PaLM-E: An Embodied Multimodal Language Model

Danny Driess [1 2] **Fei Xia** [1] **Mehdi S. M. Sajjadi** [3] **Corey Lynch** [1] **Aakanksha Chowdhery** [3]
Brian Ichter [1] **Ayzaan Wahid** [1] **Jonathan Tompson** [1] **Quan Vuong** [1] **Tianhe Yu** [1] **Wenlong Huang** [1]
Yevgen Chebotar [1] **Pierre Sermanet** [1] **Daniel Duckworth** [3] **Sergey Levine** [1] **Vincent Vanhoucke** [1]
Karol Hausman [1] **Marc Toussaint** [2] **Klaus Greff** [3] **Andy Zeng** [1] **Igor Mordatch** [3] **Pete Florence** [1]

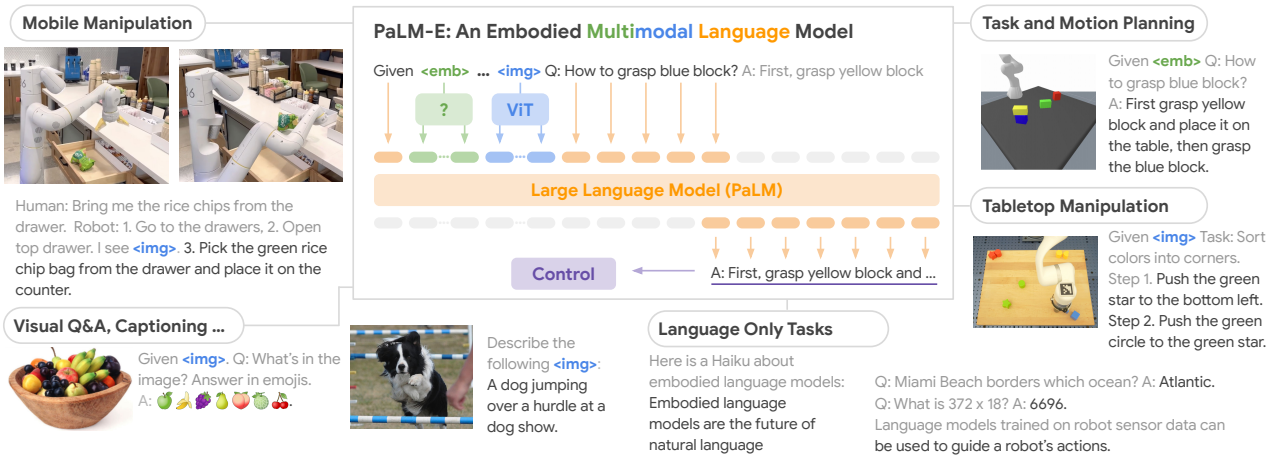

Figure 1: PaLM-E is a single general-purpose multimodal language model for embodied reasoning tasks, visual-language tasks, and language tasks. PaLM-E *transfers* knowledge from visual-language domains into embodied reasoning – from robot planning in environments with complex dynamics and physical constraints, to answering questions about the observable world. PaLM-E operates on *multimodal sentences*, i.e. sequences of tokens where inputs from arbitrary modalities (e.g. images, neural 3D representations, or states, in green and blue) are inserted alongside text tokens (in orange) as input to an LLM, trained end-to-end.

Abstract

Large language models excel at a wide range of complex tasks. However, enabling general inference in the real world, e.g. for robotics problems, raises the challenge of grounding. We propose embodied language models to directly incorporate real-world continuous sensor modalities into language models and thereby establish the link between words and percepts. Input to our embodied language model are multimodal sentences that interleave visual, continuous state estimation, and textual input encodings. We train these encodings end-to-end, in conjunction with a pre-trained large language model, for multiple embodied tasks including sequential robotic manipulation planning, visual question answering, and captioning. Our evaluations show that PaLM-E, a single large embodied multimodal model, can address a variety of embodied reasoning tasks, from a variety of observation modalities, on multiple embodiments, and further, exhibits positive transfer: the model benefits from diverse joint training across internet-scale language, vision, and visual-language domains. Our largest model with 562B parameters, in addition to being trained on robotics tasks, is a visual-language generalist with state-of-the-art performance on OK-VQA, and retains generalist language capabilities with increasing scale.

1. Introduction

Large language models (LLMs) demonstrate strong reasoning capabilities across various domains, including dialogue (Glaese et al., 2022; Thoppilan et al., 2022), step-by-step reasoning (Wei et al., 2022; Kojima et al., 2022), math problem solving (Lewkowycz et al., 2022; Polu et al., 2022), and code writing (Chen et al., 2021a). However, a limitation of such models for inference in the real world is the issue of grounding: while training LLMs on massive textual data

[1]Robotics at Google [2]TU Berlin [3]Google Research. Correspondence to: Danny Driess <danny.driess@gmail.com>, Pete Florence <peteflorence@google.com>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

may lead to representations that relate to our physical world, *connecting* those representations *to* real-world visual and physical sensor modalities is essential to solving a wider range of *grounded* real-world problems in computer vision and robotics (Tellex et al., 2020). Previous work (Ahn et al., 2022) interfaces the output of LLMs with learned robotic policies and affordance functions to make decisions, but is limited in that the LLM itself is only provided with textual input, which is insufficient for many tasks where the spatial layout of the scene is important. Further, in our experiments we show that current state-of-the-art *visual*-language models trained on typical vision-language tasks such as visual-question-answering (VQA) cannot directly solve robotic reasoning tasks.

In this paper we propose embodied language models, which directly incorporate continuous inputs from sensor modalities of an embodied agent and thereby enable the language model *itself* to make more grounded inferences for sequential decision making in the real world. Inputs such as images and state estimates are embedded into the same latent embedding as language tokens and processed by the self-attention layers of a Transformer-based LLM in the same way as text. We start from a pre-trained LLM in which we inject the continuous inputs through an encoder. These encoders are trained end-to-end to output sequential decisions in terms of natural text that can be interpreted by the embodied agent by conditioning low-level policies or give an answer to an embodied question. We evaluate the approach in a variety of settings, comparing different input representations (e.g. standard vs. object-centric ViT encodings for visual input), freezing vs. finetuning the language model while training the encoders, and investigating whether co-training on multiple tasks enables transfer.

The approach enables a broad set of capabilities, as we demonstrate on three robotic manipulation domains (two of which are closed-loop in the real-world) and a set of standard visual-language tasks such as VQA and image captioning, while simultaneously retaining the strong pure-language abilities of PaLM. Our results indicate that multi-task training improves performance compared to training models on individual tasks. We show that this *transfer* across tasks can lead to high data-efficiency for robotics tasks, e.g. significantly increasing learning success from handfuls of training examples, and even demonstrating one-shot or zero-shot generalization to novel combinations of objects or unseen objects.

We scale PaLM-E up to 562B parameters, integrating the 540B PaLM (Chowdhery et al., 2022) LLM and the 22B Vision Transformer (ViT) (Dehghani et al., 2023) into, to our knowledge, the largest vision-language model currently reported. PaLM-E-562B achieves state-of-the-art performance on the OK-VQA (Marino et al., 2019) benchmark, without relying on task-specific finetuning. Although not the focus of our experimentation, we also find (Fig. 6) that PaLM-E-562B exhibits a wide array of capabilities including zero-shot multimodal chain-of-thought reasoning, few-shot prompting, and multi-image reasoning, despite being trained on only single-image examples.

To summarize our main contributions, we (1) propose the methodological idea to train a generalist vision, language, and robotics model that addresses robotics tasks through vision-language modeling. We also (2) demonstrate the novel scientific result of demonstrating *positive transfer* across both vision and language into robotics tasks, which is enabled by the methodological idea mentioned prior. In studying how to best train such models, we (3) introduce novel architectural ideas such as neural scene representations and entity-labeling multimodal tokens. In addition to our focus on PaLM-E as an embodied reasoner we (4) show that PaLM-E is also a quantitatively competent vision and language generalist, and (5) demonstrate that scaling the language model size enables multimodal finetuning with less catastrophic forgetting.

2. Related Work

General vision-language modeling. Building on successes in large language (Brown et al., 2020; Devlin et al., 2018) and vision (Dosovitskiy et al., 2020) models, recent years have seen a growing interest in large vision-language models (VLMs) (Li et al., 2019; Lu et al., 2019; Hao et al., 2022; Gan et al., 2022). Unlike their predecessors, VLMs are capable of simultaneously understanding both images and text, and can be applied to tasks such as visual question answering (Zhou et al., 2020; Zellers et al., 2021b), captioning (Hu et al., 2022), optical character recognition (Li et al., 2021), and object detection (Chen et al., 2021b). The methods by which images are integrated varies. For example, Alayrac et al. (2022) introduces cross-attention layers to fuse images into a pretrained language model. In contrast, PaLM-E represents images and text as "multimodal sentences" where both image and text tokens are input to the self-attention layers of the language model. This also allows it to process multiple images in a flexible way within any part of a sentence. More closely related to our work is Frozen (Tsimpoukelli et al., 2021) where vision encoder parameters are optimized via backpropagation through a frozen LLM (Lu et al., 2021). Inspired by this work, we investigate the design in a broader scope by introducing alternative input modalities (e.g. neural scene representations), and our proposed approach empirically outperforms Frozen by more than 45% on the VQAv2 benchmark. More importantly we demonstrate that PaLM-E is applicable not only to perceptual but also embodied tasks.

Actions-output models. Prior works focus on combining vision and language inputs in an embodied setting with the

goal of direct action prediction (Guhur et al., 2022; Shridhar et al., 2022b;a; Zhang & Chai, 2021; Silva et al., 2021; Jang et al., 2022; Nair et al., 2022; Lynch et al., 2022; Brohan et al., 2022). Among these methods, VIMA (Jiang et al., 2022) explores multimodal prompts similar to PaLM-E. The role of language is perhaps most aptly described as task specification in these works. In contrast, PaLM-E generates high-level instructions as text; in doing so, the model is able to naturally condition upon its own predictions and directly leverage the world knowledge embedded in its parameters. This enables not only embodied reasoning but also question answering, as demonstrated in our experiments. Among works that output actions, perhaps most similar is the approach proposed in Gato (Reed et al., 2022) which, like PaLM-E, is a generalist multi-embodiment agent. In contrast to Gato, we demonstrate positive transfer across different tasks where the model benefits from diverse joint training across multiple domains.

LLMs in embodied task planning. There have been several methods proposed to leverage LLMs in embodied domains. While many works focus on understanding natural language *goals* (Lynch & Sermanet, 2020; Shridhar et al., 2022a; Nair et al., 2022; Lynch et al., 2022), fewer consider natural language as a representation for *planning* – the focus of this work. LLMs contain vast amounts of internalized knowledge about the world (Bommasani et al., 2021), but without grounding, generated plans may be impossible to execute. One line of research has employed prompting to elicit a sequence of instructions directly from an LLM either by leveraging semantic similarity between an LLM's generation and an eligible set of instructions (Huang et al., 2022b), incorporating affordance functions (Ahn et al., 2022), visual feedback (Huang et al., 2022c), generating world models (Nottingham et al., 2023; Zellers et al., 2021a), planning over graphs and maps (Shah et al., 2022; Huang et al., 2022a), visual explanations (Wang et al., 2023), program generation (Liang et al., 2022; Singh et al., 2022), or injecting information into the prompt (Zeng et al., 2022). In contrast, PaLM-E is trained to generate plans directly without relying on auxiliary models for grounding. This in turn enables direct integration of the rich semantic knowledge stored in pretrained LLMs into the planning process.

