V-JEPA 2: Self-Supervised Video Models Enable Understanding, Prediction and Planning

Mahmoud Assran[1,*], **Adrien Bardes**[1,*], **David Fan**[1,*], **Quentin Garrido**[1,*], **Russell Howes**[1,*], **Mojtaba Komeili**[1,*], **Matthew Muckley**[1,*], **Ammar Rizvi**[1,*], **Claire Roberts**[1,*], **Koustuv Sinha**[1,*], **Artem Zholus**[1,2,*], **Sergio Arnaud**[1,*], **Abha Gejji**[1,*], **Ada Martin**[1,*], **Francois Robert Hogan**[1,*], **Daniel Dugas**[1,*], **Piotr Bojanowski**[1], **Vasil Khalidov**[1], **Patrick Labatut**[1], **Francisco Massa**[1], **Marc Szafraniec**[1], **Kapil Krishnakumar**[1], **Yong Li**[1], **Xiaodong Ma**[1], **Sarath Chandar**[2], **Franziska Meier**[1,*], **Yann LeCun**[1,*], **Michael Rabbat**[1,*], **Nicolas Ballas**[1,*]

[1]FAIR at Meta, [2]Mila – Quebec AI Institute and Polytechnique Montréal
[*]Core Team

A major challenge for modern AI is to learn to understand the world and learn to act largely by observation (LeCun, 2022). This paper explores a self-supervised approach that combines internet-scale video data with a small amount of interaction data (robot trajectories), to develop models capable of understanding, predicting, and planning in the physical world. We first pre-train an action-free joint-embedding-predictive architecture, V-JEPA 2, on a video and image dataset comprising over 1 million hours of internet video. V-JEPA 2 achieves strong performance on motion understanding (77.3 top-1 accuracy on Something-Something v2) and state-of-the-art performance on human action anticipation (39.7 recall-at-5 on Epic-Kitchens-100) surpassing previous task-specific models. Additionally, after aligning V-JEPA 2 with a large language model, we demonstrate state-of-the-art performance on multiple video question-answering tasks at the 8 billion parameter scale (e.g., 84.0 on PerceptionTest, 76.9 on TempCompass). Finally, we show how self-supervised learning can be applied to robotic planning tasks by post-training a latent action-conditioned world model, V-JEPA 2-AC, using less than 62 hours of unlabeled robot videos from the Droid dataset. We deploy V-JEPA 2-AC zero-shot on Franka arms in two different labs and enable picking and placing of objects using planning with image goals. Notably, this is achieved without collecting any data from the robots in these environments, and without any task-specific training or reward. This work demonstrates how self-supervised learning from web-scale data and a small amount of robot interaction data can yield a world model capable of planning in the physical world.

Date: June 12, 2025
Correspondence: Nicolas Ballas <ballasn@meta.com> and Michael Rabbat <mikerabbat@meta.com>
Code: https://github.com/facebookresearch/vjepa2
Blogpost: https://ai.meta.com/blog/v-jepa-2-world-model-benchmarks

Meta

1 Introduction

Humans have the ability to adapt and generalize when taking on new tasks and operating in unfamiliar environments. Several cognitive learning theories suggest that humans learn an internal model of the world by integrating low-level sensory inputs to represent and predict future states (Craik, 1967; Rao and Ballard, 1999), and they further posit that this world model shapes our perception at any given moment, playing a crucial role in informing our understanding of reality (Friston, 2010; Clark, 2013; Nortmann et al., 2015). Moreover, our ability to predict the effects of our actions on future states of the world is also essential for goal-oriented planning (Sutton and Barto, 1981, 1998; Ha and Schmidhuber, 2018; Wolpert and Ghahramani, 2000). Building artificial agents that learn a world model from sensory data, such as video, could enable them to *understand* the physical world, *predict* future states, and effectively — like humans — *plan* in new situations, resulting in systems capable of tackling tasks that have not been encountered before.

Previous works have explored the development of predictive world models from interaction data consisting

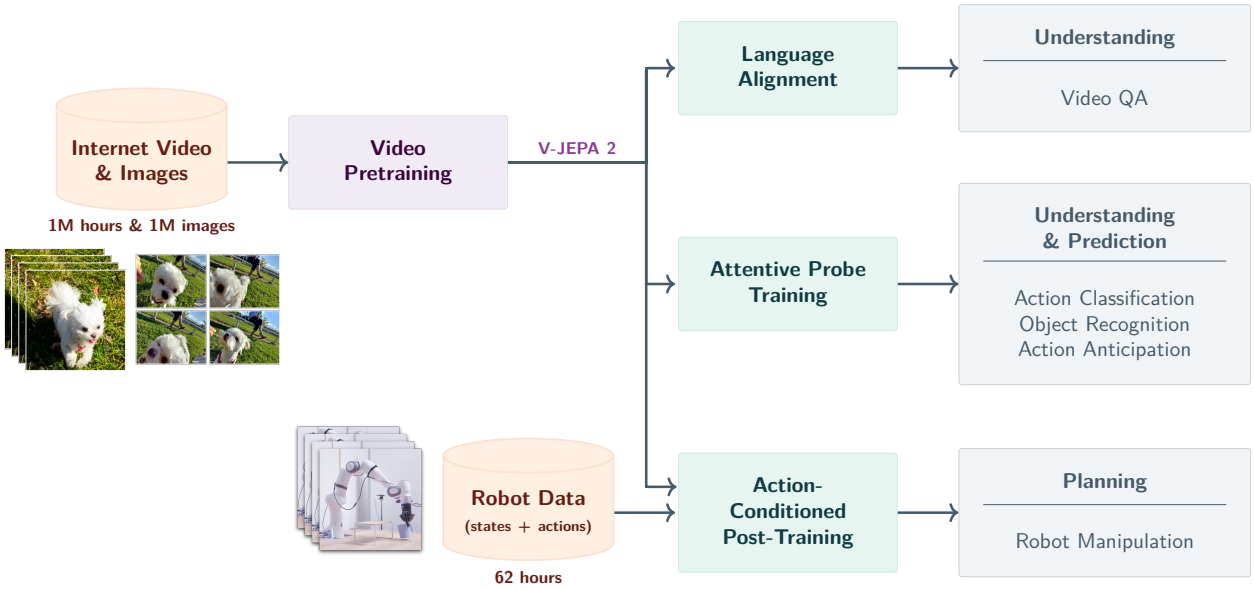


Figure 1 V-JEPA 2 Overview. Leveraging 1M hours of internet-scale video and 1M images, we pretrain the V-JEPA 2 video model using a visual mask denoising objective (Bardes et al., 2024; Assran et al., 2023), and leverage this model for downstream tasks such as action classification, object recognition, action anticipation, and Video Question Answering by aligning the model with an LLM backbone. After pretraining, we can also freeze the video encoder and train a new action-conditioned predictor with a small amount of robot interaction data on top of the learned representations, and leverage this action-conditioned model, V-JEPA 2-AC, for downstream robot manipulation tasks using planning within a model predictive control loop.

