Robotics: Science and Systems 2023
Daegu, Republic of Korea, July 10-July 14, 2023

Language-Driven Representation Learning for Robotics

Siddharth Karamcheti[1], Suraj Nair[1], Annie S. Chen[1],
Thomas Kollar[2], Chelsea Finn[1], Dorsa Sadigh[1], Percy Liang[1]

[1]Stanford University [2]Toyota Research Institute

***Abstract*—Recent work in visual representation learning for robotics demonstrates the viability of learning from large video datasets of humans performing everyday tasks. Leveraging methods such as masked autoencoding and contrastive learning, these representations exhibit strong transfer to policy learning for visuomotor control. But, robot learning encompasses a diverse set of problems beyond control including grasp affordance prediction, language-conditioned imitation learning, and intent scoring for human-robot collaboration, amongst others. First, we demonstrate that existing representations yield inconsistent results across these tasks: masked autoencoding approaches pick up on low-level spatial features at the cost of high-level semantics, while contrastive learning approaches capture the opposite. We then introduce $\mathcal{V}$oltron, a framework for language-driven representation learning from videos and associated captions. $\mathcal{V}$oltron trades off language-conditioned visual reconstruction to learn low-level visual patterns, and visually-grounded language generation to encode high-level semantics. We also construct an evaluation suite spanning five distinct robot learning problems – a unified platform for holistically evaluating visual representations for robotics. Through comprehensive, controlled experiments across all five problems, we find that $\mathcal{V}$oltron's language-driven representations outperform the prior state-of-the-art, especially on targeted problems requiring higher-level features.**[1]

I. Introduction

Good words are worth much, and cost little.
— George Herbert

Realizing a future of ubiquitous, broadly capable robots is predicated on systems capable of generalizable perception and interaction [95, 12, 47]. Towards this goal, recent work in robotics present approaches for learning visual representations to bootstrap learning for visuomotor control [62, 58, 65]. Critically, these approaches show that we can learn such representations from *real-world videos of human behavior* – specifically, egocentric video datasets such as Something-Something-v2 and Ego4D [24, 25] – instead of solely relying on in-domain robotics data that is scarce and expensive. While prior work has developed and evaluated representations for visuomotor control, robot learning is an expansive discipline, spanning *a diverse spectrum of problems*: predicting grasp proposals from visual input [71, 54], language-conditioned imitation learning [89] and belief/intent tracking for human-robot interaction [27, 38], amongst others. Broadening our focus to problems beyond learning for control enables us to develop flexible, generalizable representations that capture *both* low-level spatial reasoning and high-level semantic understanding – a flexibility that is a key prerequisite to realizing a foundation model for robotics [11]. Thus, we ask: *how can we learn visual representations that generalize across the diverse spectrum of problems in robot learning?*

Recent approaches for learning visual representations for robotics use pretraining objectives that reflect different inductive biases for what the learned representations should capture. Masked Visual Pretraining [MVP; 65] proposes using masked autoencoding [29] to prioritize visual reconstruction from heavily masked video frames, encoding representations that facilitate per-pixel reconstruction. Separately, Reusable Representations for Robotic Manipulation [R3M; 58] eschews pixel reconstruction for two contrastive learning objectives: time contrastive learning [74] and video-language alignment. These approaches show strong performance on imitation learning in simulated and real-world settings, with sizeable improvements over strong alternatives such as ResNet or CLIP features [28, 64]; however, they have not been evaluated beyond these settings. As a first contribution, we evaluate these representations on problems beyond control and identify *inconsistent evaluation performance*, with huge penalties depending on the approach and specific application. MVP performs well on problems such as grasp affordance prediction, but struggles with higher-level problems such as language-conditioned imitation. R3M instead excels at the higher-level problems, but degrades completely on problems such as grasp affordance prediction.

Motivated by this, we present $\mathcal{V}$**oltron**, a framework for language-driven visual representation learning for robotics that learns representations that capture both low-level and high-level features, empirically outperforming prior approaches over *all* applications. $\mathcal{V}$oltron models take videos and associated language captions as input to a masked autoencoding pipeline, reconstructing one (or more) frames from a masked context. *The novelty of our framework is in how we use language supervision*. Depending on a tunable probability α, we either condition on $(\alpha = 0)$, or generate $(\alpha > 0)$ the associated caption. Explicitly *conditioning* on words in different contexts allows for low-level pattern recognition at the local, spatial level, while *generating* language from our learned visual encoding allow us to infer higher-level features around affordances and intents. Furthermore, guided by the hypothesis that language is especially useful in describing *change*, we study *dual-frame* contexts consisting of the initial and current observation in multi-timestep tasks.

[1]Project Page: https://sites.google.com/view/voltron-robotics
Models & Pretraining Code: https://github.com/siddk/voltron-robotics
Evaluation Suite: https://github.com/siddk/voltron-evaluation

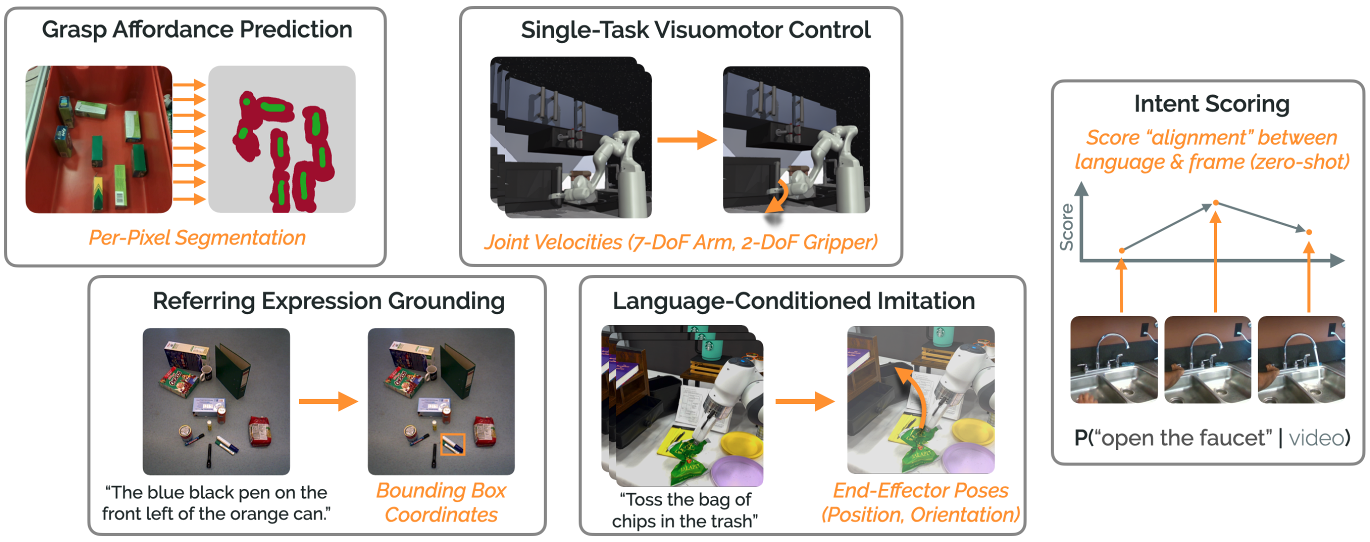

Grasp Affordance Prediction
Per-Pixel Segmentation
Single-Task Visuomotor Control
Joint Velocities (7-DoF Arm, 2-DoF Gripper)
Intent Scoring
Score "alignment" between language & frame (zero-shot)
Score
P("open the faucet" | video)
Referring Expression Grounding
"The blue black pen on the front left of the orange can."
Bounding Box Coordinates
Language-Conditioned Imitation
"Toss the bag of chips in the trash"
End-Effector Poses (Position, Orientation)

Fig. 1: **$\mathcal{V}$oltron Evaluation Suite.** We introduce a suite of evaluation problems spanning *five applications within robotics*, including grasp affordance prediction, referring expression grounding, single-task visuomotor control (in simulation), language-conditioned imitation learning (on a real robot), and intent scoring.

Altogether, we examine *three different $\mathcal{V}$oltron variants*:

1) $\mathcal{V}$ – **Cond** (*Language Conditioning*: single frame, $\alpha = 0$).
2) $\mathcal{V}$ – **Dual** (*Context*: dual-frame conditioning, $\alpha = 0$).
3) $\mathcal{V}$ – **Gen** (*Language Generation*: dual-frame, $\alpha = 0.5$).

To evaluate $\mathcal{V}$oltron and other visual representation learning approaches, we assemble a new evaluation suite (depicted in Fig. 1) spanning five problem domains within robotics: 1) dense segmentation for grasp affordance prediction [101], 2) object detection from referring expressions (e.g., "the blue coffee mug to the left of the plate") in cluttered scenes [94], 3) imitation learning for visuomotor control (in simulation) [58], 4) learning multi-task language-conditioned policies for real-world manipulation [86] (on a real-world Franka Emika fixed-arm manipulator), and 5) zero-shot intent scoring [38, 13]. We choose these tasks for their broad coverage; tasks such as grasp affordance prediction and referring expression grounding require reasoning over low-level spatial features, while language-conditioned imitation and intent scoring require a deeper understanding of semantics.

Through experiments controlling for pretraining data and model capacity, we show that the simplest $\mathcal{V}$oltron representations (from $\mathcal{V}$ – **Cond**) strictly outperform both MVP and R3M representations across *all* evaluation domains. Furthermore, by adapting our models to learn from multiple frame contexts and that favor generation (e.g., with $\mathcal{V}$ – **Dual** and $\mathcal{V}$ – **Gen**), we show that we can further boost performance on evaluations requiring higher-level features such as with language-conditioned policy learning (on a real robot) and intent scoring. Though language-conditioning offers universal performance gains, there are tradeoffs *between $\mathcal{V}$oltron models*; adding language generation hurts performance on some control tasks, even though its necessary for strong performance on intent scoring. Furthermore, $\mathcal{V}$oltron with single-frame language conditioning performs well on non-episodic tasks (e.g., grasping), but underperforms multi-frame models on control tasks. There is not yet a silver bullet – a single representation strong on all tasks – but the ability to balance tradeoffs between encoding low and high-level features offers a net win over restrictions of past work.

Contributions. 1) We present **$\mathcal{V}$oltron**, a framework for language-driven visual representation learning. Through controlled experiments and comprehensive ablations we demonstrate that $\mathcal{V}$oltron's representations strictly outperform the prior art across **2)** a new evaluation suite composed of five distinct problem domains within robotics. Finally, **3)** we analyze the tradeoffs between different $\mathcal{V}$oltron models that balance different types of feature learning, outlining several directions for future work. We release all models, the evaluation suite, code (pretraining and adaptation), and preprocessed data at: https://sites.google.com/view/voltron-robotics.

Limitations. We do not have access to the compute resources to train models of the same scale and data used in prior work [65, 58]. Instead, we carefully reproduce MVP and R3M – the current state-of-the-art approaches – by pretraining on the Something-Something-v2 dataset [24], further controlling for batch ordering, model capacity, and other sources of randomness (full details are in §IV). However, for full context we also include results from the official release artifacts from both these works, as well as other methods such as CLIP [64], though we note these results *in gray* or with dashed lines as to indicate they are not directly comparable.

II. Related Work

$\mathcal{V}$oltron is situated within a rich body of work in visual representation learning for robotics and multimodal pretraining.

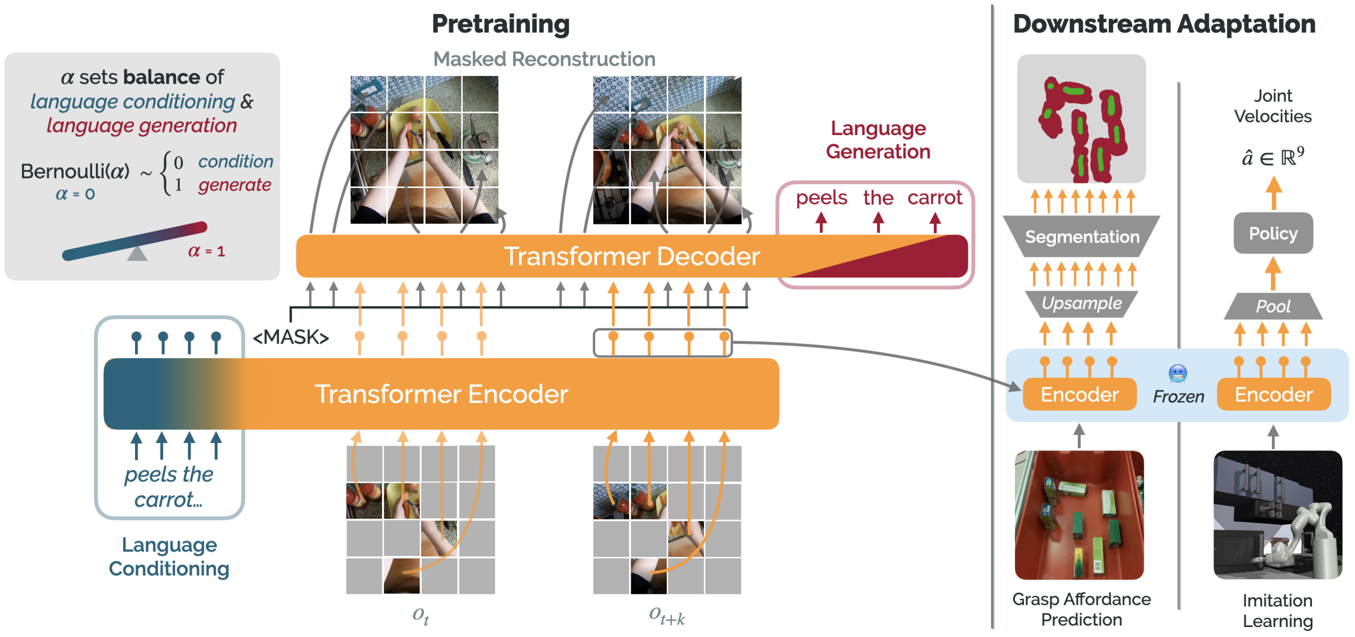


Fig. 2: **The $\mathcal{V}$oltron Framework**. Central to our approach is *language-driven learning* on top of a masked autoencoding backbone. We incorporate language in two ways, following §III-B: 1) as a *conditioning variable* fed to a multimodal encoder that also encodes one or more video frames, or 2) as a *generation target* for the language generator **[Left]**. During downstream evaluation, we use the (frozen) outputs from the encoder, adapting evaluation-specific "heads" on top **[Right]**.

Visual Representation Learning for Robotics. An emerging body of work in robot learning studies learning visual state representations for control. A wealth of prior approaches learn representations from *in-domain* data taken directly from the target environment (and corresponding task); these techniques range from using data augmentation [45, 84, 44, 61] to modeling forward dynamics [22, 26] to using task-specific information [39, 103]. Unlike these approaches, we move beyond task-specific data, instead leveraging large, accessible datasets such as videos of humans performing everyday tasks. Work in this paradigm has exploded in recent years. A number of approaches find that existing representations such as features from models trained on ImageNet [19], or features from CLIP [64] enable more efficient learning [75, 42]. More recently, multiple approaches have shown increased dividends in applying such representations to visuomotor control, for example by combining features at different layers of pretrained ResNets [62] or by pretraining such representations on human videos, conjecturing that such data captures features useful for robotic manipulation [58, 97, 65, 53]. Missing from these approaches however is a notion of semantics; works such as MVP [97, 65] purely learn to perform masked reconstruction, and even works that leverage *some* temporal and linguistic signals do so in a limited way [58, 53]. Instead, our work is motivated by the hypothesis that language understanding – via conditioning *and* generation – is an essential component for learning generalizable representations. It is not enough that a representation summarizes an observation; instead, for generalization to new contexts, it must capture how observations (and *changes* thereof) relate to higher-level semantic abstractions.

$\mathcal{V}$oltron aims to do this with its language-driven representation learning objective: by jointly modeling sequences of frames *and* language, we enable a range of capabilities, from producing representations of single images in isolation, to providing the capability to *generate* language grounded in visual contexts. We demonstrate the benefits of language-driven learning in our evaluation (see §V): in head-to-head comparisons controlling for data and model capacity, $\mathcal{V}$oltron models strictly outperform prior approaches across *all* evaluation domains.

Learning Multimodal Foundation Models. Our work draws further inspiration from a wave of progress in multimodal foundation models such as CLIP, Multimodal Masked Autoencoders (M3AE), Flamingo, CoCa, and Gato, amongst many others [64, 23, 3, 99, 68, 51, 1]. These approaches highlight the myriad benefits of multimodal pretraining: language supervision works to enrich visual representations (even *in the absence of language downstream*), while visual supervision similarly enriches language representations [50, 82]. Of the many capabilities afforded by these models, many have applications in embodied AI and robotics. CLIP representations have shown to be effective in applications to various robotics tasks [80, 42, 17], while multimodal transformer models have proven effective initializations for training control policies [69, 49]. These approaches are similar to $\mathcal{V}$oltron in their joint use of visual and language inputs; where $\mathcal{V}$oltron differs, however, is in our novel representation learning objective that balances language conditioning and generation, enabling learning representations that transfer to a wide range of applications within robotics.

III. $\mathcal{V}$oltron – Language-Driven Learning

We assume access to a dataset of videos paired with natural language annotations; in each video-language pair (v, c), language can take the form of a caption (e.g., "peels the carrot" in Fig. 2), narration, or even coarse textual label of a behavior. We assume each video $v \in \mathbb{R}^{T \times H \times W \times C}$ consists of a sequence of frames $v = [o_1, \ldots, o_T]$, where each frame $o_i \in \mathbb{R}^{H \times W \times C}$ is RGB-encoded. We tokenize and one-hot encode each utterance into a vocabulary V of cardinality $|V|$, padding to a max length L such that $c \in \mathbb{R}^{L \times |V|}$. We define a `<NULL>` token (separate from the `<PAD>` token) as a placeholder for an empty language context. Furthermore, following the MAE work, we define a visual masking function $\texttt{Mask}(v, \gamma) \rightarrow (v_{\text{visible}} \in \mathbb{R}^{(1-\gamma)(T \times H \times W \times C)}, v_{\text{masked}} \in \mathbb{R}^{\gamma(T \times H \times W \times C)})$ that partitions the regions of a video into a set of visible and masked-out regions subject to a fixed masking ratio γ. This mask is held constant across timesteps in a given clip. We sample a mask once, and apply it uniformly across *all* frames in the video to prevent leakage [90]; if the masks were sampled independently, a masked region in one frame could be visible in another, allowing the encoder to "cheat" by looking ahead.