With few exceptions, the parameters of the LLMs employed in many of these works are employed as-is without further training. In LID (Li et al., 2022), this constraint is relaxed and LLM parameters are finetuned to produce a planning network for generating high-level instructions. $(SL)^3$ (Sharma et al., 2021) tackles the more challenging task of simultaneously finetuning two LLMs: a planning network, which produces high-level instructions, and a low-level policy network, which selects actions. With PaLM-E, our interests are distinct and complementary: we investigate a generalist, multi-embodiment model, across multiple modalities.

3. Background on Large Language Models

Decoder-only LLMs. Decoder-only large language models (LLMs) are generative models trained to predict the probability $p(w_{1:L})$ of a piece of text $w_{1:L} = (w_1, \ldots, w_L)$ that is represented as a sequence of tokens $w_i \in \mathcal{W}$. Typical neural architectures realize this by factorizing into

$$p(w_{1:L}) = \prod_{l=1}^{L} p_{\text{LM}}(w_l | w_{1:l-1}), \tag{1}$$

where p_{LM} is a large transformer network.

Prefix-decoder-only LLMs. Since the LLM is autoregressive, a pre-trained model can be conditioned on a prefix $w_{1:n}$ without the necessity to change the architecture

$$p(w_{n+1:L} | w_{1:n}) = \prod_{l=n+1}^{L} p_{\text{LM}}(w_l | w_{1:l-1}). \tag{2}$$

The prefix or *prompt* $w_{1:n}$ provides the context based on which the LLM continues to predict the subsequent tokens $w_{n+1:L}$. This is often used for inference to steer the predictions of the model. For example, the prompt can contain a description of the task the LLM should solve or examples of desired text completions for similar tasks.

Token embedding space. The tokens w_i are elements of a fixed vocabulary $\mathcal{W}$ which is a discrete, finite set corresponding to (sub)words in natural language. Internally, the LLM embeds w_i into a word token embedding space $\mathcal{X} \subset \mathbb{R}^k$ via $\gamma : \mathcal{W} \to \mathcal{X}$, i.e. $p_{\text{LM}}(w_l | x_{1:l-1})$ with $x_i = \gamma(w_i) \in \mathbb{R}^k$. The mapping γ is typically represented as a large embedding matrix of size $k \times |\mathcal{W}|$ and trained end-to-end. In our case, $|\mathcal{W}| = 256\,000$ (Chowdhery et al., 2022).

4. Methodology: An Embodied Multimodal Language Model

The main architectural idea of PaLM-E is to inject continuous, embodied observations such as images, state estimates, or other sensor modalities into the language embedding space of a pre-trained language model. This is realized by encoding the continuous observations into a sequence of vectors with the same dimension as the embedding space of the language tokens. The continuous information is hence injected into the language model in an analogous way to language tokens. PaLM-E is a decoder-only LLM that generates textual completions autoregressively given a prefix or prompt. We call our model PaLM-**E**, since we use PaLM (Chowdhery et al., 2022) as the pre-trained language model, and make it **E**mbodied.

The *inputs* to PaLM-E consist of text and (multiple) continuous observations. The multimodal tokens corresponding to these observations are interleaved with the text

to form *multimodal sentences*. An example of such a multimodal sentence is `Q: What happened between <img_1> and <img_2>?` where `<img_i>` represents an embedding of an image. The *output* of PaLM-E is text generated auto-regressively by the model, which could be an answer to a question, or a sequence of decisions produced by PaLM-E in textual form that should be executed by a robot. When PaLM-E is tasked with producing decisions or plans, we assume that there exists a low-level policy or planner that can translate these decisions into low-level actions. Prior work has discussed a variety of ways to train such low-level policies (Lynch & Sermanet, 2020; Brohan et al., 2022), and we use these prior methods directly without modification. In the following, we describe our approach more formally.

Multimodal sentences: injection of continuous observations. Multimodal information such as image observations can be injected into the LLM by skipping the discrete token level and directly mapping the continuous observations into the language embedding space $\mathcal{X}$. To this end, we train an encoder $\phi : \mathcal{O} \to \mathcal{X}^q$ that maps a (continuous) observation space $\mathcal{O}$ (refer to Sec. 4.1 for details) into a *sequence* of q-many vectors in $\mathcal{X}$. These vectors are then interleaved with normal embedded text tokens to form the prefix for the LLM. This means that each vector x_i in the prefix is formed from either the word token embedder γ or an encoder ϕ_i:

$$x_i = \begin{cases} \gamma(w_i) & \text{if } i \text{ is a text token, or} \\ \phi_j(O_j)_i & \text{if } i \text{ corresponds to observation } O_j. \end{cases} \quad (3)$$

Note that a single observation O_j is usually encoded into multiple embedding vectors. It is possible to interleave different encoders ϕ_i at different locations in the prefix to combine, e.g., information from different observation spaces. Injecting the continuous information this way into the LLM reuses its existing positional encodings. In contrast to other VLM approaches (e.g, (Chen et al., 2022)), the observation embeddings are not inserted at fixed positions, but instead placed dynamically within the surrounding text.

Embodying the output: PaLM-E in a robot control loop. PaLM-E is a generative model producing text based on multimodal sentences as input. In order to connect the output of the model to an embodiment, we distinguish two cases. If the task can be accomplished by outputting text only as, e.g., in embodied question answering or scene description tasks, then the output of the model is directly considered to be the solution for the task.

Alternatively, if PaLM-E is used to solve an embodied planning or control task, it generates text that conditions low-level commands. In particular, we assume to have access to policies that can perform low-level skills from some (small) vocabulary, and a successful plan from PaLM-E must consist of a sequence of such skills. Note that PaLM-E must determine on its own which skills are available based on the training data and the prompt, and no other mechanism is used to constrain or filter its outputs. Although these policies are language conditioned, they are not capable of solving long-horizon tasks or taking in complex instructions. PaLM-E is hence integrated into a control-loop, where its predicted decisions are executed through the low-level policies by a robot, leading to new observations based on which PaLM-E is able to replan if necessary. In this sense, PaLM-E can be understood as a high-level policy that sequences and controls the low-level policies.

4.1. Input & Scene Representations for Different Sensor Modalities

In this section, we describe the individual modalities that we incorporate into PaLM-E, and how we set up their encoders. We propose different architectural choices for each encoder $\phi : \mathcal{O} \to \mathcal{X}$ to map the corresponding modality into the language embedding space. We investigate state estimation vectors, Vision Transformers (ViTs) (Dosovitskiy et al., 2020; Chen et al., 2022; Ryoo et al., 2021) for 2D image features, and the 3D-aware Object Scene Representation Transformer (OSRT) (Sajjadi et al., 2022a). In addition to encoders that represent the input scene globally, we consider object-centric representations that factor observations into tokens that represent individual objects in the scene.

State estimation vectors. State vectors, e.g. from a robot or a state estimate for objects, are perhaps the simplest to input into PaLM-E. Let $s \in \mathbb{R}^S$ be a vector describing the state of the objects in a scene. For example, s could contain the pose, size, color etc. of those objects. Then, the MLP ϕ_{state} maps s into the language embedding space.

Vision Transformer (ViT). ViT $\tilde{\phi}_{\text{ViT}}$ (Dosovitskiy et al., 2020) is a transformer architecture mapping an image I into a number of token embeddings $\tilde{x}_{1:m} = \tilde{\phi}_{\text{ViT}}(I) \in \mathbb{R}^{m \times \tilde{k}}$. We consider several variants, including the 4 billion parameter model from Chen et al. (2022), which we refer to as ViT-4B, and a similar 22 billion parameter model, ViT-22B (Dehghani et al., 2023), both of which have been pre-trained on image classification. We further investigate the ViT token learner architecture (ViT + TL) (Ryoo et al., 2021) which is trained end-to-end from scratch. Note that the dimensionality $\tilde{k}$ of the ViT embeddings is not necessarily the same as that of the language model. We therefore project each embedding into $x_i = \phi_{\text{ViT}}(I)_i = \psi(\tilde{\phi}_{\text{ViT}}(I)_i)$ with ψ being a learned affine transformation.

Object-centric representations. Unlike language, visual input is not pre-structured into meaningful entities and relationships: while ViT may capture semantics, the structure of the representation resembles a static grid rather than a collection of object instances. This poses a challenge both for interfacing with LLMs which have been pre-trained on symbols, and for solving embodied reasoning which requires interaction with physical objects. We therefore also explore

structured encoders that aim to separate visual inputs into distinct objects before injecting them into the LLM. Given ground-truth object instance masks M_j, we can decompose ViT's representation into $x^j_{1:m} = \phi_{\text{ViT}}(M_j \circ I)$ for object j.

Object Scene Representation Transformer (OSRT). An alternative that does not require ground-truth segmentations is OSRT (Sajjadi et al., 2022a): rather than relying on external knowledge about objects, they are discovered in an unsupervised way through inductive biases in the architecture (Locatello et al., 2020). Based on SRT (Sajjadi et al., 2022b), OSRT learns 3D-centric neural scene representations through a novel view synthesis task. Its scene representations consist of object slots $o_j = \bar{\phi}_{\text{OSRT}}(I_{1:v})_j \in \mathbb{R}^{\bar{k}}$. We project each of these slots into $x^j_{1:m} = \psi(\bar{\phi}_{\text{OSRT}}(I_{1:v})_j)$ with an MLP ψ. Note that individual objects are always tokenized into *multiple* embeddings each, i.e. $\psi : \mathbb{R}^{\bar{k}} \to \mathbb{R}^{m \times k}$ for OSRT maps into m-many embeddings.

Entity referrals. For embodied planning tasks, PaLM-E must be able to reference objects in its generated plan. In many cases, including the majority of our experiments, objects in a scene can be identified in natural language by some of their unique properties. However, there also exist settings where objects are not easily identifiable by language in few words, e.g. if there are multiple blocks on a table of the same color at different locations. For object-centric representations such as OSRT, we label the multimodal tokens corresponding to an object in the input prompt as follows: `Object 1 is <obj_1>. ... Object j is <obj_j>.` This enables PaLM-E to reference objects via special tokens of the form `obj_j` in its generated output sentences. In this case, we assume that the low-level policies operate on these tokens as well.