of state-action sequences, often also relying on explicit reward feedback from the environment to infer goals (Sutton and Barto, 1981; Fragkiadaki et al., 2015; Ha and Schmidhuber, 2018; Hafner et al., 2019b; Hansen et al., 2022). However, the limited availability of real-world interaction data constrains the scalability of these methods. To address this limitation, more recent works have leveraged both internet-scale video and interaction data towards training action-conditioned video generation models for robot control, but only demonstrate limited results in robot execution using model-based control (Hu et al., 2023; Yang et al., 2024b; Bruce et al., 2024; Agarwal et al., 2025). In particular, this line of research often emphasizes the evaluation of the faithfulness of the predictions and visual quality instead of planning capabilities, perhaps due to the computational cost of planning by generating video.

In this work, we build upon the self-supervised hypothesis as a means to learn world models that capture background knowledge of the world largely from observation. Specifically, we leverage the joint-embedding predictive architecture (JEPA) (LeCun, 2022), which learns by making predictions in a learned representation space. In contrast to approaches that focus on learning entirely from interaction data, self-supervised learning enables us to make use of internet-scale video — depicting sequences of states without direct observations of the actions — to learn to both represent video observations and learn a predictive model for world dynamics in this learned representation space. Furthermore, in contrast to approaches based on video generation, the JEPA approach focuses on learning representations for predictable aspects of a scene (e.g., the trajectory of an object in motion) while ignoring unpredictable details that generative objectives emphasize, since they make pixel-level predictions (e.g., the precise location of each blade of grass in a field, or each leaf on a tree). By scaling JEPA pretraining, we demonstrate that it yields video representations with state-of-the-art understanding and prediction capabilities, and that such representations can be leveraged as a basis for action-conditioned predictive models and enable zero-shot planning.

Our approach, V-JEPA 2, utilizes a stage-wise training procedure, beginning with action-free pre-training on internet-scale video, followed by post-training with a small amount of interaction data (see Figure 1). In the first stage, we use a mask-denoising feature prediction objective (Assran et al., 2023; Bardes et al., 2024), where the model predicts masked segments of a video in a learned representation space. We train

the V-JEPA 2 encoder with up to 1 billion parameters and with more than 1 million hours of video. Our experiments confirm that scaling self-supervised video pretraining enhances the encoder's ability to achieve visual understanding, including broad motion and appearance recognition capabilities, through probe-based evaluations and by aligning the encoder with a language model for video question-answering (Krojer et al., 2024; Pătrăucean et al., 2023; Liu et al., 2024c; Cai et al., 2024; Shangguan et al., 2024).

Following pretraining on internet-scale video, we train an action-conditioned world model, V-JEPA 2-AC, on a small set of interaction data using the representations learned in the first stage. Our action-conditioned world model is a 300M-parameter transformer network employing a block-causal attention mechanism, which autoregressively predicts the representation of the next video frame conditioned on an action and previous states. With as little as 62 hours of unlabeled interaction data from the Droid dataset (Khazatsky et al., 2024), we demonstrate the feasibility of training a latent world model that, given sub-goals, can be leveraged to plan actions on a Franka robot arm and perform prehensile manipulation tasks from a monocular RGB camera zero-shot in a new environment.

To summarize, we show that joint-embedding predictive architectures learning from videos can be used to build a world model that enables *understanding* the physical world, *predicting* future states, and effectively *planning* in new situations; this is achieved by leveraging internet-scale video and a small amount of interaction data. Specifically:

- *Understanding — Probe-based Classification:* Scaling self-supervised video pretraining results in video representations applicable to many tasks. V-JEPA 2 excels at encoding fine-grained motion information, achieving strong performance on tasks requiring motion understanding, such as Something-Something v2, with 77.3 top-1 accuracy using an attentive probe.
- *Understanding — Video Question-Answering:* V-JEPA 2 encoder can be used to train a multi-modal large language model, to tackle video-question answering tasks. We observe state-of-the-art performance on 8B language model class on multiple benchmarks that require physical world understanding and temporal reasoning, such as MVP (44.5 paired accuracy), PerceptionTest (84.0 test set accuracy), TempCompass (76.9 multi-choice accuracy), TemporalBench (36.7 multi-binary short-QA accuracy) and TOMATO (40.3 accuracy). In particular, we show that a video encoder pre-trained *without* language supervision can be aligned with a language model and achieve state-of-the-art performance, contrary to conventional wisdom (Yuan et al., 2025; Wang et al., 2024b).
- *Prediction:* Large-scale self-supervised video pretraining enhances prediction capabilities. V-JEPA 2 achieves state-of-the-art performance on the Epic-Kitchens-100 human-action anticipation task using an attentive probe, with 39.7 recall-at-5, which is a 44% relative improvement over the previous best model.
- *Planning:* We demonstrate that V-JEPA 2-AC, obtained by post-training V-JEPA 2 with only 62 hours of unlabeled robot manipulation data from the popular Droid dataset, can be deployed in new environments to solve prehensile manipulation tasks using planning with given subgoals. Without training on any additional data from robots in our labs, and without any task-specific training or reward, the model successfully handles prehensile manipulation tasks, such as Grasp and Pick-and-Place with novel objects and in new environments.

