Language Conditioned Imitation Learning over Unstructured Data

Corey Lynch *
Robotics at Google
coreylynch@google.com

Pierre Sermanet *
Robotics at Google
sermanet@google.com

***Abstract*—Natural language is perhaps the most flexible and intuitive way for humans to communicate tasks to a robot. Prior work in imitation learning typically requires each task be specified with a task id or goal image—something that is often impractical in open-world environments. On the other hand, previous approaches in instruction following allow agent behavior to be guided by language, but typically assume structure in the observations, actuators, or language that limit their applicability to complex settings like robotics. In this work, we present a method for incorporating free-form natural language conditioning into imitation learning. Our approach learns perception from pixels, natural language understanding, and multitask continuous control end-to-end as a single neural network. Unlike prior work in imitation learning, our method is able to incorporate unlabeled and unstructured demonstration data (i.e. no task or language labels). We show this dramatically improves language conditioned performance, while reducing the cost of language annotation to less than 1% of total data. At test time, a single language conditioned visuomotor policy trained with our method can perform a wide variety of robotic manipulation skills in a 3D environment, specified only with natural language descriptions of each task (e.g. "open the drawer...now pick up the block...now press the green button...") (see video). To scale up the number of instructions an agent can follow, we propose combining text conditioned policies with large pretrained neural language models. We find this allows a policy to be robust to many out-of-distribution synonym instructions, without requiring new demonstrations. See videos of a human typing live text commands to our agent at https://language-play.github.io**

I. Introduction

Imitation learning [4, 3] is a popular framework for acquiring complex robotic skills from raw onboard sensors. Traditionally, imitation learning has been applied to learning individual skills from structured and isolated human demonstrations [39, 1, 53].

These collection requirements are difficult to scale to real world scenarios, where robots are expected to be *generalists*—capable of autonomously performing a wide variety of skills. As we consider deploying imitation learning in this setting, a critical challenge is scale: how can we lessen the data requirements of learning each new skill? Is it possible instead to learn many skills simultaneously from large amounts of unstructured, *unlabeled* demonstration data [16, 26, 14]?

Recent works have focused on scaling up multitask imitation learning over unstructured data [8, 26]. However, these approaches typically assume that tasks are specified to the agent at test time via mechanisms like one-hot task selectors [37, 43], goal images [29, 11, 26], or target configurations of the state space [14]. While these types of conditioning are straightforward to provide in simulation, they are often impractical to provide in open-world settings. Thus, another important consideration when deploying imitation learning in everyday settings is scalable *task specification*: how can untrained users instruct robot behavior? This motivates robots that can follow free-form natural language instructions.

Training agents to follow instructions is an active area of research in the broader machine learning community [25], yet it remains a difficult open challenge. Prior approaches have trained agents that map observations and language inputs directly to actions using neural networks, but often make assumptions that limit their applicability to robotics. Typical studies involve 2D observation spaces (e.g. games [24, 32, 27, 12] and gridworlds [52]), simplified actuators, (e.g. binary pick and place primitives [18, 48]), or synthetic predefined language [17, 6, 20, 54].

In this work, we study the setting of human language conditioned robotic manipulation (Fig. 1, step 3). In this setting, a single agent must execute a series of visual manipulation tasks in a row, each expressed in free-form natural language, e.g. "open the door all the way to the right...now grab the block...now push the red button...now open the drawer". Furthermore, agents in this scenario are expected to be able to perform any combination of subtasks in any order. This is the first version of instruction following, to our knowledge, that combines: natural language conditioning, high-dimensional pixel inputs, 8-DOF continuous control, and complex tasks like long-horizon robotic object manipulation. Text conditioning is also a new and difficult setting for goal-directed imitation learning [26, 8], which introduces important new research questions. For example, how can we learn the mapping between language and actions with the fewest language labels? How can we leverage large unstructured demonstration datasets that have no language labels? How can we follow the maximum number of instructions at test time, given a finite training set?

To address this setting, we propose a simple approach for combining imitation learning with free-form text conditioning (Fig. 1). Our method consists of only standard supervised learning subroutines, and learns perception, language understanding, and control end-to-end as a single neural network. Critically, unlike prior work in instruction following, our method can leverage large amounts of unstructured and

* Equal contribution

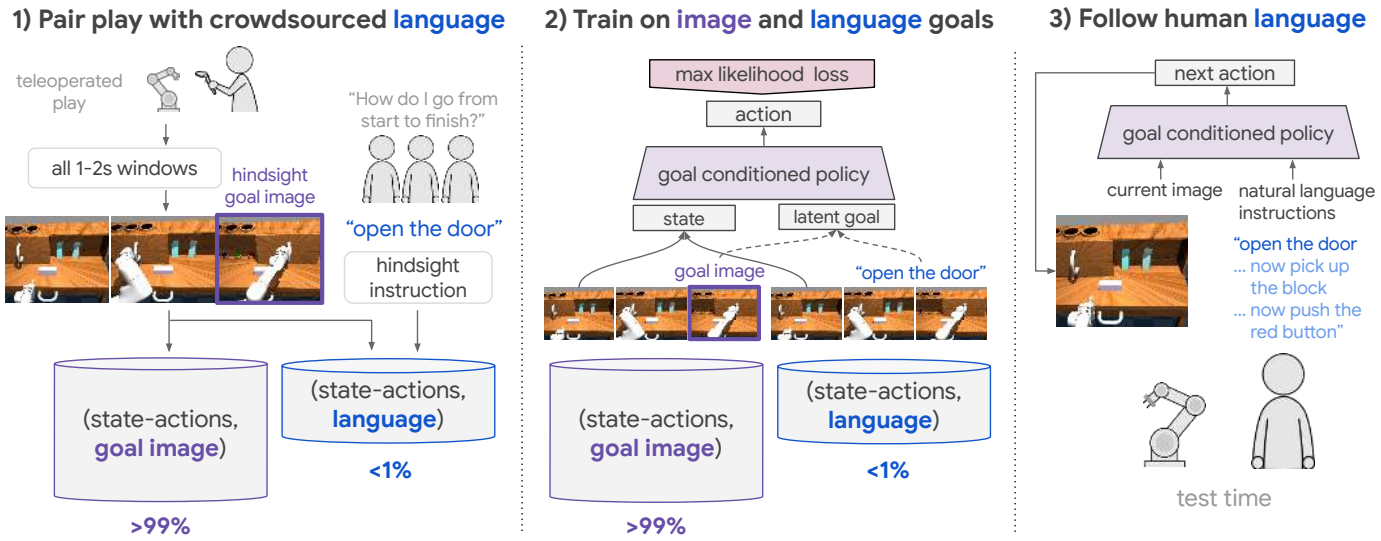

Fig. 1: **Scaling up free-form natural language instruction following with unstructured data.** 1) First, relabel unstructured teleoperated "play" data (no task labels) into hindsight goal image examples. Next, pair a small amount of play with hindsight instructions. 2) Train a single policy to follow either image or language goals. 3) Use only language conditioning at test time.

unlabeled demonstration data (i.e. with no language or task labels). We show this reduces the burden of language annotation to less than 1% of total data. To scale up the maximum amount of instructions our agent can follow at test time, we introduce a simple technique for combining any language conditioned policy with *large pretrained language models* [7, 36]. We find that this simple modification allows our agent to follow thousands of new synonym instructions at test time, without requiring that new robot demonstrations be collected for each synonym. We believe that the capabilities proposed here of learning from unlabeled demonstrations, end-to-end learning of text conditioned visuomotor policies, and following new synonym instructions without new robot data constitute important steps towards scalable robot learning systems.

We evaluate our method in a dynamically accurate simulated 3D tabletop environment with a fixed set of objects. Our experiments show that a language conditioned visuomotor policy trained with our method can perform many complex robotic manipulation skills in a row specified entirely with natural language (see video), outperforming conventional imitation baselines trained on structured data.

Contributions To summarize the contributions of this work, we:

- introduced a setting of human language conditioned robotic visual manipulation.
- introduced a simple learning method for combining free-form text conditioning with multitask imitation learning.
- introduced *multicontext imitation learning*, applicable to any contextual imitation learning setup, allowing us to train a language conditioned policy over mostly unstructured and unlabeled demonstrations (i.e. with no task or language labels).
- demonstrated that the resulting language conditioned visuomotor policy can follow many free-form human text instructions over a long horizon in a simulated 3D tabletop setting, i.e. "open the door... now pick up the block... now press the red button" (see video).
- introduced a simple way to combine any language conditioned policy with large pretrained language models. We show this improves manipulation performance and allows an agent to be robust to thousands of new synonym instructions at test time, in 16 languages, without requiring new demonstrations for each synonym.

II. Related Work

Robotic learning from general sensors. In general, learning complex robotic skills from low-level sensors is possible, but requires substantial human supervision. Two common approaches are imitation learning (IL) [3] and reinforcement learning (RL) [23]. When combined with deep function approximators, IL typically requires many human demonstrations [38, 37, 53] to drive supervised learning of a policy. In RL, supervision takes the form of hand-designed task rewards. Reward design is non-trivial in complex environments, often requiring either task-specific instrumentation [13] or learned perceptual rewards [41, 42]. Additionally, RL agents often require hand-designed strategies [22, 9] or human demonstrations [38] to overcome hard exploration problems. Finally, even under multitask formulations of RL [46] and IL [37], each new task considered requires a corresponding and sizable human effort. This makes it difficult to scale either approach naively to a broad task setting. In this work, we focus on scaling imitation learning to be multitask and language conditioned. While there are several ways to perform imitation learning such as inverse reinforcement learning [30, 50, 10] and occupancy matching [19], we restrict our attention in this work to behavior cloning [35] given its stability and ease of use.

Imitation learning from large unstructured datasets. Recent works [16, 26, 14] have sought to mitigate the costs of conventional multitask imitation learning by instead learning many skills at once over large unstructured demonstration

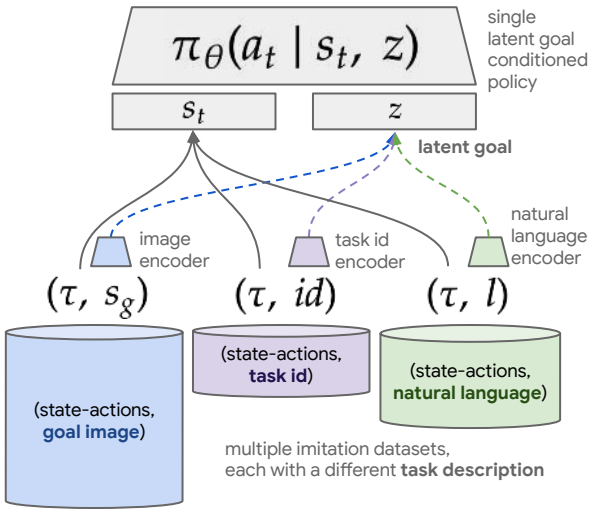

Fig. 2: **Multicontext Imitation Learning (MCIL).** We introduce a simple generalization of contextual imitation learning to multiple heterogeneous contexts (e.g. goal image, task id, natural language). MCIL trains a single *latent goal* conditioned policy over all datasets simultaneously, as well as one encoder per dataset, each mapping to the shared latent goal space. This allows training a language conditioned policy over both labeled and unlabeled demonstration datasets.