4.2. Model Training

PaLM-E is trained on a dataset of the form $D = \left\{\left(I^i_{1:u_i}, w^i_{1:L_i}, n_i\right)\right\}_{i=1}^{N}$, where each example i consists of u_i-many continuous observations I^i_j, a text $w^i_{1:L_i}$, and an index n_i. Despite being a decoder-only model, the text consists of a prefix part up to index n_i that is formed from multimodal sentences, and the prediction target, which only contains text tokens. The loss function is therefore a cross-entropy loss averaged over the individual non-prefix tokens $w^i_{n_i+1:L_i}$. To form the multimodal sentences within the model, we have special tokens in the text that get replaced by the embedding vectors of the encoders at the locations in the text of those tokens. We base PaLM-E on the pre-trained 8B, 62B, and 540B parameter variants of PaLM as the decoder-only LLM into which we inject the continuous observations through the input encoders. Those encoders are either pre-trained or trained from scratch, see Sec. 4.1. We refer to an 8B LLM combined with a 4B ViT as PaLM-E-12B, similarly a 62B LLM + 22B ViT as PaLM-E-84B, and 540B LLM + 22B ViT as PaLM-E-562B.

Variation with Model freezing. Most of our architectures consist of three parts, an encoder $\tilde{\phi}$, a projector ψ, and the LLM p_{LM}. When training PaLM-E, one way is to update the parameters of all these components. However, LLMs show impressive reasoning capabilities if supplied with a suitable prompt (Wei et al., 2022). Therefore, we investigate whether it is possible to *freeze* the LLM and to just train the input encoders, and if so, how different-modality encoders compare. In this case, the encoder has to produce embedding vectors such that the frozen LLM is grounded on the observations, and also propagate information to the LLM about the capabilities of an embodiment. Training such encodings can be understood as a form of input-conditioned soft-prompting (Tsimpoukelli et al., 2021), in relation to normal soft prompts (Lester et al., 2021). In experiments with ϕ_{OSRT}, we also freeze the slot representation, i.e. we only update the small projector ψ which serves as the interface between OSRT and the LLM.

Co-training across tasks. In our experiments, we investigate the effects of co-training our models on a variety of diverse data. The "full mixture", see App. D, consists primarily of a diverse set of internet-scale vision-and-language data, from a variety of tasks. The sampling frequencies are set such that only 8.9% of the full mixture is embodied data, and there are several tasks for each embodiment.

5. Experiments

Our experiments consider diverse robotic (mobile) manipulation tasks across three different robot embodiments, in simulation and with two different real robots. We refer to `https://palm-e.github.io` for videos showing the capabilities of PaLM-E on those tasks. Although not the focus of our work, we evaluate PaLM-E also on general vision-language tasks such as visual-question-answering (VQA), image captioning, and established language tasks.

We split our experimental investigation into two broad categories. First, we compare the different input representations from Sec. 4.1 with respect to performance, generalization, and data-efficiency. The second thread of experiments focuses on one architecture, the main PaLM-E version, consisting of a pre-trained ViT and PaLM LLM that takes in raw images as the continuous inputs. Here we show that a single model, trained on a mixture of many datasets, across diverse tasks, and across robot embodiments, can simultaneously achieve high performance on all of those tasks. Crucially, we investigate whether co-training on these datasets enables *transfer* (Fig. 2): despite different tasks and embodiments, the performance on the individual tasks increases by training on the mixture of tasks. We study the influence on performance, generalization, and data efficiency with respect to co-training strategies and model parameter size. Finally, we consider if freezing the LLM and just training the ViT that injects vision into the LLM is a viable path.

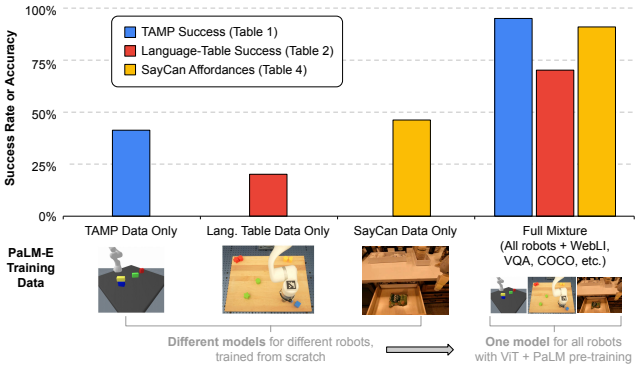

Figure 2: Overview of *transfer* learning demonstrated by PaLM-E: across three different robotics domains, using PaLM and ViT pretraining together with the full mixture of robotics and general visual-language data provides a significant performance increase compared to only training on the respective in-domain data. See Tab. 1, Fig. 3, Tab. 2, Tab. 4 for additional data in each domain.

As baselines, we consider the state-of-the art visual language model PaLI (Chen et al., 2022), which has not been trained on embodiment robot data, as well as the SayCan algorithm (Ahn et al., 2022), supplied with oracle affordances.

5.1. Robot Environments / Tasks

Our three robot environments (Fig. 1) include a Task and Motion Planning (TAMP) domain where a robot has to manipulate (grasp and stack) objects, a table-top pushing environment, and a mobile manipulation domain. In each domain, PaLM-E is trained on expert data from that domain. In many cases, this is a sparse amount of data per task. The TAMP tasks involve large combinatorics over possible plans, and many decision sequences are infeasible. PaLM-E has to generate plans that consist of multiple steps, with complicated decision boundaries. The multi-object tabletop pushing environment is taken from the publicly available Language-Table dataset (Lynch et al., 2022) and is challenging since it includes several objects, large cardinality of language, and complex pushing dynamics. For both the TAMP and Language-Table environment, PaLM-E has to reason about the poses of the objects. It is not sufficient to know which objects are on the table or knowing their rough relationships, the more fine-grained details about the scene geometry are important for solving the tasks. Finally, we consider a mobile manipulation domain similar to SayCan (Ahn et al., 2022), where a robot has to solve a variety of tasks in a kitchen environment, including finding objects in drawers, picking them, and bringing them to a human. For all domains we consider both planning and VQA tasks in those environments. For the mobile manipulation and Language-Table environments, PaLM-E is integrated into the control loop to execute the plans in the real world, and has to adjust the plan in presence of external disturbances or failures of the low-level control policies.

5.2. TAMP Environment

Tab. 8 (appendix) shows planning success rates and VQA performance for the TAMP environment. The LLM is frozen here (for pre-trained LLM). For the results reported in Tab. 8, the input representations are trained on a dataset containing 96,000 training scenes of solely the TAMP environment, i.e. no other data is part of the mixture. When 3-5 objects are in the scene, as in the training set, most input representations perform similarly well. However, when increasing the number of objects, it turns out that using a pre-trained LLM improves performance considerably, especially with entity referrals. Furthermore, a 62B LLM shows better out-of-distribution generalization compared to the 8B variant, while a non-pretrained LLM shows no out-of-distribution generalization. The SayCan baseline (Ahn et al., 2022) utilizes oracle affordance functions and has difficulties solving this environment, since affordance functions only constrain what is possible right now, but are not informative enough for the LLM to construct long-horizon plans in TAMP environments. Additionally, the short-horizon skills (Tab. 8, first row) are not sufficient to solve these tasks.

Tab. 1 shows results for 3-5 objects when training on 1% of the dataset, which corresponds to only 320 examples for each of the two planning tasks. Here we see that there are significant differences between the input representations, especially for the planning tasks. First, pre-training the LLM is beneficial in the low data regime for state inputs. Second, both ViT variants (ViT+TL, ViT-4B) do not perform well in solving the planning tasks for this little data. However, if we co-train on all other robot environments as well as general vision-language datasets (ViT-4B generalist), then the performance of the ViT-4B more than doubles. This shows a significant transfer effect between different robot embodiments and tasks. Finally, using OSRT as the input representation leads to the best performance here, demonstrating the strengths of 3D-aware object representations. We also observe another instance of transfer here: when we remove the TAMP VQA data and only train on the 640 planning tasks examples, there is a (slight) drop in performance. The state-of-the art vision-language model PaLI (Chen et al., 2022) that was not trained on robot data is not able to solve the tasks. We only evaluated it on q_2 (objects left/right/center on the table) and q_3 (vertical object relations), since those most resemble typical VQA tasks.

5.3. Language-Table Environment

Tab. 2 reports success rates on long-horizon tasks from the Language-Table environment (Lynch et al., 2022). PaLM-E is integrated into a control loop that takes as input the long-horizon task and the current image, and outputs an instruction for the low-level policy. We see that joint training on internet-scale vision and language results in a more effective model for robot planning, particularly in the few-shot

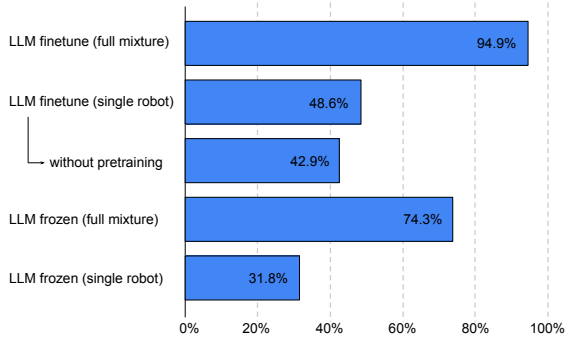

Figure 3: Planning success results in the TAMP environment (1% data) for PaLM-E-12B, comparing of the effects of PaLM-E models (i) using the full training mixture, (ii) pre-training (ViT and PaLM), and (iii) freezing or finetuning the language model. Transfer from full mixture is particularly effective. Note that full mixture contains only 1% of the training data (320 examples each) for the tasks evaluated here. Shown is the mean of tasks p_1, p_2.

	Object-centric	LLM pre-train	Embodied VQA				Planning	
			q_1	q_2	q_3	q_4	p_1	p_2
SayCan (oracle afford.) (Ahn et al., 2022)		✓	-	-	-	-	38.7	33.3
PaLI (zero-shot) (Chen et al., 2022)		✓	-	0.0	0.0	-	-	-
PaLM-E (ours) w/ input enc:								
State	✓(GT)	✗	99.4	89.8	90.3	88.3	45.0	46.1
State	✓(GT)	✓	**100.0**	96.3	95.1	93.1	55.9	49.7
ViT + TL	✓(GT)	✓	34.7	54.6	74.6	91.6	24.0	14.7
ViT-4B single robot	✗	✓	-	45.9	78.4	92.2	30.6	32.9
ViT-4B full mixture	✗	✓	-	70.7	93.4	92.1	74.1	74.6
OSRT (no VQA)	✓	✓	-	-	-	-	71.9	75.1
OSRT	✓	✓	99.7	**98.2**	**100.0**	**93.7**	**82.5**	**76.2**

Table 1: Comparison of different input representations on TAMP environment (in terms of success rates), where data from TAMP constitutes only 1% (i.e., 320 samples for p_1, p_2 each) of total training data. PaLM-E outperforms both PaLI and SayCan. Cross-domain *transfer* is observed, since the PaLM-E with ViT-4B trained on our full data mixture improves planning performance. OSRT, despite using no large-scale data, provides the most effective input encodings for learning. (GT) means ground-truth object-centric information provided. In all experiments, the LLM is frozen. The non-object centric ViT-4B variant utilizes color to reference objects, hence q_1 cannot be evaluated here. The LLM is frozen in these experiments (except for the case where it is not pre-trained). Sec. E.1 describes the tasks q_1-q_4, p_1, q_2.

regime with only 10 demos per task. Scaling the 12B model to the 84B model leads to improvements on 2 of 3 tasks. As with the TAMP environment, neither SayCan nor zero-shot PaLI are effective, unable to solve the easiest task tested.