A. $\mathcal{V}$oltron – Core Components

A $\mathcal{V}$oltron model comprises 1) a *multimodal encoder* that takes in a visual context and (optional) language utterance producing a dense representation, 2) a *visual reconstructor* that attempts to reconstruct the masked-out visual context from the encoder's representation of what is visible, and 3) a *language generator* that predicts the language annotation for the video given the encoded visual context. The visual reconstructor and language generator crucially act to shape the representations by first erasing portions of a (v, c) pair, then attempting to reconstruct the missing parts; we show in our experiments (see §V) that this bottleneck helps focus on more low-level features when we favor reconstruction over generation, and more high-level, semantic features when we favor generation over reconstruction. We step through each component below.

Multimodal Encoder: $\text{E}_\theta(\tilde{v}, u) \rightarrow h \in \mathbb{R}^{S \times d}$

The multimodal encoder (Fig. 2; lower half in blue and orange) is the core of a $\mathcal{V}$oltron model. It takes as input $(\tilde{v}, u)$ where $\tilde{v} \in \{v_{\text{visible}}, v\}$ denotes either the *masked* or *unmasked* (full) visual context respectively, and u represents a (possibly `<NULL>`) utterance to condition on. As output, the encoder produces a dense representation $h \in \mathbb{R}^{S \times d}$ where S denotes the number of encoded regions, and d is a hyperparameter denoting the dimensionality of the representation. Keeping with the original MAE work, we divide each image $o_i \in \mathbb{R}^{H \times W \times C}$ into a set of non-overlapping regions R, where each region is a $p \times p$ patch; this results in $|R| = HW/p^2$ regions. Given a k-frame context, $S = (1 - \gamma)k|R|$.

Visual Reconstructor: $\text{R}_\theta(h) \rightarrow \hat{v}_{\text{masked}} \in \mathbb{R}^{\gamma(k \times H \times W \times C)}$

The visual reconstructor (Fig. 2; upper half in orange) takes as input the encoded representation of the *visible* visual context $h = \text{E}_\theta(v_{\text{visible}}, c)$. It attempts to reconstruct the missing visual regions v_{masked}, conditioned on language context c, producing a prediction $\hat{v}_{\text{masked}}$. Following prior work, the elements of $\hat{v}_{\text{masked}}$ are the normalized pixel targets from the original image. We use mean-squared error as the reconstruction loss $\mathcal{L}_{\text{reconstruct}}(\theta)$.

Language Generator: $\text{G}_\theta(h) \rightarrow \hat{c} \in \mathbb{R}^{L \times C}$

The language generator (Fig. 2; upper half in red) takes the encoded representation of the *visible* context and the `<NULL>` language token, $h = \text{E}_\theta(v_{\text{visible}}, \texttt{<NULL>})$. It generates the language annotation, producing $\hat{c} \in \mathbb{R}^{L \times |V|}$, with each of the L elements corresponding to a probability distribution over the vocabulary. We use the negative log-likelihood (cross-entropy) of the annotation c under the generator as our loss $\mathcal{L}_{\text{generate}}$.

The language generator crucially takes the `<NULL>` token as input instead of the annotation c; inputting the same c that the generator is trying to output can lead to trivial collapse where the encoder learns to memorize the tokens to aid the generator. As a result, for each example during training we need to *either* condition *or* generate language; this further motivates the parameter α in Fig. 2 and in the training objective.

B. Balancing Reconstruction & Generation

The $\mathcal{V}$oltron learning objective trades off language-conditioned *reconstruction* and visually-grounded *language generation* to shape the features captured by the encoder's learned representation. The reconstruction objective prioritizes low-level spatial information conducive to filling in missing textures, colors, or edges; likewise, the generation objective captures higher-level semantic information, encouraging the encoder to encode features that are predictive of the language caption. We make this tradeoff explicit by minimizing the following loss, characterized by the parameter $\alpha \in [0, 1]$:

$$\begin{aligned}
\mathcal{L}(\theta) &= \mathcal{L}_{\text{reconstruct}}(\theta) + \mathcal{L}_{\text{generate}}(\theta) \\
&= \begin{cases} \text{MSE}(v_{\text{masked}}, \text{R}_\theta(\text{E}_\theta(v_{\text{visible}}, c))) & \text{if } z = 0 \\ \text{MSE}(v_{\text{masked}}, \text{R}_\theta(\text{E}_\theta(v_{\text{visible}}, \texttt{<NULL>}))) & \text{if } z = 1 \\ \quad + \text{ NLL}(c, \text{G}_\theta(\text{E}_\theta(v_{\text{visible}}, \texttt{<NULL>}))) & \end{cases} \\
&\text{and } \ z \sim \text{Bernoulli}(\alpha)
\end{aligned}$$

For each example (v, c) seen at training, we draw $z \sim \text{Bernoulli}(\alpha)$: with $z = 0$ we *condition* on the original language utterance, while with $z = 1$, we *generate* the original language utterance, conditioning the encoder on the `<NULL>` token. We limit our exploration in this work to at most two frame contexts $k = 2$ due to computational cost; even four frame contexts exceed the memory on the compute available to us. In selecting the two frame contexts, we sample at least five frames from each video clip in our dataset (with random intervals between). We enforce a heuristic such that the first frame in each dual-frame context comes from the first 20% of the clip, with the other frame appearing in the remaining 80%.

Driven by the hypothesis that different values of α and frame-contexts k shape the balance of low-level and high-level features in our representations, we evaluate three different instantiations of $\mathcal{V}$oltron (as mentioned in §I):

- $\mathcal{V}$ **– Cond**: $\alpha = 0$, $k = 1$ *single-frame conditioning*.
- $\mathcal{V}$ **– Dual**: $\alpha = 0$, $k = 2$ *dual-frame conditioning*; identical to $\mathcal{V}$ **– Cond** but trained on dual-frame pairs (initial frame, random subsequent frame).
- $\mathcal{V}$ **– Gen**: $\alpha = 0.5$, $k = 2$; condition *and* generate with equal probability, trained on dual-frame contexts as above.

Note that we *do not evaluate* $\alpha = 1$; preliminary experiments show that some language conditioning is always helpful.

IV. Implementation & Reproducibility

In addition to our framework, a core contribution of this work is a comprehensive set of controlled experiments. To do this, we *reimplement* both MVP and R3M using code released by the authors, controlling for the pretraining data (at the level of the individual frames seen per epoch) and model capacity.

Baselines – Preliminaries. Throughout this work, we have mentioned both MVP and R3M in terms of their tradeoffs; here, we make their pretraining objectives explicit. Both prior approaches use video datasets, but only learn *single-frame encoders*, choosing to use the video structure in different ways (detailed below). Of the two approaches, we note that only R3M uses language supervision.

MVP follows a masked autoencoding backbone, similar to that depicted in Fig. 2 (without language). MVP does not offer any special consideration to the temporal structure of videos, instead treating each frame in the dataset as as standalone input. Given a single frame, MVP masks out regions subject to a fixed mask ratio γ (same as in $\mathcal{V}$oltron), encoding the visible context with a Transformer encoder, then attempting to reconstruct the missing context with a separate Transformer decoder – also using mean-squared error for reconstruction.

R3M is different in that it does not contain a reconstruction component, instead combining *two contrastive objectives* on top of a single-frame visual encoder – time contrastive learning [74] and image-language temporal alignment [64, 57]. These objectives explicitly use the *temporal* structure of videos. Given an encoding of a visual context, the time-contrastive objective seeks to maximize the score of encodings between frames close together in time (e.g., within a few frames of each other), contrasted against frames from the same video that are further away. R3M also *uses language supervision*. Given a separate encoder that fuses a language caption with the encoding dual-frames contexts (consisting of an initial and subsequent frame) the image-language alignment objective attempts to assign scores that capture "task progress:" the score of a subsequent frame occurring later in a video subject to a language caption should be higher than the score of a frame occurring earlier. The two key differences between $\mathcal{V}$oltron and R3M are 1) using visual reconstruction as a dense objective vs. time contrastive learning, and 2) explicitly conditioning on or generating language in $\mathcal{V}$oltron vs. matching visual and language embeddings as a contrastive objective.

Pretraining Dataset Construction. We use Something-Something-v2 [Sth-Sth; 24] as our pretraining dataset, motivated by prior work [77, 13, 97]. All models see the *exact same image frames*. We extract 5 frames per video, per training epoch to ensure we are learning from multiple visual inputs of the same context and to facilitate R3M's time contrastive learning objective [74]; we serialize the processed frames, and store index files with the video/frame indices per epoch.

Data-Equivalent Reproductions. Though prior works release trained model artifacts, they do not provide sufficient details for reproduction, such as the exact frames sampled from videos, preprocessing applied, or hardware/compute used. We thus reimplement MVP and R3M in a controlled setting on Sth-Sth using the released code from the original papers where possible and clarifying additional details with the authors directly as needed. We implement all models with a Vision Transformer (ViT) backbone and additionally implement R3M with a ResNet-50 backbone based on discussions with the authors of the original work. They suggested that there may be slight differences in the inductive bias of ResNets vs. Vision Transformers [67] that would be worth investigating. We use the ViT-Small/16 variant, with patch size $p \times p = 16 \times 16$ and a Transformer with 12 blocks, 6 attention heads per block, and hidden dimension $d = 384$ [96]. We refer to our reproductions as "R-MVP," "R-R3M (ViT-S)," and "R-R3M (RN-50)."

We pretrain all models in this work on TPU v3-8 compute, generously granted to us by the TPU Research Cloud program (TRC). We run 400 epochs of training for all models with a batch size of 1024, each epoch comprised of a pass through 844K frames (168K clips in Sth-Sth, 5 frames per clip). We do not use dropout or data augmentation. All code and reproducibility details are in our open-source code repositories, linked from our project page.

Additional Comparisons. Though we lack the compute resources to train on models on the same scale data, we further contextualize our results by evaluating the official R3M and MVP models released in the original works. We note that the released R3M model uses an unspecified subset of the Ego4D dataset [25], comprised of over 3000 hours of videos, spanning over 3M individual clips (constituting a dataset **more than 20x** larger than that used in this work). The released MVP also uses an unspecified subset of Ego4D, but add Sth-Sth, Epic-Kitchens, and more [18, 76], while also scaling models up to 86M and 307M parameters, (**4-10x** the size of ViT-Small). We also evaluate OpenAI's CLIP model (ViT-Base) as a strong baseline that leverages language supervision. We refer to these models as *"R3M (Ego4D)," "MVP (EgoSoup),"* and *"CLIP (ViT-B),"* following naming conventions from the original work and denote them with *gray text* and dashed lines in plots.

$\mathcal{V}$oltron Architecture Details. $\mathcal{V}$oltron follows the masked autoencoding pipeline detailed above, with simple extensions for incorporating language. We implement the $\mathcal{V}$oltron encoder E_θ by jointly embedding the language u and visual inputs v_{visible} with a Transformer [93]. We initialize language embeddings

TABLE I: **Summary of Evaluation Suite & Results.** While some of our evaluation domains use language input, grasp affordance prediction and single-task visuomotor control *do not*. $\mathcal{V}$oltron models obtain strong performance over *all applications*, whereas R-R3M and R-MVP exhibit variable performance depending on the application subset.

	Input Format	**Train Dataset Size**	**Best Model**	**Best Baseline**
Grasp §V-A	Single Frame	1470	$\mathcal{V}$ – **Cond**	R-MVP
Referring Expressions §V-B	Single Frame, **Language Expression**	259,839	$\mathcal{V}$ – **Cond**	R-R3M (ViT)
Single-Task Control §V-C	Frame History	$n \in [5, 10, 25]$ Demos	$\mathcal{V}$ – **Dual**	R-R3M (RN-50)
Language-Conditioned Imitation §V-D	Frame History, **Language Instruction**	100 = 5 x 20 Demos	$\mathcal{V}$ – **Dual** / $\mathcal{V}$ – **Gen**	R-R3M (ViT)
Intent Scoring §V-E	Frame History, **Language Intent**	N/A (Zero-Shot)	$\mathcal{V}$ – **Gen**	N/A

from DistilBERT [70], learning a separate linear projection into the encoder's embedding space, similar to R3M. For the visual reconstructor R_θ and language generator G_θ, we use a separate Transformer with a small addition to enable language generation. In a standard MAE decoder, patches are generated independently, attending to all patch embeddings from the encoder. To enable generation, we append a causal (lower triangular) attention mask for preventing our language decoder from "peeking" at the future inputs to generate (visualized by the red triangle in Fig. 2). This is akin to prefix language modeling [66]; all embeddings can attend to the visual inputs (as in a traditional MAE decoder), but language embeddings can only attend to the preceding language input.

$\mathcal{V}$oltron uses a combination of different language objectives on top of the standard MAE pipeline, adding complexity. To help ensure stable and reliable training, we follow best practices from the NLP community and make a series of small changes to the Transformer architecture including: 1) switching the default LayerNorm to root-mean square normalization [104, 59] (stability, no learned parameters), 2) switching from the default GELU to the more performant SwishGLU activation [79, 15] (performance), and 3) adopting LayerScale for scaling down the magnitude of each residual connection [91, 40] (prevents overflow). To ensure that any gains in evaluation performance stem from our insights around language-driven learning rather than this modified architecture, we run an ablation experiment in §VI. We find that these changes do not change downstream evaluation results, but significantly improve training stability. We present further details, including a sketch of the implementation differences in §B-A.

Adapting Representations. Unfortunately, there is not a standard way to extract representations from learned Vision Transformer encoders, especially for those trained via masked autoencoding. However, Zhai et al. [102] suggest that multi-headed attention pooling [MAP; 46] is a strong and versatile approach. We choose to use MAP as the sole feature extraction approach in all our ViT experiments, finding it to *universally improve performance for all ViT models*, relative to the "default" extraction approaches suggested in prior work. Notably, we find that just switching to MAP-based extraction over the procedure used in the original MVP work *almost doubles success rate* on visuomotor control tasks; we provide this analysis in §D-B. We also use MAP when evaluating *CLIP (ViT-Base/16)* and *MVP (EgoSoup)* for the strongest possible comparison.

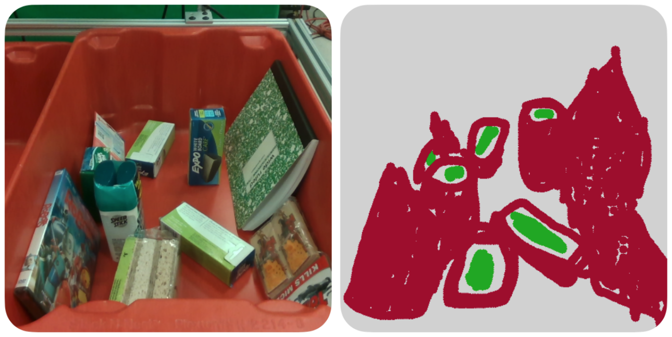


Fig. 3: **Grasp Affordance Prediction** [ARC Grasping; 101]. Given objects in cluttered bins, segment the image corresponding to "graspable" (green), vs. "non-graspable" (red) regions; note that these regions are labeled for use with *suction grippers*.

V. Evaluation Suite: Construction & Results

We outline our evaluation suite (Table I) comprised of five problem domains within robotics. Each evaluation consists of *adaptation data* and *evaluation metrics*. The adaptation data consists of visual input(s) (as RGB frames) and in some cases, language (e.g., an instruction for language-conditioned imitation). We evaluate representations from $\mathcal{V}$oltron and various baseline models by *freezing the pretrained vision and language encoders*, instead *adapting* evaluation-specific "heads"(lightweight networks) on top of the extracted representations. We choose evaluations that capture different types of visual understanding; in the following sections, we motivate the role of each application and provide experimental results.

TABLE II: **Results on Grasp Affordance Prediction.** We report average precision at various confidence intervals following the original procedure described in Zeng et al. [101].

	Architecture	**Top-1**	**Top 1%**	**Top 5%**
R-R3M	ViT-Small	40.38	40.55	28.66
R-MVP	ViT-Small	72.94	61.47	39.77
$\mathcal{V}$ – **Cond** [Ours]	ViT-Small	**85.15**	**80.71**	47.45
$\mathcal{V}$ – **Cond** [Ours]	ViT-Base	**90.00**	**82.44**	**62.33**
CLIP	*ViT-Base*	*43.20*	*44.11*	*29.66*
MVP (EgoSoup)	*ViT-Base*	*77.49*	*72.87*	*51.28*

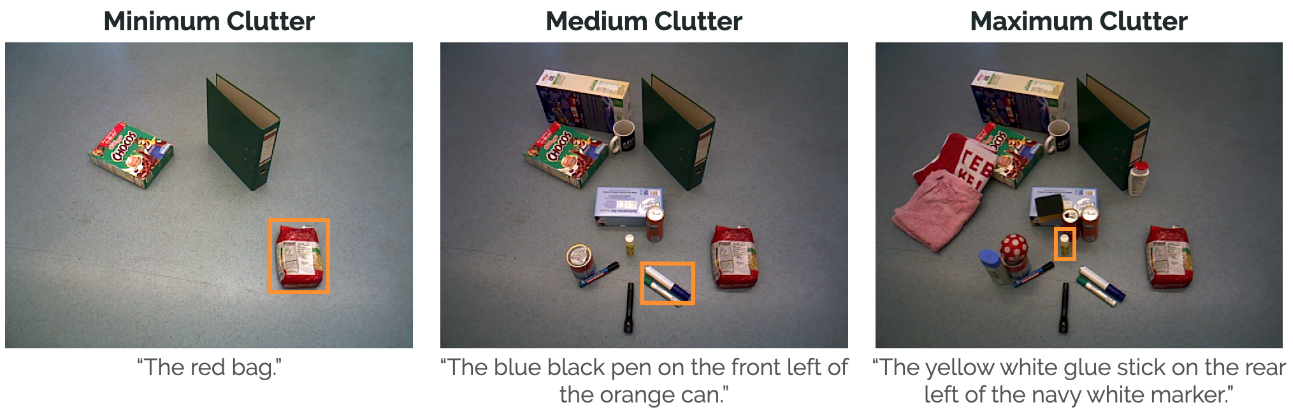

Minimum Clutter
Medium Clutter
Maximum Clutter
"The red bag."
"The blue black pen on the front left of the orange can."
"The yellow white glue stick on the rear left of the navy white marker."