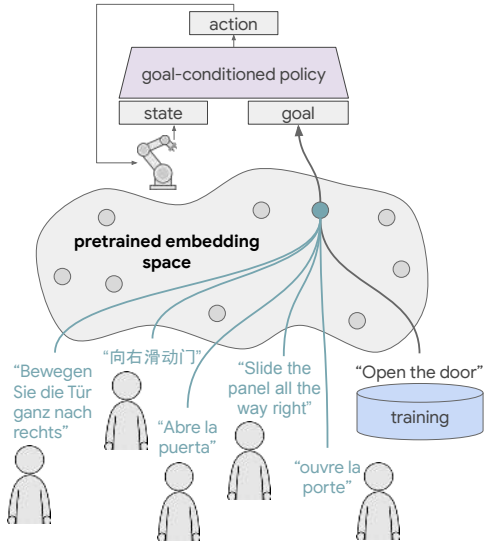

Fig. 3: **Pretrained language models make language conditioned policies robust to out-of-distribution synonyms**. Simply by training on top of pretrained language embeddings, we can give a language conditioned policy the ability to follow out-of-distribution synonym instructions at test time. The pretrained embedding space is responsible for relating new synonym instructions (green) to ones from the agent's training set (black).

datasets. Like these works, we incorporate unstructured demonstration data into our method. Unlike these works, the resulting goal conditioned policies can be conditioned with natural language.

Task agnostic control. This paper builds on the setting of task agnostic control, where a single agent is trained to reach any reachable goal state in its environment upon command [21, 40]. One way of acquiring this kind of control is to first learn a model of the environment through interaction [31, 9] then use it for planning. However, these approaches rely on accurate forward models of visual dynamics, a challenging open problem. A powerful model-free strategy for task agnostic control is goal relabeling [21, 2]. This technique trains goal conditioned policies to reach any previously visited state upon demand, with many recent examples in RL [29, 14, 28] and IL [26, 8]. A limitation to models combining relabeling with image observation spaces is that tasks must be specified with goal images at test time. The present work builds on relabeled imitation, but additionally equips policies with natural language conditioning.

Multicontext learning. A number of previous methods have focused on generalizing across tasks [5], or generalizing across goals [40]. We introduce multicontext learning, a framework for generalizing across heterogeneous task and goal descriptions. When one of the training sources is plentiful and the other scarce, multicontext learning can be seen as transfer learning [45] through a shared goal space. Multicontext imitation is a central component of our method, as it reduces the cost of human language supervision to the point where it can be practically applied.

Instruction following. There is a long history of research into agents that not only learn a grounded language understanding [49], but demonstrate that understanding by following instructions (survey [25]). Recently, authors have had success using deep learning to directly map raw input and text instructions to actions. However, prior work has often studied 2D environments [24, 32, 27, 52] and simplified actuators [48, 18, 12]. Additionally, learning to follow *natural* language is still not the standard in instruction following research [25], with typical implementations instead assuming access to simulator-provided instructions drawn from a restricted vocabulary and grammar [17]. This work, in contrast, studies 1) natural language instructions, 2) high-dimensional continuous sensory inputs and actuators, and 3) complex tasks like long-horizon 3D robotic object manipulation. Furthermore, unlike existing RL approaches to instruction following [18], our IL method is sample efficient, requires no reward definition, and scales easily to the multitask setting. Unlike concurrent IL approaches to robotic instruction following [44], which assume access to labeled task demonstrations and pretrained object detection, our method learns from unstructured and unlabeled demonstration data and learns perception, language understanding, and control fully end-to-end via a single imitation loss.

III. Problem Formulation

We consider the problem of learning a natural language conditioned control policy $\pi_\theta(a|s, l)$, which outputs next action $a \in A$, conditioned on current state $s \in S$ and free-form natural language $l \in L$ describing a short-horizon task. Note that S is not the true environment state, but rather the high dimensional onboard observations of the robot, e.g. $S = \{\text{image}, \text{proprioception}\}$. l is provided by humans and has no limits on vocabulary or grammar.

At test time, a human gives a robot a series of instructions

in a row, one at a time: $\{l_0, ..., l_N\}$. The language conditioned visuomotor policy $\pi_\theta(a|s, l)$ issues high frequency continuous control in closed loop to obtain the desired behavior described in l. The human may provide new instructions l to the agent at any time, either commanding a new subtask or providing guidance (i.e. "move your hand back slightly").

Standard imitation learning setups learn single task policies $\pi_\theta(a|s)$ using supervised learning over a dataset $\mathcal{D} = \{\tau_i\}_i^N$, of expert state-action trajectories $\tau = \{(s_0, a_0), ...\}$. Closest to our problem setting is goal conditioned imitation learning [21], which instead aims to learn a policy $\pi_\theta(a|s, g)$, conditioned on $s \in S$ and a task descriptor $g \in G$ (often a one-hot task encoding [37]). When tasks can be described as a goal state $g = s_g \in S$ to reach, this allows any state visited during collection to be *relabeled* [21, 2, 8] as a "reached goal state", with the preceding states and actions treated as optimal behavior for reaching that goal. At test time, learned behaviors are conditioned using a goal image s_g.

Relabeled imitation learning can be applied to unstructured and unlabeled demonstration data [26] to learn general purpose goal-reaching policies. Here, demonstrators are not constrained to a predefined set of tasks, but rather engage in every available object manipulation in a scene (see example). This yields one long unsegmented demonstration of semantically meaningful behaviors, which can be relabeled using Algorithm 2 into a training set $D_{\text{play}} = \{(\tau, s_g)_i\}_{i=0}^{D_{\text{play}}}$. These short horizon goal image conditioned demonstrations are fed to a maximum likelihood goal conditioned imitation objective: $\mathcal{L}_{\text{GCIL}} = \mathbb{E}_{(\tau,\ s_g)\sim D_{\text{play}}}\left[\sum_{t=0}^{|\tau|} \log \pi_\theta\left(a_t|s_t, s_g\right)\right]$.

However, when learning language conditioned policies $\pi_\theta(a|s, l)$, we cannot easily relabel any visited state s to a natural language goal as the goal space G is no longer equivalent to the observation space L. Consequently, this limits language conditioned imitation learning from being able to incorporate large unstructured demonstration datasets D_{play}.

Next, we describe our approach that relaxes this limitation.

IV. Learning to Follow Human Instructions from Unstructured Data

We present a framework that aims for a scalable combination of self-supervised relabeled imitation learning and labeled instruction following. Our approach (Fig. 1, Sec. IV-C) can be summarized as follows:

1) Collection: Collect a large unstructured "play" demonstration dataset.
2) Relabeling: Relabel unstructured data into goal image demonstrations. Pair a small number of random windows with language after-the-fact. (Sec. IV-A)
3) Multicontext imitation: Train a single imitation policy to solve for either goal image or language goals. (Sec. IV-B)
4) Use only language conditioning at test time.

A. Pairing unstructured demonstrations with natural language

Learning to follow language instructions involves addressing a difficult language grounding problem [15]: how do agents relate unstructured language l to their onboard perceptions s and actions a? We take a statistical machine learning approach, creating a paired corpora of (demonstration, language) by pairing random windows from unstructured data D_{play} with *hindsight instructions* after-the-fact (Algorithm 3, part 1 of Fig. 1). Here, we show untrained human annotators onboard videos, then ask them "what instruction would you give the agent to get from first frame to last frame?" See training examples in Table V, video examples here, and a full description of the collection process in Appendix M. Annotators were encouraged to use free-form natural language and not constrain themselves to predefined vocabulary or grammar. This yields a new dataset $D_{(\text{play,lang})} = \{(\tau, l)_i\}_{i=0}^{D_{(\text{play,lang})}}$, a dataset of text conditioned demonstrations. Crucially, we do not need pair every window from play with language to learn to follow instructions. This is made possible with Multicontext Imitation Learning, described next.

Algorithm 1 Multicontext imitation learning

Input: $\mathcal{D} = \{D^0, ...,\ D^K\}, D^k = \{(\tau_i^k, c_i^k)\}_{i=0}^{D^k}$, One dataset per context type (e.g. goal image, language instruction, task id), each holding pairs of (demonstration, context).
Input: $\mathcal{F} = \{f_\theta^0, ...,\ f_\theta^K\}$, One encoder per context type, mapping context to shared latent goal space, i.e. $z = f_\theta^k(c^k)$.
Input: $\pi_\theta(a_t|s_t, z)$, Single latent goal conditioned policy.
Input: Randomly initialize parameters $\theta = \{\theta_\pi,\ \theta_{f^0},\ ...,\ \theta_{f^K}\}$

while True **do**
  $\mathcal{L}_{\text{MCIL}} \leftarrow 0$
  # Loop over datasets.
  for $k = 0...K$ **do**
    # Sample a (demonstration, context) batch from this dataset.
    $(\tau^k, c^k) \sim D^k$
    # Encode context in shared latent goal space.
    $z = f_\theta^k(c^k)$
    # Accumulate imitation loss.
    $\mathcal{L}_{\text{MCIL}} \mathrel{+}= \sum_{t=0}^{|\tau^k|} \log \pi_\theta\left(a_t|s_t, z\right)$
  end for
  # Average gradients over context types.
  $\mathcal{L}_{\text{MCIL}} \mathrel{*}= \frac{1}{|\mathcal{D}|}$
  # Train policy and all encoders end-to-end.
  Update θ by taking a gradient step w.r.t. $\mathcal{L}_{\text{MCIL}}$
end while

B. Multicontext Imitation Learning

So far, we have described a way to create two contextual imitation datasets: D_{play}, holding goal image demonstrations and $D_{(\text{play,lang})}$, holding language demonstrations. Ideally, we could train a single imitation policy that could be conditioned with *either* task description. Critically, this would enable language conditioned imitation to make use of self-supervised imitation over unstructured data.

With this motivation, we introduce *multicontext imitation learning* (MCIL), a simple and universally applicable generalization of contextual imitation to multiple heterogeneous contexts. The main idea is to represent a large set of policies by a single, unified function approximator that generalizes over states, tasks, and *task descriptions*. Concretely, MCIL

assumes access to multiple contextual imitation datasets $\mathcal{D} = \{D^0, ..., D^K\}$, each with a different way of describing tasks. Each $D^k = \{(\tau_i^k, c_i^k)\}_{i=0}^{D^k}$ holds demonstrations τ paired with some context $c \in C^k$. For example, D^0 might contain one-hot task demonstrations (a conventional multitask imitation dataset), D^1 might contain goal image demonstrations (obtained by hindsight relabeling), and D^2 might contain language goal demonstrations. MCIL trains a single *latent goal* conditioned policy $\pi_\theta(a_t|s_t, z)$ over *all* datasets simultaneously, as well as one parameterized encoder per dataset $\mathcal{F} = \{f_\theta^0, ..., f_\theta^K\}$, learning to map raw task descriptions to a common latent goal space $z \in \mathbb{R}^d$. See Fig. 2. For instance, these could be a one-hot task embedding lookup, an image encoder, and a language encoder respectively.