Real Robot Results and Few-Shot Generalization. In Fig. 7, a), we see PaLM-E is capable of guiding a real robot through a multi-stage tabletop manipulation task, while remaining robust to adversarial disturbances. Given the observed image and a long-horizon goal, e.g. "sort the blocks by colors into corners", PaLM-E outputs language subgoals at 1 Hz to the policies from Lynch et al. (2022), that output low-level robot actions at 5 Hz. Prior work (Lynch et al., 2022) instead involved a human in the loop to interactively guide subgoals and corrections. In Fig. 4, b) we see PaLM-E is capable of one-shot and zero-shot learning. Here, we finetuned PaLM-E on 100 different long horizon tasks with a single training example each, e.g. "put all the blocks in the center", "remove the blue blocks from the line". We additionally see that PaLM-E can generalize zero-shot to tasks involving novel object pairs (Fig. 7, c) and to tasks involving objects that were unseen in either the original robot dataset or the finetuning datasets, e.g. a toy turtle (Fig. 4, d).

5.4. Mobile Manipulation Environment

We demonstrate the performance of PaLM-E on challenging and diverse mobile manipulation tasks. We largely follow the setup in Ahn et al. (2022), where the robot needs to plan a sequence of navigation and manipulation actions based on a human instruction. For example, given the instruction "I spilled my drink, can you bring me something to clean it up?", the robot needs to plan a sequence containing "1. Find a sponge, 2. Pick up the sponge, 3. Bring it to the user, 4. Put down the sponge." Inspired by these tasks, we develop 3 use cases to test the embodied reasoning abilities of PaLM-E: affordance prediction, failure detection, and long-horizon planning. The low-level policies are from RT-1 (Brohan et al., 2022), with RGB image and language instruction as input, and end-effector controls as outputs.

Affordance prediction. We investigate PaLM-E's performance at affordance prediction, i.e. whether a `skill` of the low-level policy can be executed in the current environment. This can be formulated as the VQA problem `Given <img>. Q: Is it possible to <skill> here?`. PaLM-E outperforms PaLI (zero-shot), as well as thresholding on value functions trained with QT-OPT (Tab. 4).

Failure detection. For a robot to do closed-loop planning, it is important to detect failures, as shown in (Huang et al., 2022c). The multimodal prompt is `Given <img>. Q: Was <skill> successful?`. Tab. 4 shows that PaLM-E outperforms PaLI (zero-shot), and a fine-tuned version of CLIP on this data. PaLM-E also outperforms the algorithm of Xiao et al. (2022) that leverages two CLIP models trained with hindsight relabeled data. This method has access to more information than our method, and was specifically designed to just solve failure detection on this dataset.

Real robot results: Long-horizon planning. Finally, we use PaLM-E to perform *embodied planning* end-to-end for mobile manipulation tasks. The prompt structure for this task is `Human: <instruction> Robot: <step history>. I see <img>`. PaLM-E is trained to generate the next step of the plan, conditioned on the history of taken steps and the current image observation of the scene. After each step is decoded, we map them to a low-level policy as defined in Ahn et al. (2022). This process is done in an autoregressive manner, until PaLM-E outputs "terminate". We train the model by using the runs from (Ahn et al., 2022), which contains 2912 sequences. We qualitatively

Zero-shot Baselines						Task 1			Task 2			Task 3		
SayCan (oracle afford.) (Ahn et al., 2022)						0.0			-			-		
PaLI (Chen et al., 2022)						0.0			-			-		
PaLM-E-	trained on	from scratch	LLM+ViT pretrain	LLM frozen	Task finetune	*# Demos* *10*	*20*	*40*	*10*	*20*	*40*	*10*	*20*	*80*
12B	Single robot	✓	✗	n/a	✓	20.0	30.0	50.0	2.5	6.3	2.5	11.3	16.9	28.3
12B	Full mixture	✗	✓	✓	✗	-	-	20.0	-	-	36.3	-	-	29.4
12B	Full mixture	✗	✓	✗	✗	-	-	80.0	-	-	57.5	-	-	50.0
12B	Full mixture	✗	✓	✗	✓	**70.0**	**80.0**	80.0	**31.3**	**58.8**	**58.8**	**57.5**	**54.4**	56.3
84B	Full mixture	✗	✓	✗	✗	-	-	**90.0**	-	-	53.8	-	-	**64.4**

Table 2: Results on planning tasks in the simulated environment from Lynch et al. (2022).

Task 1. Q: There is a block that is closest to {*i.e., top right corner*}. Push that block to the other block of the same color.

Task 2. Q: How to sort the blocks by colors into corners?

Task 3. Q: How to push all the blocks that are on the {*left/right*} side together, without bringing over any of the blocks that are on the {*right/left*} side?

Table 3: Task prompts for Tab. 2.

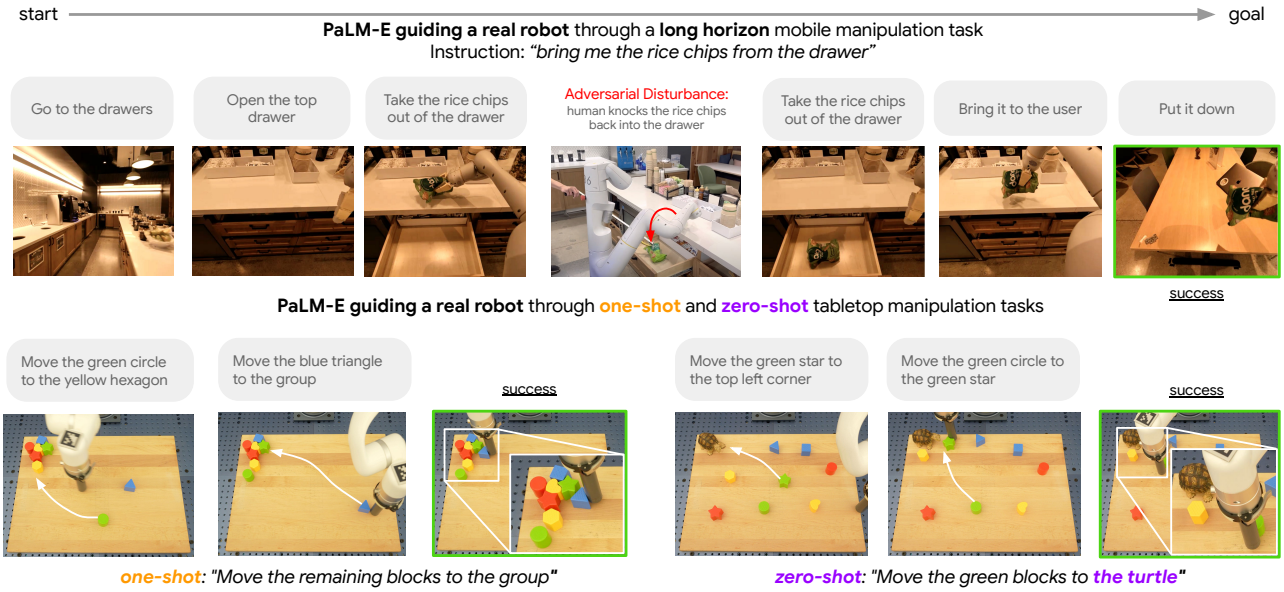

Figure 4: A single PaLM-E model directs the low-level policies of two real robots. Shown is a long-horizon mobile manipulation task in a kitchen, and one-shot / zero-shot generalization with a tabletop manipulation robot.

Baselines				Failure det.	Affordance
PaLI (Zero-shot) (Chen et al., 2022)				0.73	0.62
CLIP-FT (Xiao et al., 2022)				0.65	-
CLIP-FT-hindsight (Xiao et al., 2022)				0.89	-
QT-OPT (Kalashnikov et al., 2018)				-	0.63
PaLM-E-12B trained on	from scratch	LLM+ViT pretrain	LLM frozen		
Single robot	✓	✗	n/a	0.54	0.46
Single robot	✗	✓	✓	**0.91**	0.78
Full mixture	✗	✓	✓	**0.91**	0.87
Full mixture	✗	✓	✗	0.77	**0.91**

Table 4: Mobile manipulation environment: failure detection and affordance prediction (F1 score) in out-of-distribution scenes.

Model	VQAv2 test-dev	VQAv2 test-std	OK-VQA val	COCO Karpathy test
Generalist (one model)				
PaLM-E-12B	76.2	-	55.5	135.0
PaLM-E-562B	80.0	-	**66.1**	138.7
Task-specific finetuned models				
Flamingo (Alayrac et al., 2022)	82.0	82.1	57.8†	138.1
PaLI (Chen et al., 2022)	84.3	84.3	64.5	149.1
PaLM-E-12B	77.7	77.9	60.1	136.0
PaLM-E-66B	-	-	62.9	-
PaLM-E-84B	80.5	-	63.3	138.0
Generalist (one model), with frozen LLM				
(Tsimpoukelli et al., 2021)	48.4	-	-	-
PaLM-E-12B frozen	70.3	-	51.5	128.0

Table 5: Results on general visual-language tasks. For the generalist models, they are the same checkpoint across the different evaluations, while task-specific finetuned models use different-finetuned models for the different tasks. COCO uses Karpathy splits. † is 32-shot on OK-VQA (not finetuned).

evaluated the model in a real kitchen and found the model can carry out long-horizon mobile manipulation tasks, even under adversarial disturbances (Fig. 4).

5.5. Performance on General Visual-Language Tasks

Although it is not the focus of our work, we report in Tab. 5 results on general vision-language tasks, including OK-VQA (Marino et al., 2019), VQA v2 (Goyal et al., 2017) and COCO captioning (Chen et al., 2015). A single, generalist PaLM-E-562B model achieves the highest reported number on OK-VQA, including outperforming models finetuned specifically on OK-VQA. Compared to (Tsimpoukelli et al., 2021), PaLM-E achieves the highest performance on VQA v2 with a frozen LLM to our knowledge. This establishes that PaLM-E is a competitive visual-language generalist, in addition to being an embodied reasoner on robotic tasks.

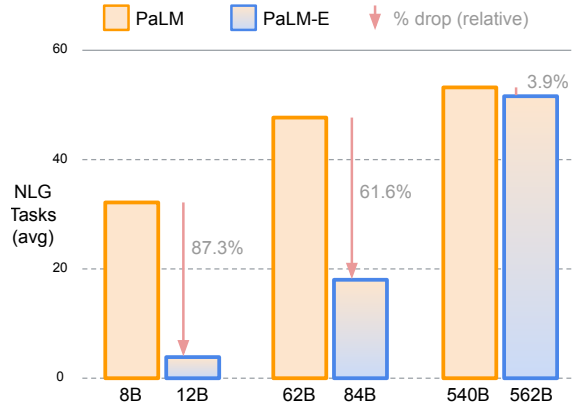

Figure 5: Results on general language tasks (NLG = natural language generation): increasing scale leads to less catastrophic forgetting between a corresponding PaLM-E model and its inherited PaLM model. See full suite of tasks and results in Tab. 10.