Fig. 4: **Referring Expression Grounding (Object Detection) from the OCID-Ref Dataset** [94]. Given a referring expression in natural language, the goal is to predict the bounding box coordinates around the respective object. An important feature of OCID-Ref are the various dataset splits, corresponding to three increasing amounts of clutter, depicted left-to-right.

A. *Grasp Affordance Prediction*

We consider the problem of grasp affordance prediction: given an image of a set of objects on a cluttered workspace, predict a dense segmentation mask corresponding to "graspable" and "non-graspable" locations for a suction-based gripper.

Motivation. Grasp affordance prediction from visual input is a foundational task in robot learning, and is often a key component of many modular systems [10, 16]. Including this evaluation allows us to probe the low-level spatial features retained by various representations.

Evaluation Details. We specifically consider the problem as formulated in the Amazon Robotics Challenge Grasping Dataset (ARC-Grasping) introduced by Zeng et al. [101]. We choose this dataset over alternatives as it is readily available and consists of 1800+ images of multiple real-world objects in cluttered bins (Fig. 3; left). We focus on the RGB-only, suction-grasping split of the dataset. We implement models for grasp affordance prediction following recent work on semantic segmentation with Transformers [105, 87, 7], specifically by introducing a Progressive Upsampling (SETR-PUP) head on top of our frozen visual features. We omit results from all ResNet models – R-R3M (RN-50) and *R3M (Ego4D)*; unfortunately, training with simple PUP-style on the final ResNet-50 7×7 spatial grid did not converge, possibly indicating a need for more complex architectures with significant added parameters (beyond the scope of this work). As this task only takes a single frame as input, we do not evaluate $\mathcal{V}$ – **Dual** and $\mathcal{V}$ – **Gen**. Following the original work, we report average precision at various confidences: Top-1 precision, Top-1% precision, and Top-5% precision. We select models via 5-fold cross validation. This task *does not have a language component*. We provide additional details around the adaptation procedure in §E and the open-source code repositories.

Experimental Results. Looking at Table II, representations from MVP and $\mathcal{V}$oltron models perform well across the board, while contrastive representations (e.g., from CLIP and R-R3M) perform quite poorly. Interestingly, $\mathcal{V}$ – **Cond** outperforms R-MVP and *MVP (EgoSoup)* on this task, *despite the absence of language input*, demonstrating that language supervision during pretraining can improve low-level feature learning, even relative to larger-scale models trained on much more data.

B. *Referring Expression Grounding*

Given a cluttered scene and language expression, the goal is to predict a bounding box around an object (e.g., "the blue black pen on the front left of the orange can" in Fig. 4; middle).

Motivation. Capturing object-centric priors and high-level semantics around properties such as color and spatial relationships is crucial across the entire robotics stack. More importantly, this is a *language-conditioned* task, allowing us to evaluate the impact of pretraining with language supervision.

TABLE III: **Results on Referring Expression Grounding.** We report average precision @ 0.25 IoU following Wang et al. [94] (OCID-Ref). This is a *language-conditioned* task; across various clutter levels, $\mathcal{V}$oltron models are substantially more performant than baselines, as well as models trained on more data and with alternative language supervision (e.g., CLIP).

	Architecture	Total	Minimum Clutter	Medium Clutter	Maximum Clutter
R-R3M	ViT-Small	63.30	63.87	68.34	55.33
R-MVP + DistilBERT	ViT-Small	49.58	50.98	53.83	41.94
$\mathcal{V}$ – **Cond** [Ours]	ViT-Small	**89.38**	**85.88**	**95.39**	**89.12**
$\mathcal{V}$ – **Cond** [Ours]	ViT-Base	**90.77**	**87.56**	**96.58**	**90.17**
CLIP	*ViT-Base*	*68.35*	*67.01*	*76.61*	*60.33*
MVP (EgoSoup) + DistilBERT	*ViT-Base*	*49.25*	*51.46*	*52.15*	*40.50*

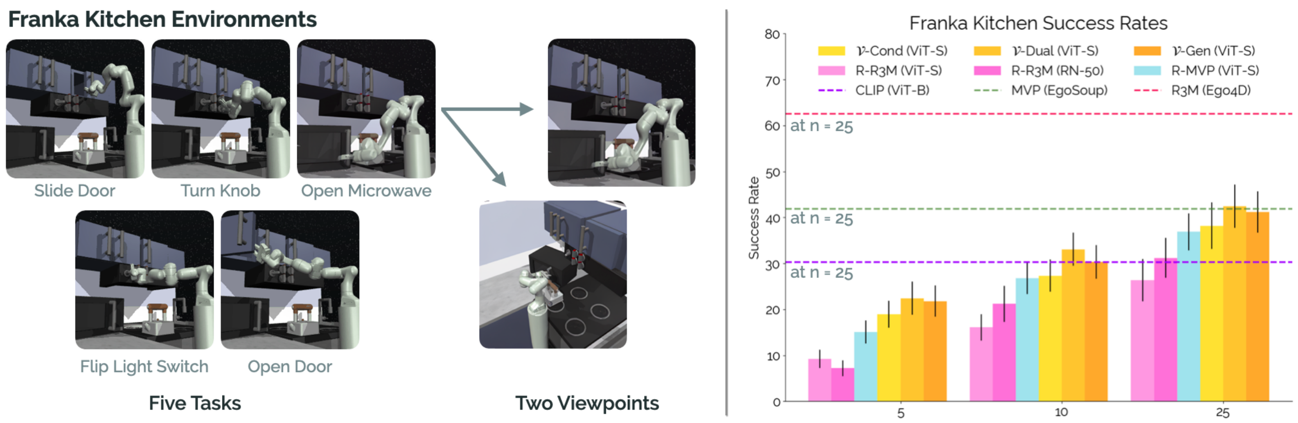


Fig. 5: **Franka Kitchen – Single-Task Visuomotor Control Results**. Visualization of the Franka Kitchen evaluation environments, comprised of five unique tasks, with two camera viewpoints **[Left]**. Results (success rate for each of n demonstrations) for $\mathcal{V}$oltron and baselines, showing the benefit of language-driven learning (over 3 seeds) **[Right]**. In dashed lines (not directly comparable), we plot *CLIP (ViT-B)*, *MVP (EgoSoup)*, and *R3M (Ego4D)* trained with $n = 25$ demonstrations.

Evaluation Details. We use the OCID-Ref Dataset [94] grounded in scenes that are representative of robotics settings; other datasets such as RefCoCo [100] are grounded in more global scenes (e.g., multiple humans playing frisbee on a field) that are less informative for robot learning. OCID-Ref also provides splits based on the clutter level of the underlying scene, letting us further evaluate robustness. We regress bounding box coordinates directly from our frozen features using a shallow MLP. All approaches condition on language (see expressions in Fig. 4), using the given language encoder where possible. This means using the multimodal encoder for $\mathcal{V}$ **– Cond** and the default learned text encoder for CLIP or R3M. However, for approaches that only learn visual representations (e.g., MVP), we append pretrained language features from DistilBERT – the same language model used to initialize $\mathcal{V}$oltron. We note again that we omit ResNet results; though this task did not require upsampling, we find trained models obtained no better than random performance, again indicating a need for a more sophisticated adaptation architecture (beyond the scope of this work). We report average precision at 0.25 IoU for each split following the evaluation procedure outlined in Wang et al. [94]. We provide additional details around the adaptation procedure in §E and the open-source code repositories.

Experimental Results. Results for each model across the various clutter splits are in Table III. $\mathcal{V}$oltron models are especially strong, vastly outperforming R-MVP by 40% and R-R3M by over 25% on all splits, showing that multimodal pretraining – even just conditioning on language when optimizing for masked reconstruction – can lead to substantial gains on downstream multimodal tasks. We isolate the massive performance gains of $\mathcal{V}$oltron models over prior work due to the multimodal encoder that learns *fused* embeddings of vision and language, allowing language to shape the visual representations during pretraining. In contrast, R3M, and CLIP models learn *independent* text encodings that are only fused post-hoc, during adaptation. This is even worse for MVP: these models need to learn to fuse their strong visual embeddings with the language embeddings from a completely different model (DistilBERT).

C. Single-Task Visuomotor Control

Learn a policy for a given task, predicting continuous joint actions given the proprioceptive state of a robotic arm, and a visual observation of the scene from an external camera.

Motivation. Imitation learning for visuomotor control has been the de-facto evaluation for prior work [62, 58, 65], giving us the closest comparison to the evaluations used in MVP and R3M. This evaluation focuses on *sample-efficient generalization*, measuring how well visual representations help in learning policies from limited demonstrations $n \in \{5, 10, 25\}$. This evaluation takes place in simulation.

Evaluation Details. We look at policy learning in the Franka Kitchen simulation environments as defined by Nair et al. [58]. This domain consists of 5 tasks, with 2 distinct camera viewpoints (Fig. 5). We learn shallow MLP policy heads via behavioral cloning that predict 9-DoF joint velocities (7 joints, 2 gripper) from our (frozen) visual features and proprioceptive state. We follow the R3M evaluation, reporting average success rates for each setting with n demonstrations across the 5 tasks, 2 viewpoints, and 3 random seeds. We train separate policies per task, with *no language conditioning* – using the exact code provided by Nair et al. [58]. Additional details are in §E and the open-source code.

Experimental Results. Most approaches perform similarly across the various number of training demonstrations (Fig. 5; right). However, we see some promising trends; $\mathcal{V}$oltron models perform better than both baselines, with approaches that learn from multiple frame contexts $\mathcal{V}$ **– Dual** and $\mathcal{V}$ **– Gen** showing *significant* improvements over single-frame approaches. Yet, the absolute success rates are low; learning for control is difficult, and while good visual representations can help, learning closed-loop policies from limited data remains an open challenge.

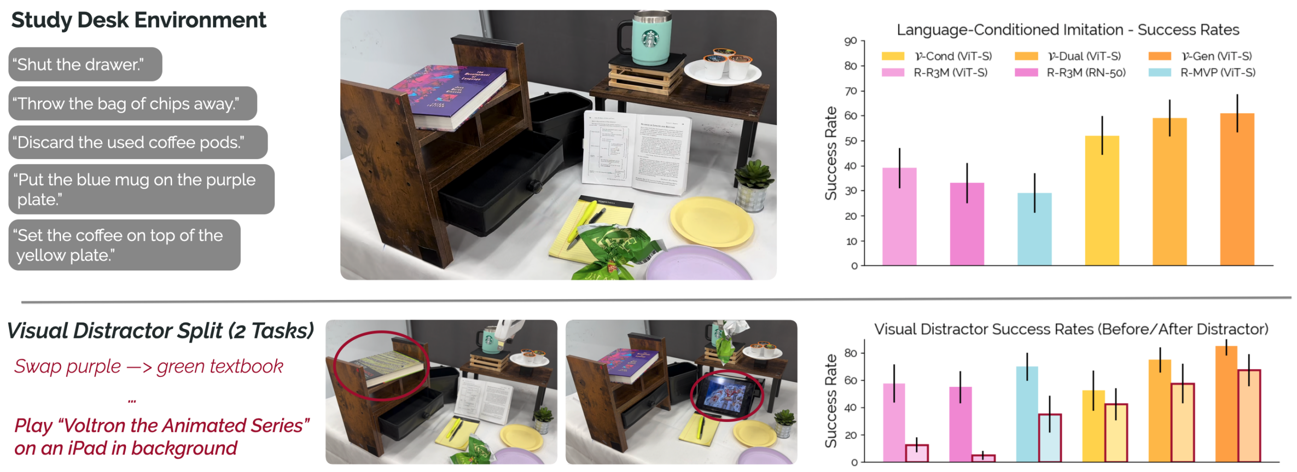


Fig. 6: **Real-World Language-Conditioned Imitation Learning Results**. The real-world "Study Desk" environment, with sample language instructions corresponding to the five behaviors we evaluate. **[Top]** The challenging *visual distractor* split for evaluating robustness to novel distractors, ranging from simple color swapping of background objects (e.g., purple to green textbook), to more drastic changes such as playing a clip from *"Voltron – the Animated Series"* in the background **[Bottom]**.

D. *Real-World Language-Conditioned Imitation*

Given a dataset of language instructions (e.g. "throw the bag of chips away") paired with demonstrations (in a real-world tabletop setting), learn an instruction following policy via behavioral cloning. Fig. 6 depicts the real-world environment.

Motivation. A large body of work looks at learning language-conditioned policies for human-robot collaborative settings [4, 86, 52, 41, 2]. This evaluation gets at the robustness and reliability of learned representations, with the goal of validating different approaches in real-robot settings.

Evaluation Details. We construct a "study desk" environment (Fig. 6) with five prototypical "tasks": 1) closing the drawer, 2) throwing the green bag of chips in the trash can, 3) discarding the used coffee pods, 4) moving the cyan coffee mug to the purple plate, and 5) moving the same mug to the yellow plate. For each task, we collect 20 teleoperated demonstrations at 10 Hz, randomly resetting the scene between episodes. We adopt the keyframe-based action space proposed in James and Davison [36] for learning. This approach heuristically breaks a demonstration into 4-5 "waypoints" (end-effector poses) that are used as action targets during behavior cloning; during policy execution, we plan min-jerk trajectories from the current position to the predicted waypoint, feeding the subsequent state and visual observation back to our policy [37, 81]. To collect diverse instructions, we prompt ChatGPT [version dated Jan 9th, 2023; 60] with simple task descriptions, asking it to generate diverse language instructions, collecting 25 utterances total (20 train, 5 held-out) per task.[2] We parameterize our policy similarly to §V-C, adding a shallow MLP on top of the extracted (frozen) visual representations [56]. This task is *language-conditioned*; as in OCID-Ref, we use the given language encoders for each approach where possible, appending DistilBERT features to pure visual representations otherwise. We report success rates with partial credit – 0.25 points for achieving each of the following "milestones": reaching an object, interacting with it, transporting it, and completing the task. We provide additional details in §E, and include videos of policy rollouts on the project page.

[2]ChatGPT Prompt (additional details and generated instructions on project page): *I'm trying to train a robot assistant that can follow diverse language instructions. One task requires moving an empty chip bag (a green bag of those jalapeno chips) to the garbage. Can you generate 25 natural-sounding instructions (e.g., "throw away the chips")?*

Experimental Results. Looking at success rates of the various representations (Fig. 6; top right) we see an exaggerated version of the trends exhibited in the single-task control setting; $\mathcal{V}$oltron models obtain an extra boost in performance across the board given that this task is language-conditioned, highlighting the strength of its fused representations. Similarly, R-R3M models exhibit the next best performance. Due to shared resource constraints, we do not run out *MVP (EgoSoup)*, *R3M (Ego4D)*, or *CLIP (ViT-B/16)*, though we expect similar trends.

E. *Qualitative: Zero-Shot Intent Scoring*

We perform a qualitative evaluation for the problem of language-based *intent scoring*; given a language expression describing an intent or behavior (e.g., "opening the faucet") and a corresponding video (that may or may not show the described behavior), predict an "alignment score" for each frame of a video. This alignment score should capture how well the current visual context matches the described behavior – ideally reflecting calibrated confidence over time (an example language/video is shown in Fig. 7; left).

Motivation. This evaluation is motivated by two active areas of research: reward learning from language and demonstrations [83, 77, 13, 5], and belief modeling for human-robot collaboration [31, 27, 6]. This evaluation probes for the ability to reason over intents and visual behaviors *jointly, without the need for additional data or supervision.*

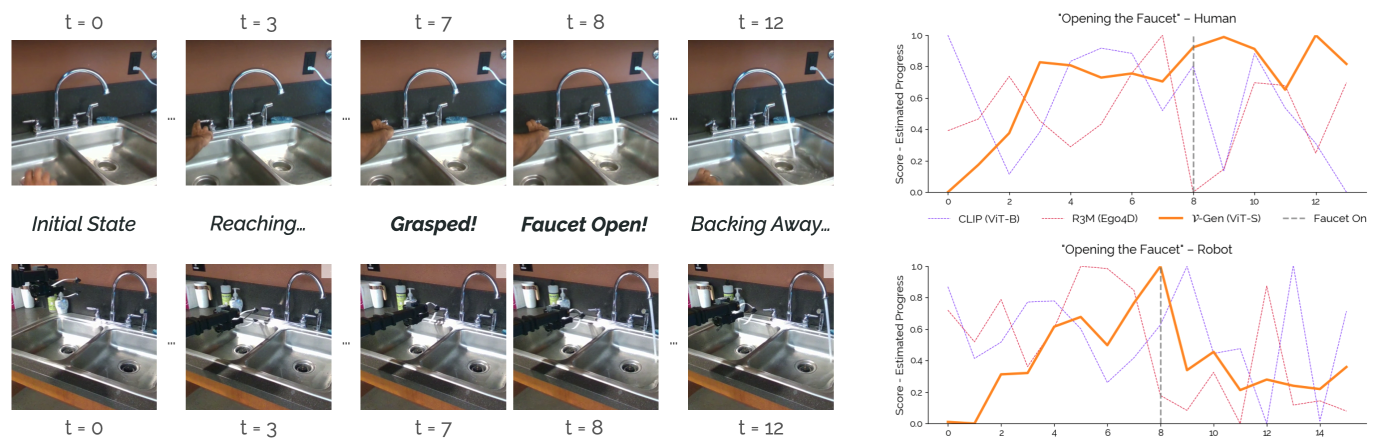


Fig. 7: **Qualitative Zero-Shot Intent Scoring Results.** Given a pair of videos from the WHiRL dataset [5] of a human and robot performing a task, we evaluate the ability of **$\mathcal{V}$ – Gen**, R3M (from Nair et al. [58]) and CLIP in scoring various frames subject to the utterance "opening the faucet." While CLIP and R3M produce extremely noisy scores, **$\mathcal{V}$ – Gen** is *calibrated*, successfully tracking progress over time – both for the human user, *as well as for the robot*.