The remainder of this paper is organized as follows. Section 2 describes the V-JEPA 2 pretraining procedure, including the key ingredients enabling scaling beyond the original V-JEPA recipe of Bardes et al. (2024). Section 3 then introduces our approach to training a task-agnostic action-conditioned world model, V-JEPA 2-AC, leveraging the pretrained V-JEPA 2 model. Section 4 demonstrates using V-JEPA 2-AC for robot control via model-based planning. Because V-JEPA 2-AC models world dynamics in a learned representation space, its capabilities fundamentally depend on the information captured in the V-JEPA 2 representation space, and so we further explore the performance of V-JEPA 2 for video understanding in Section 5 and prediction tasks in Section 6. Finally, in Section 7 we show that V-JEPA 2 can be aligned with a language model for video question answering. Section 8 discusses related work, and we conclude in Section 9.

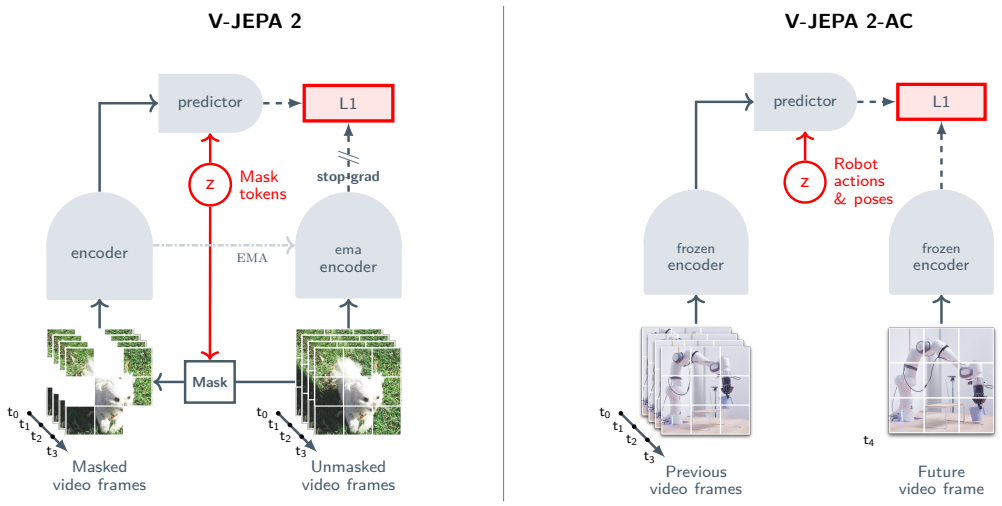


Figure 2 Multistage training. (Left) We first pretrain the V-JEPA 2 video encoder on internet-scale image and video data using a visual mask denoising objective (Bardes et al., 2024; Assran et al., 2023). A video clip is patchified into a sequence of tokens and a mask is applied by dropping a subset of the tokens. The encoder then processes the masked video sequence and outputs an embedding vector for each input token. Next, the outputs of the encoder are concatenated with a set of learnable mask tokens that specify the position of the masked patches, and subsequently processed by the predictor. The outputs of the predictor are then regressed to the prediction targets using an L1 loss. The prediction targets are computed by an ema-encoder, the weights of which are defined as an exponential moving average of the encoder weights. **(Right)** After pretraining, we freeze the video encoder and learn a new action-conditioned predictor, V-JEPA 2-AC, on top of the learned representation. We leverage an autoregressive feature prediction objective that involves predicting the representations of future video frames conditioned on past video frames, actions, and end-effector states. Our action-conditioned predictor uses a block-causal attention pattern such that each patch feature at a given time step can attend to the patch features, actions, and end-effector states from current and previous time steps.

2 V-JEPA 2: Scaling Self-Supervised Video Pretraining

We pretrain V-JEPA 2 on a visual dataset that includes over 1 million hours of video. The self-supervised training task is based on mask denoising in representation space and builds upon the V-JEPA framework (Bardes et al., 2024). In this paper, we extend the V-JEPA framework by exploring larger-scale models, increasing the size of the pretraining data, and introducing a spatial and temporal progressive resolution training strategy that enables us to efficiently pretrain models beyond short 16-frame video clips.

2.1 Methodology

Mask-Denoising in Representation Space. The V-JEPA objective aims to predict the learned representation of a video y from a view x of that video that has been masked, i.e., from which patches have been randomly dropped (Figure 2, left). The task meta-architecture consists of an encoder, $E_\theta(\cdot)$, which extracts video representations, and a predictor, $P_\phi(\cdot)$, which predicts the representation of masked video parts. The encoder and predictor are trained simultaneously using the objective,

$$\text{minimize}_{\theta,\phi,\Delta_y} \quad \|P_\phi(\Delta_y, E_\theta(x)) - \text{sg}(E_{\overline{\theta}}(y))\|_1, \tag{1}$$

where Δ_y is a learnable mask token that indicates the locations of the dropped patches. The loss uses a stop-gradient operation, $\text{sg}(\cdot)$, and an exponential moving average, $\overline{\theta}$, of the weights θ of the encoder network to prevent representation collapse. The loss is applied only to the predictions of the masked patches.

Architecture. The encoder, $E_\theta(\cdot)$, and predictor, $P_\phi(\cdot)$, are each parameterized as a vision transformer (Dosovitskiy et al., 2020) (or ViT). To encode relative position information in the vision transformer, we leverage RoPE (Rotary Position Embedding) instead of the absolute sincos position embedding used in Bardes et al.

(2024). We use a 3D extension of traditional 1D-RoPE (Su et al., 2024) by partitioning the feature dimension into three approximately equal segments (for the temporal, height, and width axes) and applying the 1D rotations separately to the segment for each axis. We found that using 3D-RoPE instead of absolute sincos position embeddings (Vaswani et al., 2017) helps stabilize training for the largest models. To process a video with our transformer encoder, we first patchify it as a sequence of tubelets of size $2 \times 16 \times 16$ ($T \times H \times W$) and employ the same multiblock masking strategy as in Bardes et al. (2024).

Key Scaling Ingredients. In this section we introduce and study four additional key ingredients which enable scaling the V-JEPA pre-training principle to obtain our V-JEPA 2 model.

1. *Data scaling:* We increase the dataset size from 2 million to 22 million videos by leveraging and curating additional data sources.
2. *Model scaling:* We scale the encoder architecture from 300 million to over 1 billion parameters, going from a ViT-L to a ViT-g (Zhai et al., 2022).
3. *Longer training:* Adopting a warmup-constant-decay learning rate schedule simplifies hyperparameter tuning and enables us to extend training from 90 thousand up to 252 thousand iterations, effectively leveraging the additional data.
4. *Higher resolution:* We leverage the warmup-constant-decay schedule to efficiently scale to higher resolution video and longer video clips by training on shorter, lower-resolution clips during the warmup and constant phases, and then increasing resolution and/or clip-length during the final decay phase.