MCIL has a simple training procedure, shown in Algorithm 1: at each training step, sample a batch of contextual imitation examples from each dataset, encode contexts in the shared latent space using the respective encoders, then compute a latent goal-conditioned imitation loss, averaged over all datasets. The policy and goal encoders are trained end-to-end to maximize this objective.

MCIL allows for a highly efficient training scheme, broadly useful beyond this paper: learn the majority of control from the data source that is cheapest to collect, while simultaneously learning scalable task conditioning from a small number of labeled examples. As we will see empirically, MCIL allows us to train an instruction following agent with less than 1% of data requiring language annotation, with the majority of perception and control learned instead from relabeled goal image imitation.

C. LangLfP: following image or language goals.

We now have all the components to introduce *LangLfP* (language conditioned learning from play). LangLfP is a special case of MCIL (Sec. IV-B) applied to our problem setting, and learns perception from pixels, natural language understanding, and control end-to-end with no auxiliary losses.

Training LangLfP. LangLfP trains a single multicontext policy $\pi_\theta(a_t|s_t, z)$ over datasets $\mathcal{D} = \{D_{\text{play}}, D_{(\text{play,lang})}\}$. We define $\mathcal{F} = \{g_{\text{enc}}, s_{\text{enc}}\}$, neural network encoders mapping from goal images (Appendix D) and text instructions (Appendix E) respectively to z. Fig. 4 compares LangLfP training to prior LfP training. At each training step, LangLfP: 1) samples a batch of image goal tasks from D_{play} and a batch of language goal tasks from $D_{(\text{play,lang})}$, 2) encodes image and language goals into z, and 3) computes the MCIL objective. We then take a combined gradient step with respect to all modules—perception (Appendix C), language (Appendix E), and control (Appendix F)—optimizing the whole architecture end-to-end as a single neural network. See full training details in Appendix G.

Following natural language instructions at test time. At test time, LangLfP conditions only on onboard pixel observations and a free-form natural language task description to solve user-defined tasks in closed loop. See picture in part 3 of Fig. 1 and details in Appendix H.

Leveraging pretrained language models While LangLfP offers a straightforward way for learning natural language end-to-end from imitation, this may not always be desirable. In open-world scenarios, instruction following robots are likely to be given instructions that are *synonyms* to ones they have been trained to follow, but do not overlap exactly with a finite training set. For example, "*shut* the door" is a valid, but potentially out-of-distribution way of describing training task "*close* the door". Many recent works have successfully transferred knowledge from unlabeled text to downstream NLP tasks via pretrained embeddings [7, 36]. Can we achieve similar knowledge transfer to *robotic manipulation*? There are two motivations for this kind of transfer: 1) improve language conditioned manipulation and 2) allow an agent to follow out of distribution synonym instructions (Fig. 3). To test these hypotheses, we augment LangLfP, encoding language inputs l to the policy at training and test time in the pretrained embedding space of a multilingual neural language model [51]. We refer to this augmented model as *TransferLangLfP*. See Appendix E for details.

V. Experiments

Our experiments aim to answer the following questions: Q0) Can a policy trained with our method learn perception, natural language understanding, and control end-to-end to solve many free-form text conditioned tasks in a 3D tabletop environment? Q1) Can our language conditioned multitask imitation learning (LangLfP) match the performance of goal image conditioned imitation learning (LfP)? Q2) Does our method's ability to leverage large unlabeled imitation datasets improve language conditioned performance? Q3) Does incorporating pretrained language models grant any benefit to manipulation performance? Q4) Does incorporating pretrained language models allow our agent to be robust to out of distribution synonym instructions?

Dataset and tasks. We conduct our experiments in the simulated 3D Playroom environment introduced in [26], shown in Fig. 10. It consists of an 8-DOF robotic arm controlled with continuous position control from visual observations at 30Hz to accomplish manipulation tasks from 18 distinct task families. See [26] for a complete discussion of tasks. We modify the environment to include a text channel which the agent observes at each timestep, allowing humans to type unconstrained language commands (details in Appendix H). We define two sets of experiments for each baseline: **pixel experiments**, where models receive pixel observations and must learn perception end-to-end, and **state experiments**, where models instead receive simulator state consisting of positions and orientations for all objects in the scene. The latter provides an upper bound on how all well the various methods can learn language conditioned control, independent of a difficult perception problem (which might be improved upon independently with self-supervised representation learning methods (e.g. [41, 33, 34])).

Methods. We compare the following methods (details on each in Appendix O): **LangBC** ("language, but no play"):

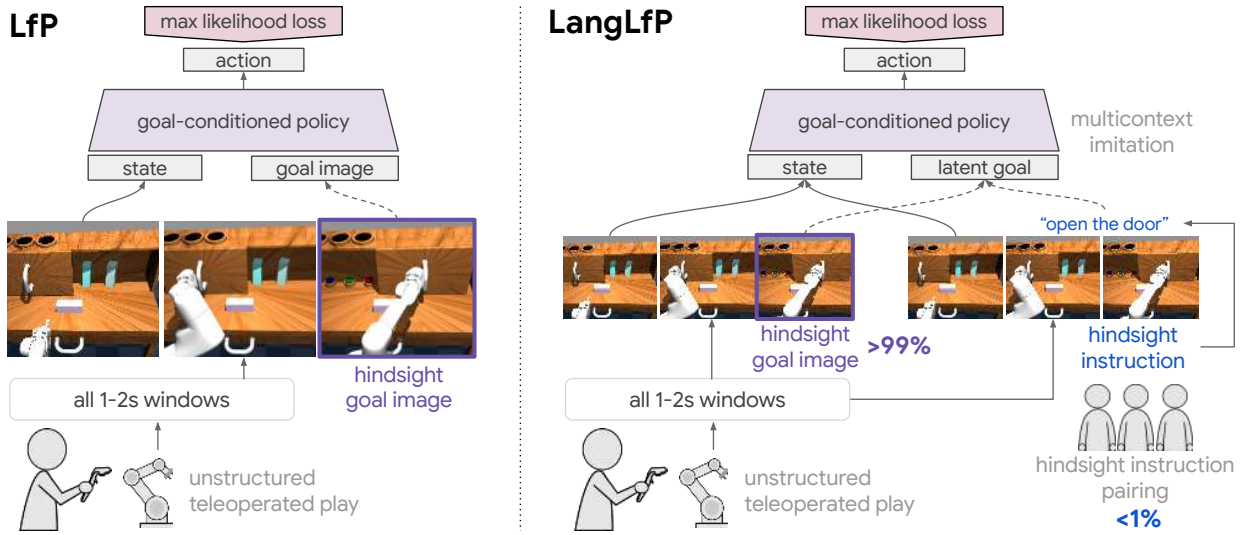

Fig. 4: **Comparing prior LfP (left) to our LangLfP (right)**. Both are trained on unstructured and unsegmented demonstrations, relabeled into goal image demonstrations. LangLfP is additionally trained on random windows paired with hindsight natural language instructions.

a baseline natural language conditioned multitask imitation policy [37], trained on $D_{(\text{demo,lang})}$, 100 expert demonstrations for each of the 18 evaluation tasks paired with hindsight instructions. **LfP** ("play, but no language"): a baseline LfP model trained on D_{play}, conditioned on goal images at test time. **LangLfP (ours)** ("play and language"): Multicontext imitation trained on unstructured data D_{play} and unstructured data paired with language $D_{(\text{play,lang})}$. Tasks are specified at test time using only natural language. **Restricted LangLfP**: LangLfP trained on "restricted D_{play}", a play dataset restricted to be the same size as $D_{(\text{demo,lang})}$. This restriction is somewhat artificial, as more unsegmented play can be collected for the same budget of time. This provides a controlled comparison to LangBC. **TransferLangLfP (ours)**: LangLfP trained on top of pretrained language embeddings. To perform a controlled comparison, we use the same network architecture (details in Sec. B) across all baselines. See Appendix K for a detailed description of all data sources.

A. Human Language Conditioned Visual Manipulation

We construct a large number of multi-stage human language conditioned manipulation tasks by treating the original 18 evaluation tasks in [26] as subtasks, then considering all valid N-stage transitions between them. This results in 2-stage, 3-stage, and 4-stage benchmarks, referred to here as Chain-2, Chain-3, and Chain-4. See Appendix O for benchmark details. We obtain language instructions for each subtask in the same way as training (Sec. IV-A), by presenting human annotators with videos of the completed tasks and asking for hindsight instructions. We present results in Table I and Fig. 5 and discuss below.

Language conditioned manipulation results. In Table I, we see that LangLfP can achieve 68.6% success on individual free-form instructions (covering 18 different tasks), 52.1% success on 925 4-stage instructions. This helps answer the first main question of this work (Q0), whether our approach can learn perception, control, and language understanding end-to-end to follow many free-form human instructions. We also see that LangLfP matches the performance of prior goal image conditioned LfP on all benchmarks within margin of error (Q1). This is important, as it shows a more scalable mode of task conditioning can be achieved with only ∼0.1% of demonstration data requiring language annotation.

Large unlabeled imitation datasets improve language conditioned performance. We see in Table I and Fig. 5 that LangLfP outperforms LangBC on every benchmark. This is important because it shows that by allowing our policy to train over unlabeled demonstration data (via MCIL), it achieves significantly better performance than baseline policies trained only on typical predefined task demonstrations (Q2). We note this holds even when unlabeled training sources are restricted to the same size as labeled ones (Restricted LangLfP vs. LangBC). Qualitatively (videos), we see clear differences between models that incorporate unstructured data and those that do not. We find that on long-horizon evaluations, MCIL-trained policies tend to transition well between tasks and recover from initial failures, whereas baseline policies trained on conventional demonstrations tend to quickly encounter compounding imitation error.

High capacity models make the most use of unlabeled data. In Fig. 6 and Appendix Q, we additionally find the phenomenon that performance scales linearly with model size for policies that leverage unstructured data, whereas performance peaks and then declines for models trained on conventional predefined demonstrations. We see this holds even when datasets are restricted to the same size. This suggests that the simple recipe of collecting large unstructured imitation datasets, pairing them with a small amount of language data, then training large capacity imitation learning models may be a valid way to scale up language conditioned control.