5.6. Performance on General Language Tasks

Tab. 10 reports the averaged performance of PaLM-E on 21 general language benchmarks for Natural Language Understanding (NLU) and Natural Language Generation (NLG) tasks. The notable trend is that with increasing model scale, there is considerably less catastrophic forgetting of language capabilities. As seen in Fig. 5, while for the smallest (PaLM-E-12B) model 87.3% of its NLG performance (relative) has degraded during multimodal training, merely 3.9% have been degraded for the largest model (PaLM-E-562B).

6. Summary of Experiments & Discussion

Generalist vs specialist models – transfer. As summarized in Fig. 2, we have shown several instances of *transfer*, meaning that PaLM-E trained on different tasks and datasets at the same time leads to significantly increased performance relative to models trained separately on the different tasks alone. In Fig. 3, co-training on the "full mixture" achieves more than double the performance. In Tab. 11, we see significant improvements in performance if we add LLM/ViT pre-training, and training on the full mixture instead of the mobile manipulation data alone. For the Language-Table experiment in Tab. 2, we observe similar behaviour.

Data efficiency. Compared to available massive language or vision-language datasets, robotics data is significantly less abundant. As discussed in the last paragraph, our model exhibits transfer, which aids PaLM-E to solve robotics tasks from very few training examples in the robotics domain, e.g. between 10 and 80 for Language Table or 320 for TAMP. The OSRT results show another instance of data-efficiency by using a geometric input representation. A promising opportunity for future work is to combine this with a method benefitting from large-scale visual data.

Retaining language capabilities. We have shown two paths to retain the language capabilities of the model during multimodal training. As one option, freezing the LLM and only training the input encoders is a viable path for building embodied language models, although this approach occasionally struggled for robotics tasks (Tab. 2). As an alternative route, when the whole model is trained end-to-end, the model retains significantly more of its original language performance with increasing model scale (Fig. 5).

Limitations and Impact. In App. B and C we discuss technical limitations and considerations for broader impact.

7. Conclusion

We proposed to build an embodied language model by injecting multimodal information such as images into the embedding space of a pre-trained LLM. Experiments showed that off-the-shelf state-of-the-art vision-language models trained on general VQA and captioning tasks are not sufficient for embodied reasoning tasks, as well as limitations of a recent proposal for grounding language models through affordances. To overcome these limitations, we proposed PaLM-E, a single model that is able to control different robots in simulation and in the real world, while at the same time being quantitatively competent at general VQA and captioning tasks. In particular the novel architectural idea of ingesting neural scene representations (i.e., OSRT) into the model is particularly effective, even without large-scale data. PaLM-E is trained on a mixture of diverse tasks across multiple robot embodiments as well as general vision-language tasks. Importantly, we have demonstrated that this diverse training leads to several avenues of *transfer* from the vision-language domains into embodied decision making, enabling robot planning tasks to be achieved data efficiently. While our results indicate that frozen language models are a viable path towards general-purpose embodied multimodal models that fully retain their language capabilities, we have also surfaced an alternative route with unfrozen models: scaling up the language model size leads to significantly less catastrophic forgetting while becoming an embodied agent. Our largest model, PaLM-E-562B, showcases emergent capabilities like multimodal chain-of-thought reasoning, and the ability to reason over multiple images, despite being trained on only single-image prompts.

Acknowledgements

The authors would like to thank, for their advice, help and support: Xi Chen, Etienne Pot, Sebastian Goodman, Maria Attarian, Ted Xiao, Keerthana Gopalakrishnan, Kehang Han, Henryk Michalewski, Mario Lucic, Neil Houlsby, Basil Mustafa, Justin Gilmer, Yonghui Wu, Erica Moreira, Victor Gomes, Tom Duerig, Henning Meyer, and Kendra Byrne. DD and MT acknowledge support by the Deutsche Forschungsgemeinschaft EXC 2002/1, project number 390523135 and the International Max-Planck Research School for Intelligent Systems (IMPRS-IS).

References

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

Alayrac, J.-B., Donahue, J., Luc, P., Miech, A., Barr, I., Hasson, Y., Lenc, K., Mensch, A., Millican, K., Reynolds, M., et al. Flamingo: a visual language model for few-shot learning. *arXiv preprint arXiv:2204.14198*, 2022.

Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., Bernstein, M. S., Bohg, J., Bosselut, A., Brunskill, E., et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

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

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33: 1877–1901, 2020.

Changpinyo, S., Kukliansky, D., Szpektor, I., Chen, X., Ding, N., and Soricut, R. All you may need for vqa are image captions, 2022. URL https://arxiv.org/abs/2205.01883.

Chen, M., Tworek, J., Jun, H., Yuan, Q., Pinto, H. P. d. O., Kaplan, J., Edwards, H., Burda, Y., Joseph, N., Brockman, G., et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021a.

Chen, T., Saxena, S., Li, L., Fleet, D. J., and Hinton, G. Pix2seq: A language modeling framework for object detection. *arXiv preprint arXiv:2109.10852*, 2021b.

Chen, X., Fang, H., Lin, T., Vedantam, R., Gupta, S., Dollár, P., and Zitnick, C. L. Microsoft COCO captions: Data collection and evaluation server. *CoRR*, abs/1504.00325, 2015.

Chen, X., Wang, X., Changpinyo, S., Piergiovanni, A., Padlewski, P., Salz, D., Goodman, S., Grycner, A., Mustafa, B., Beyer, L., et al. Pali: A jointly-scaled multilingual language-image model. *arXiv preprint arXiv:2209.06794*, 2022.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., et al. Palm: Scaling language modeling with pathways. *arXiv preprint arXiv:2204.02311*, 2022.

Dehghani, M., Djolonga, J., Mustafa, B., Padlewski, P., Heek, J., Gilmer, J., Steiner, A., Caron, M., Geirhos, R., Alabdulmohsin, I., et al. Scaling vision transformers to 22 billion parameters. *arXiv preprint arXiv:2302.05442*, 2023.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Driess, D., Ha, J.-S., and Toussaint, M. Deep visual reasoning: Learning to predict action sequences for task and motion planning from an initial scene image. In *Proc. of Robotics: Science and Systems (R:SS)*, 2020.

Gan, Z., Li, L., Li, C., Wang, L., Liu, Z., Gao, J., et al. Vision-language pre-training: Basics, recent advances, and future trends. *Foundations and Trends® in Computer Graphics and Vision*, 14(3–4):163–352, 2022.

Glaese, A., McAleese, N., Trebacz, M., Aslanides, J., Firoiu, V., Ewalds, T., Rauh, M., Weidinger, L., Chadwick, M., Thacker, P., et al. Improving alignment of dialogue agents via targeted human judgements. *arXiv preprint arXiv:2209.14375*, 2022.

Goyal, Y., Khot, T., Summers-Stay, D., Batra, D., and Parikh, D. Making the V in VQA matter: Elevating the role of image understanding in Visual Question Answering. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Guhur, P.-L., Chen, S., Garcia, R., Tapaswi, M., Laptev, I., and Schmid, C. Instruction-driven history-aware policies for robotic manipulations. *arXiv preprint arXiv:2209.04899*, 2022.

Hao, Y., Song, H., Dong, L., Huang, S., Chi, Z., Wang, W., Ma, S., and Wei, F. Language models are general-purpose interfaces. *arXiv preprint arXiv:2206.06336*, 2022.

Hu, X., Gan, Z., Wang, J., Yang, Z., Liu, Z., Lu, Y., and Wang, L. Scaling up vision-language pre-training for image captioning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17980–17989, 2022.

Huang, C., Mees, O., Zeng, A., and Burgard, W. Visual language maps for robot navigation. *arXiv preprint arXiv:2210.05714*, 2022a.

Huang, W., Abbeel, P., Pathak, D., and Mordatch, I. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. *arXiv preprint arXiv:2201.07207*, 2022b.

Huang, W., Xia, F., Xiao, T., Chan, H., Liang, J., Florence, P., Zeng, A., Tompson, J., Mordatch, I., Chebotar, Y., et al. Inner monologue: Embodied reasoning through planning with language models. *arXiv preprint arXiv:2207.05608*, 2022c.

Ilharco, G., Wortsman, M., Wightman, R., Gordon, C., Carlini, N., Taori, R., Dave, A., Shankar, V., Namkoong, H., Miller, J., Hajishirzi, H., Farhadi, A., and Schmidt, L. Openclip, 2021.

Jang, E., Irpan, A., Khansari, M., Kappler, D., Ebert, F., Lynch, C., Levine, S., and Finn, C. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, pp. 991–1002. PMLR, 2022.

Jiang, Y., Gupta, A., Zhang, Z., Wang, G., Dou, Y., Chen, Y., Fei-Fei, L., Anandkumar, A., Zhu, Y., and Fan, L. Vima: General robot manipulation with multimodal prompts. *arXiv preprint arXiv:2210.03094*, 2022.

Kalashnikov, D., Irpan, A., Pastor, P., Ibarz, J., Herzog, A., Jang, E., Quillen, D., Holly, E., Kalakrishnan, M., Vanhoucke, V., et al. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*, pp. 651–673. PMLR, 2018.

Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., and Iwasawa, Y. Large language models are zero-shot reasoners. *arXiv preprint arXiv:2205.11916*, 2022.

Lester, B., Al-Rfou, R., and Constant, N. The power of scale for parameter-efficient prompt tuning. *arXiv preprint arXiv:2104.08691*, 2021.

Lewkowycz, A., Andreassen, A., Dohan, D., Dyer, E., Michalewski, H., Ramasesh, V., Slone, A., Anil, C., Schlag, I., Gutman-Solo, T., et al. Solving quantitative reasoning problems with language models. *arXiv preprint arXiv:2206.14858*, 2022.

Li, L. H., Yatskar, M., Yin, D., Hsieh, C.-J., and Chang, K.-W. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*, 2019.

Li, M., Lv, T., Chen, J., Cui, L., Lu, Y., Florencio, D., Zhang, C., Li, Z., and Wei, F. Trocr: Transformer-based optical character recognition with pre-trained models. *arXiv preprint arXiv:2109.10282*, 2021.

Li, S., Puig, X., Du, Y., Wang, C., Akyurek, E., Torralba, A., Andreas, J., and Mordatch, I. Pre-trained language models for interactive decision-making. *arXiv preprint arXiv:2202.01771*, 2022.

Liang, J., Huang, W., Xia, F., Xu, P., Hausman, K., Ichter, B., Florence, P., and Zeng, A. Code as policies: Language model programs for embodied control. *arXiv preprint arXiv:2209.07753*, 2022.

Locatello, F., Weissenborn, D., Unterthiner, T., Mahendran, A., Heigold, G., Uszkoreit, J., Dosovitskiy, A., and Kipf, T. Object-centric learning with slot attention. *Advances in Neural Information Processing Systems*, 33:11525–11538, 2020.

Lu, J., Batra, D., Parikh, D., and Lee, S. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32, 2019.

