2022. URL https://github.com/kevinzakka/mujoco_scanned_objects.