Language unlocks human assistance. Natural language conditioning allows for new modes of interactive test time

Method	Input	Training source	Task conditioning	Multi-18 Success (18 tasks)	Chain-4 Success (925 long-horizon tasks)
LangBC	pixels	predefined demos	text	20.0% ±3.0	7.1% ±1.5
Restricted LangLfP	pixels	unstructured demos	text	47.1% ±2.0	25.0% ±2.0
LfP	pixels	unstructured demos	image	66.4% ±2.2	53.0% ±5.0
LangLfP (ours)	pixels	unstructured demos	text	68.6% ±1.7	52.1% ±2.0
TransferLangLfP (ours)	pixels	**unstructured demos**	**text**	**74.1%** ±1.5	**61.8%** ±1.1
LangBC	states	predefined demos	text	38.5% ±6.3	13.9% ±1.4
Restricted LangLfP	states	unstructured demos	text	88.0% ±1.4	64.2% ±1.5
LangLfP (ours)	states	unstructured demos	text	88.5% ±2.9	63.2% ±0.9
TransferLangLfP (ours)	states	**unstructured demos**	**text**	**90.5%** ±0.8	**71.8%** ±1.6

TABLE I: **Human language conditioned visual manipulation experiments**

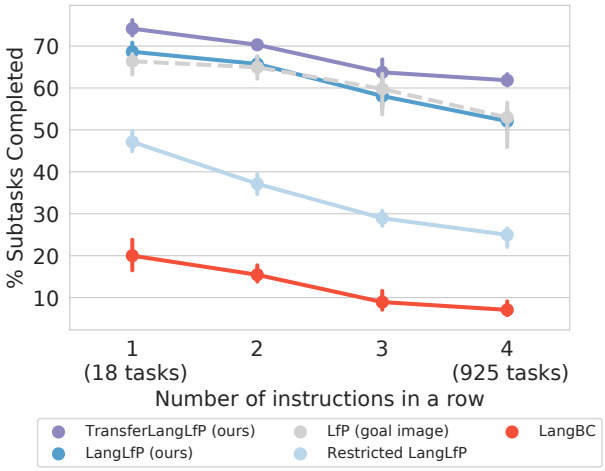

Fig. 5: **Long horizon language conditioned visual manipulation results.**

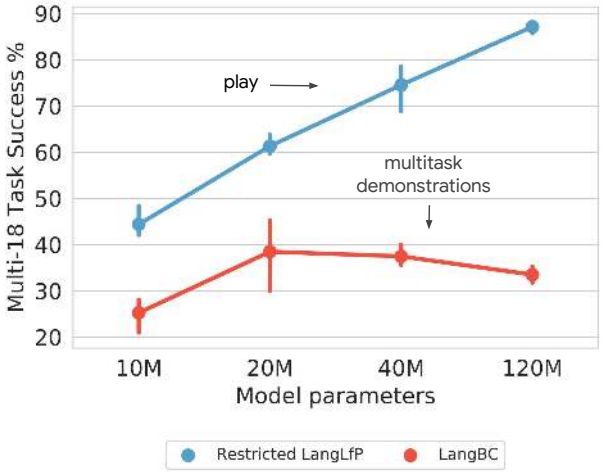

Fig. 6: **Performance scales linearly with model capacity, but only for unstructured demonstrations.** We see that for the same amount of data, more diverse unstructured imitation data is better utilized by large model learning.

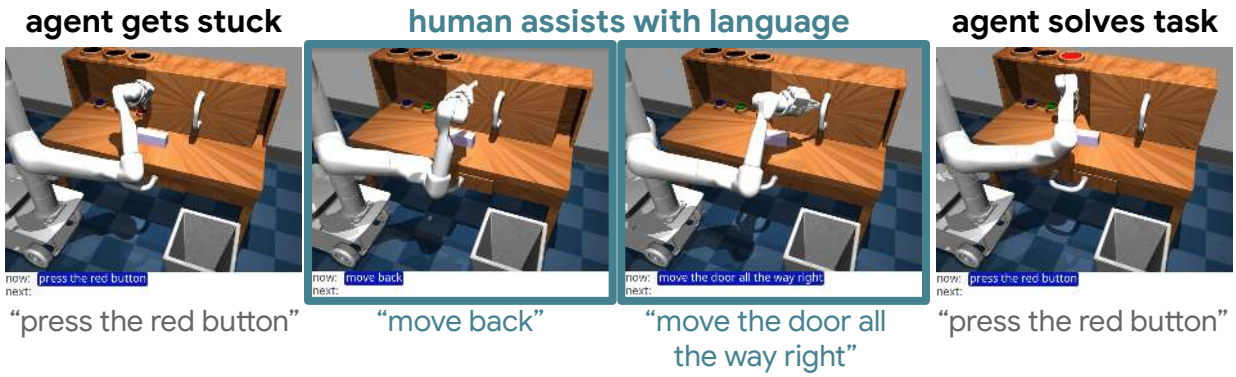

Fig. 7: **Getting out of trouble with human language assistance:** Unlike other forms of task conditioning, natural language conditioning allows a human operator to offer quick interactive assistance when an agent gets stuck.

behavior, allowing humans to give guidance to agents that would be impractical to give via goal image or one-hot task conditioned control. See Fig. 7, this video and Sec. R for a concrete example. Sec. O additionally shows how humans can quickly compose tasks with language that are outside the 18-task benchmark (but covered by the training set), like "put the block in the trash bin".

B. Instruction Following with Large Pretrained Language Models

Positive transfer to robotic manipulation. In Table I and Fig. 8, we see that TransferLangLfP systematically outperforms LangLfP and LfP. This is the first evidence, to our knowledge, that sentence embeddings obtained from large pretrained language models can significantly improve the convergence of language-guided robotic control policies (Q3).

Robustness to out-of-distribution synonym instructions. In Table II, we study how robust our language conditioned

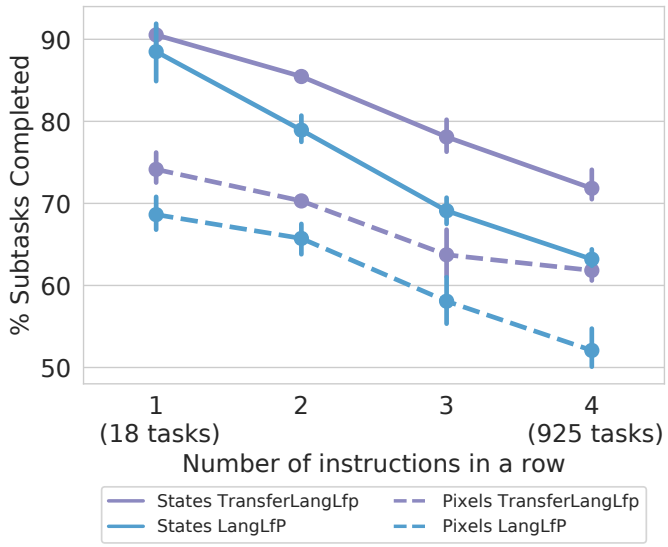

Fig. 8: **Knowledge transfer from generic text corpora benefits robotic manipulation.** We see models that take pretrained embeddings as language input (purple) converge to higher performance than those that must learn language from scratch (blue).

Method	**OOD-syn (∼15k instructions)**	**OOD-16-lang (∼240k instructions)**
Random Policy	0.0% ± 0.0	0.0% ± 0.0
LangLfP	37.6% ± 2.3	27.94% ± 3.5
TransferLangLfP	**60.2%** ± 3.2	**56.0%** ± 1.4

TABLE II: **Out of distribution synonym robustness.**

policies are to synonyms not found in the training set. We see that only agents equipped with large pretrained language models (TransferLangLfP) are robust to these out of distribution synonyms (OOD-syn), giving affirmative support to Q4. Additionally, just by choosing a multilingual pretrained model [51], we see this kind of robustness extends to 16 different languages, which have no vocabulary overlap with training (OOD-16-lang). See videos (link) of TransferLangLfP following multilingual synonym instructions. **We stress that these experiments do not test the ability of a policy to generalize to new *kinds* of manipulation tasks than the ones seen in training**. Rather, this shows that a simple training modification increases the number of *ways* a language-guided agent can be conditioned to execute the same *fixed set* of behaviors from its training distribution, effectively expanding the training instruction set. Find more details on these experiments in Appendix S.

C. Limitations and Future Work

Although the coverage of unstructured play demonstration data mitigates failure modes in conventional imitation setups, we observe several limitations in our policies at test time. In this video, we see the policy make multiple attempts to solve the task, but times out before it is able to do so. We see in this video that the agent encounters a particular kind of compounding error, where the arm flips into an awkward configuration, likely avoided by humans during teleoperated play. This is potentially mitigated by a more stable choice of rotation representation, or more varied play collection. We note that the human is free to help the agent out of these awkward configurations using language assistance, as demonstrated in Sec. R. More examples of failures can be seen here. While LangLfP relaxes important constraints around task specification, it is fundamentally a goal-directed imitation method and lacks a mechanism for autonomous policy improvement. An exciting area for future work may be one that combines the coverage of teleoperated play, the scalability and flexibility of multicontext imitation pretraining, and the autonomous improvement of reinforcement learning, similar to prior successful combinations of LfP and RL [14]. Additionally, the scope of this work is task agnostic control in a single simulated environment with a fixed set of objects. We note this is consistent with the standard imitation assumptions that training and test tasks are drawn i.i.d. from the same distribution. An interesting question for future work is whether training on a large play corpora covering many rooms and objects allows for generalization to unseen rooms or objects.

VI. Conclusion

We proposed a scalable framework for combining multitask imitation with free-form text conditioning. Our method can learn language conditioned visuomotor policies, capable of following multiple human instructions over a long horizon in a dynamically accurate 3D tabletop setting. Key to our method is the ability to learn over unstructured and unlabeled imitation data, a property we made possible by introducing Multicontext Imitation Learning. Critically, the ability to learn from unstructured data reduced the cost of language annotation to less than 1% of total data, while also resulting in much higher language conditioned task success. Finally, we showed a simple, but effective way to combine any language conditioned policy with large pretrained language models. We found that this small modification allowed our policy to be robust to many out-of-distribution synonym instructions, without requiring the collection of additional demonstration data.

Acknowledgments

We thank Karol Hausman, Eric Jang, Mohi Khansari, Kanishka Rao, Jonathan Thompson, Luke Metz, Anelia Angelova, Sergey Levine, and Vincent Vanhoucke for providing helpful feedback on this manuscript. We additionally thank the annotators for providing paired language instructions.

References

[1] Baris Akgun, Maya Cakmak, Karl Jiang, and Andrea L Thomaz. Keyframe-based learning from demonstration. *International Journal of Social Robotics*, 4(4):343–355, 2012.

[2] Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. In *Advances in neural information processing systems*, pages 5048–5058, 2017.

[3] Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57(5):469–483, 2009.

[4] Aude Billard, Sylvain Calinon, Ruediger Dillmann, and Stefan Schaal. Survey: Robot programming by demonstration. Technical report, Springrer, 2008.

[5] Rich Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.

[6] Maxime Chevalier-Boisvert, Dzmitry Bahdanau, Salem Lahlou, Lucas Willems, Chitwan Saharia, Thien Huu Nguyen, and Yoshua Bengio. Babyai: A platform to study the sample efficiency of grounded language learning. In *International Conference on Learning Representations*, 2018.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[8] Yiming Ding, Carlos Florensa, Pieter Abbeel, and Mariano Phielipp. Goal-conditioned imitation learning. In *Advances in Neural Information Processing Systems*, pages 15298–15309, 2019.

[9] Frederik Ebert, Chelsea Finn, Sudeep Dasari, Annie Xie, Alex Lee, and Sergey Levine. Visual foresight: Model-based deep reinforcement learning for vision-based robotic control. *arXiv preprint arXiv:1812.00568*, 2018.