Lu, K., Grover, A., Abbeel, P., and Mordatch, I. Pretrained transformers as universal computation engines. *arXiv preprint arXiv:2103.05247*, 1, 2021.

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

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

Marino, K., Rastegari, M., Farhadi, A., and Mottaghi, R. Ok-vqa: A visual question answering benchmark requiring external knowledge. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

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

Nottingham, K., Ammanabrolu, P., Suhr, A., Choi, Y., Hajishirzi, H., Singh, S., and Fox, R. Do embodied agents dream of pixelated sheep?: Embodied decision making using language guided world modelling. *arXiv preprint arXiv:2301.12050*, 2023.

Piergiovanni, A., Kuo, W., and Angelova, A. Pre-training image-language transformers for open-vocabulary tasks, 2022. URL https://arxiv.org/abs/2209.04372.

Polu, S., Han, J. M., Zheng, K., Baksys, M., Babuschkin, I., and Sutskever, I. Formal mathematics statement curriculum learning. *arXiv preprint arXiv:2202.01344*, 2022.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Reed, S., Zolna, K., Parisotto, E., Colmenarejo, S. G., Novikov, A., Barth-Maron, G., Gimenez, M., Sulsky, Y., Kay, J., Springenberg, J. T., et al. A generalist agent. *arXiv preprint arXiv:2205.06175*, 2022.

Ryoo, M. S., Piergiovanni, A., Arnab, A., Dehghani, M., and Angelova, A. Tokenlearner: What can 8 learned tokens do for images and videos? *arXiv preprint arXiv:2106.11297*, 2021.

Sajjadi, M. S. M., Duckworth, D., Mahendran, A., van Steenkiste, S., Pavetić, F., Lučić, M., Guibas, L. J., Greff, K., and Kipf, T. Object Scene Representation Transformer. *NeurIPS*, 2022a. URL https://osrt-paper.github.io/.

Sajjadi, M. S. M., Meyer, H., Pot, E., Bergmann, U., Greff, K., Radwan, N., Vora, S., Lučić, M., Duckworth, D., Dosovitskiy, A., et al. Scene representation transformer: Geometry-free novel view synthesis through set-latent scene representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6229–6238, 2022b.

Shah, D., Osinski, B., Ichter, B., and Levine, S. Lm-nav: Robotic navigation with large pre-trained models of language, vision, and action. *arXiv preprint arXiv:2207.04429*, 2022.

Sharma, P., Ding, N., Goodman, S., and Soricut, R. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of ACL*, 2018.

Sharma, P., Torralba, A., and Andreas, J. Skill induction and planning with latent language. *arXiv preprint arXiv:2110.01517*, 2021.

Shridhar, M., Manuelli, L., and Fox, D. Cliport: What and where pathways for robotic manipulation. In *Conference on Robot Learning*, pp. 894–906. PMLR, 2022a.

Shridhar, M., Manuelli, L., and Fox, D. Perceiver-actor: A multi-task transformer for robotic manipulation. *arXiv preprint arXiv:2209.05451*, 2022b.

Silva, A., Moorman, N., Silva, W., Zaidi, Z., Gopalan, N., and Gombolay, M. Lancon-learn: Learning with language to enable generalization in multi-task manipulation. *IEEE Robotics and Automation Letters*, 7(2):1635–1642, 2021.

Singh, I., Blukis, V., Mousavian, A., Goyal, A., Xu, D., Tremblay, J., Fox, D., Thomason, J., and Garg, A. ProgPrompt: Generating situated robot task plans using large language models. *arXiv preprint arXiv:2209.11302*, 2022.

Tellex, S., Gopalan, N., Kress-Gazit, H., and Matuszek, C. Robots that use language. *Annual Review of Control, Robotics, and Autonomous Systems*, 3:25–55, 2020.

Thoppilan, R., De Freitas, D., Hall, J., Shazeer, N., Kulshreshtha, A., Cheng, H.-T., Jin, A., Bos, T., Baker, L., Du, Y., et al. Lamda: Language models for dialog applications. *arXiv preprint arXiv:2201.08239*, 2022.

Tsimpoukelli, M., Menick, J. L., Cabi, S., Eslami, S., Vinyals, O., and Hill, F. Multimodal few-shot learning with frozen language models. *Advances in Neural Information Processing Systems*, 34:200–212, 2021.

Wang, Z., Cai, S., Liu, A., Ma, X., and Liang, Y. Describe, explain, plan and select: Interactive planning with large language models enables open-world multi-task agents. *arXiv preprint arXiv:2302.01560*, 2023.

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Chi, E., Le, Q., and Zhou, D. Chain of thought prompting elicits reasoning in large language models. *arXiv preprint arXiv:2201.11903*, 2022.

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

Zellers, R., Holtzman, A., Peters, M., Mottaghi, R., Kembhavi, A., Farhadi, A., and Choi, Y. Piglet: Language grounding through neuro-symbolic interaction in a 3d world. *arXiv preprint arXiv:2106.00188*, 2021a.

Zellers, R., Lu, X., Hessel, J., Yu, Y., Park, J. S., Cao, J., Farhadi, A., and Choi, Y. Merlot: Multimodal neural script knowledge models. *Advances in Neural Information Processing Systems*, 34:23634–23651, 2021b.

Zeng, A., Wong, A., Welker, S., Choromanski, K., Tombari, F., Purohit, A., Ryoo, M., Sindhwani, V., Lee, J., Vanhoucke, V., et al. Socratic models: Composing zero-shot multimodal reasoning with language. *arXiv preprint arXiv:2204.00598*, 2022.

Zhang, Y. and Chai, J. Hierarchical task learning from language instructions with unified transformers and self-monitoring. *arXiv preprint arXiv:2106.03427*, 2021.

Zhou, L., Palangi, H., Zhang, L., Hu, H., Corso, J., and Gao, J. Unified vision-language pre-training for image captioning and vqa. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

A. Qualitative Results – Emergent Capabilities

PaLM-E demonstrates several emergent capabilities, including zero-shot multimodal chain-of-thought (CoT), few-shot multimodal prompting, OCR-free math reasoning, as well as multi-image reasoning capabilities despite being only trained on single-image prompts (Fig. 6). Zero-shot CoT (Kojima et al., 2022), originally a language-only concept, has been shown on *multimodal* data with language-composed programs (Zeng et al., 2022) but to our knowledge, not via an end-to-end model.

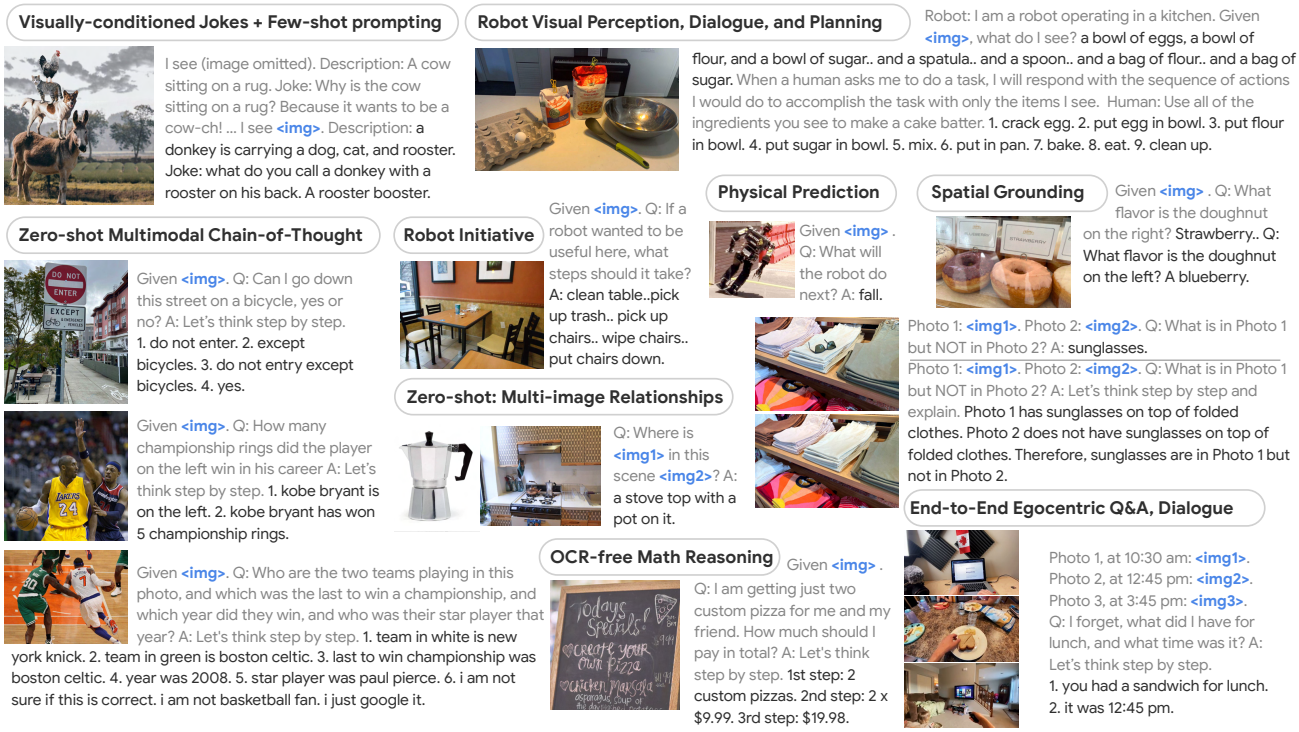

Figure 6: PaLM-E-562B is able to perform *zero-shot multimodal chain-of-thought reasoning*, can tell visually-conditioned jokes given an image, and demonstrates an array of robot-relevant multimodal-informed capabilities including perception, visually-grounded dialogue, and planning. PaLM-E also generalizes, zero-shot, to multi-image prompts despite only being trained on single-image prompts. PaLM-E can also perform math given an image with textually-interleaved handwritten numbers. In addition, the model can perform, zero-shot, question and answering on temporally-annotated egocentric vision, similar to what was shown in (Zeng et al., 2022) but end-to-end all in one model.

B. Limitations

One limitation of PaLM-E's formulation is that it relies on low-level language-conditioned policies to solve robotics tasks. For example, if the underlying policy is not capable of peeling a banana when asked to "peel the banana", then it would be difficult for PaLM-E to guide the low-level policy with simple textual primitives in such a dexterous task. Additionally, if the goal requires interacting with specific parts of a scene that is difficult to describe verbally, then outputting only text can also be limiting. To address these cases, we have proposed the idea of using self-supervised entity-centric labeling, which allows the high-level PaLM-E policy to refer the low-level policy to specific entities without describing them in natural language.

C. Impact

While the positive societal impacts of artificial intelligence may be profound, we also discuss inherent risks. Since PaLM-E is a model capable of solving general vision-language and language-only tasks, we inherit many of the risks associated with large language models, including hallucination, non-factual answers, and biases. Expanding the set of capabilities into the vision-language domain has the potential to further increase these biases based on the vision-language datasets the model has been trained on. One of our scientific goals of this research is to evaluate whether training on many tasks at the same time improves the performance on individual tasks. As we have shown, PaLM-E shows positive transfer across tasks.