Evaluation Details. This is a qualitative evaluation that focuses on measuring how well existing approaches "track" progress conditioned on a language intent over time. Doing this zero-shot means that we can only evaluate models that can produce alignment scores given language and visual context: 1) *CLIP (ViT-B/16)* through cosine similarity of learned vision and text representations, 2) *R3M (Ego4D)* through the "video-language alignment" head, and 3) our **$\mathcal{V}$ – Gen** model (by measuring the likelihood of a given language utterance conditioned on visual context under the language generator). Given a video of an agent performing some behavior described in language (e.g., "opening the faucet"), we estimate and plot scores under each model across a sequence of video frames. We use videos from WHiRL [5] of humans and robots performing the same tasks from different views; we choose to evaluate intent scoring for both agents to better capture the robustness and transfer potential for these approaches in similar real-world settings.

Experimental Results. The two curves in Fig. 7 show the predicted scores over time for the language intent "opening the faucet." Even though it has never been trained for this task, we find that **$\mathcal{V}$ – Gen** is able to coherently predict not only the exact frames corresponding to "keypoints" in each video (e.g., touching the handle, observing when the water starts running), but is also capable of measuring *partial progress* – akin to a shaped, dense reward; however, both *R3M (Ego4D)* and *CLIP (ViT-B/16)* fail at this task, predicting random scores with high variance across sequential time steps. Note that the intent scores are not perfect; after turning the faucet on for the human video, predicted scores remain high, while for the robot, the scores taper off. It is not clear why this happens, but given a small amount of adaptation data, one could ensure consistent behavior. We provide more examples from WHiRL in §C-E, and additional evaluation details in §E.

VI. Ablations, Extensions, & Further Analysis

The comparative results across the various evaluation problem domains paint $\mathcal{V}$oltron's language-driven representations in a favorable light relative to MVP and R3M baselines. Yet, there remain key questions that we address in this section: is language supervision actually driving these results? Why generative language modeling over masked language modeling? Will $\mathcal{V}$oltron scale?

Ablation: The Impact of Language Supervision. The second row of Table IV shows a subset of evaluation results across three different problem domains when training a "no-language" variant of the **$\mathcal{V}$ – Cond** architecture – this variant is in essence an alternate version of a masked autoencoder that uses the small architecture modifications we added for training stability in §IV. As such, it also serves as an *architecture ablation* when compared to the R-MVP results, enabling us to isolate the impact of the small stability modifications described in §IV. Indeed, the results confirm our hypotheses: first, removing language results in a definitive drop in performance across all evaluation applications. Second, the respective results for each evaluation application are on par with the corresponding results for the R-MVP model, demonstrating that the performance of $\mathcal{V}$oltron models does not stem from the architecture. We delve further into this ablation in §C-A.

Ablation: Generative vs. Masked Language Modeling. Looking at the $\mathcal{V}$oltron objective, a natural question to ask is why we chose *language generation* over *masked language modeling*. Furthermore, recent and concurrent work propose learning multimodal masked autoencoders (M3AE) both within and outside of robotics [23, 49], showing promising results in learning visual representations for image classification tasks, amongst others. To assess the differences, we choose to reproduce the M3AE model in a manner similar to our reproduction of MVP and R3M; we keep the same Something-

Something-v2 pretraining data, adopting the exact procedure described in Geng et al. [23], then evaluating the resulting representations on the same subset of evaluation domains as in the prior ablation (third row of Table IV). Surprisingly, we see drastic drops in performance *across the board*. Looking at the pretraining curves, we identify a possible reason for this failure: in optimizing M3AE on Sth-Sth, we see the language modeling loss go to zero almost *immediately*, leading to overfitting. A possible explanation is that the masked language modeling conditioned on visual contexts in datasets annotated with *short, predictable narrations* leads to degenerate representations, while generative language modeling is not susceptible to the same types of collapse; looking at ways to mitigate this seems like a promising direction for future work. Explicit details around pretraining and evaluating R-M3AE, with an in-depth discussion are in §C-B.

Extension: Scaling Up. Prior approaches have shown gains in scaling model capacity; here, we present preliminary evidence that $\mathcal{V}$oltron models behave similarly. For each evaluation in §V, we evaluate a ViT-Base variant of $\mathcal{V}$ **– Cond** (86M parameters vs. the 22M in the ViT-Small). We see universal improvement: Top-5% precision for grasping (Table II; middle row) increases by 15%, expression grounding accuracy improves (Table III; middle row), as does performance on control.

Extension: Robustness to Real-World Distractors. Factors such as lighting conditions, time of day, and accidental environment perturbations (e.g., a colleague knocking over the camera) can have a profound impact on performance of robotic systems, especially if learned representations are not robust. We run a limited "robustness" evaluation after training language-conditioned policies from the demonstrations described in §V-D. Success rates before and after introducing visual distractors for two of the "meta-tasks" are in Fig. 6 (bottom right).[3] We find that $\mathcal{V}$oltron and R-MVP models are robust to even the most extreme distractors – seemingly a benefit of per-patch masking coupled with MAP-based extraction.

TABLE IV: **Ablation Experiments.** We select a subset of evaluations from §V – grasp affordance prediction, referring expression grounding, and single-task visuomotor control.

	Grasp PR @ Top-1%	**Refer** Total Accuracy	**Imitate** (n = 25)
$\mathcal{V}$ + Lang [Ours]	80.71	89.38	38.2 ± 5.09
No-Language ↓	65.83	*53.44*	33.1 ± 4.79
R-M3AE ↓↓	*52.79*	*51.61*	*24.0 ± 4.21*

[3]We try five distractors spanning simple changes such as swapping the purple textbook in the background for a green one, to more extreme distractors such as playing a clip from *"Voltron, the Animated Series"* on a tablet in the middle of the workspace. Videos are on the project page.

VII. Discussion & Conclusion

We propose $\mathcal{V}$oltron, a framework for language-driven representation learning that balances *conditioning* and *generation* to shape the balance of low and high-level features captured. We introduce an evaluation suite spanning five diverse problems within robotics for holistically evaluating visual representations. Through controlled experiments and ablations, we validate the strengths of our representations; across all evaluation tasks, $\mathcal{V}$oltron models that balance language conditioning and generation strictly outperform prior approaches such as R3M and MVP, and in many cases show performance competitive with or exceeding that of approaches that use orders of magnitude more data or more expressive models.

Yet, while language is a pivotal source of supervision, there are still key questions to answer. Why is language-based pretraining helpful on tasks that have nothing to do with language? Why not try to learn *one model* that can encode both low-level and high-level features, without tradeoffs? While there is not a silver bullet *yet* we hope that future work takes a deep, grounded look at these questions, identifying what existing representations capture – and more importantly, what they miss. Our hope is that $\mathcal{V}$oltron serves as a starting point; a flexible, unified framework for future improvements in visual representation learning for robotics.

Acknowledgements

This work would not have been possible without the support of entire communities of students, engineers, and various domain experts; our gratitude cannot be understated. We would specifically like to thank Shyamal Buch, David Hall, and John Thickstun for their invaluable advice and suggestions around pretraining and evaluation. We further thank Dilip Arumugam, Suneel Belkhale, Masha Itkina, Tyler Lum, Vivek Myers, Karl Pertsch, and Blake Wulfe for their feedback on earlier drafts.

Toyota Research Institute ("TRI") provided funds to support this work. This research – specifically model pretraining – was supported with Cloud TPUs from Google's TPU Research Cloud (TRC). Siddharth Karamcheti is grateful to be supported by the Open Philanthropy Project AI Fellowship. Annie Chen is supported by the NSF Graduate Research Fellowship (NSF GRFP). Finally, we thank the Stanford ILIAD, IRIS, and NLP groups for their unwavering support and valuable discussions.

References

[1] Armen Aghajanyan, Bernie Huang, Candace Ross, Vladimir Karpukhin, Hu Xu, Naman Goyal, Dmytro Okhonko, Mandar Joshi, Gargi Ghosh, Mike Lewis, and Luke Zettlemoyer. Cm3: A causal masked multimodal model of the internet. *arXiv preprint arXiv:2201.07520*, 2022.

[2] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Daniel Ho, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Eric Jang, Rosario Jauregui Ruano, Kyle Jeffrey, Sally Jesmonth, Nikhil Jayant Joshi, Ryan C. Julian, Dmitry Kalashnikov, Yuheng Kuang, Kuang-Huei Lee, Sergey Levine, Yao Lu, Linda Luu, Carolina Parada, Peter Pastor, Jornell Quiambao, Kanishka Rao, Jarek Rettinghouse, Diego M Reyes, Pierre Sermanet, Nicolas Sievers, Clayton Tan, Alexander Toshev, Vincent Vanhoucke, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, and Mengyuan Yan. Do as I can, not as I say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

[3] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andy Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. Flamingo: a visual language model for few-shot learning. *arXiv preprint arXiv:2204.14198*, 2022.

[4] Dilip Arumugam, Siddharth Karamcheti, Nakul Gopalan, Lawson L. S. Wong, and Stefanie Tellex. Accurately and efficiently interpreting human-robot instructions of varying granularities. In *Robotics: Science and Systems (RSS)*, 2017.

[5] Shikhar Bahl, Abhi Gupta, and Deepak Pathak. Human-to-robot imitation in the wild. In *Robotics: Science and Systems (RSS)*, 2022.

[6] Tirthankar Bandyopadhyay, Kok Sung Won, Emilio Frazzoli, David Hsu, Wee Sun Lee, and Daniela Rus. Intention-aware motion planning. In *Workshop for the Algorithmic Foundations of Robotics (WAFR)*, 2013.

[7] Hangbo Bao, Li Dong, and Furu Wei. BEiT: BERT pre-training of image transformers. In *International Conference on Learning Representations (ICLR)*, 2022.

[8] Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[9] Elad Ben-Zaken, Shauli Ravfogel, and Yoav Goldberg. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Association for Computational Linguistics (ACL)*, 2022.

[10] Jeannette Bohg, Antonio Morales, Tamim Asfour, and Danica Kragic. Data-driven grasp synthesis—a survey. *IEEE Transactions on Robotics (T-RO)*, 30:289–309, 2013.

[11] Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie Chen, Kathleen Creel, Jared Quincy Davis, Dorottya Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Ben Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, Julian Nyarko, Giray Ogut, Laurel Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Rob Reich, Hongyu Ren, Frieda Rong, Yusuf Roohani, Camilo Ruiz, Jack Ryan, Christopher Ré, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishnan Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

[12] François Chaumette and Seth A. Hutchinson. Visual servo control. i. basic approaches. *IEEE Robotics & Automation Magazine*, 13:82–90, 2006.

[13] Annie S. Chen, Suraj Nair, and Chelsea Finn. Learning generalizable robotic reward functions from "in-the-wild" human videos. In *Robotics: Science and Systems (RSS)*, 2021.

[14] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning (ICML)*, pages 1597–1607, 2020.

[15] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, A. Rao, Parker Barnes,

Yi Tay, Noam M. Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, B. Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, M. Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, S. Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier García, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, D. Luan, Hyeontaek Lim, Barret Zoph, A. Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, T. S. Pillai, Marie Pellat, Aitor Lewkowycz, E. Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, K. Meier-Hellstern, D. Eck, J. Dean, Slav Petrov, and Noah Fiedel. PaLM: Scaling language modeling with pathways. *arXiv*, 2022.

[16] Nikolaus Correll, Kostas E. Bekris, Dmitry Berenson, Oliver Brock, Albert J. Causo, Kris K. Hauser, Kei Okada, Alberto Rodriguez, Joseph M. Romano, and Peter R. Wurman. Analysis and observations from the first amazon picking challenge. *Science*, 15:172–188, 2016.

[17] Yuchen Cui, Scott Niekum, Abhi Gupta, Vikash Kumar, and Aravind Rajeswaran. Can foundation models perform zero-shot task specification for robot manipulation? In *Learning for Dynamics & Control Conference (L4DC)*, 2022.

[18] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Scaling egocentric vision: The EPIC-KITCHENS dataset. In *European Conference on Computer Vision (ECCV)*, 2018.

[19] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, 2009.

[20] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Association for Computational Linguistics (ACL)*, pages 4171–4186, 2019.

[21] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? The KITTI vision benchmark suite. In *Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361, 2012.

[22] Carles Gelada, Saurabh Kumar, Jacob Buckman, Ofir Nachum, and Marc G. Bellemare. Deepmdp: Learning continuous latent space models for representation learning. In *International Conference on Machine Learning (ICML)*, 2019.

[23] Xinyang Geng, Hao Liu, Lisa Lee, Dale Schuurams, Sergey Levine, and P. Abbeel. Multimodal masked autoencoders learn transferable representations. *arXiv preprint arXiv:2205.14204*, 2022.

[24] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzynska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fründ, Peter N. Yianilos, Moritz Mueller-Freitag, Florian Hoppe, Christian Thurau, Ingo Bax, and Roland Memisevic. The "something something video database for learning and evaluating visual common sense. In *International Conference on Computer Vision (ICCV)*, 2017.

[25] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Q. Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, Miguel Martin, Tushar Nagarajan, Ilija Radosavovic, Santhosh K. Ramakrishnan, F. Ryan, Jayant Sharma, Michael Wray, Mengmeng Xu, Eric Z. Xu, Chen Zhao, Siddhant Bansal, Dhruv Batra, Vincent Cartillier, Sean Crane, Tien Do, Morrie Doulaty, Akshay Erapalli, Christoph Feichtenhofer, Adriano Fragomeni, Qichen Fu, Christian Fuegen, Abrham Gebreselasie, Cristina González, James M. Hillis, Xuhua Huang, Yifei Huang, Wenqi Jia, Weslie Yu Heng Khoo, Jáchym Kolár, Satwik Kottur, Anurag Kumar, Federico Landini, Chao Li, Yanghao Li, Zhenqiang Li, Karttikeya Mangalam, Raghava Modhugu, Jonathan Munro, Tullie Murrell, Takumi Nishiyasu, Will Price, Paola Ruiz Puentes, Merey Ramazanova, Leda Sari, Kiran K. Somasundaram, Audrey Southerland, Yusuke Sugano, Ruijie Tao, Minh Vo, Yuchen Wang, Xindi Wu, Takuma Yagi, Yunyi Zhu, Pablo Arbeláez, David J. Crandall, Dima Damen, Giovanni Maria Farinella, Bernard Ghanem, Vamsi Krishna Ithapu, C. V. Jawahar, Hanbyul Joo, Kris Kitani, Haizhou Li, Richard A. Newcombe, Aude Oliva, Hyun Soo Park, James M. Rehg, Yoichi Sato, Jianbo Shi, Mike Zheng Shou, Antonio Torralba, Lorenzo Torresani, Mingfei Yan, and Jitendra Malik. Ego4D: Around the world in 3,000 hours of egocentric video. In *Computer Vision and Pattern Recognition (CVPR)*, 2022.

[26] Danijar Hafner, Timothy P. Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations (ICLR)*, 2020.

[27] Kris K. Hauser. Recognition, prediction, and planning for assisted teleoperation of freeform tasks. *Autonomous Robots (AURO)*, pages 241–254, 2012.

[28] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition (CVPR)*, 2016.

[29] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross B. Girshick. Masked autoencoders are scalable vision learners. In *Computer Vision and Pattern Recognition (CVPR)*, 2022.

[30] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[31] Guy Hoffman and Cynthia Breazeal. Cost-based anticipatory action selection for human–robot fluency. *IEEE Transactions on Robotics (T-RO)*, 23:952–961, 2007.

[32] Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-

efficient transfer learning for NLP. *arXiv*, 2019.

[33] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

[34] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning (ICML)*, pages 448–456, 2015.

[35] Andrew Jaegle, Felix Gimeno, Andrew Brock, Andrew Zisserman, Oriol Vinyals, and João Carreira. Perceiver: General perception with iterative attention. In *International Conference on Machine Learning (ICML)*, 2021.

[36] Stephen James and Andrew J. Davison. Q-Attention: Enabling efficient learning for vision-based robotic manipulation. *IEEE Robotics and Automation Letters (RA-L)*, 7:1612–1619, 2022.

[37] Stephen James, Kentaro Wada, Tristan Laidlow, and Andrew J. Davison. Coarse-to-fine Q-Attention: Efficient learning for visual robotic manipulation via discretisation. In *Computer Vision and Pattern Recognition (CVPR)*, pages 13729–13738, 2022.

[38] Shervin Javdani, Henny Admoni, Stefania Pellegrinelli, Siddhartha S Srinivasa, and J Andrew Bagnell. Shared autonomy via hindsight optimization for teleoperation and teaming. *International Journal of Robotics Research (IJRR)*, 37:717–742, 2018.

[39] Rico Jonschkowski and Oliver Brock. Learning state representations with robotic priors. *Autonomous Robots*, 39:407–428, 2015.

[40] Siddharth Karamcheti, Laurel Orr, Jason Bolton, Tianyi Zhang, Karan Goel, Avanika Narayan, Rishi Bommasani, Deepak Narayanan, Tatsunori Hashimoto, Dan Jurafsky, Christopher D. Manning, Christopher Potts, Christopher Ré, and Percy Liang. Mistral - a journey towards reproducible language model training, 2021.

[41] Siddharth Karamcheti, Megha Srivastava, Percy Liang, and Dorsa Sadigh. LILA: Language-informed latent actions. In *Conference on Robot Learning (CoRL)*, 2021.

[42] Apoorv Khandelwal, Luca Weihs, Roozbeh Mottaghi, and Aniruddha Kembhavi. Simple but effective: CLIP embeddings for embodied AI. In *Computer Vision and Pattern Recognition (CVPR)*, pages 14809–14818, 2021.

[43] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015.

[44] Ilya Kostrikov, Denis Yarats, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *International Conference on Learning Representations (ICLR)*, 2021.