[10] Chelsea Finn, Sergey Levine, and Pieter Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In *International conference on machine learning*, pages 49–58. PMLR, 2016.

[11] Carlos Florensa, Jonas Degrave, Nicolas Heess, Jost Tobias Springenberg, and Martin Riedmiller. Self-supervised learning of image embedding for continuous control. *arXiv preprint arXiv:1901.00943*, 2019.

[12] Prasoon Goyal, Scott Niekum, and Raymond J. Mooney. Using natural language for reward shaping in reinforcement learning, 2019.

[13] Shixiang Gu, Ethan Holly, Timothy Lillicrap, and Sergey Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 3389–3396. IEEE, 2017.

[14] Abhishek Gupta, Vikash Kumar, Corey Lynch, Sergey Levine, and Karol Hausman. Relay policy learning: Solving long horizon tasks via imitation and reinforcement learning. *Conference on Robot Learning (CoRL)*, 2019.

[15] Stevan Harnad. The symbol grounding problem. *Physica D: Nonlinear Phenomena*, 42(1-3):335–346, 1990.

[16] Karol Hausman, Yevgen Chebotar, Stefan Schaal, Gaurav Sukhatme, and Joseph Lim. Multi-modal imitation learning from unstructured demonstrations using generative adversarial nets. *arXiv preprint arXiv:1705.10479*, 2017.

[17] Karl Moritz Hermann, Felix Hill, Simon Green, Fumin Wang, Ryan Faulkner, Hubert Soyer, David Szepesvari, Wojciech Marian Czarnecki, Max Jaderberg, Denis Teplyashin, et al. Grounded language learning in a simulated 3d world. *arXiv preprint arXiv:1706.06551*, 2017.

[18] Felix Hill, Andrew Lampinen, Rosalia Schneider, Stephen Clark, Matthew Botvinick, James L McClelland, and Adam Santoro. Emergent systematic generalization in a situated agent. *arXiv preprint arXiv:1910.00571*, 2019.

[19] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *arXiv preprint arXiv:1606.03476*, 2016.

[20] Yiding Jiang, Shixiang Gu, Kevin Murphy, and Chelsea Finn. Language as an abstraction for hierarchical deep reinforcement learning, 2019.

[21] Leslie Pack Kaelbling. Learning to achieve goals. In *IJCAI*, pages 1094–1099. Citeseer, 1993.

[22] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *arXiv preprint arXiv:1806.10293*, 2018.

[23] Jens Kober, J Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

[24] Thomas Kollar, Stefanie Tellex, Deb Roy, and Nicholas Roy. Toward understanding natural language directions. In *2010 5th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 259–266. IEEE, 2010.

[25] Jelena Luketina, Nantas Nardelli, Gregory Farquhar, Jakob Foerster, Jacob Andreas, Edward Grefenstette, Shimon Whiteson, and Tim Rocktäschel. A survey of reinforcement learning informed by natural language. *arXiv preprint arXiv:1906.03926*, 2019.

[26] Corey Lynch, Mohi Khansari, Ted Xiao, Vikash Kumar, Jonathan Tompson, Sergey Levine, and Pierre Sermanet. Learning latent plans from play. *Conference on Robot Learning (CoRL)*, 2019. URL https://arxiv.org/abs/1903.01973.

[27] Dipendra Misra, John Langford, and Yoav Artzi. Mapping instructions and visual observations to actions with reinforcement learning. *arXiv preprint arXiv:1704.08795*, 2017.

[28] Ashvin Nair, Shikhar Bahl, Alexander Khazatsky, Vitchyr Pong, Glen Berseth, and Sergey Levine. Contextual imagined goals for self-supervised robotic learning. In *Conference on Robot Learning*, pages 530–539. PMLR, 2020.

[29] Ashvin V Nair, Vitchyr Pong, Murtaza Dalal, Shikhar Bahl, Steven Lin, and Sergey Levine. Visual reinforcement learning with imagined goals. In *Advances in Neural Information Processing Systems*, pages 9191–9200, 2018.

[30] Andrew Y Ng, Stuart J Russell, et al. Algorithms for inverse reinforcement learning. In *Icml*, volume 1, page 2, 2000.

[31] Junhyuk Oh, Xiaoxiao Guo, Honglak Lee, Richard L Lewis, and Satinder Singh. Action-conditional video prediction using deep networks in atari games. In *Advances in neural information processing systems*, pages 2863–2871, 2015.

[32] Junhyuk Oh, Satinder Singh, Honglak Lee, and Pushmeet Kohli. Zero-shot task generalization with multi-task deep reinforcement learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 2661–2670. JMLR. org, 2017.

[33] Sherjil Ozair, Corey Lynch, Yoshua Bengio, Aaron Van den Oord, Sergey Levine, and Pierre Sermanet. Wasserstein dependency measure for representation learning. In *Advances in Neural Information Processing Systems*, pages 15578–15588, 2019.

[34] Sören Pirk, Mohi Khansari, Yunfei Bai, Corey Lynch, and Pierre Sermanet. Online object representations with contrastive learning. *arXiv preprint arXiv:1906.04312*, 2019.

[35] Dean A Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural computation*, 3(1):88–97, 1991.

[36] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8):9, 2019.

[37] Rouhollah Rahmatizadeh, Pooya Abolghasemi, Ladislau Bölöni, and Sergey Levine. Vision-based multi-task manipulation for inexpensive robots using end-to-end learning from demonstration. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3758–3765. IEEE, 2018.

[38] Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. *arXiv preprint arXiv:1709.10087*, 2017.

[39] Stefan Schaal. Is imitation learning the route to humanoid robots? *Trends in cognitive sciences*, 3(6):233–242, 1999.

[40] Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In *International conference on machine learning*, pages 1312–1320, 2015.

[41] Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, and Sergey Levine. Time-contrastive networks: Self-supervised learning from video. *Proceedings of International Conference in Robotics and Automation (ICRA)*, 2018. URL http://arxiv.org/abs/1704.06888.

[42] Avi Singh, Larry Yang, Kristian Hartikainen, Chelsea Finn, and Sergey Levine. End-to-end robotic reinforcement learning without reward engineering. *arXiv preprint arXiv:1904.07854*, 2019.

[43] Avi Singh, Eric Jang, Alexander Irpan, Daniel Kappler, Murtaza Dalal, Sergey Levine, Mohi Khansari, and Chelsea Finn. Scalable multi-task imitation learning with autonomous improvement. *arXiv preprint arXiv:2003.02636*, 2020.

[44] Simon Stepputtis, Joseph Campbell, Mariano Phielipp, Stefan Lee, Chitta Baral, and Heni Ben Amor. Language-conditioned imitation learning for robot manipulation tasks. *arXiv preprint arXiv:2010.12083*, 2020.

[45] Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. A survey on deep transfer learning. In *International conference on artificial neural networks*, pages 270–279. Springer, 2018.

[46] Yee Teh, Victor Bapst, Wojciech M Czarnecki, John Quan, James Kirkpatrick, Raia Hadsell, Nicolas Heess, and Razvan Pascanu. Distral: Robust multitask reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 4496–4506, 2017.

[47] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *IROS*, pages 5026–5033. IEEE, 2012. ISBN 978-1-4673-1737-5. URL http://dblp.uni-trier.de/db/conf/iros/iros2012.html#TodorovET12.

[48] Xin Wang, Qiuyuan Huang, Asli Celikyilmaz, Jianfeng Gao, Dinghan Shen, Yuan-Fang Wang, William Yang Wang, and Lei Zhang. Reinforced cross-modal matching and self-supervised imitation learning for vision-language navigation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6629–6638, 2019.

[49] Terry Winograd. Understanding natural language. *Cognitive psychology*, 3(1):1–191, 1972.

[50] Markus Wulfmeier, Peter Ondruska, and Ingmar Posner. Maximum entropy deep inverse reinforcement learning. *arXiv preprint arXiv:1507.04888*, 2015.

[51] Yinfei Yang, Daniel Cer, Amin Ahmad, Mandy Guo, Jax Law, Noah Constant, Gustavo Hernandez Abrego, Steve Yuan, Chris Tar, Yun-Hsuan Sung, et al. Multilingual universal sentence encoder for semantic retrieval. *arXiv preprint arXiv:1907.04307*, 2019.

[52] Haonan Yu, Haichao Zhang, and Wei Xu. Interactive grounded language acquisition and generalization in a 2d world. *arXiv preprint arXiv:1802.01433*, 2018.

[53] Tianhao Zhang, Zoe McCarthy, Owen Jow, Dennis Lee, Xi Chen, Ken Goldberg, and Pieter Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–8. IEEE, 2018.

[54] Victor Zhong, Tim Rocktäschel, and Edward Grefenstette. Rtfm: Generalising to novel environment dynamics via reading. *arXiv preprint arXiv:1910.08210*, 2019.

APPENDIX

A. Relabeling Play

Algorithm 2 Creating many goal image conditioned imitation examples from teleoperated play.

Input: $\mathcal{S} = \{(s_{0:t}, a_{0:t})^n\}_n^\infty$, the unsegmented stream of observations and actions recorded during play.
Input: $D_{\text{play}} \leftarrow \{\}$.
Input: $w_{\text{low}},\ w_{\text{high}}$, bounds on hindsight window size.
while True **do**
  # Get next play episode from stream.
  $(s_{0:t}, a_{0:t}) \sim \mathcal{S}$
  for $w = w_{\text{low}}...w_{\text{high}}$ **do**
    for $i = 0...(t - w)$ **do**
      # Select each w-sized window.
      $\tau = (s_{i:i+w}, a_{i:i+w})$
      # Treat last observation in window as goal.
      $s_g = s_w$
      Add (τ, s_g) to D_{play}
    end for
  end for
end while

B. Pairing Play with Language

Algorithm 3 Pairing robot sensor data with natural language instructions.

Input: D_{play}, a relabeled play dataset holding (τ, s_g) pairs.
Input: $D_{(\text{play,lang})} \leftarrow \{\}$.
Input: get_hindsight_instruction(): human overseer, providing after-the-fact natural language instructions for a given τ.
Input: K, number of pairs to generate, $K << |D_{\text{play}}|$.
for $0...K$ **do**
  # Sample random trajectory from play.
  $(\tau, \_) \sim D_{\text{play}}$
  # Ask human for instruction making τ optimal.
  l = get_hindsight_instruction(τ)
  Add $(\tau,\ l)$ to $D_{(\text{play,lang})}$
end for

Below we describe the networks we use for perception, natural language understanding, and control—all trained end-to-end as a single neural network to maximize our multicontext imitation objective. We stress that this is only one implementation of possibly many for LangLfP, which is a general high-level framework 1) combining relabeled play, 2) play paired with language, and 3) multicontext imitation.

C. Perception Module

We map raw observations (image and proprioception) to low dimensional perceptual embeddings that are fed to the rest of the network. To do this we use the perception module described in Fig. 9. We pass the image through the network described in Table III, obtaining a 64-dimensional visual embedding. We then concatenate this with the proprioception observation (normalized to zero mean, unit variance using training statistics). This results in a combined perceptual embedding of size 72. This perception module is trained end to end to maximize the final imitation loss. We do no photometric augmentation on the image inputs, but anticipate this may lead to better performance.