The remainder of this section describes each of these ingredients in further detail and also quantifies the impact of each ingredient using the evaluation protocol described next.

Evaluation Protocol. Our goal with model pretraining is to infuse general visual understanding into our encoder. We therefore evaluate our model and data design choices by assessing the quality of the model's learned representation on a set of six motion and appearance classification tasks: Something-Something v2 (Goyal et al., 2017), Diving-48 (Li et al., 2018), Jester (Materzynska et al., 2019), Kinetics (Kay et al., 2017), COIN (Tang et al., 2019), and ImageNet (Deng et al., 2009). We use a frozen evaluation protocol: we freeze the encoder weights and train a task-specific 4-layers attentive probe on its representation to output a predicted class. In this section, we focus mainly on the average accuracy across the six understanding tasks. Refer to Section 5 for additional details about the tasks, evaluation protocol, and results.

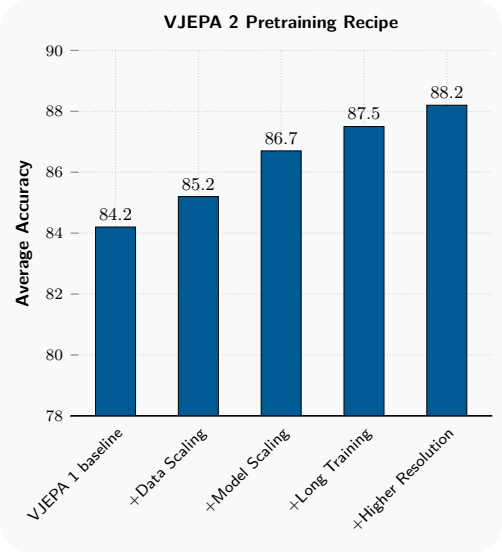


Figure 3 Scaling Ingredients. The effects of scaling interventions on average accuracy across 6 image and video classification tasks (SSv2, Diving-48, Jester, Kinetics, COIN, ImageNet) using a ViT-L/16 model as baseline.

2.2 Scaling Self-Supervised Video Learning

We first present a summary of the key findings of our scaling analysis, where we investigate the impact of the four key ingredients on downstream task average performance. Figure 3 illustrates the effects of these scaling interventions on average accuracy across 6 classification tasks, using a ViT-L/16 model pretrained on 2 million videos with the V-JEPA objective as our baseline. Increasing the dataset from 2 million to 22 million videos (VM22M) yields a 1.0-point improvement. Scaling the model from 300 million to 1 billion parameters (ViT-g/16) provides an additional 1.5-point gain. Extending training from 90K to 252K iterations contributes another 0.8-point improvement. Finally, enhancing both spatial resolution ($256 \rightarrow 384$) and temporal duration ($16 \rightarrow 64$ frames), during both pretraining and evaluation, boosts performance to 88.2%, representing a

Table 1 VideoMix22M (VM22M) Pretraining Dataset. To build our observation pretraining dataset, we combined four different video sources and one image dataset. We use a source-specific sampling probability during training and apply retrieval-based curation on YT1B to reduce noisy content (e.g., cartoon- or clipart-style).

Source	Samples	Type	Total Hours	Apply Curation	Weight
SSv2 (Goyal et al., 2017)	168K	EgoVideo	168	No	0.056
Kinetics (Carreira et al., 2019)	733K	ExoVideo	614	No	0.188
Howto100M (Miech et al., 2019)	1.1M	ExoVideo	134K	No	0.318
YT-Temporal-1B (Zellers et al., 2022)	19M	ExoVideo	1.6M	Yes	0.188
ImageNet (Deng et al., 2009)	1M	Images	n/a	No	0.250

cumulative 4.0-point improvement over the ViT-L/16 baseline. Each individual change provides a positive impact, confirming the potential of scaling in video self-supervised learning (SSL).

2.3 Pretraining Dataset

Next, we describe the sources of videos and images that make up our pretraining dataset, and our approach to curating the dataset.

Scaling Dataset Size. We construct a large-scale video dataset by combining publicly available data sources. Using publicly-available sources in this work enables other researchers to reproduce these results. The overall dataset includes ego-centric videos from the Something-Something v2 dataset (SSv2) introduced in Goyal et al. (2017), exo-centric action videos from the Kinetics 400, 600, and 700 datasets (Kay et al., 2017; Carreira et al., 2018, 2019), YouTube tutorial videos from HowTo100M (Miech et al., 2019), and general YouTube videos from YT-Temporal-1B (Zellers et al., 2022), which we refer to as YT1B. We also include images from the ImageNet dataset (Deng et al., 2009) to increase the visual coverage of the pretraining data. To enable joint image and video pretraining, we duplicate an image temporally and treat it as a 16-frame video where all frames are identical. During training, we sample from each data source with a weighting coefficient that we determined empirically via manual tuning. The resulting dataset, which we refer to as VideoMix22M (or VM22M), consists of 22 million samples. Table 1 lists these data sources and their weights.

Figure 4 (Left) compares the performance of a ViT-L/16 pretrained on VM22M with a similar model trained on the smaller (2 million) VideoMix2M dataset from Bardes et al. (2024). Training on VM22M leads to a +1 point improvement on average performance on visual understanding tasks, compared to VM2M. Performance improvement is more prominent on appearance-based tasks such as Kinetics-400, COIN, and ImageNet, showing the importance of increasing visual coverage for those tasks.

Data Curation. YT1B is a large video dataset, consisting of 1.4 million video-hours, with no curation and minimal filtering compared to smaller video datasets (like Kinetics and Something-Something v2). Because uncurated and unbalanced data can hinder model performance (Assran et al., 2022; Oquab et al., 2023), we filter YT1B by adapting an existing retrieval-based curation pipeline to handle videos. Specifically, we extract scenes from YT1B videos, compute an embedding vector for each scene, and then use a cluster-based retrieval process (Oquab et al., 2023) to select video scenes according to a target distribution, which is composed of the Kinetics, Something-Something v2, COIN and EpicKitchen training datasets. We describe the details of the dataset construction procedure in Appendix A.2. Similar to Oquab et al. (2023), we ensure that none of the videos from the target validation sets are contained in the initial, uncurated data pool.