[45] Michael Laskin, Kimin Lee, Adam Stooke, Lerrel Pinto, P. Abbeel, and A. Srinivas. Reinforcement learning with augmented data. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[46] Juho Lee, Yoonho Lee, Jungtaek Kim, Adam R. Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International Conference on Machine Learning (ICML)*, 2018.

[47] S. Levine, Chelsea Finn, Trevor Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research (JMLR)*, 17, 2016.

[48] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common objects in context. In *European Conference on Computer Vision (ECCV)*, pages 740–755, 2014.

[49] Hao Liu, Lisa Lee, Kimin Lee, and Pieter Abbeel. Instructrl: Simple yet effective instruction-following agents with multimodal transformer. *arXiv preprint arXiv:2210.13431*, 2022.

[50] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. ViLBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[51] Jiasen Lu, Christopher Clark, Rowan Zellers, Roozbeh Mottaghi, and Aniruddha Kembhavi. Unified-io: A unified model for vision, language, and multi-modal tasks. In *International Conference on Learning Representations (ICLR)*, 2023.

[52] Corey Lynch and Pierre Sermanet. Grounding language in play. *arXiv preprint arXiv:2005.07648*, 2020.

[53] Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. Vip: Towards universal visual reward and representation via value-implicit pre-training. *arXiv preprint arXiv:2210.00030*, 2022.

[54] Jeffrey Mahler, Jacky Liang, Sherdil Niyaz, Michael Laskey, Richard Doan, Xinyu Liu, Juan Aparicio Ojea, and Ken Goldberg. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. In *Robotics: Science and Systems (RSS)*, 2017.

[55] Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. In *Conference on Robot Learning (CoRL)*, 2021.

[56] Dipendra K. Misra, John Langford, and Yoav Artzi. Mapping instructions and visual observations to actions with reinforcement learning. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2017.

[57] Suraj Nair, Eric Mitchell, Kevin Chen, Brian Ichter, Silvio Savarese, and Chelsea Finn. Learning language-conditioned robot behavior from offline data and crowd-sourced annotation. In *Conference on Robot Learning (CoRL)*, 2021.

[58] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint*

arXiv:2203.12601, 2022.

[59] Sharan Narang, Hyung Won Chung, Yi Tay, William Fedus, Thibault Févry, Michael Matena, Karishma Malkan, Noah Fiedel, Noam M. Shazeer, Zhenzhong Lan, Yanqi Zhou, Wei Li, Nan Ding, Jake Marcus, Adam Roberts, and Colin Raffel. Do transformer modifications transfer across implementations and applications? In *Empirical Methods in Natural Language Processing (EMNLP)*, 2021.

[60] OpenAI. Chatgpt: Optimizing language models for dialogue, 2022.

[61] Jyothish Pari, Nur Muhammad (Mahi) Shafiullah, Sridhar Pandian Arunachalam, and Lerrel Pinto. The surprising effectiveness of representation learning for visual imitation. In *Robotics: Science and Systems (RSS)*, 2022.

[62] Simone Parisi, Aravind Rajeswaran, Senthil Purushwalkam, and Abhinav Kumar Gupta. The unsurprising effectiveness of pre-trained vision models for control. *arXiv preprint arXiv:2203.03580*, 2022.

[63] Ofir Press, Noah A. Smith, and Mike Lewis. Train short, test long: Attention with linear biases enables input length extrapolation. In *International Conference on Learning Representations (ICLR)*, 2022.

[64] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning (ICML)*, volume 139, pages 8748–8763, 2021.

[65] Ilija Radosavovic, Tete Xiao, Stephen James, P. Abbeel, Jitendra Malik, and Trevor Darrell. Real-world robot learning with masked visual pre-training. In *Conference on Robot Learning (CoRL)*, 2022.

[66] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019.

[67] Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do vision transformers see like convolutional neural networks? In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

[68] Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, Tom Eccles, Jake Bruce, Ali Razavi, Ashley D. Edwards, Nicolas Manfred Otto Heess, Yutian Chen, Raia Hadsell, Oriol Vinyals, Mahyar Bordbar, and Nando de Freitas. A generalist agent. *arXiv preprint arXiv:2205.06175*, 2022.

[69] Machel Reid, Yutaro Yamada, and Shixiang Shane Gu. Can Wikipedia help offline reinforcement learning? *arXiv preprint arXiv:2201.12122*, 2022.

[70] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

[71] Ashutosh Saxena, Justin Driemeyer, and A. Ng. Robotic grasping of novel objects using vision. *International Journal of Robotics Research (IJRR)*, 27:157–173, 2008.

[72] Christoph Schuhmann, Richard Vencu, Romain Beaumont, Robert Kaczmarczyk, Clayton Mullis, Aarush Katta, Theo Coombes, Jenia Jitsev, and Aran Komatsuzaki. LAION-400M: Open dataset of CLIP-filtered 400 million image-text pairs. *arXiv preprint arXiv:2111.02114*, 2021.

[73] Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, Patrick Schramowski, Srivatsa Kundurthy, Katherine Crowson, Ludwig Schmidt, Robert Kaczmarczyk, and Jenia Jitsev. LAION-5B: An open large-scale dataset for training next generation image-text models. In *Neural Information Processing Systems Track on Datasets and Benchmarks (NeurIPS Datasets and Benchmarks)*, 2022.

[74] Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, and Sergey Levine. Time-contrastive networks: Self-supervised learning from video. In *International Conference on Robotics and Automation (ICRA)*, pages 1134–1141, 2018.

[75] Rutav Shah and Vikash Kumar. Rrl: Resnet as representation for reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2021.

[76] Dandan Shan, Jiaqi Geng, Michelle Shu, and David F. Fouhey. Understanding human hands in contact at internet scale. In *Computer Vision and Pattern Recognition (CVPR)*, pages 9866–9875, 2020.

[77] Lin Shao, Toki Migimatsu, Q. Zhang, Karen Yang, and Jeannette Bohg. Concept2robot: Learning manipulation concepts from instructions and human demonstrations. In *Robotics: Science and Systems (RSS)*, 2020.

[78] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Association for Computational Linguistics (ACL)*, 2018.

[79] Noam M. Shazeer. GLU variants improve transformer. *arXiv preprint arXiv:2002.05202*, 2020.

[80] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Cliport: What and where pathways for robotic manipulation. In *Conference on Robot Learning (CoRL)*, 2021.

[81] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning (CoRL)*, 2022.

[82] Amanpreet Singh, Ronghang Hu, Vedanuj Goswami, Guillaume Couairon, Wojciech Galuba, Marcus Rohrbach, and Douwe Kiela. FLAVA: A foundational

language and vision alignment model. In *Computer Vision and Pattern Recognition (CVPR)*, pages 15617–15629, 2022.

[83] Laura Smith, Nikita Dhawan, Marvin Zhang, P. Abbeel, and Sergey Levine. Avid: Learning multi-stage tasks via pixel-level translation of human videos. In *Robotics: Science and Systems (RSS)*, 2020.

[84] A. Srinivas, Michael Laskin, and P. Abbeel. CURL: Contrastive unsupervised representations for reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2020.

[85] Krishna Srinivasan, Karthik Raman, Jiecao Chen, Michael Bendersky, and Marc Najork. Wit: Wikipedia-based image text dataset for multimodal multilingual machine learning. In *ACM Special Interest Group on Information Retreival (SIGIR)*, 2021.

[86] Simon Stepputtis, J. Campbell, Mariano Phielipp, Stefan Lee, Chitta Baral, and H. B. Amor. Language-conditioned imitation learning for robot manipulation tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[87] Robin Strudel, Ricardo Garcia Pinel, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for semantic segmentation. In *International Conference on Computer Vision (ICCV)*, pages 7242–7252, 2021.

[88] Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021.

[89] Stefanie Tellex, Thomas Kollar, Steven Dickerson, Matthew R Walter, Ashis Gopal Banerjee, Seth J Teller, and Nicholas Roy. Understanding natural language commands for robotic navigation and mobile manipulation. In *Association for the Advancement of Artificial Intelligence (AAAI)*, 2011.

[90] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. VideoMAE: Masked autoencoders are data-efficient learners for self-supervised video pre-training. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[91] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with image transformers. In *International Conference on Computer Vision (ICCV)*, pages 32–42, 2021.

[92] Ashish Vaswani, Yinggong Zhao, Victoria Fossum, and David Chiang. Decoding with large-scale neural language models improves translation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1387–1392, 2013.

[93] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

[94] Ke-Jyun Wang, Yun-Hsuan Liu, Hung-Ting Su, Jen-Wei Wang, Yu-Siang Wang, Winston H. Hsu, and Wen-Chin Chen. OCID-Ref: A 3d robotic dataset with embodied language for clutter scene grounding. In *Association for Computational Linguistics (ACL)*, 2021.

[95] Lee E. Weiss, Arthur C. Sanderson, and Charles P. Neuman. Dynamic sensor-based control of robots with visual feedback. *IEEE Robotics and Automation Letters (RA-L)*, 3:404–417, 1987.

[96] Ross Wightman. Pytorch image models. https://github.com/rwightman/pytorch-image-models, 2019.

[97] Tete Xiao, Ilija Radosavovic, Trevor Darrell, and Jitendra Malik. Masked visual pre-training for motor control. *arXiv preprint arXiv:2203.06173*, 2022.

[98] Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

[99] Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, and Yonghui Wu. Coca: Contrastive captioners are image-text foundation models. *arXiv preprint arXiv:2205.01917*, 2022.

[100] Licheng Yu, Patrick Poirson, Shan Yang, Alexander C. Berg, and Tamara L. Berg. Modeling context in referring expressions. In *European Conference on Computer Vision (ECCV)*, 2016.

[101] Andy Zeng, Shuran Song, Kuan-Ting Yu, Elliott Donlon, Francois Robert Hogan, Maria Bauzá, Daolin Ma, Orion Taylor, Melody Liu, Eudald Romo, Nima Fazeli, Ferran Alet, Nikhil Chavan Dafle, Rachel Holladay, Isabella Morona, Prem Qu Nair, Druck Green, Ian Taylor, Weber Liu, Thomas A. Funkhouser, and Alberto Rodriguez. Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching. *International Journal of Robotics Research (IJRR)*, 41: 690–705, 2017.

[102] Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *Computer Vision and Pattern Recognition (CVPR)*, pages 1204–1213, 2022.

[103] Amy Zhang, Rowan McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning invariant representations for reinforcement learning without reconstruction. In *International Conference on Learning Representations (ICLR)*, 2021.

[104] Biao Zhang and Rico Sennrich. Root mean square layer normalization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[105] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip H. S. Torr, and Li Zhang. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *Computer Vision and Pattern Recognition (CVPR)*, 2021.

Overview

In the appendices below, we provide additional details around the implementation, pretraining, and adaptation procedures described in the main text, in addition to delving deeper into various discussions. Finally, we add additional results and visualizations that further complement the findings from the main text.

We provide open-source code for loading and using pretraining models, hosted links for our preprocessing splits (including the actual batches seen during training), and a separate, standalone open-source code repository for our evaluation suite. Our hope is that the evaluation suite especially is general and easy to use for downstream work on evaluating learned representations.

The full manifest of resources are as follows:

- Project Page (videos & additional links): https://sites.google.com/view/voltron-robotics
- Open-Source Modeling Repository (pretraining & model usage): https://github.com/siddk/voltron-robotics
- Open-Source Evaluation Suite (API for automated evaluation): https://github.com/siddk/voltron-evaluation

All code is in PyTorch; however, the evaluation code can be easily extended to support various frameworks and infrastructure.

An overview of each appendix can be found below. We further indicate which parts of the appendices are best viewed here in the text or on the project page; for videos and visualizations, we highly recommend navigating to the latter.

---

§C – *Additional Results & Visualization*

We report additional results and visualizations from experiments mentioned in the main text, as well as other experiments that further support our conclusions.

§C-A – *Analysis: Impact of Language-Conditioning on Reconstruction Loss*

We revisit the language vs. no-language ablation from the main text, looking at pretraining curves to help explain why language is so helpful as a supervision signal. We find that language-conditioning significantly lowers reconstruction loss, allowing models to pick up on more low-level features.

§C-B – *Analysis: Generative vs. Masked Language Modeling*

We look further at the masked language modeling ablation from the main text, via the reproduction of Multimodal Masked Autoencoders [M3AE; 23]. We find in the pretraining curves high evidence of overfitting with masked models early in training, impacting the learned representations.

§C-C – *Results: Adroit Visuomotor Control*

We present results on the Adroit Visumotor Control environments from Nair et al. [58], finding that while language is again superior, *higher-level features perform better*. This is preliminary evidence that even for individual evaluation domains (e.g., single-task visuomotor control), there is no silver bullet; different types of representations perform differently.

§C-D – *Qualitative: Real-Robot Language-Conditioned Policy Rollouts*

Visualizations of real-world policy rollouts from the various representation learning approaches.

§C-E – *Qualitative: Additional Intent Scoring Visualizations*

Additional intent scoring visualizations using videos from the WHiRL dataset [5].

§D – *Data-Equivalent Reproductions & Reproducibility*

We provide additional discussion around the reproductions of MVP and R3M on the Something-Something-v2 dataset:

§D-A – *Additional Preprocessing Discussion*

Additional discussion of how we preprocess Something-Something-v2 [Sth-Sth; 24] for pretraining, with a comparison of how prior work such as MVP source and process pretraining data.

§D-B – *Multiheaded Attention Pooling – Feature Extraction*

Detailed explanation of the Multiheaded Attention Pooling [MAP; 46] feature extraction strategy, with analysis and results comparing to alternative methods.

§E – *Adapting Representations for Evaluation*

We provide further descriptions of the adaptation pipeline for each of the five evaluation domains.

Appendix A
Motivating Questions

Q1. *From the results, some $\mathcal{V}$oltron models outperform larger models such as MVP-Base trained on significantly more data, even on tasks that do not necessarily need language information. How do you make sense of this?*

We find that in many of our evaluation domains, especially domains with episodic tasks such as single-task and language-conditioned imitation learning, it is important to discern differences *across frames in the same overall visual context*, or otherwise pay attention to small visual distinctions. Looking at the original MVP work [65], we see that the original pretraining datasets are compiled by sampling frames from various video datasets *once, in a single-step procedure*, at low sampling rates. For many datasets (such as Sth-Sth and Ego4D), this means only seeing 1-2 frames per video clip in total during training.

In contrast, when we sample data from Sth-Sth, we ensure to sample *at least 5 frames per clip, per epoch*; while the aggregate amount of diverse contexts is much lower than in the original MVP work, seeing multiple frames per context seems to significantly help learning, *and not just for $\mathcal{V}$oltron models*! On the tasks where $\mathcal{V}$oltron models outperform *MVP (EgoSoup)* (with a larger ViT-Base encoder), we also see commensurate gains in our reproductions R-MVP and R-R3M. For example, R-MVP is at par with or only slightly less performant than *MVP (EgoSoup)* on grasp affordance prediction and single-task control. We offer further discussion in §D-A.

Q2. *Why don't you evaluate models trained with $\alpha = 1$ (pure language generation)?*

In preliminary experiments, we partially pretrained variants of $\mathcal{V}$ **– Gen** with values $\alpha = 0.25, 0.5, 0.75$; we focused on evaluating the downstream performance of these representations in the context of the single task visuomotor control evaluation. With $\alpha = 0.75$ we observed significant performance degradation on control tasks; furthermore, looking at the pretraining loss curves, we saw the reconstruction error plateau early in training. We found that $\alpha = 0.5$ balanced learning, and allowed us to continue to push reconstruction error down while also pushing the language generator loss (cross-entropy) lower; with $\alpha = 0.25$, we saw the opposite trend as with $\alpha = 0.75$.

These results are to be taken with a grain of salt, given the limited pretraining duration. However, we worry that with $\alpha = 1$, we might suffer doubly for 1) never conditioning on language, which is so clearly helpful from our results, and 2) potentially fall into the same failure mode as the R-M3AE multimodal masked autoencoder from Section §VI in the main text, overfitting to the language loss. In general, $\mathcal{V}$ **– Gen** with $\alpha = 0.5$ already converges to a substantially higher reconstruction loss as $\mathcal{V}$ **– Cond** and $\mathcal{V}$ **– Dual**, as shown in the pretraining curves in §B-C. That being said, it is a promising avenue for future work to understand if this is inherent or a problem with the specific optimization procedure we used – perhaps changing the relative scaling of the two losses over the course of pretraining may mitigate this issue, or even adaptively clipping the gradient updates depending on the relative contribution of the visual reconstructor or language generator.

Q3. *Why does language during pretraining help for downstream tasks that don't use language?*

Consider a masked visual input of a "black, curved object above a wooden surface." Given this information – and this information alone – what is a plausible reconstruction? There are myriad objects that fit those percepts – a black, curved object: we could lbe looking at the side of a bowl, the handle of a briefcase, the arm of a chair or stool, or in general, any number of possible options. A masked autoencoder optimizing for reconstruction must embed in the representation of this input as many of the features possible to enable good downstream reconstruction loss. It needs to model *everything*, as the visual context is under-specified and ambiguous. This compressed bottleneck is core to learning a masked autoencoder, but the unfortunate byproduct of this – in light of a vast world of possibilities – are representations that try to capture *everything* they possibly can.

Contrast this with a world in which you are told that the same visual context is associated with the language caption "lifting a black coffee mug on the table." What changes? The posterior over possible objects collapses down to the narrow slice of possibilities captured by "black coffee mug"; under this new set of possibilities, what does the encoder focus on? What *type* of black coffee mug is on the table? If it is being lifted, *how* is it being lifted? From what part of the object – the handle (seen in frame), or somewhere else? What are the features that help further reconstruct the black coffee mug? The other nearby surfaces – what is the mug resting on (a wooden table? the wooden arm of a chair?), is it at an angle? The additional visual context – what type of scene are we in – a living room, a coffeehouse? What else can I specifically encode that helps me reconstruct this cup in high-fidelity? The edges of the cup, its texture, the way the light is reflecting off of it in this particular visible context?