Layer	Details
Input RGB Image	200 x 200 x 3
Conv2D	32 filters, 8 x 8, stride 4
Conv2D	64 filters, 4 x 4, stride 2
Conv2D	64 filters, 3 x 3, stride 1
Spatial softmax	
Flatten	
Fully connected	512 hidden units, relu
Fully connected	64 output units, linear

TABLE III: Hyperparameters for vision network.

D. Image goal encoder

g_{enc} takes as input image goal observation O_g and outputs latent goal z. To implement this, we reuse the perception module P_θ, described in Appendix C, which maps O_g to s_g. We then pass s_g through a 2 layer 2048 unit ReLU MLP to obtain z.

E. Language understanding module

From scratch. For LangLfP, our latent language goal encoder, s_{enc}, is described in Fig. 9. It maps raw text to a 32 dimensional embedding in goal space as follows: 1) apply subword tokenization, 2) retrieve learned 8-dim subword embeddings from a lookup table, 3) summarize the sequence of subword embeddings (we considered average pooling and RNN mechanisms for this, choosing average pooling based on validation), and 4) pass the summary through a 2 layer 2048 unit ReLU MLP. Out-of-distribution subwords are initialized as random embeddings at test time.

Transfer learning. For our experiments, we chose Multilingual Universal Sentence Encoder described in [51]. MUSE is a multitask language architecture trained on generic multilingual corpora (e.g. Wikipedia, mined question-answer pair datasets), and has a vocabulary of 200,000 subwords. MUSE maps sentences of any length to a 512-dim vector. We simply treat these embeddings as language observations and do not finetune the weights of the model. This vector is fed to a 2 layer 2048 unit ReLU MLP, projecting to latent goal space.

We note there are many choices available for encoding raw text in a semantic pretrained vector space. MUSE showed results immediately, allowing us to focus on other parts of the problem setting. We look forward to experimenting with different choices of pretrained embedder in the future.

F. Control Module

We could in principle use any goal-directed imitation network architecture to implement the policy $\pi_\theta(a_t|s_t, z)$. For direct comparison to prior work, we use Latent Motor Plans (LMP)

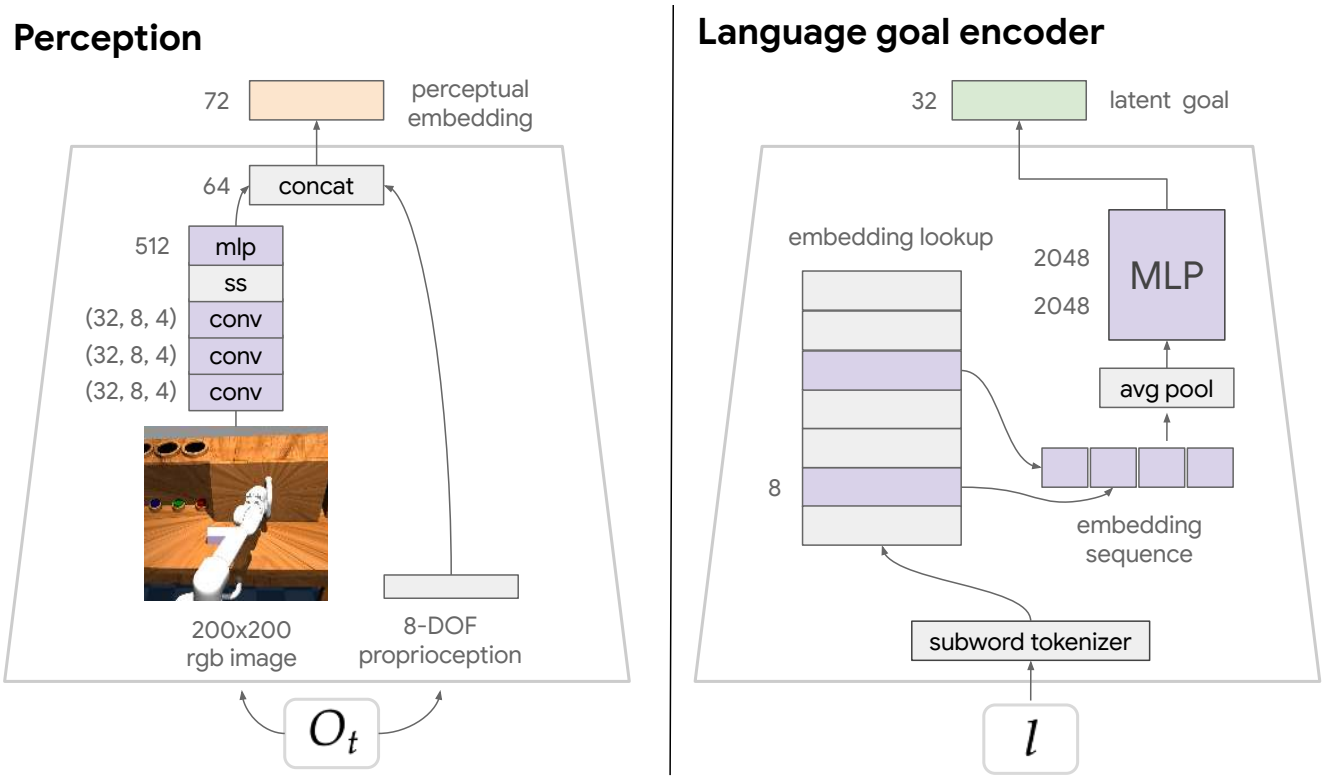

Fig. 9: **Perception and language embedder modules**

architecture Lynch et al. [26] to implement this network. LMP uses latent variables to model the large amount of multimodality inherent in freeform imitation datasets. Concretely it is a sequence-to-sequence conditional variational autoencoder (seq2seq CVAE), autoencoding contextual demonstrations through a latent "plan" space. The decoder is a goal and latent plan conditioned policy. As a CVAE, LMP lower bounds maximum likelihood contextual imitation ($\mathcal{L}_{\text{LfP}}$), and is easily adapted to our multicontext setting.

Multicontext LMP. Here we describe "multicontext LMP": LMP adapted to be able to take either image or language goals. This imitation architecture learns both an abstract visuolingual goal space z^g, as well as a plan space z^p capturing the many ways to reach a particular goal. We describe this implementation now in detail.

As a conditional seq2seq VAE, original LMP trains 3 networks. 1) A posterior $q(z^p|\tau)$, mapping from full state-action demonstration τ to a distribution over recognized plans. 2) A learned prior $p(z^p|s_0, s_g)$, mapping from initial state in the demonstration and goal to a distribution over possible plans for reaching the goal. 3) A goal and plan conditioned policy $p(a_t|s_t, s_g, z^p)$, decoding the recognized plan with teacher forcing to reconstruct actions that reach the goal. See [26] for details.

To train multicontext LMP, we simply replace the goal conditioning on s_g everywhere with conditioning on z^g, the latent goal output by multicontext encoders $\mathcal{F} = \{g_{\text{enc}}, s_{\text{enc}}\}$. In our experiments, g_{enc} is a 2 layer 2048 unit ReLU MLP, mapping from encoded goal image s_g to a 32 dimensional latent goal embedding. s_{enc} is a subword embedding summarizer described in Sec. E. Unless specified otherwise, the LMP implementation in this paper uses the same hyperparameters and network architecture as in [26]. See appendix there for details on the posterior, conditional prior, and decoder architecture, consisting of a RNN, MLP, and RNN respectively.

G. Training Details

At each training step, we compute two contextual imitation losses: image goal and language goal. The image goal forward pass is described in Fig. 16. The language goal forward pass is described in Fig. 17. We share the perception network and LMP networks (posterior, prior, and policy) across both steps. We average minibatch gradients from image goal and language goal passes and we train everything—perception networks, g_{enc}, s_{enc}, posterior, prior, and policy—end-to-end as a single neural network to maximize the combined training objective. We describe all training hyperparameters in Table IV.

H. Test time details

At the beginning of a test episode the agent receives as input its onboard observation O_t and a human-specified natural language goal l. The agent encodes l in latent goal space z using the trained sentence encoder s_{enc}. The agent then solves for the goal in closed loop, repeatedly feeding the current observation and goal to the learned policy $\pi_\theta(a|s_t, z)$, sampling actions, and executing them in the environment. The human operator can type a new language goal l at any time.

Hyperparameter	Value
hardware configuration	8 NVIDIA V100 GPUs
action distribution	discretized logistic, 256 bins per action dimension
optimizer	Adam
learning rate	2e-4
hindsight window low	16
hindsight window high	32
LMP β	0.01
batch size (per GPU)	64 sequences * padded max length 32 = 2048 frames
training time	3 days

TABLE IV: **Training Hyperparameters**

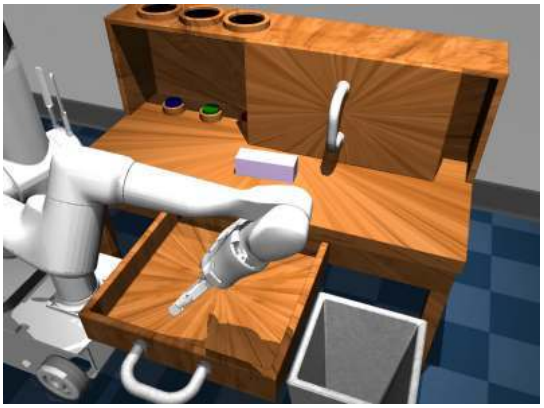

Fig. 10: **The Playground environment.**

We use the same simulated 3D playground environment as in [26], keeping the same observation and action spaces. We define these below for completeness.

I. 3D simulated environment

The environment contains a desk with a sliding door and drawer that can be opened and closed. On the desk is a movable rectangular block and 3 buttons that can be pushed to control different colored lights. Next to the desk is a trash bin. Physics are simulated using the MuJoCo physics engine [47]. Videos here of play data collected in this environment give a good idea of the available complexity. In front of the desk is a realistic simulated 8-DOF robot (arm and parallel gripper). The agent perceives its surroundings from egocentric high-dimensional RGB video sensors. It additionally has access to continuous internal proprioceptive sensors, relaying the cartesian position and rotation of its end effector and gripper angles. We modify the environment to include a text channel which the agent observes at each timestep, allowing humans to type unconstrained language commands. The agent must perform high-frequency, closed-loop continuous control to solve for user-described manipulation tasks, sending continuous position, rotation, and gripper commands to its arm at 30hz.

J. Observation space

We consider two types of experiments: pixel and state experiments. In the pixel experiments, observations consist of (image, proprioception) pairs of 200x200x3 RGB images and 8-DOF internal proprioceptive state. Proprioceptive state consists of 3D cartesian position and 3D euler orientation of the end effector, as well as 2 gripper angles. In the state experiments, observations consist of the 8D proprioceptive state, the position and euler angle orientation of a movable block, and a continuous 1-d sensor describing: door open amount, drawer open amount, red button pushed amount, blue button pushed amount, green button pushed amount. In both experiments, the agent additionally observes the raw string value of a natural language text channel at each timestep.