In Figure 4 (Right), we compare the average performance on visual understanding evaluations between a ViT-L model pretrained on uncurated YT-1B data and a comparable model trained on our Curated-YT-1B dataset. Training with the curated dataset yields a +1.4 point average performance improvement over the uncurated baseline. Notably, the Curated-YT-1B-trained model achieves competitive performance relative to the full VM22M dataset at the ViT-L scale. However, larger-scale models benefit more from VM22M training (see Appendix A.2), suggesting that combining Curated-YT-1B with other data sources enhances scalability.

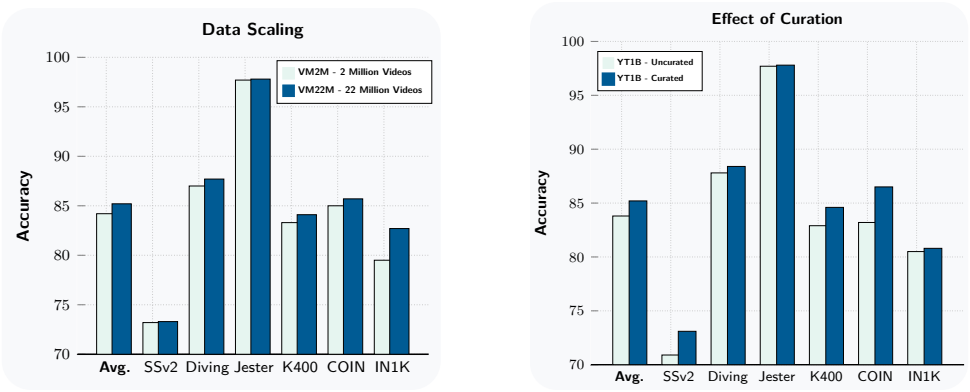


Figure 4 Data Scaling & Curation. We train and compare models on different data-mixes. Models are ViT-L/16 trained for 90K iterations using a cosine learning schedule following Bardes et al. (2024). **(Left)** We compare the performance of a ViT-L/16 model pretrained on the VM2M dataset and our VM22M dataset. Training on the VM22M dataset leads to a +1 point improvement in average performance. Performance improvement is more pronounced on appearance-based tasks such as Kinetics-400, COIN, and ImageNet **(Right)** We compare the performance of a ViT-L/16 model pretrained on YT1B and a model pretrained on our Curated-YT1B dataset, which leverages our cluster-based curation. Training on the curated dataset leads to a +1.4 point improvement on average performances, showing the effectiveness of data-curation.

2.4 Pretraining Recipe

Scaling Model Size. To explore the scaling behavior of our model, we trained a family of encoder models with parameter counts ranging from 300 million (ViT-L) to 1 billion (ViT-g) parameters. All encoder architecture details are provided in Table 12 in the appendix. Note that each encoder uses the same predictor architecture, similar to a ViT-small. We report the average performance of these encoders on visual understanding tasks in Figure 5 (Left). Scaling the model size from 300 million (ViT-L) to 1 billion (ViT-g) parameters yields a +1.5 points average performance improvement. Both motion and appearance understanding tasks benefit from scaling, with SSv2 improving by +1.6 points and Kinetics by +1.5 points (cf.Table 4). These results confirm that self-supervised video pretraining effectively leverages larger model capacities, up to the 1B-parameter ViT-g.

Training Schedule. V-JEPA 2 model training employs a warmup-constant learning rate schedule followed by a cooldown phase (Zhai et al., 2022; Hägele et al., 2024). Similarly to Hägele et al. (2024), we found that this schedule performs comparably to a half-cosine schedule (Loshchilov and Hutter, 2016); it also makes exploring long training runs more cost-effective, since multiple cooldown runs can be started from different checkpoints of the constant phase. We simplified the recipe from Bardes et al. (2024) by maintaining fixed teacher EMA and weight decay coefficients instead of using ramp-up schedule, as these variations showed minimal impact on downstream understanding tasks. Figure 3 shows that extending the training schedule from 90K to 252K iterations yields a +0.8 average performance improvement with ViT-g models, validating the benefits of extended training durations. This schedule also facilitates progressive training by incrementally increasing video resolution during the cooldown phase.

Efficient Progressive-Resolution Training. While most previous video encoders focus on short clips of 16 frames (roughly seconds) (Bardes et al., 2024; Wang et al., 2024b, 2023), we explore training with longer clips of up to 64 frames (16 seconds) at higher spatial resolutions. However, training time increases dramatically with longer durations and higher resolutions — training our ViT-g model on $64 \times 384 \times 384$ inputs would require roughly 60 GPU-years (see Figure 5, Middle). To reduce this, we adopt a progressive resolution strategy (Touvron et al., 2019; Oquab et al., 2023) that boosts training efficiency while maintaining downstream performance. Our training process begins with a warmup phase where we train on 16-frame, 256×256-resolution videos with linear learning rate warmup over 12K iterations, followed by a main training phase with a constant learning rate for 228K iterations. Then, during the cooldown phase, we increase video duration and resolution while linearly decaying the learning rate over 12K iterations. Hence the additional