Conditioning on a language description both *simplifies* and *focuses* what I need to represent. My encoded features are no longer general enough to cover the full range of objects that could follow from the visible context alone; instead, I can use that same capacity to represent *this specific context, as denoted by language*. The encoder can focus on all of things left *unspecified* by language – arguably, the very things we *want* a visual encoder for robotics to represent. Because we know that it is a "black coffee mug," we can encode features around different types of black coffee mugs as a first level, and at a second level, go deeper, and actually model the *low-level* features that are not tied to semantics, but tied to core, perceptual primitives: the texture of the mug, the edges/boundaries of the object relative to other objects, even the way light reflects off of the surface. These are the features that help in tasks like grasp affordance prediction (the edges of objects), and when we learn joint representations of language and vision, the features that help with localization (grounding referring expressions) and detection. Though speculative, we can attempt to make this concrete with results: if language is indeed reducing the space over plausible reconstructions (and *focusing* the encoder), we might expect lower reconstruction error when language-conditioning vs. when we condition solely on the visual context alone. This is exactly what we show in §C-A, and a hint at why $\mathcal{V}$oltron is able to perform so strongly downstream (even without language input). The simple presence of language during pretraining refocuses the features in our representations.

Q4. *Why only evaluate frozen representations? Why not fully finetune the backbones for each downstream evaluation?*

Both MVP and R3M [65, 58] only evaluate frozen visual representations, following a precedent set by a long tradition of work in self-supervised learning from the computer vision community [14, 64, 29]. There are two reasons for the validity of evaluating frozen representations. First, the hope is that evaluating frozen representations (via adapted per-evaluation "heads" on top) help us isolate the relative impact of what the representations contain – otherwise, the separation between the standalone representations and the downstream evaluation parameters (and the co-mingling of the two when optimizing all weights via gradient descent) becomes much less clear. Second, for many of the evaluations we look at, we have extremely small amounts of data – on the order of 1000 examples for grasp affordance prediction, 10 - 20 demonstrations for single task and language-conditioned imitation. There is a valid fear that full-finetuning the sizeable visual encoders vs. just the adaptation parameters (< 50K parameters) could lead to extreme overfitting. In general, finetuning Transformers with minimal data is an active area of research in and of itself, with work like adapters, low-rank approximations, and partial finetuning [32, 33, 9].

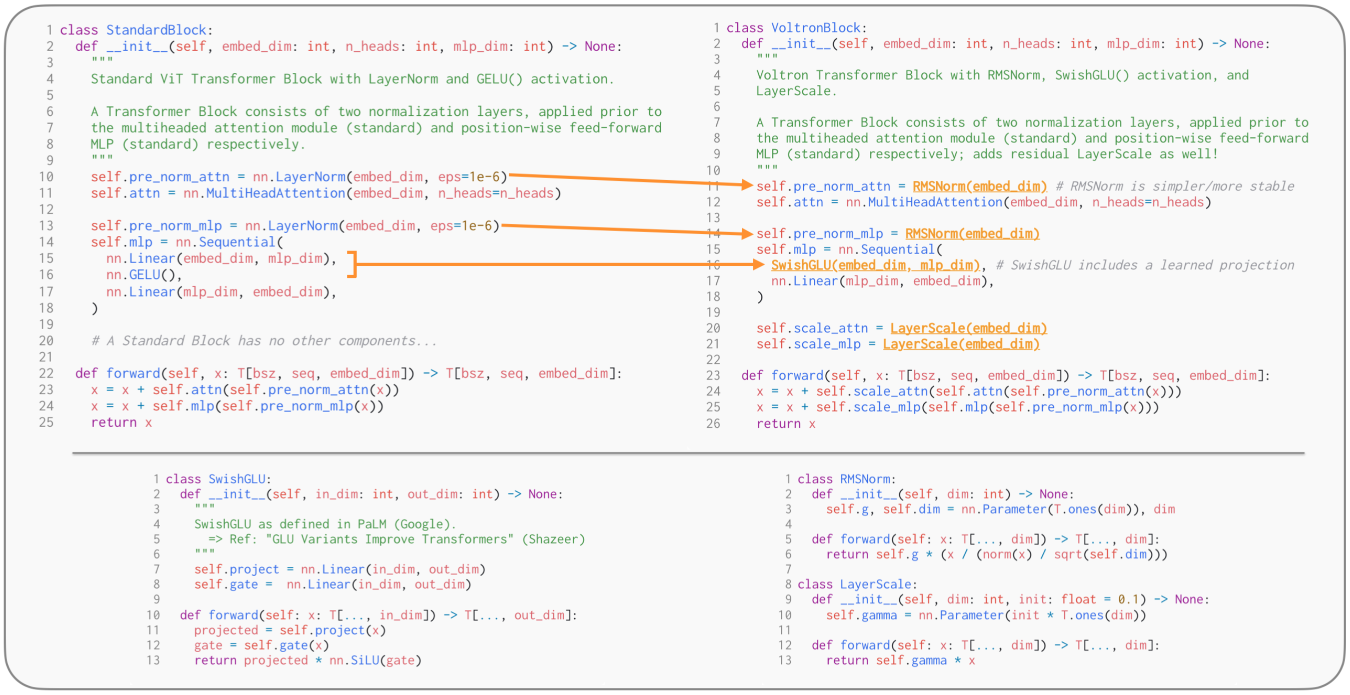
Q5. *Assuming pretraining datasets of (video, language) pairs feels restrictive; is there a way to leverage other sources of data?*

While $\mathcal{V}$oltron expects a dataset of videos and associated language narrations, there is a wealth of *visually diverse and relevant data* that does not subscribe to this type signature:: datasets of standalone images from curated datasets [19, 21, 98], curated images paired with language captions as in Conceptual Captions [48, 78], and large in-the-wild datasets of images paired with text scraped from the internet [72, 85, 73].

Luckily (though beyond the scope of this initial work), incorporating this data into the existing $\mathcal{V}$oltron learning pipeline is straightforward; for image data without language, we can simply "annotate" each example with an empty `<NULL>` token in the worst case, or alternatively, with some minimal textual metadata (e.g., a class label, dataset descriptor, or even a URL if available). To accommodate for training on variable length image contexts, a naive solution would be adopting frame dropout or padding; there are myriad ways to do this efficiently – from Perceiver-based resampling of large patch sequences [35, 3] to different position encoding schemes [88, 63], to more efficient attention variants [8].

Appendix B
𝒱oltron Implementation & Artifacts

We provide complete implementation details for the various 𝒱oltron models, from the small modifications to the Transformer block for added pretraining stability, to the added structural components for embedding multimodal (vision and language) inputs. All of these details are made explicit in our code release, linked on our project page.



```python
class StandardBlock:
  def __init__(self, embed_dim: int, n_heads: int, mlp_dim: int) -> None:
    """
    Standard ViT Transformer Block with LayerNorm and GELU() activation.

    A Transformer Block consists of two normalization layers, applied prior to
    the multiheaded attention module (standard) and position-wise feed-forward
    MLP (standard) respectively.
    """
    self.pre_norm_attn = nn.LayerNorm(embed_dim, eps=1e-6)
    self.attn = nn.MultiHeadAttention(embed_dim, n_heads=n_heads)

    self.pre_norm_mlp = nn.LayerNorm(embed_dim, eps=1e-6)
    self.mlp = nn.Sequential(
      nn.Linear(embed_dim, mlp_dim),
      nn.GELU(),
      nn.Linear(mlp_dim, embed_dim),
    )

    # A Standard Block has no other components...

  def forward(self, x: T[bsz, seq, embed_dim]) -> T[bsz, seq, embed_dim]:
    x = x + self.attn(self.pre_norm_attn(x))
    x = x + self.mlp(self.pre_norm_mlp(x))
    return x
```

```python
class VoltronBlock:
  def __init__(self, embed_dim: int, n_heads: int, mlp_dim: int) -> None:
    """
    Voltron Transformer Block with RMSNorm, SwishGLU() activation, and
    LayerScale.

    A Transformer Block consists of two normalization layers, applied prior to
    the multiheaded attention module (standard) and position-wise feed-forward
    MLP (standard) respectively; adds residual LayerScale as well!
    """
    self.pre_norm_attn = RMSNorm(embed_dim) # RMSNorm is simpler/more stable
    self.attn = nn.MultiHeadAttention(embed_dim, n_heads=n_heads)

    self.pre_norm_mlp = RMSNorm(embed_dim)
    self.mlp = nn.Sequential(
      SwishGLU(embed_dim, mlp_dim), # SwishGLU includes a learned projection
      nn.Linear(mlp_dim, embed_dim),
    )

    self.scale_attn = LayerScale(embed_dim)
    self.scale_mlp = LayerScale(embed_dim)

  def forward(self, x: T[bsz, seq, embed_dim]) -> T[bsz, seq, embed_dim]:
    x = x + self.scale_attn(self.attn(self.pre_norm_attn(x)))
    x = x + self.scale_mlp(self.mlp(self.pre_norm_mlp(x)))
    return x
```

```python
class SwishGLU:
  def __init__(self, in_dim: int, out_dim: int) -> None:
    """
    SwishGLU as defined in PaLM (Google).
      => Ref: "GLU Variants Improve Transformers" (Shazeer)
    """
    self.project = nn.Linear(in_dim, out_dim)
    self.gate =  nn.Linear(in_dim, out_dim)

  def forward(self: x: T[..., in_dim]) -> T[..., out_dim]:
    projected = self.project(x)
    gate = self.gate(x)
    return projected * nn.SiLU(gate)
```

```python
class RMSNorm:
  def __init__(self, dim: int) -> None:
    self.g, self.dim = nn.Parameter(T.ones(dim)), dim

  def forward(self: x: T[..., dim]) -> T[..., dim]:
    return self.g * (x / (norm(x) / sqrt(self.dim)))

class LayerScale:
  def __init__(self, dim: int, init: float = 0.1) -> None:
    self.gamma = nn.Parameter(init * T.ones(dim))

  def forward(self: x: T[..., dim]) -> T[..., dim]:
    return self.gamma * x
```

Fig. 8: **Standard vs. 𝒱oltron Transformer Implementation.** The 𝒱oltron Transformer Block is near-identical to the "standard" Transformer block used in prior work in Vision Transformers, with exceptions marked in orange. Notably, we switch `LayerNorm` for `RMSNorm`, a standard MLP with a `GELU` activation [30] with a `SwishGLU` activation, and adopt `LayerScale` for each residual connection; these components are defined explicitly below the block definitions. In ablating these architecture modifications, we find *no impact on downstream performance, but increased pretraining stability*.

A. *𝒱oltron Transformer Implementation*

As mentioned in §IV, we perform a series of modifications to the typical Transformer block used in prior work in the Vision Transformer and Masked Autoencoding literature to help with pretraining stability; these changes are motivated by recent work from the NLP community on training stable and performant Transformer models [59, 40, 15].

We show the side-by-side comparison of the "standard" Transformer block implementation vs. the 𝒱oltron Transformer block in Fig. 8. The changes are three-fold:

- Using Root Mean-Square Normalization [104] over the default LayerNorm; not only does RMSNorm have fewer parameters, but it has been shown to increase stability and performance [59].
- Using the SwishGLU activation [79, 15] over the default GELU [30].
- Using LayerScale [91] for scaling down the magnitude of residual connections; prior work has found this to have a powerful stabilizing effect during pretraining [40].

We also provide pseudocode for implementing the various modifications in Fig. 8 (bottom); these modifications are all simple and transferable across Transformer implementations. Furthermore, as part of the no-language implementation in §VI, we ablate the effects of these modifications on performance; we find *that these modifications do not change downstream performance, but significantly increase pretraining stability*, following our initial motivation.

B. Jointly Processing Vision & Language

To incorporate language into the typical masked autoencoding pipeline, we add a series of small structural changes to handle 1) multi-modality, 2) sharing a Transformer decoder for both visual reconstruction and language generation, and 3) handling position encoding for both visual patch embeddings and textual tokens.

Multimodal Encoder. We make the following adjustments to enable a Transformer encoder to embed multiple modalities. First, we project both our learned "patch embeddings" (obtained as in a standard ViT, by learning a linear transformation of our flattened RGB patches of size $p \times p \times 3$) and our pretrained language embeddings to the same space $\mathbb{R}^d$, where d is the Transformer dimensionality (e.g., $d = 384$ for a ViT-Small). While we learn our patch embedding end-to-end, we initialize our language embeddings from a pretrained (and frozen) DistilBERT model [70]; this is following R3M [58]. We pad each language annotation c in our dataset to a maximum length $L = 20$ tokens, additionally storing a binary length mask to ensure that each Transformer block does not attend to padding.

Once projected into the Transformer's embedding space, we add learned modality embeddings (e.g., an embedding for `<IMG>` and `<LANG>`) to each of the respective inputs; we find that this better allows the Transformer to reason over different modalities. We initialize these learnable embeddings via a truncated normal distribution, with scale $\sigma = 0.02$, following how other special embeddings are initialized in the MAE and Vision Transformer literature [29].

The final step is for handling multi-frame contexts; we learn a set of frame index embeddings (e.g., for `FRAME-1`, `FRAME-2`, etc.) and add these to the corresponding patch embeddings – i.e. we add the `FRAME-i` embedding to all patch embeddings from the first frame and so on. This further allows us to distinguish individual frame patches from one another.

At this point, we concatenate the full sequence of flattened visual patch embeddings and language token embeddings, and feed them through the stack of Transformer blocks that form the multimodal encoder. This output is fed to the decoder, in the same fashion as a traditional masked autoencoder.

Shared Transformer for Reconstruction & Generation. As mentioned in §IV, we make one crucial change to the standard Transformer decoder in a masked autoencoder to additionally allow for language generation: namely adding a *prefix mask over the language inputs* [66]. The goal of this mask (as stated in the main text) is to prevent information leakage when decoding; this mask selectively zeroes out dependencies in the multiheaded attention during training such that when generating language given a visual context, each language embedding at a given timestep t can only attend to prior generated language at timesteps $< t$, as well as the entire visual context. This masking operates in the same way as the original decoder masking described in Vaswani et al. [92]; the attention scores for all "invalid" inputs ($> t$) are set to 0, restricting the model from incorporating future predictions as it processes the sequence.

Apart from this, the only other change we make to the MAE decoder is learning a separate set of modality embeddings (as described in the prior section) – i.e. embeddings for `<IMG-DECODER>` and `<LANG-DECODER>`; the reason for this is that the Decoder sees a series of `<MASK>` embeddings representing the "unseen" visible context to reconstruct, as well as the new language context to generate (recall that because of the α gating, the language generator *never* sees language embeddings from the encoder). We add these to the corresponding embeddings fed to the decoder, then resume the standard MAE decoding pipeline (reconstructing visual patches), and the language generation pipeline (autoregressively generating the original annotation).

Position Encoding. We follow standard pratice in the masked autoencoding literature (and the same practice used by MVP), as position encode each of the patch embeddings subject to a fixed (deterministic) 2D sinusoidal embedding that reflects both vertical and horizontal positioning of each patch within a grid – this is taken directly from the original MAE codebase. To encode text, we use a similar strategy, using a 1D sinusoidal embedding added to each token embedding in a sequence.

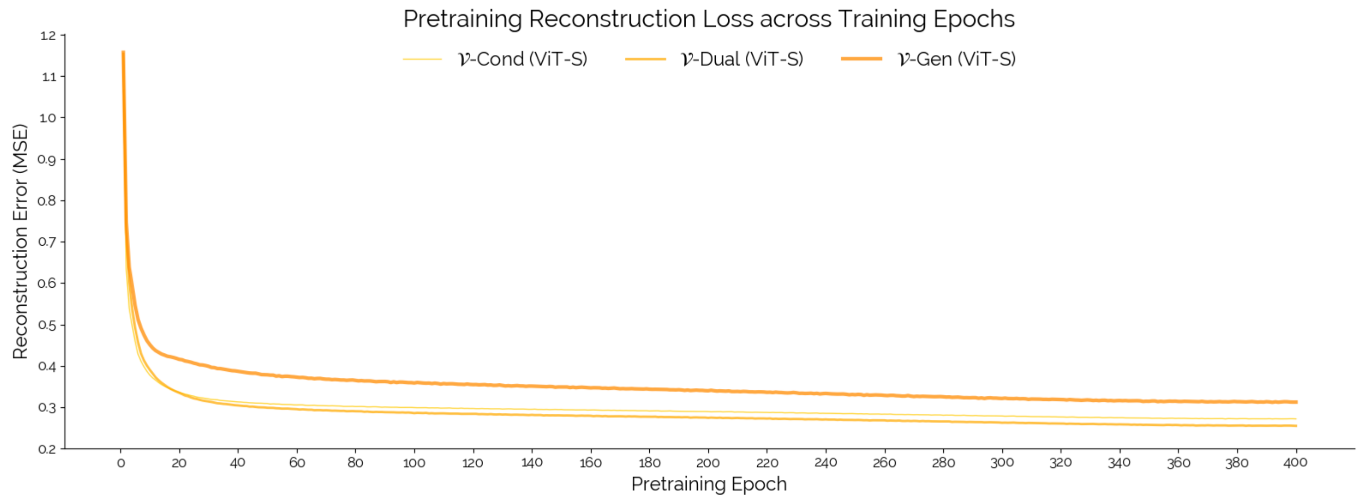


Fig. 9: $\mathcal{V}$**oltron Pretraining Learning Curves (Reconstruction Error).** We visualize the reconstruction error over pretraining epoch for each of the $\mathcal{V}$oltron models. Note that each model learns differently, converging to different reconstruction errors: both the language-conditioned models ($\alpha = 0$) converge to low reconstruction error, with $\mathcal{V}$ **– Dual** showing that encoding and learning over multi-frame contexts allowing for a better fit. The language generative model $\mathcal{V}$ **– Gen** ($\alpha = 0.5$) converges to a relatively higher reconstruction error, showing the tension between balancing two disparate objectives.