K. Action space

We use the same action space as [26]: 8-DOF cartesian position, euler, and gripper angle of the end effector. Similarly, during training we quantize each action element into 256 bins. All stochastic policy outputs are represented as discretized logistic distributions over quantization bins.

L. Play dataset

We use the same play logs collected in [26] as the basis for all relabeled goal image conditioned learning. This consists of ∼7h of play relabeled in to D_{play}: ∼10M short-shorizon windows, each spanning 1-2 seconds.

M. (Play, language) dataset

We pair 10K windows from D_{play} with natural language hindsight instructions using the interface shown in Fig. 11 to obtain $D_{(\text{play},\text{lang})}$. See real examples below in Table V. Note the language collected has significant diversity in length and content.

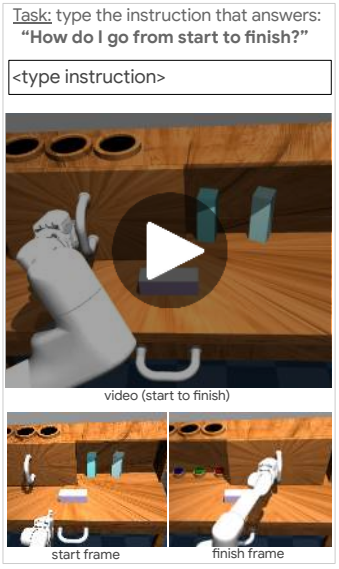

Fig. 11: A schematic rendering of our hindsight language collection tool.

Fig. 11 demonstrates how hindsight instructions are collected. Overseers are presented with a webpage that contains a looping video clip of 1 to 2 seconds of play, along with the first and the last frames of that video, to help them identify the beginning and the end of the clip if necessary. Overseers are asked to type in a text box the sentence that best answers the question "How do I go from start to finish?". Several considerations can go into the design of this interface. First, we can ask overseers to type in multiple instructions that are as different from each other as possible, to increase generalization and diversity of the language dataset. After experimenting with one or multiple text boxes, we opted for using only one, while asking users to write as diverse sentences as possible throughout the collection process. A disadvantage of having multiple boxes is that it can sometimes be challenging to come up with diverse sentences when the observed action is simple. It also leads to less video-language pairs for the same budget. Thus we decided one sentence per video was most beneficial in our case. Another collection consideration is the level of details in a sentence. For example, for a generalist robot application, it seems "open the drawer" is a more likely use case than "move your hand 10 cm to the right above the drawer handle, then grasp the drawer handle, then pull on the drawer handle". Similarly, an instruction geared toward a function such as "open the drawer" is more likely useful than one detached from it function, e.g. "put your fingers around the drawer handle, pull your hand back". Finally, given the temporal horizon of our video clips, multiple things can be happening within one clip. How many events should be described? For example it might be important to describe multiple motions such as "close the drawer then reach for the object". With all these observations in mind, we instructed the overseers to only describe the main actions, while asking themselves: "which instructions would I give to another human being so that they could produce a similar video if they were in that scene?".

N. (Demo, language) dataset

To define $D_{(\text{demo,lang})}$, we pair each of the 100 demonstrations of the 18 evaluation tasks from [26] with hindsight instructions using the same process as in IV-A. We similarly pair the 20 test time demonstrations of each of the 18 tasks. See examples for each of the tasks in Table VI. 74.3% of the natural language instructions in the test dataset appear at least once in D_{play}. 85.08% of the instructions in $D_{(\text{play,lang})}$ never never appear in the test set.

O. Restricted play dataset

For a controlled comparison between LangLfP and LangBC, we train on a play dataset restricted to the same size as the aggregate multitask demonstration dataset ($\sim$1.5h). This was obtained by randomly subsampling the original play logs $\sim$7h to $\sim$1.5h before relabeling.

Below we describe our various baselines and their training sources. Note that for a controlled comparison, all methods are trained using the exact same architecture described in Sec. B, differing only on the source of data.

"grasp the object from the drawer and drop it on the table"
"pick the object and then lift it up."
"pull the drawer."
"drag the object into the drawer"
"drop the object,and again pickup the object high"
"close the drawer"
"do nothing."
"move your hand to the right"
"push the door to the right."
"reach for the green button"
"slightly close your hand."
"go grasp the door handle"
"close the drawer"
"drop the object and push it inside the shelf"
"grasp the block and drop it on top of the table"
"place the block on top of the table"
"rotate the block 90 degrees to the right"
"move the cabinet door to the right and then release your fingers."
"open the drawer"
"slightly close the drawer, then drag it a little bit , close it and then let go"
"reach for the object."
"press the red button and then release it"
"move your hand towards the door handle"
"drop the object"
"slightly pull the drawer and then reach for its handle."
"drop the object in the drawer"
"release the block and move your hand upwards."
"drop the object on the table and reach for the door handle"
"push the object into to the drawer"
"slightly move the door to the right, then drop the object and again turn the object to the left"
"slowly move the door to the left"

TABLE V: Randomly sampled examples from the training set of hindsight instructions paired with play. As hindsight instruction pairing sits on top of play, the language we collect is similarly not constrained by predefined tasks, and instead covers both specific functional behaviors and general, non task-specific behaviors.

Method	Training Data
LangBC	$D_{(\text{demo,lang})}$
LfP (goal image)	D_{play}
LangLfP Restricted	restricted D_{play}, $D_{(\text{play,lang})}$
LangLfP (ours)	D_{play}, $D_{(\text{play,lang})}$
TransferLangLfP (ours)	D_{play}, $D_{(\text{play,lang})}$

TABLE VII: Methods and their training data sources. All baselines are trained to maximize the same generalized contextual imitation objective MCIL.

Task construction. We construct long-horizon evaluations by considering transitions between the 18 subtasks defined in [26]. These span multiple diverse families, e.g. opening and closing doors and drawers, pressing buttons, picking and placing a movable block, etc. For example, one of the 925 Chain-4 tasks may be "open_sliding, push_blue, open_drawer, sweep". We exclude as invalid any transitions that would result in instructions that would have necessarily already been satisfied, e.g. "open_drawer, press_red, open_drawer".

Note this multi-stage scenario subsumes the original Multi-18 in Lynch et al. [26], which can be seen as just testing the first stage. This allows for direct comparison to prior work.

To allow natural language conditioning we pair 20 test demonstrations of each subtask with human instructions using the same process as in Sec. IV-A (example sentences in Table VI).

Success is reported with confidence intervals over 3 seeded

Task	Natural language instructions
open sliding door	"move the door all the way to the right" "slide the door to the right" "move the sliding door all the way to the right and let go"
close sliding door	"Grasp the door handle, then slide the door to the left" "move the door all the way to the left"
open drawer	"'open the cabinet drawer" "open the drawer and let go"
close drawer	"'close the drawer and let go" "'close the drawer"
grasp flat	"Pick up the block" "grasp the object and lift it up" "grasp the object and move your hand up"
grasp lift	"Pick up the object from the drawer and drop it on the table." "hold the block and place it on top of the table"
grasp upright	"Pick the object and lift it up" "grasp the object and lift"
knock	"push the block forward" "push the object towards the door"
pull out shelf	"Drag the block from the shelf towards the drawer" "pick up the object from the shelf and drop it on the table"
put in shelf	"grasp the object and place it inside the cabinet shelf" "Pick the object, move it into the shelf and then drop it."
push red	"go press the red button" "Press the red button"
push green	"press the green button" "push the green button"
push blue	"push the blue button" "press down on the blue button."
rotate left	"Pick up the object, rotate it 90 degrees to the left and drop it on the table" "rotate the object 90 degrees to the left"
rotate right	"turn the object to the right" "rotate the block90 degrees to the right"
sweep	"roll the object into the drawer" "drag the block into the drawer"
sweep left	"Roll the block to the left" "close your fingers and roll the object to the left"
sweep right	"roll the object to the right" "Push the block to the right."

TABLE VI: Example natural language instructions used to specify the 18 test-time visual manipulation tasks.

training runs.

Evaluation walkthrough. The long horizon evaluation happens as follows. For each of the N-stage tasks in a given benchmark, we start by resetting the scene to a neutral state. This is discussed more in the next section. Next, we sample a natural language instruction for each task in the N-stage sequence from a test set. Next, for each subtask in a row, we condition the agent on the current subtask instruction and rollout the policy. If at any timestep the agent successfully completes the current subtask (according to the environment-defined reward), we transition to the next subtask in the sequence (after a half second delay). This attempts to mimic the qualitative scenario where a human provides one instruction after another, queuing up the next instruction, then entering it only once the human has deemed the current subtask solved. If the agent does not complete a given subtask in 8 seconds, the entire episode ends in a timeout. We score each multi-stage rollout by the percent of subtasks completed, averaging over all multi-stage tasks in the benchmark to arrive at the final N-stage number.

The importance of neutral starting conditions in multitask evaluation. When evaluating any context conditioned policy (goal-image, task id, natural language, etc.), a natural question that arises is: how much is the policy relying on the context to infer and solve the task? In [26], evaluation began by resetting the simulator to the first state of a test demonstration, ensuring that the commanded task was valid. We find that under this reset scheme, the initial pose of the agent can in some cases become correlated with the task, e.g. the arm starts nearby the drawer for a drawer opening task. This is problematic, as it potentially reveals task information to the policy through the *initial state*, rather than the context.

In this work, we instead reset the arm to a fixed neutral position at the beginning of every test episode. This "neutral" initialization scheme is used to run all evaluations in this paper. We note this is a fairer, but significantly more difficult benchmark. Neutral initialization breaks correlation between initial state and task, forcing the agent to rely entirely on language to infer and solve the task. For completeness, we present evaluation curves for both LangLfP and LangBC under this and the prior possibly "correlated" initialization scheme in Fig. 12. We see that when LangBC is allowed to start from the exact first state of a demonstration (a rigid real world assumption), it does well on the first task, but fails to transition to the other tasks, with performance degrading quickly after the first instruction (Chain-2, Chain-3, Chain-4). Play-trained LangLfP on the other hand, is far more robust to change in starting position, experiencing only a minor degradation in performance when switching from the correlated initialization to neutral.

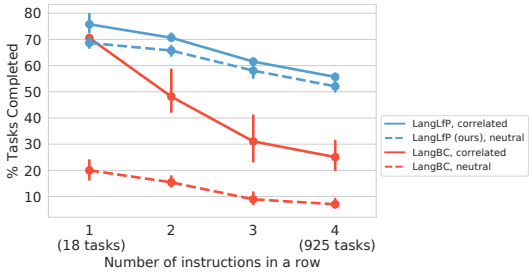

Fig. 12: **Training on relabeled play leads to robustness.** Models trained on relabeled play (LangLfP) are robust to a fixed neutral starting position during multitask evaluation, models trained on conventional predefined demonstrations (LangBC) are not.