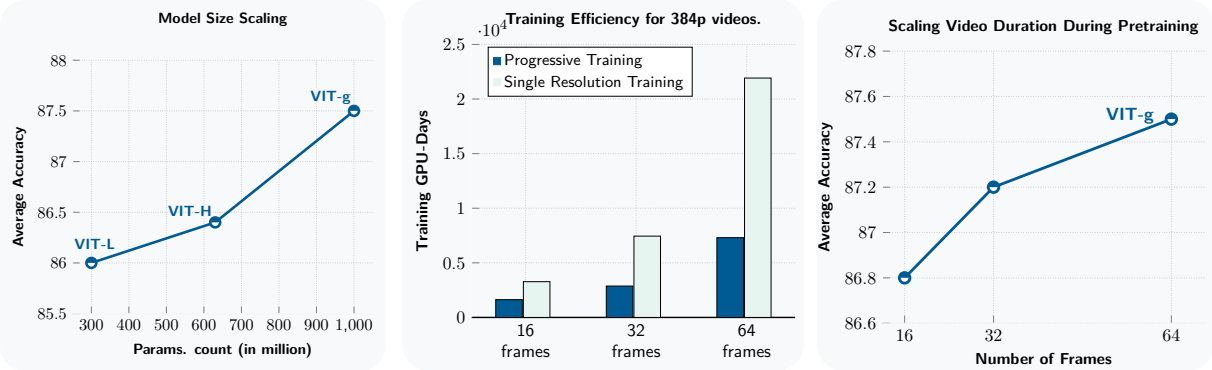


Figure 5 Model Scaling. We explore the impact of scaling model size and input video resolution. All models are trained on the VideoMix22M pretraining dataset. **(Left)** Average performance across six understanding tasks as a function of model scale. Models are trained with a constant learning rate until performance plateaus on downstream tasks. We then cool down the model using 64 frames at 256×256 resolution and report post-cooldown performance. Scaling the model size from 300M to 1B parameters yields a +1.7 point average improvement. **(Middle)** Training times (GPU-days) for ViT-g on A100 GPUs when training videos at 384×384 resolution with different numbers of frames per clip. We compare progressive resolution training (252K iterations at 16 frames / 256×256 resolution, followed by 12K cooldown iterations at 384×384 resolution) to the projected time for full-resolution training. Progressive training provides up to 8× speedup, significantly reducing the pretraining compute requirement. **(Right)** Effect of inscreasing video duration at cooldown on downstream performance for ViT-g. Even when only using 16-frame clips during inference/evaluation, increasing video duration during the cooldown phase of training improves average task performance by +0.7 points.

computational overhead associated with training on longer-duration, higher-resolution videos is only incurred during the final cooldown phase. This approach enables efficient high-resolution training: as shown in Figure 5 (Middle), we achieve an 8.4× reduction in GPU time for a model that can ingest 64-frame, 384×384 resolution inputs, compared to directly training such a model from scratch at full resolution throughout all phases of training. Furthermore, we still observe the benefits of a model that can process longer-duration and higher-resolution inputs as discussed next.

Scaling temporal and spatial video resolution. Figure 5 examines how input video resolution affects downstream task performance. When increasing clip duration from 16 to 64 frames during pretraining while maintaining a fixed 16-frame evaluation duration, we observe a +0.7 percentage point average performance improvement (Figure 5, Right). Additionally, we see that increasing the video duration and resolution during evaluation leads to a significant improvement across the tasks (refer to Table 4 and Appendix A.4.2). These results demonstrate that video self-supervised pretraining benefits from increased temporal resolution during both training and evaluation. Although we experimented with scaling to even longer video clips (128 and 256 frames), we did not observe any further improvement beyond 64 frames on this set of understanding tasks.

3 V-JEPA 2-AC: Learning an Action-Conditioned World Model

After pre-training, the V-JEPA 2 model can make predictions about missing part in videos. However, these predictions do not directly take into account the causal effect of actions that an agent might take. In the next stage of training, described in this section, we focus on making the model useful for planning by leveraging a small amount of interaction data. To that end, we learn a frame-causal action-conditioned predictor on top of the frozen V-JEPA 2 video encoder (Figure 2, right). We train our model on data from the Droid dataset (Khazatsky et al., 2024) consisting of data from experiments with a table-top Franka Panda robot arm collected through teleoperation. We refer to the resulting action-conditioned model as V-JEPA 2-AC, and in Section 4 we show that V-JEPA 2-AC can be used within a model-predictive control planning loop to plan actions in new environments.

3.1 Action-Conditioned World Model Training

Our goal is to take the V-JEPA 2 model after pre-training and obtain a latent world model that can be used for control of an embodied agentic system via closed-loop model-predictive control. To achieve this, we train V-JEPA 2-AC, an autoregressive model that predicts representations of future video observations conditioned on control actions and proprioceptive observations.

In this section we describe a concrete instantiation of this framework for a tabletop arm with a fixed exocentric camera, and where control actions correspond to end-effector commands. The model is trained using approximately 62 hours of unlabeled video from the raw Droid dataset, which consists of short videos, typically 3–4 seconds long, of a 7-DoF Franka Emika Panda arm equipped with a two-finger gripper. Here, *unlabeled* video refers to the fact that we do not use additional meta-data indicating any reward, what type of task was being performed in each demonstration, or whether the demonstration was successful or not in completing the task being attempted. Rather, we only use the raw video and end-effector state signals from the dataset (each video in the dataset is accompanied by meta-data indicating the end-effector state in each frame — three dimensions for position, three for orientation, and one for the gripper state).

Model inputs. In each iteration of training we randomly sample a mini-batch of 4 second video clips from the Droid dataset, and, for simplicity, discard any videos shorter than 4 seconds, leaving us with a smaller subset of the dataset comprising under 62 hours of video. The video clips are sampled with resolution 256×256 and a frame-rate of 4 frames-per-second (fps), yielding 16 frame clips denoted by $(x_k)_{k\in[16]}$, where each x_k represents a single video frame. The robot's end-effector state in each observation is denoted by the sequence $(s_k)_{k\in[16]}$, where s_k is a real-valued 7D vector defined relative to the base of the robot. The first three dimensions of s_k encode the cartesian position of the end-effector, the next three dimensions encode its orientation in the form of extrinsic Euler angles, and the last dimension encodes the gripper state. We construct a sequence of actions $(a_k)_{k\in[15]}$ by computing the change in end-effector state between adjacent frames. Specifically, each action a_k is a real-valued 7-dimensional vector representing the change in end-effector state between frames k and $k+1$. We apply random-resize-crop augmentations to the sampled video clips with the aspect-ratio sampled in the range (0.75, 1.35).