C. Pretraining Curves

To further contextualize our results and enrich some of the discussion §VI (and further on in the appendices), we include the pretraining loss curves for each of the three Voltron models we train in this work – $\mathcal{V}$ **– Cond**, $\mathcal{V}$ **– Dual**, and $\mathcal{V}$ **– Gen**. The reconstruction error curves for the three models can be found in Fig. 9. In general, we find that the "trade-off" between language-conditioned reconstruction and visually-grounded language generation is made concrete in the pretraining loss – both purely language-conditioned models ($\mathcal{V}$ **– Cond**, $\mathcal{V}$ **– Dual** with $\alpha = 0$) converge to fairly low reconstruction error; however, $\mathcal{V}$ **– Gen** (with $\alpha = 0.5$) converges to a much higher reconstruction error – due to the tension between optimizing for both reconstruction and language generation. We additionally note that adding even simple, dual-frame contexts enables lower reconstruction error – even with the ViT-Small models, on the Sth-Sth dataset.

D. Index of Released Artifacts

All of the following are linked in our code release and project page:

- Checkpoints for $\mathcal{V}$ **– Cond**, $\mathcal{V}$ **– Dual**, and $\mathcal{V}$ **– Gen** after 400 epochs of training on Sth-Sth.
- Checkpoints for our reproductions R-MVP and R-R3M (both with a ViT-S and RN-50 backbone).
- All index files (serialized frames/order seen during training) for reproducible pretraining.
- Intermediate checkpoints every 20 epochs for each of the three $\mathcal{V}$oltron models – along with optimizer states.
- Checkpoints for the ViT-Base variant of $\mathcal{V}$ **– Cond** (86M parameters vs. 22M for a ViT-Small).

The modeling code release additionally provides documentation and scripts for 1) training these models from scratch, and 2) downloading and extracting representations from the pretrained models. The evaluation code release provides a unified API for the various problem domains we evaluate on in this work.

Appendix C
Additional Results & Visualizations

We present additional results and visualizations to further support our claims from the main text. We provide additional discussion of 1) the impact of language supervision (in the context of pretraining reconstruction loss), 2) a further discussion of masked vs. generative language modeling as an objective, with an analysis of pretraining language modeling loss, 3) additional single task control results on the Adroit dexterous manipulation environments, 4) qualitative trajectory rollouts from the $\mathcal{V}$ – **Gen** language-conditioned imitation policy, and 5) additional qualitative intent scoring results.

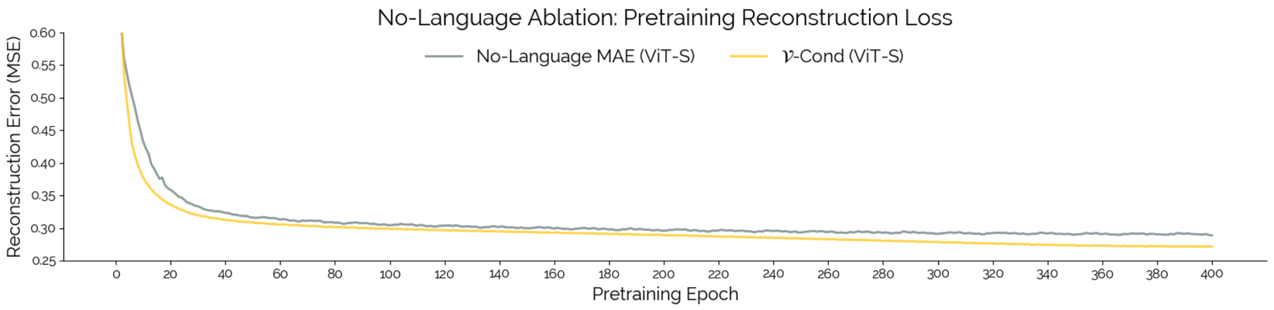


Fig. 10: **Pretraining Curves for the No-Language Ablation Experiment.** Training with language-conditioning ($\mathcal{V}$ – **Cond**) converges to a *lower reconstruction error* while also learning *faster*, compared to no-language (single-frame MAE) pretraining.

A. Analysis: Impact of Language-Conditioning on Reconstruction Loss

As part of the ablation experiments in §VI, we evaluate the impact of language-supervision during pretraining via a no-language ablation, training a single-frame masked autoencoder with the $\mathcal{V}$oltron Transformer architecture as described in §B-A; this resulting model *does not condition on language at all*, but is otherwise identical to $\mathcal{V}$ – **Cond**. In the main text, we evaluated the corresponding no-language model on a subset of evaluation tasks, showing a noticeable drop in performance across *every evaluated application* (even those without language input) – thereby showing concrete evidence as to the value of language-driven pretraining. Here we expand on those results by characterizing the behavior of both $\mathcal{V}$ – **Cond** and the no-language ablation thereof in terms of their *pretraining behavior*.

Fig. 10 shows the reconstruction error for both $\mathcal{V}$ – **Cond** (yellow) and the no-language ablation (gray) over the course of pretraining. There are two noticeable properties of these curves: first, $\mathcal{V}$ – **Cond** converges to a substantially lower reconstruction error than the same model trained *without language*. Second, $\mathcal{V}$ – **Cond** is able to learn *faster*, showing a steeper decline in reconstruction error earlier on in training. Taken together, these curves suggest that language-conditioning is able to focus feature learning in a way that allows the learned visual encoder to better encode masked contexts – especially considering that the visual reconstructor is by definition *not language-conditioned*. Furthermore, from the aggregate evaluation results, the features learned as a result somehow generalize better across the board, from low-level tasks like grasp affordance prediction, to high-level tasks such as control.

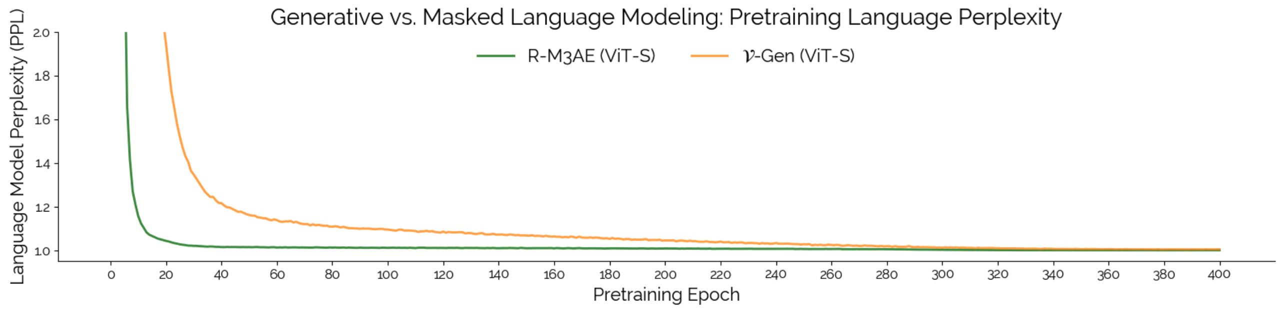


Fig. 11: **Pretraining Curves for the Generative vs. Masked Language Ablation Experiment.** Compared to multimodal masked language modeling (R-M3AE), $\mathcal{V}$ – **Gen** ($\alpha = 0.5$) shows that with language generation as an objective, language modeling perplexity (PPL = $\exp(\text{NLL})$) gradually decreases. R-M3AE overfits to language prediction almost immediately (PPL = 1), impacting its learned representations.

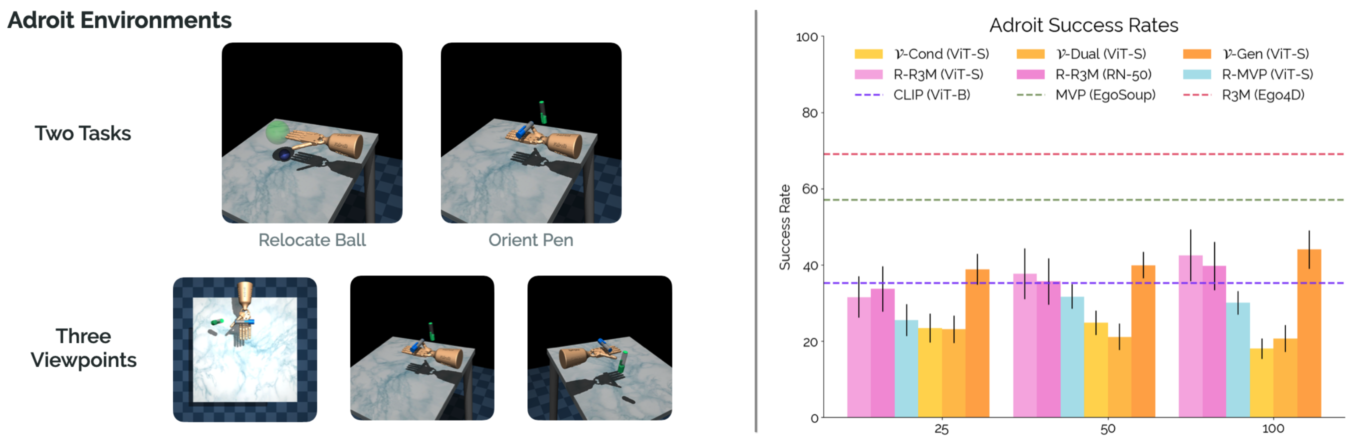


Fig. 12: **Adroit – Single-Task Visuomotor Control Results.** Visualization of the high-dimensional Adroit environments, comprised of two dexterous manipulation tasks, with three camera viewpoints **[Left]**. Results (success rate for each of n demonstrations with $n \in [25, 50, 100]$) for $\mathcal{V}$oltron and baselines (over 3 seeds) **[Right]**. Note the flipped trends relative to the Franka Kitchen results – notably, the more "high-level" representations (from CLIP, R3M, or $\mathcal{V}$ **– Gen**) tend to do better on this task; yet, $\mathcal{V}$ **– Gen** is still outperforming R-R3M and CLIP, showing the benefit of language-driven flexible learning.

B. Analysis: Generative vs. Masked Language Modeling

Later in §VI, we raise the question: why generative (autoregressive) language modeling over masked language modeling? To help contextualize this choice, we look at recent work on combining masked autoencoders (for vision) with masked language modeling (for text), through multimodal masked autoencoders [M3AE; 23]. We reimplement this M3AE model, pretraining on the same Sth-Sth dataset used throughout this work, following the same standard of quality as for R-MVP and R-R3M. When we evaluate the corresponding R-M3AE model, we notice *substantially worse performance across all evaluation domains*; in the main text we attributed this to overfitting during pretraining – here, we provide that concrete evidence.

Fig. 11 shows the *language model perplexity* over time for both the R-M3AE, and the $\mathcal{V}$ **– Gen** model (trained with $\alpha = 0.5$). Perplexity (PPL) = $\exp(\mathrm{NLL})$ is a monotonic function of the cross-entropy loss; lower values are "better" with a lower bound value of 1.0. Almost immediately, the R-M3AE model overfits to the masked language modeling task, hitting a "perfect" perplexity of 1 (loss of 0.0) within the first 20 epochs. Contrast this with $\mathcal{V}$ **– Gen** that learns to gradually lower perplexity of the entire course of training, almost driving down to a PPL of 1.0 by the 400th epoch. We attribute R-M3AE's poor performance to this extremely early overfitting of the language loss, again echoing the hypothesis that language generation is slightly more robust to these settings – predict short language captions given visual context – than a masked language modeling objective. We note that this pretraining data (Sth-Sth) is significantly different than the data used to train the original M3AE model in Geng et al. [23]; the original M3AE work used Conceptual Captions 12M [78], a rich dataset of images paired with long, descriptive captions. Further work on extending M3AE models as in Liu et al. [49] further pretrain on *text-only* datasets such as Wikipedia and Toronto Books [20] suggesting the need for diverse, broad coverage text when training (multimodal) masked language models.

C. Results: Adroit Visuomotor Control

To supplement our single-task visuomotor control results, we run out evaluations on the Adroit dexterous manipulation tasks from the R3M paper [58]. The two tasks we evaluate on, depicted in Fig. 12 (left) consist of controlling a high degree-of-freedom robotic hand (24-DoF) for the task of 1) relocating a ball on the table to a specified target position, and 2) reorienting a pen within the hand to reach a target orientation. Given the innate difficulty of controlling a high-dimensional dexterous robotic hand over a 9-DoF fixed arm manipulator, these tasks are evaluated with $n \in [25, 50, 100]$ demonstrations instead of $n \in [5, 10, 25]$ as with the Franka Kitchen evaluation. In general, learning policies in this environment is *difficult*, especially from limited data.

Looking to the results we see that on this environment, $\mathcal{V}$ **– Gen** and R-R3M models tend to be the most performant, in contrast with the Franka Kitchen results which favored $\mathcal{V}$ **– Cond** and $\mathcal{V}$ **– Dual** (the reconstruction-leaning models). Interestingly, this flipped trend seems to suggest that even within single-task control, different tasks and environments seems to prefer different visual features to perform well – in this case, the more high-level features under models such as R-R3M and $\mathcal{V}$ **– Gen** seem to be preferred. In a way, this makes sense; unlike with Franka Kitchen, the actual background objects and interactions thereof – turning knobs, opening microwaves, or sliding doors with clearly marked handles – seem more sensitive

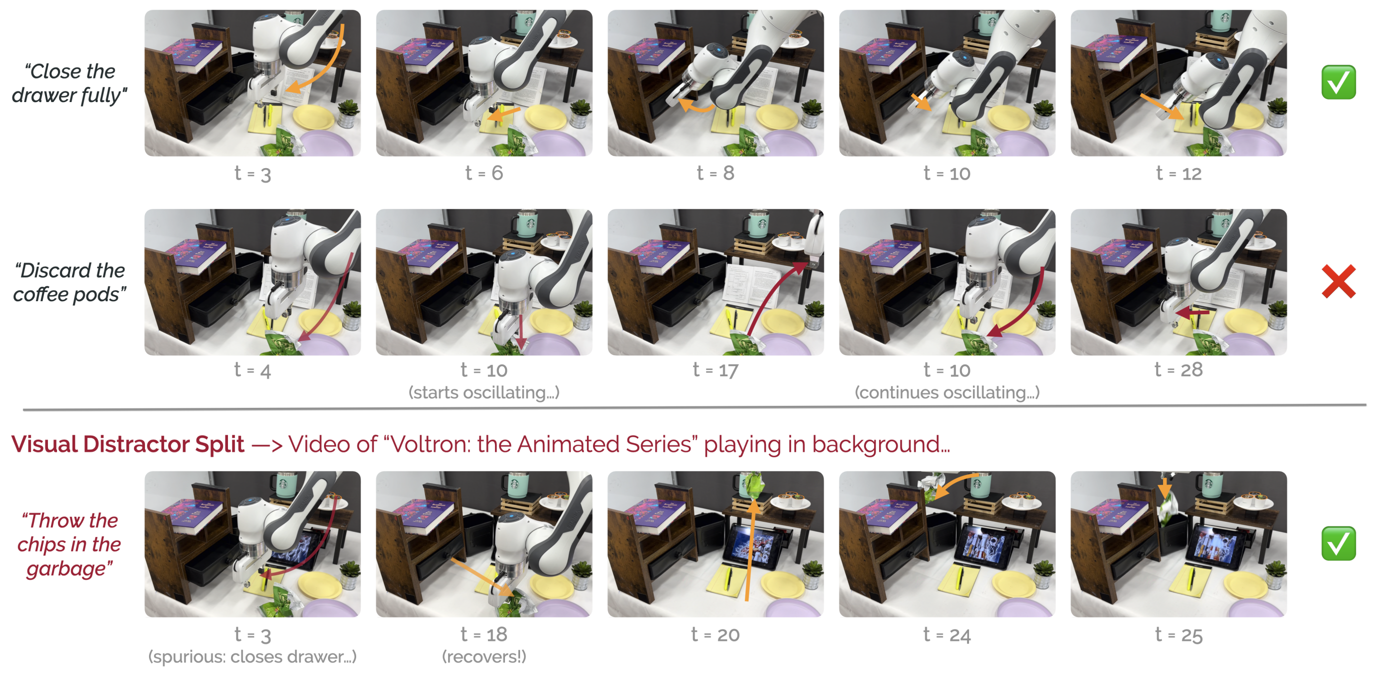

"Close the drawer fully"
t = 3
t = 6
t = 8
t = 10
t = 12
"Discard the coffee pods"
t = 4
t = 10
(starts oscillating...)
t = 17
t = 10
(continues oscillating...)
t = 28
Visual Distractor Split —> Video of "Voltron: the Animated Series" playing in background...
"Throw the chips in the garbage"
t = 3
(spurious: closes drawer...)
t = 18
(recovers!)
t = 20
t = 24
t = 25

Fig. 13: **Real-World Language-Conditioned Imitation Rollouts from $\mathcal{V}$ – Gen.** We visualize some rollouts from the best-performing real-world language-conditioned imitation learning model, $\mathcal{V}$ – **Gen**. While some tasks – e.g., discarding the plate of used coffee pods in the trash – prove hard for all methods, $\mathcal{V}$ – **Gen** shows smooth motion on a series of tasks, even when challenging visual distractors are present. Videos with evaluation rollouts for each method are on our project page.

to low-level features (where on the microwave is the handle, which knob of the various possible needs to be turned). In Adroit however, these tasks are on clean backgrounds, with individual objects; the high-level behaviors instead that are more important (e.g., "is the ball getting closer to the target location?"). It would be an interesting direction for future work to further profile other "common" visuomotor control tasks along this axis, to get a better understanding of what visual representations must capture to be generally useful (predictive of performance on downstream real-world control problems).

D. *Qualitative: Real-Robot Language-Conditioned Policy Rollouts*

While the experimental results in §V capture the quantitative success rates of various methods for language-conditioned imitation, they do not paint a picture of *how* these policies behave. In Fig. 13 we show three different rollouts for the best-performing $\mathcal{V}$ – **Gen** model: a task success (in-distribution), a task failure (in-distribution), and an example rollout from the visual distractor split. With the waypoint-based action space described in §V, we generally see smooth motions; however, the failure mode of these policies are "oscillations" (Fig. 13; middle) where the policy collapses to predicting the same two waypoints repeatedly. *Full videos of rollouts from each representation learning approach* are all on our project page.

E. *Qualitative: Additional Intent Scoring Visualizations*

Fig. 14 presents additional intent scoring qualitative visualizations for two other tasks from the WHiRL dataset [5] – specifically "lifting the lid off a pot" and "stacking cups." In both scenarios, we see similar behavior to the results from §V of the main text: $\mathcal{V}$ – **Gen** shows a propensity for not only tracking the key progress points in the videos for *both human and robot* agents, but also providing a dense and smooth measure of intermediate progress. Both *CLIP (ViT-Base)* and *R3M (Ego4D)* unfortunately predict high-variance scores, seemingly random across the video.