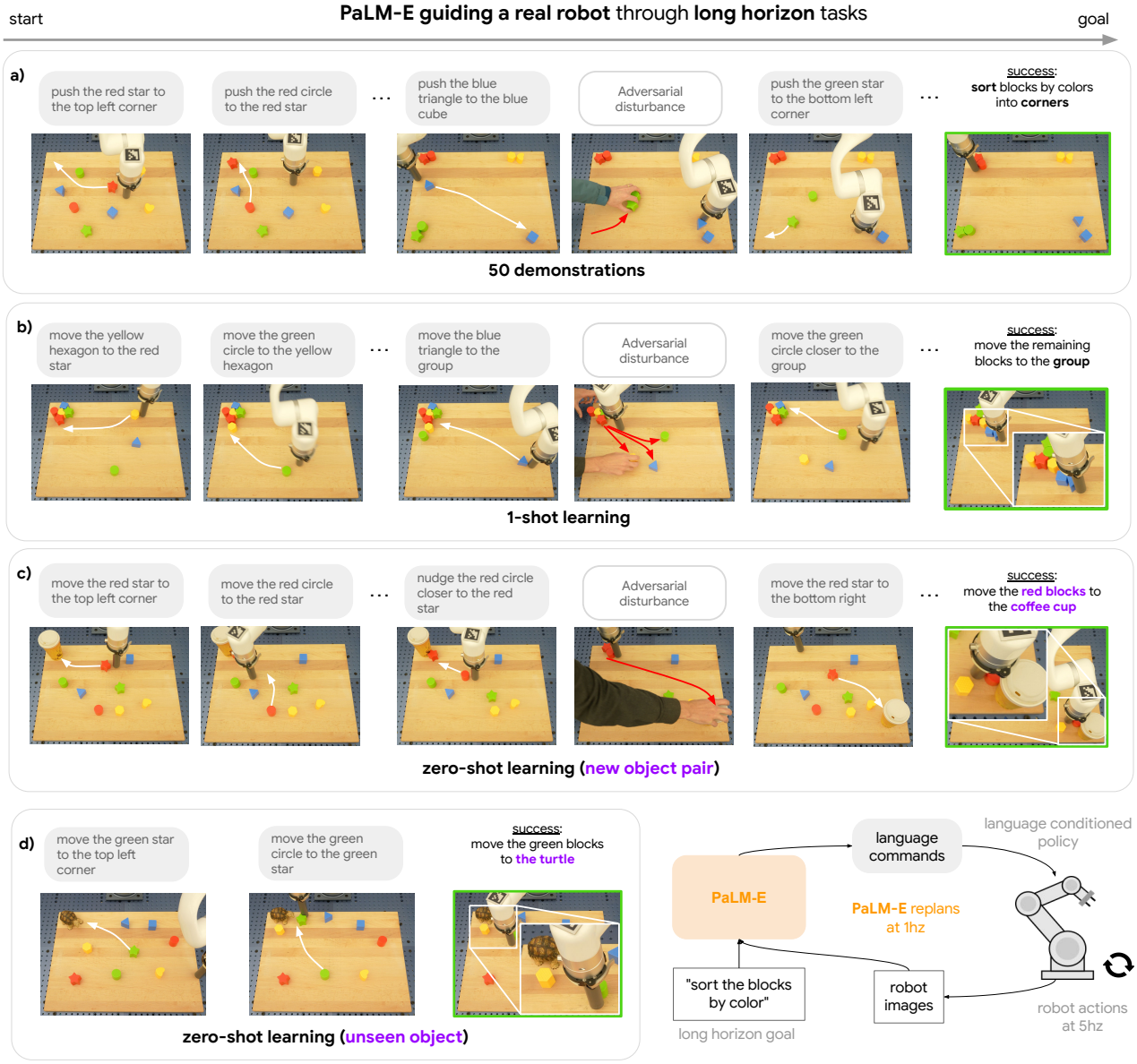

Figure 7: PaLM-E interactively guides a real robot through long-horizon manipulation tasks on Language-Table, while remaining robust to adversarial disturbances. We find evidence that PaLM-E is capable of one-shot and zero shot generalization.

This paves a path towards more powerful models. However, the effects of transfer across tasks with respect to potential biases induced from training on other tasks is not yet well understood. We integrate PaLM-E into control loops controlling robots in the real world. This induces additional risks, as the decisions made by the model are impacting the behavior of real physical systems. If a model like PaLM-E is to be deployed in other settings, especially around non-experts, this should involve additional physical risk assessment. The fact that PaLM-E does not directly control robot actuators, but provides textual instructions for low-level policies enables introspection and interpretability of its outputs.

D. Data Mixture

Tab. 6 shows the dataset and sampling frequency for the "full mixture" as referred to in the experiments. The majority of the data distribution is general vision-language tasks, with less than 10% robot data.

Dataset in full mixture	Sampling frequency	%
Webli (Chen et al., 2022)	100	52.4
VQ^2A (Changpinyo et al., 2022)	25	13.1
VQG (Changpinyo et al., 2022)	10	5.2
CC3M (Sharma et al., 2018)	25	13.1
Object Aware (Piergiovanni et al., 2022)	10	5.2
OK-VQA (Marino et al., 2019)	1	0.5
VQAv2 (Goyal et al., 2017)	1	0.5
COCO (Chen et al., 2015)	1	0.5
Wikipedia text	1	0.5
(robot) Mobile Manipulator, real	6	3.1
(robot) Language Table (Lynch et al., 2022), sim and real	8	4.2
(robot) TAMP, sim	3	1.6

Table 6: Dataset sampling frequency and ratio for the "full mixture" referred to in experiments.

In Tab. 7, we vary the amount of robot data in the training mixture. The remaining data has the same relative distribution as in Tab. 6. As one can see, varying the amount of robot data in the mixture mainly influences the performance on general vision-language tasks. Note that the metrics in Tab. 7 for the general vision-language tasks are the next-token-prediction accuracies, which are different from the CIDER score or VQA accuracy reported in Tab. 5. Using just robot data (100%) is not sufficient to achieve good performance on general vision-language tasks, which is expected, as the robot data does not contain the same amount of variety. The robot task performance is largely unaffected by the data mixture. For the TAMP planning task p_2 (stacking blocks), we find a small advantage of having more general vision-language tasks in the mixture. Note that even when just robot data (100%) is used, the model can still benefit from cross-robot transfer learning.

Amount of robot data in overall mixture	Robotics tasks				General vision-language tasks		
	TAMP		Mobile manipulation		OK-VQA	VQAv2	COCO
	p_1	p_2	failure detection	affordance detection			
10%	96.5[a]	93.5[a]	95.9[b]	96.3[b]	65.8[c]	74.5[c]	53.5[c]
90%	97.0[a]	91.6[a]	98.1[b]	98.1[b]	61.4[c]	70.9[c]	49.6[c]
100%	96.9[a]	88.7[a]	98.4[b]	98.2[b]	26.1[c]	31.2[c]	15.1[c]

[a]success rate (1% of TAMP training data). [b]accuracy (%), in-domain test set. [c]next token accuracy (proxy metric).

Table 7: Performance on different tasks when varying the amount of robot data in the training mixture.

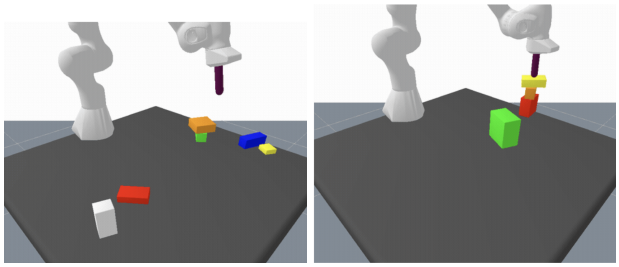

Figure 8: Two TAMP environment test examples. Left with 6 objects (training data contains 3-5 objects), right with 4 objects.

E. Environment Details

E.1. Task and Motion Planning (TAMP)

The training scenes for the TAMP environment contain 3-5 cube-shaped objects of different sizes, colors and sampled initial poses. Fig. 8 show an example test scene that contains 6 objects.

In the global version, we consider the following three VQA tasks:

- q_2: object-table relation. Example prompt: `Given <img>. Q: Is the red object left, right, or center of the table?`. Target: `A: The red object is in the center of the table.`
- q_3: object-object relations. Example prompt: `Given <img>. Q: Is the yellow object below the blue object?`. Target: `A: No, the yellow object is not below the blue object.`
- q_4: plan feasibility. Example prompt: `Given <img>. Q: Is it possible to first grasp the blue object, then place it on the yellow object, and then grasp the yellow object?`. Target: `A: No, this is not possible.`

as well as the two planning tasks

- p_1: grasping. Example prompt: `Given <img>. Q: How to grasp the green object?`. Target: `A: First grasp the orange object and place it on the table, then grasp the green object.`
- p_2: stacking. Example prompt: `Given <img>. Q: How to stack the white object on top of the red object?`. Target: `A: First grasp the green object and place it on the table, then grasp the white object and place it on the red object.`

For the object-centric version with entity referrals, all prompts contain the prefix `<prefix>` = `Obj 1 is <obj`$_1$`>. ... Obj j is <obj`$_j$`>.`, and the VQA task q_1 is about the color of an object. The other tasks (except with the different prefix, and entity referrals), remain the same.

We utilize the planner from Driess et al. (2020) to generate the dataset for the planning tasks. The low-level policies are also obtained with the method of Driess et al. (2020).

E.2. Interactive Language Table

We use the Language-Table real-world tabletop setup and simulated environment from Interactive Language (Lynch et al., 2022).

Data collection. For each task, given the long horizon instruction, we prompt a labeler to enter a short horizon command every 4 seconds. We pass the short horizon instructions to an Interactive Language policy trained using the same procedure as in Lynch et al. (2022). The policy executes 40 steps (10Hz for 4 seconds) before requiring another command from the labeler. This is repeated until the labeler determines the long horizon instruction is complete and issues a 'done' instruction. The data collection procedure for the real world experiments are the same as in simulation.