Loss function. We use V-JEPA 2 encoder $E(\cdot)$ as an image encoder and encode each frame independently in a given clip to obtain a sequence of feature maps $(z_k)_{k\in[16]}$, where $z_k \coloneqq E(x_k) \in \mathbb{R}^{H\times W\times D}$ with $H \times W$ denoting the spatial resolution of the feature map, and D the embedding dimension. In practice, our feature maps are encoded using the ViT-g encoder and have the shape $16 \times 16 \times 1408$. Note that the encoder is kept frozen during this post-training phase. The sequence of feature maps, end-effector states, and actions are temporally interleaved as $(a_k, s_k, z_k)_{k\in[15]}$ and processed with the transformer predictor network $P_\phi(\cdot)$ to obtain a sequence of next state representation predictions $(\hat{z}_{k+1})_{k\in[15]}$. The scalar-valued teacher-forcing loss function is finally computed as

$$\mathcal{L}_{\text{teacher-forcing}}(\phi) \coloneqq \frac{1}{T}\sum_{k=1}^{T}\|\hat{z}_{k+1} - z_{k+1}\|_1 = \frac{1}{T}\sum_{k=1}^{T}\left\|P_\phi\left((a_t, s_t, E(x_t))_{t\leq k}\right) - E(x_{k+1})\right\|_1, \tag{2}$$

with $T = 15$. We also compute a two-step rollout loss to improve the model's ability to perform autoregressive rollouts at inference time. For simplicity of exposition and with slight overloading of notation, let $P_\phi(\hat{a}_{1:T}; s_k, z_k) \in \mathbb{R}^{H\times W\times D}$ denote the final predicted state representation obtained by autoregressively running V-JEPA 2-AC with an action sequence $(\hat{a}_i)_{i\in[T]}$, starting from $(s_k,\, z_k)$. We can now denote the rollout loss as:

$$\mathcal{L}_{\text{rollout}}(\phi) \coloneqq \|P_\phi(a_{1:T}, s_1, z_1) - z_{T+1}\|_1. \tag{3}$$

In practice we use $T = 2$ for computing the rollout loss, such that we only differentiate the predictor through one recurrent step.

The overall training objective is thus given by

$$L(\phi) \coloneqq \mathcal{L}_{\text{teacher-forcing}}(\phi) + \mathcal{L}_{\text{rollout}}(\phi), \tag{4}$$

and is minimized with respect to the predictor weights ϕ. For illustrative purposes, the training procedure is depicted in Figure 6 with $T = 4$ for both the teacher forcing and rollout loss.

Figure 6 V-JEPA 2-AC training. V-JEPA 2-AC is trained in an autoregressive fashion, utilizing a teacher forcing loss and a rollout loss. (**Left**) In the teacher forcing loss, the predictor takes the encoding of the current frame representation as input and learns to predict the representation of the next timestep. (**Right**) The rollout loss involves feeding the predictor's output back as input, allowing the model to be trained to predict several timesteps ahead. By optimizing the sum of these two losses, V-JEPA 2-AC enhances its ability to accurately forecast the future by reducing error accumulation during rollouts.

Architecture. The predictor network $P_\phi(\cdot)$ is a ∼300M parameter transformer network with 24-layers, 16 heads, 1024 hidden dimension, and GELU activations. The action, end-effector state, and flattened feature maps input to the predictor are processed with separate learnable affine transformations to map them to the hidden dimension of the predictor. Similarly, the outputs of the last attention block of the predictor go through a learnable affine transformation to map them back to the embedding dimension of the encoder. We use our 3D-RoPE implementation to represent the spatiotemporal position of each video patch in the flattened feature map, while only applying the temporal rotary positional embeddings to the action and pose tokens. We use a block-causal attention pattern in the predictor so that each patch feature at a given time step can attend to the action, end-effector state, and other patch features from the same timestep, as well as those from previous time steps.

3.2 Inferring Actions by Planning

Energy minimization. Given an image of the goal state, we leverage V-JEPA 2-AC for downstream tasks by planning. Specifically, at each time step, we plan an action sequence for a fixed time horizon by minimizing a goal-conditioned energy function. We then execute the first action, observe the new state, and repeat the process. Let s_k denote the current end-effector state, and x_k and x_g denote the current observed frame and goal image, respectively, which are separately encoded with the video encoder to obtain the feature maps z_k and z_g. Given a planning horizon, T, we optimize a sequence of robot actions, $(a_i^\star)_{i\in[T]}$, by minimizing a goal-conditioned energy function,

$$\mathcal{E}(\hat{a}_{1:T};\ z_k, s_k, z_g) \coloneqq \|P(\hat{a}_{1:T}; s_k, z_k) - z_g\|_1, \tag{5}$$

such that $(a_i^\star)_{i\in[T]} \coloneqq \operatorname{argmin}_{\hat{a}_{1:T}} \mathcal{E}(\hat{a}_{1:T};\ z_k, s_k, z_g)$. As illustrated in Figure 7, the model infers an action sequence $(a_i^\star)_{i\in[T]}$ by selecting a trajectory that minimizes the L1 distance between the world model's imagined state representation T steps into the future and its goal representation. In practice, we minimize (5) in each

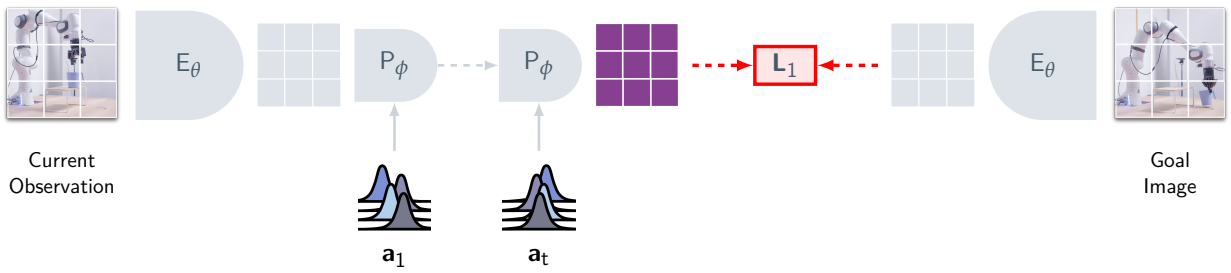


Figure 7 Planning. We plan an action sequence for a fixed time horizon T by minimizing the L1 distance between the world model's imagined state representation T steps into the future and its goal representation. The L1 loss is optimized with respect to the actions $(a_k)_{k\in[T]}$ using the cross-entropy method (Rubinstein, 1997). Specifically, in each planning step, we sample the action coordinates at each point in the planning horizon from a sequence of Gaussian distributions initialized with zero mean and unit variance. The population statistics of the top-k actions trajectories are used to update the mean and variance of the Gaussian distributions. This process is repeated for several iterations before finally returning the mean of the sequence of Gaussians as the selected action trajectory.

planning step using the Cross-Entropy Method (Rubinstein, 1997), and only execute the first action on the robot before re-planning, as in receding horizon control.