Appendix D
Data-Equivalent Reproductions & Reproducibility

In this section we provide additional discussion around two aspects of the reproduction and pretraining procedure discussed in §IV: 1) preprocessing, and specifically the *importance of selecting multiple images from the same context*, and 2) how to operationalize the representations from the visual encoder for downstream learning.

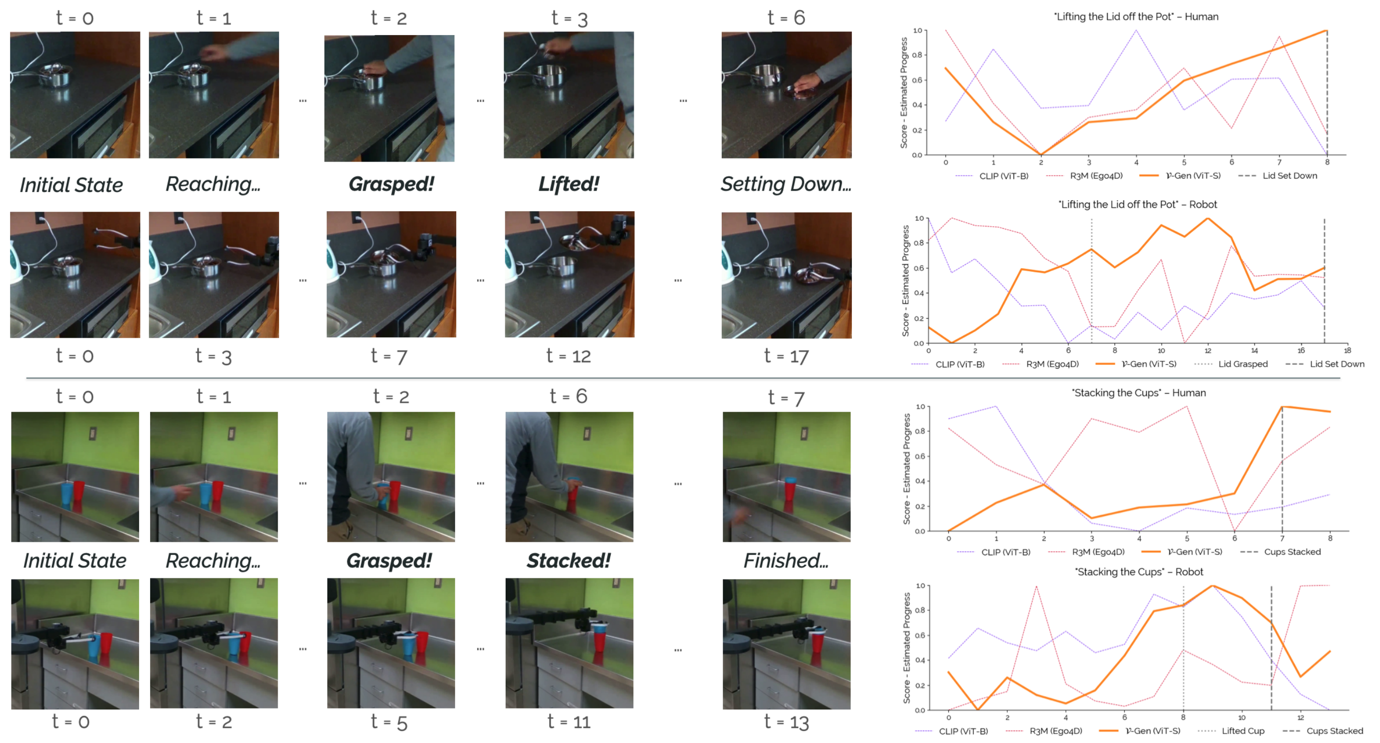


Fig. 14: **Additional Qualitative Zero-Shot Intent Scoring Examples.** Given more videos of humans and robots performing similar behaviors from the WHiRL dataset [5], we evaluate the zero-shot intent scoring capabilities of $\mathcal{V}$ **– Gen**, *R3M (Ego4D)* and *CLIP (ViT-Base)*. In general, $\mathcal{V}$ **– Gen** continues to show a nuanced understanding of semantics over time, in general tracking key points in each video smoothly, whereas both baselines are for the most part predicting random scores.

A. *Additional Preprocessing Discussion*

We described our preprocessing approach in §IV: following the R3M paper, we sample *five frames from each video clip* for each epoch of pretraining. Seeing multiple frames from the same visual context is minimally necessary for the R3M time-contrastive learning objective, but we posit in this discussion (following the questions in §A) that repeatedly sampling from the same visual context – even with a reconstruction objective – allows for picking up on finer-grained changes *within* a context. The best evidence we have for this is in looking at how prior work constructs their pretraining datasets.

The original MVP work [97, 65] constructs *static datasets of images* by iterating through the various video clips in their pretraining datasets – Sth-Sth, Ego4D [25], 100 Days of Hands [76] – at a fixed rate, usually from 0.2 to 1 frames per second. Given video clip lengths of 2 seconds, this means that *in aggregate* these pretraining datasets comprise maybe 2-3 frames sampled from the same clip, if that. Contrast that with this work and R3M, sampling multiple frames from *each video clip* for *every pretraining epoch* (for 400 epochs). This not only means that we are seeing the same context repeatedly, but also that we are seeing different *views* of the same context; this can help tune reconstruction towards picking up on finer-grained features (e.g., if a high-capacity model is able to memorize prior contexts given enough repetition).

This offers a speculative explanation of why $\mathcal{V}$oltron models outperform *MVP (EgoSoup)* models that are both higher-capacity and trained on orders of magnitude more data – but definitely requires further experiments to prove. In the meantime, it seems as though taking steps to use as much of the pretraining datasets we have access to as possible is in our best interest.

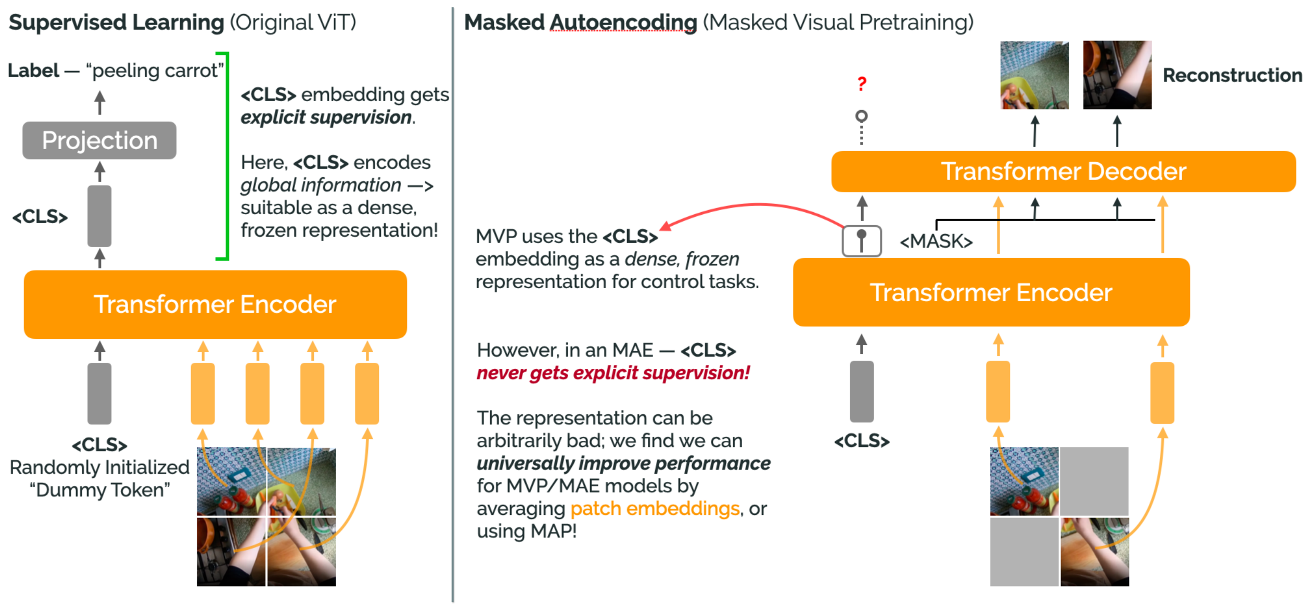

Supervised Learning (Original ViT)
Label — "peeling carrot"
Projection
<CLS>
<CLS> embedding gets explicit supervision.
Here, <CLS> encodes global information —> suitable as a dense, frozen representation!
Transformer Encoder
<CLS> Randomly Initialized "Dummy Token"
Masked Autoencoding (Masked Visual Pretraining)
?
Reconstruction
Transformer Decoder
<MASK>
MVP uses the <CLS> embedding as a dense, frozen representation for control tasks.
Transformer Encoder
However, in an MAE — <CLS> never gets explicit supervision!
The representation can be arbitrarily bad; we find we can universally improve performance for MVP/MAE models by averaging patch embeddings, or using MAP!
<CLS>

Fig. 15: **Default Feature Extraction in MAE Models.** Prior work in masked autoencoding including MVP use the embedding corresponding to a dummy <CLS> token appended to the Transformer input for downstream adaptation. While this is motivated in the *supervised learning* setting, it is not clear what this embedding captures in the MAE setting, as it never receives explicit supervision. We find that pooling the *learned patch embeddings* is strictly better.

B. Multiheaded Attention Pooling – Extracting Representations

There is a critical difference between pretraining visual representations and identifying the "right" way to use these representations for downstream adaptation tasks. Especially for Vision Transformers trained as part of a masked autoencoder – as mentioned at the end of Section §IV of the main text – identifying a method for extracting information from the learned representations is an open problem. The main text states – by fiat – that we use multiheaded attention pooling [MAP; 46] as suggested by Zhai et al. [102] to operationalize our learned representations for our downstream tasks. Here, we further contextualize that decision with a description of alternative approaches, as well as comparative results (Table V) that show the superiority of MAP-based "feature extraction" (referring to the process of taking the output of a Vision Transformer and producing a dense, summary vector for downstream learning) over alternative approaches.

MVP and prior work in masked autoencoding with Vision Transformers [29] make an interesting choice when it comes to extracting features: during pretraining, these works append a dummy <CLS> token to the input of the encoder and decoder in the masked autoencoding pipeline (depicted in Fig. 15). This "free" embedding is motivated by how Vision Transformers for supervised learning (e.g., classification) are parameterized: in these settings, after encoding an input image, the <CLS> embedding is used as (the sole) input to a linear projection into label space, thus obtaining supervision from the global loss function (e.g., the cross-entropy loss for classification). Crucially, the <CLS> embedding in these cases gets *direct supervision* during training.

TABLE V: **Feature Extraction Results.** We evaluate various feature extraction strategies on the Franka Kitchen visuomotor control tasks at $n = 10$ demonstrations. We find that MAP is strictly superior for all Vision Transformer backbones; even mean-pooling over patch embeddings outperforms the default strategy from the MVP work that uses the frozen <CLS> embedding.

	Architecture	Default Extractor	Mean-Pooling	Multiheaded Attention Pooling (MAP)
R-R3M	ViT-S	16.07 (Default = Mean-Pooling)	–	14.73
R-MVP	ViT-S	7.90 (Default = <CLS> Token)	9.50	**26.73**
$\mathcal{V}$ – **Cond**	ViT-S	–	19.07	**27.33**
$\mathcal{V}$ – **Dual**	ViT-S	–	17.40	**33.07**
$\mathcal{V}$ – **Gen**	ViT-S	–	15.67	**30.33**
$\mathcal{V}$ – **Cond**	ViT-B	–	19.40	**30.80**
$\mathcal{V}$ – **Dual**	ViT-B	–	16.40	**37.27**
$\mathcal{V}$ – **Gen**	ViT-B	–	15.73	**32.13**
CLIP	ViT-B	17.73 (Default = Pool & Normalize)	16.33	**22.20**
MVP (EgoSoup)	ViT-B	18.20 (Default = <CLS>)	20.13	**33.87**

However, in the masked autoencoding setting, this <CLS> embedding is just passed through the various Transformer layers of the encoder and decoder, *never obtaining any direct or indirect supervision*; while it does attend to all other patch embeddings as a byproduct of the multiheaded attention mechanism, there is no guarantee that this embedding captures or summarize all the useful information necessary.

Instead, recent work from the same authors of the original Vision Transformer [102] eschew the <CLS> embedding completely during training, instead identifying that two other strategies – mean-pooling *all* the patch embeddings output by the encoder, or using multiheaded attention pooling [46] – are almost always preferable. As an aside – this work is what motivates $\mathcal{V}$oltron models to also do away with the <CLS> embedding.

Multiheaded attention pooling (MAP) can be thought of as a form of cross-attention with a *learned* query. Starting with a randomly initialized query vector (or optionally, *set* of query vectors), a MAP block implements a shallow multiheaded attention operation, using the initialized query vector to cross-attend over the patch embeddings output by the Vision Transformer – the resulting output is a "weighted" combination of the individual patch embeddings that is shaped on a per-adaptation basis. We evaluate MAP-based extraction against mean-pooling and any other "default" strategy (e.g., the <CLS> embedding used in MVP, the learned dense representation under *CLIP*) in Table V. We find that MAP universally outperforms all other strategies on the Franka Kitchen control tasks (with $n = 10$ demonstrations), informing our usage of MAP as the sole feature extraction approach throughout this work. Notably, we find that MAP-based extraction when applied to the original model *MVP (EgoSoup)* released in the original work *almost doubles success rate* on downstream control tasks. We even find that simple mean-pooling over patches outperforms the <CLS> embedding, further motivating alternate strategies.

Appendix E
Adapting Representations for Evaluation

The description of the adaptation pipeline described in §V outlines all major details for the adaptation experiments for each evaluation domain; the role of this section is to clarify any potentially ambiguous details, and further motivate some of the choices we make in implementing each evaluation. In general, all of the details for adapting representations for each evaluation in the same manner used in this work are in the released evaluation code repository that provides a unified harness for evaluating arbitrary visual representations on *all evaluation domains used in this work* – this codebase is also linked from our project page.

In general, for each evaluation domain, we keep the adaptation architecture and optimization parameters as simple as possible. For all applications we use an AdamW optimizer [43] with the default learning rate of 1e-3, and weight decay of 0.01.

Grasp Affordance Prediction. We implement the adaptation head for the grasp affordance prediction task following recent work in learning segmentation heads on top of vision transformer features, specifically following the procedure outlined in Segmentation Transformers via Progressive Upsampling (SETR-PUP) [105]. A PUP block is straightforward – we first extract all patch embeddings from the output of our Vision Transformer encoder, using a shallow MAP block with the same number of seed vectors as patches output by the encoder. We then reshape the extracted features into a *grid*, then stack a series of 4 upsampling blocks (channel depths of $[128, 64, 32, 16]$, ReLU activation) that consist of a 2D convolution followed by a bilinear upsampling, until we recover a grid of the same size of the original image. We finally apply a spatial softmax, predicting distributions over each of the possible labels ("graspable," "non-graspable," "background"), and compute our loss per-pixel. We optimize with a batch size of 64, for 50 epochs in total. Given the small size of the dataset, we find that there is a great deal of variance across random initializations; we report results by running 5-fold cross-validation, taking the model with the best performance across validation folds to compute final test statistics.

Referring Expression Grounding. We use a simple adaptation head for referring expression grounding that extracts a single dense representation from our learned encoder via a shallow MAP block with a single seed vector (the default extractor for obtaining a vector representation of a visual input). For representations that are not language-conditioned, we concatenate this vector with the language embedding under the appropriate model – e.g., the CLIP text embedding for *CLIP (ViT-Base)* – or the DistilBERT language embedding for pure visual models (e.g., MVP). We then feed this context through a 4-layer MLP (hidden dimensions of $[512, 128, 128, 64]$, GELU activation) that directly predicts bounding box coordinates as $(x, y, \text{width}, \text{height})$. We use a Huber loss to compute error. We optimize with a batch size of 512, for 10 epochs in total, using the provided validation set for model selection.

Single-Task Visuomotor Control. We first extract a dense representation using a shallow MAP block (as described above), then follow the exact procedure for evaluating both Franka Kitchen and Adroit policy learning as described in the R3M work [58]. Namely, we concatenate the visual representation with the robot's proprioceptive state, followed by a BatchNorm layer [34]. These are then fed to a 2-layer MLP ($d = 256$) that directly predicts action targets for computing mean-squared error against the ground-truth actions. Following R3M, we run 20,000 gradient steps with a batch size of 32, evaluating the models online every 5000 steps on a heldout set of 50 environments (fixed seed) – we report success rate subject to the best performing model from the online evaluation. We run three seeds for each combination of viewpoint, number of demonstrations, and task.

Real-World Language-Conditioned Imitation. The full set of language instructions generated by ChatGPT can be found on our project page. For adaptation, we first extract a representation as with the referring expression evaluation by using a shallow MAP block, and concatenating the corresponding language embedding as appropriate. We concatenate this fused vector with the robot's proprioceptive state, and pass the corresponding embedding to a BatchNorm layer. Then, following recent work on real-world imitation learning [55], we only train a shallow 2-layer MLP with ($d = 64$) to predict action targets for computing mean-squared error against the ground-truth waypoint actions. We optimize with a batch size of 256, and train for 10 epochs. As policy evaluation in the real-world is expensive – especially for the five approaches we evalaute – we uniformly choose the last epoch checkpoint to perform evaluation rollouts.

Qualitative: Zero-Shot Intent Scoring. This is a zero-shot evaluation with no adaptation data, only applicable to the representation learning models capable of "scoring" joint vision-language contexts: $\mathcal{V}$ **– Gen**, *CLIP (ViT-Base)*, and *R3M (Ego4D)*. We download videos from the WHiRL dataset off of the WHiRL website: https://human2robot.github.io/. To generate plots, we sample frames at 2 FPS from each video, center cropping and resizing each frame prior to passing it to each model.