	ϕ	LLM pre-trained	q_1	q_2	q_3	q_4	p_1	p_2
3 - 5 objects	Low-level policy only	-	-	-	-	-	31.3	0.0
	SayCan (w/ oracle affordances)	✓	-	-	-	-	38.7	33.3
	state	✗	100.0	99.3	98.5	99.8	97.2	95.5
	state	✓(unfrozen)	100.0	98.8	100.0	97.6	97.7	95.3
	state	✓	100.0	98.4	99.7	98.5	97.6	96.0
	state (w/o entity referrals)	✓	100.0	98.8	97.5	98.1	94.6	90.3
	ViT + TL (obj. centric)	✓	99.6	98.7	98.4	96.8	96.2	94.5
	ViT + TL (global)	✓	-	60.7	90.8	94.3	70.7	69.2
	ViT-4B (global)	✓	-	98.2	99.4	99.0	96.0	93.4
	ViT-4B generalist	✓	-	97.1	100.0	98.9	97.5	95.2
	OSRT	✓	99.6	99.1	100.0	98.8	98.1	95.7
6 objects	state	✗	20.4	39.2	71.4	85.2	56.5	34.3
	state	✓	100.0	98.5	94.0	89.3	95.3	81.4
	state (w/o entity referrals)	✓	77.7	83.7	93.6	91.0	81.2	57.1
8 objects	state	✗	18.4	27.1	38.1	87.5	24.6	6.7
	state	✓	100.0	98.3	95.3	89.8	91.3	89.3
	state (w/o entity referrals)	✓	60.0	67.1	94.1	81.2	49.3	49.3
6 objects + OOD tasks	state (8B LLM)	✗	-	0	0	72.0	0	0
	state (8B LLM)	✓	-	49.3	89.8	68.5	28.2	15.7
	state (62B LLM)	✓	-	48.7	92.5	88.1	40.0	30.0

Table 8: Success rates on TAMP environment for different input representations. 3-5 objects in the scene correspond to the training distribution. OOD tasks means out-of-distribution tasks where the objects are referenced by color, although in the trainig data they have been referenced by their special tokens $\texttt{obj}_j$ in the object-centric case. The SayCan baseline (Ahn et al., 2022) utilizes oracle, one-step affordance functions. Compared to the results presented in Tab. 1, 100% of the training data, i.e. 100 times more than in Tab. 1, is used here.

	Task 1	Task 2	Task 3[a]
Low-level policy only	0.0	0.0	37.5[a]
# demos	*40*	*40*	*80*
PaLM-E-12B, full mixture	80.0	58.8	77.0[a]

Table 9: Results on Language Table environment comparing using *only the low-level policy* from (Lynch et al., 2022) to address the long-horizon tasks, as opposed to training PaLM-E to address the long-horizon tasks by closed-loop conditioning the low-level policy with text. Tasks are defined in Tab. 3. [a]Here the quantitative reward for Task 3 has been adjusted to more closely match the qualitative desired behavior, by penalizing blocks that were incorrectly brought from the wrong side.

Train and Evaluation. To train the finetuned versions of these models, we train a pretrained PaLM-E model for 9,000 additional steps, in order to support a data complexity sweep without training several separate models from scratch on slightly different versions of the full mixture. For Tasks 2 and 3 in simulation, we implement an automated reward to measure the success rate, and we evaluate PaLM-E by running 80 rollouts for each task. Given the current image and high level task, PaLM-E issues a text instruction which a trained low-level policy executes for 4 seconds before PaLM-E issues a new text instruction. For Task 1, we use a test-set and report validation accuracy. This is because the task only requires one step to solve, despite being a complicated visual and linguistic processing task and cannot be solved by the low-level policy from the prompt alone.

Analysis of Low-level Policies. In Tab. 9 we address the question: "since the low-level policies themselves are vision+text-conditioned policies, are they sufficient to solve the long-horizon tasks without the use of PaLM-E?". Here we directly condition the policies from (Lynch et al., 2022) with the full prompts as in Tab. 3. As shown in Tab. 9, the low-level policies are not sufficient to solve the tasks.

F. Natural Language Generation and Understanding Results

1-shot evals	**PaLM-8B**	**PaLM-E-12B** *(unfrozen)*	**PaLM-62B**	**PaLM-E-84B** *(unfrozen)*	**PaLM-540B**	**PaLM-E-562B** *(unfrozen)*	**Category**
TriviaQA (wiki) (EM)	48.5	10.1	72.7	31.8	81.4	74.6	NLG
Natural Questions (EM)	10.6	1.6	23.1	7.6	29.3	27.2	NLG
WebQuestions (EM)	12.6	3.4	19.8	7.9	22.6	21.8	NLG
Lambada	57.8	1.4	75.5	26.1	81.8	83.3	NLG
HellaSwag	68.2	48.4	79.7	75.3	83.6	83.5	NLU
StoryCloze	78.7	68.7	83.8	83.9	86.1	86.3	NLU
Winograd	82.4	71.8	85.3	86.4	87.5	89.0	NLU
Winogrande	68.3	55.3	76.8	72.5	83.7	83.0	NLU
RACE-M	57.7	43.2	64.1	57.4	69.3	70.3	NLU
RACE-H	41.6	33.2	48.7	42.3	52.1	52.8	NLU
PIQA	76.1	68.1	80.9	78.2	83.9	84.9	NLU
ARC-e	71.3	53.4	78.9	71.4	85.0	86.3	NLU
ARC-c	42.3	30.9	51.8	46.7	60.1	62.6	NLU
OpenBookQA	47.4	41.4	51.2	51.6	53.6	55.8	NLU
BoolQ	64.7	61.6	83.1	81.6	88.7	89.4	NLU
Copa	82.0	77.0	93.0	91.0	91.0	93.0	NLU
RTE	57.8	54.9	71.5	59.6	78.7	75.1	NLU
Wic	50.6	50.0	48.6	50.2	63.2	64.1	NLU
WSC	81.4	68.4	84.9	75.8	86.3	85.6	NLU
ReCoRD	87.8	71.2	91.0	78.5	92.8	92.5	NLU
CB	41.1	37.5	55.4	73.2	83.9	80.3	NLU
Avg NLU	64.7	55.0	72.3	69.2	78.2	78.5	
Avg NLG	32.4	4.1	47.8	18.4	53.8	51.7	
NLU delta (%, relative)		-15.0%		-4.3%		+0.4%	
NLG delta (%, relative)		-87.3%		-61.6%		-3.8%	

Table 10: Full language evaluation task results on both NLU and NLG tasks, for both the original PaLM models and for associated PaLM-E (unfrozen) models. The PaLM-E models with a frozen LLM have the same performance as their corresponding underlying PaLM models.

G. Additional Data for Affordance and Success Detection

Model				Precision	Recall	F1-score
PaLI (Zero-shot) (Chen et al., 2022)				0.59	0.98	0.73
CLIP-FT (Xiao et al., 2022)				0.50	0.95	0.65
CLIP-FT-hindsight (Xiao et al., 2022)				1.0	0.80	0.89
PaLM-E-12B trained on	from scratch	LLM+ViT pretrain	LLM frozen			
Single robot	✓	✗	n/a	0.52	0.55	0.54
Single robot	✗	✓	✓	0.91	0.92	**0.91**
Full mixture	✗	✓	✓	0.89	0.93	**0.91**
Full mixture	✗	✓	✗	0.66	0.91	0.77

Table 11: Mobile manipulation environment: failure detection, showing individual precision and recall scores. Results correspond to out-of-distribution scenes of Tab. 13.

Model				Precision	Recall	F1-score
PaLI (Zero-shot) (Chen et al., 2022)				0.57	0.69	0.62
QT-OPT (Kalashnikov et al., 2018)				0.60	0.67	0.63
PaLM-E-12B trained on	from scratch	LLM+ViT pretrain	LLM frozen			
Single robot	✓	✗	n/a	0.67	0.35	0.46
Single robot	✗	✓	✓	0.90	0.69	0.78
Full mixture	✗	✓	✓	0.95	0.80	0.87
Full mixture	✗	✓	✗	0.92	0.88	**0.91**

Table 12: Mobile manipulation environment: affordance prediction, showing individual precision and recall scores. Results correspond to out-of-distribution scenes of Tab. 13.

PaLM-E-12-B *trained on*	from scratch	LLM + ViT pretrain	LLM frozen	Failure detection		Affordance prediction	
				in-distribution	out-of-distribution	in-distribution	out-of-distribution
Single robot	✓	✗	n/a	0.63	0.54	0.36	0.46
Single robot	✗	✓	✓	0.96	0.91	0.97	0.78
Full mixture	✗	✓	✓	0.96	0.91	0.98	0.87
Full mixture	✗	✓	✗	0.97	0.77	0.98	0.91

Table 13: Mobile manipulation environment: Comparison between in-distribution and out-of-distribution environments. In-distribution means hold-out scenes with similar backgrounds, lighting conditions, and objects as in the training distribution. Out-of-distribution are scenes with different backgrounds, different furniture, and lighting conditions. The table shows F1-scores.

H. Image Encoders

PaLM-E-12B utilizes the ViT-4B from Chen et al. (2022) to encode images. Tab. 14 presents results using CLIP ViT models Radford et al. (2021); Ilharco et al. (2021) that have different numbers of parameters, and are pre-trained differently. Compared to the ViT-4B which is trained on an image classification as described in Chen et al. (2022), the CLIP models investigated here are trained using a contrastive objective. Further, we investigate the number of tokens to use in these experiments. The CLIP vision model was trained using a single read-out token for the embedding, and we experiment using this to convey the information of the image into the language model instead of using 256 tokens corresponding to the image patches. Alternatively, we also try using all 257 tokens (256 patch tokens, plus the 1 readout token).

As one can see, among the CLIP variations using all 257 tokens performs better than only using 1 token. Amongst the variations using 1 token, it is considerably better to finetune the vision encoder. The differences are especially pronounced on the robot tasks, and in particular on the TAMP planning tasks, which requires spatial precision to solve the tasks.

The ViT-4B model outperforms the CLIP models. One potential explanation is that the ViT-4B model has the highest number of parameters. We also note that one might, intuitively, expect the CLIP model to be better aligned to text embeddings, a pre-trained CLIP model is not necessarily aligned with the text embeddings used by the particular language model that is used to build PaLM-E.

Pre-trained vision model	Vision model frozen	# tokens input into LLM	# parameters	Robotics tasks				General vision-language tasks		
				TAMP		Mobile manipulation		OK-VQA	VQAv2	COCO
				p_1	p_2	failure detection	affordance detection			
ViT-4B	✗	256	4B	96.5[a]	93.5[a]	95.9[b]	96.3[b]	65.8[c]	74.5[c]	53.5[c]
CLIP L-14[d]	✓	1	427M	32.0[a]	35.8[a]	87.1[b]	81.6[b]	60.1[c]	66.4[c]	45.7[c]
CLIP L-14[d]	✗	1	427M	70.7[a]	72.8[a]	93.0[b]	95.2[b]	64.3[c]	71.7[c]	51.2[c]
CLIP L-14[d]	✗	257	427M	79.7[a]	78.2[a]	92.0[b]	94.5[b]	65.2[c]	73.1[c]	52.2[c]
CLIP G-14[e]	✗	257	1.8B	90.5[a]	81.2[a]	90.4[b]	95.2[b]	65.3[c]	72.9[c]	52.0[c]

[a] success rate (1% of TAMP training data). [b] accuracy (%), in-distribution test set. [c] next token accuracy (proxy metric).
[d] CLIP model from Radford et al. (2021). [e] CLIP model from Ilharco et al. (2021).

Table 14: Performance comparison for different image encoders.

I. Image Attribution

The image of the New York Knicks and Boston Celtics in Figure 2 is under the terms CC-by-2.0 (`https://creativecommons.org/licenses/by/2.0/`), and was posted to Flickr by kowarski at `https://www.flickr.com/photos/27728232@N00/8666371367`. The egocentric video images are from `https://youtu.be/-UXKmqBPk1w`, as in (Zeng et al., 2022), via permission from creator Cody Wanner.