P. Composing new tasks with language

In this section, we show that LangLfP can achieve difficult new tasks, not included in the standard 18-task benchmark, in

zero shot. In Fig. 13, the operator can compose a new task "put object in trash" at test time, simply by breaking the task into two text instructions: 1) "pick up the object" and 2) "put the object in the trash". We see similarly that the operator can ask the agent to put the object on top of a shelf in Fig. 14. Note that neither "put on shelf" nor "put in trash" are included in the 18-task benchmark.

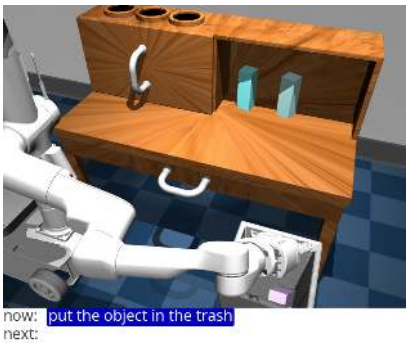

Fig. 13: **Putting the object in the trash:** An operator is able to put the object in the trash by breaking down the task into 2 smaller subtasks with the following sentences; 1) "pick up the object" 2) "put the object in the trash"

Fig. 14: **Putting the object on top of the shelf:** An operator is able to put the object in the trash by breaking down the task into 2 smaller subtasks with the following sentences; 1) "pick up the object" 2) "put the object on top of the shelf".

Q. Play scales with model capacity

In Fig. 6, we consider task performance as a function of model size for models trained on play or conventional predefined demonstrations. For fair comparison, we compare LangBC to Restricted LangLfP, trained on the same amount of data. We study both models from state inputs to understand upper bound performance. As we see, performance steadily improves as model size is increased for models trained on play, but peaks and then declines for models trained on conventional demonstrations. Our interpretation is that larger capacity is effectively utilized to absorb the diversity in play, whereas additional capacity is wasted on equivalently sized datasets constrained to predefined behaviors. This suggests that the simple recipe of collecting large play datasets and scaling up model capacity to achieve higher performance is a valid one.

R. The Operator Can Help the Robot

In this section, we show an example of the human operator adapting its instructions if the robot gets stuck, and adding extra sub-steps to get out of trouble and fix the situation in order to achieve the initial instruction. In Fig. 7, the robot gets its end-effector stuck against the door on the way to pressing the red button. The operator then changing the command to move back, then opening the door more to the right so that it has better access to the red button, then pressing the red button.

S. Pretrained language models give robustness to synonym instructions

Robustness to out of distribution synonym instructions. To study whether our proposed transfer augmentation allows an agent to follow out-of-distribution synonyms, we replace one or more words in each Multi-18 **evaluation** instruction with a synonym outside training, e.g. "drag the block from the shelf" $\rightarrow$ "*retrieve* the *brick* from the *cupboard*". Note these substitutions only happen at test time, not training time. Enumerating all substitutions results in a set of 14,609 out-of-distribution instructions covering all 18 tasks. We evaluate a random policy, LangLfP, and TransferLangLfP on this benchmark, OOD-syn, reporting the results in Table II. Success is reported with confidence intervals over 3 seeded training runs. We see while LangLfP is able to solve some novel tasks by inferring meaning from context (e.g. 'pick up the block' and 'pick up the brick' might reasonably map to the same task for one freely moving object), its performance degrades significantly. TransferLangLfP on the other hand, degrades less. This shows that the simple transfer learning technique we demonstrate greatly magnifies the test time scope of an instruction following agent. We stress that this technique only increases the number of *ways* a language-guided agent can be conditioned at test time to execute the same fixed set of training behaviors.

Following out of distribution instructions in 16 different languages. Here we show that simply choosing a multilingual pretrained language model leads to further robustness across languages not seen during training (e.g. French, Mandarin, German, etc.). To study this, we combine the original set of **evaluation** instructions from Multi-18 with the expanded synonym instruction set OOD-syn, then translate both into 16 languages using the Google translate API. This results in $\sim$240k out-of-distribution instructions covering all 18 tasks. We evaluate the previous methods on this cross-lingual manipulation benchmark, OOD-16-lang, reporting success in Table II. We see that when LangLfP receives instructions with no training overlap, it resorts to producing maximum likelihood play actions. This results in some spurious task success, but performance degrades materially. TransferLangLfP, on the other hand, is far more robust to these perturbations, degrading only slightly from the english-only benchmark. See videos of TransferLangLfP following instructions in 16 novel languages. While in this paper we do not finetune the pretrained embeddings with our imitation loss, an exciting direction for future work would be to see if grounding language embeddings in embodied imitation improves representation learning over text-only inputs.

We study how the performance of LangLfP scales with the number of collected language pairs. Fig. 15 compares models trained from pixel inputs to models trained from state inputs as we sweep the amount of collected language pairs by factors of 2. Success is reported with confidence intervals over 3 seeded training runs. Interestingly, for models trained from states, doubling the size of the language dataset from 5k to 10k has marginal benefit on performance. Models trained from pixels, on the other hand, have yet to converge and may benefit from even larger datasets. This suggests that the role of larger language pair datasets is primarily to address the complicated perceptual grounding problem.

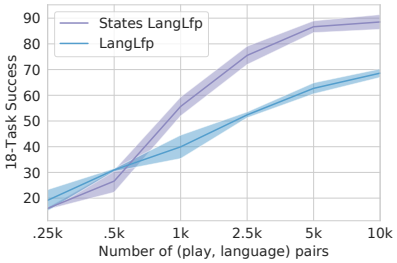

Fig. 15: **18-task success vs amount of human language pairs collected.**

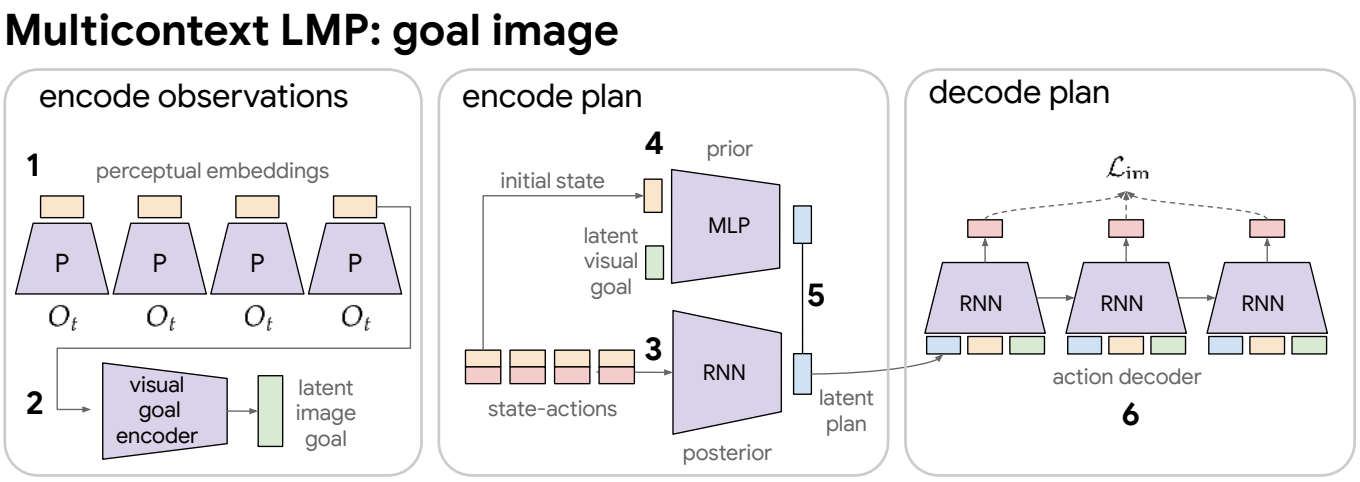

Fig. 16: **Latent image goal LMP**: This image goal conditioned forward pass of multicontext LMP. 1) Map sequence of raw observations O_t to perceptual embeddings (see Fig. 9). 2) Map image goal to latent goal space. 3) Map full state-action sequence to recognized plan through posterior. 4) Map initial state and latent goal through prior network to distribution over plans for achieving goal. 5) Minimize KL divergence between posterior and prior. 6) Compute maximum likelihood action reconstruction loss by decoding plan into actions with teacher forcing. Each action prediction is conditioned on current perceptual embedding, latent image goal, and latent plan. Note perception net, posterior, prior, and policy are shared with language goal forward pass.

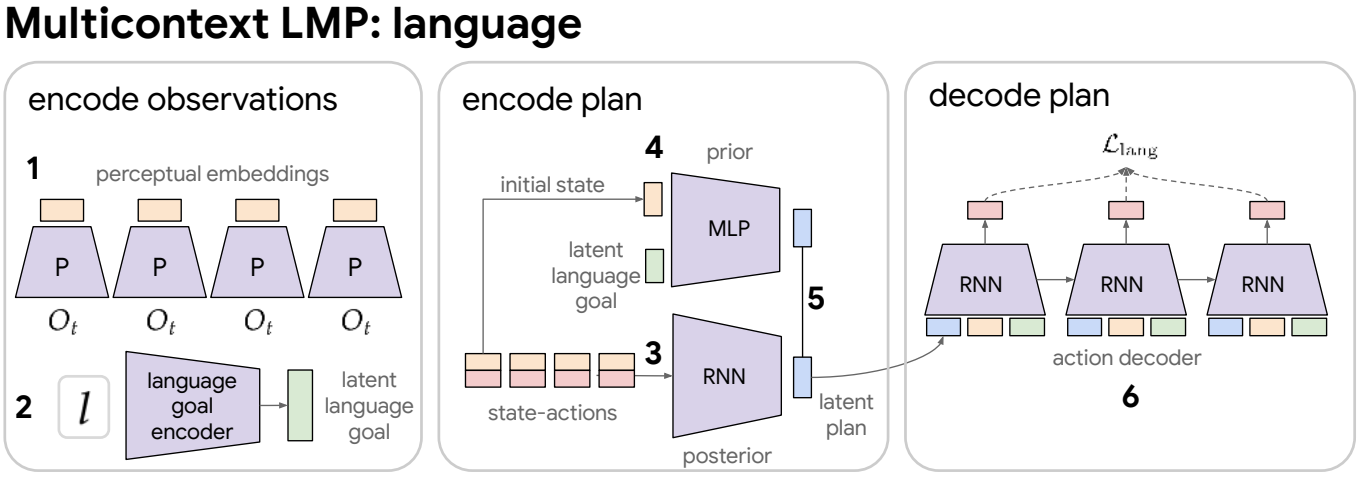

Fig. 17: **Latent language goal LMP**: This describes the language goal conditioned forward pass of multicontext LMP. 1) Map sequence of raw observations O_t to perceptual embeddings (see Fig. 9). 2) Map language observation to latent goal space. 3) Map full state-action sequence to recognized plan through posterior. 4) Map initial state and latent goal through prior network to distribution over plans for achieving goal. 5) Minimize KL divergence between posterior and prior. 6) Compute maximum likelihood action reconstruction loss by decoding plan into actions with teacher forcing. Each action prediction is conditioned on current perceptual embedding, latent language goal, and latent plan. Note perception net, posterior, prior, and policy are shared with image goal LMP.