4 Planning: Zero-shot Robot Control

In this section we demonstrate how V-JEPA 2-AC can be used to implement basic robot skills like reaching, grasping, and pick-and-place via model-predictive control. We focus on tasks with visual goal specification and show that V-JEPA 2-AC generalizes zero-shot to new environments.

4.1 Experimental Setup

Baselines. We compare the performance of V-JEPA 2-AC with two baselines, one vision-language-action model trained with behavior cloning, and one video generation-based world model.

The first baseline is based on the Octo video-language-action model that allows for goal-image conditioning (Octo Model Team et al., 2024). We start from the open-source weights of the *octo-base-1.5* version of the model, which is pretrained on the *Open-X Embodiment* dataset containing over 1M trajectories.[1] We fine-tune the Octo model with behaviour cloning on the entire Droid dataset using hindsight relabeling (Andrychowicz et al., 2017; Ghosh et al., 2019) with image goals and end-effector states. In particular, we sample random segments of trajectories from the Droid dataset during training, and uniformly sample goal images up to 20 timesteps forward in the trajectory. We use the official open-source code for fine-tuning, including all standard Droid optimization hyperparameters, and leverage single side image view inputs at 256×256 resolution, a context of two previous frames, and a horizon of 4 future actions.

The second baseline we compare with is based on the Cosmos video generation model (Agarwal et al., 2025). We start with the open-source weights for the action-free Cosmos model (latent diffusion-7B with continuous tokenizer), which was trained on 20M hours of video, and we fine-tune the model on Droid using the officially-released action-conditioned fine-tuning code.[2] To improve performance when training on Droid, we (i) lowered the learning rate to match that used in the video-conditioned Cosmos recipe, (ii) removed the dropout in the video conditioning to improve the training dynamics, and (iii) increased the noise level by a factor of e^2, as we observed that the model trained with a lower noise factor struggled to leverage the information in the conditioning frame. Although the Cosmos technical report (Agarwal et al., 2025)

[1] In comparison, we train V-JEPA 2-AC on 23k trajectories from Droid, including successes and failures.
[2] https://github.com/nvidia-cosmos/cosmos-predict1

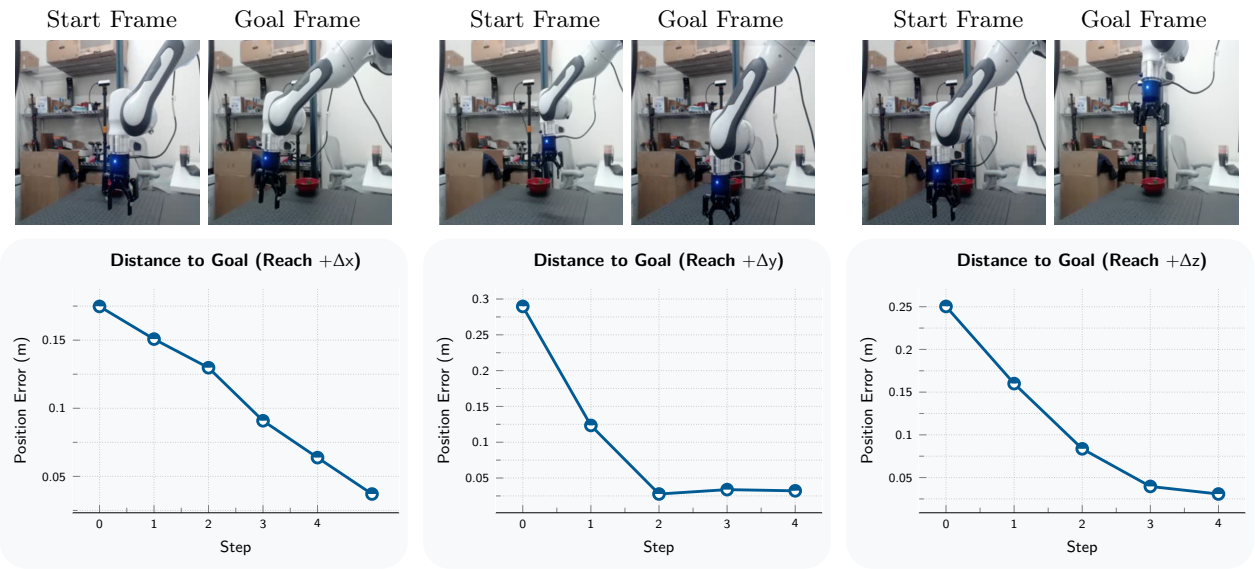


Figure 8 Single-Goal Reaching. Single-goal reaching involves moving the end-effector to a desired location in space based on a single goal image. This task measures for a basic understanding of actions as well as a 3D spatial understanding of the scene, including depth, from the monocular RGB camera. In each step, we use V-JEPA 2-AC to plan a sequence of actions by minimizing the L1 distance between the model's imagined future state representations and its representation of the goal frame. The first action is then executed before re-planning in the next time step. During planning, we only sample individual actions in the L1-Ball of radius 0.075 centered at the origin. Thus, the maximum achievable decrease in cartesian distance to the goal in a single step is 0.13 (∼13 cm).

mentions using world models for planning or model-predictive control as a future application, to the best of our knowledge this is the first reported attempt using Cosmos models for robot control.

Robot deployment. All models are deployed zero-shot on Franka Emika Panda arms with RobotiQ grippers, located in two different labs, neither of which appears in the Droid dataset. Visual input is provided through an uncalibrated low-resolution monocular RGB camera. The robots use the same exact model weights and inference code, and similar low-level controllers based on operational space control. We use blocking control for both the V-JEPA 2-AC world model and Cosmos world model (i.e., the system waits for the last commanded action to be completed before sending a new action to the controller) and experiment with both blocking and non-blocking control for Octo, and report the best performance across the two options. When planning with V-JEPA 2-AC and Cosmos, we constrain each sampled action to the L1-Ball of radius 0.075 centered at the origin, which corresponds to a maximum end-effector displacement of approximately 13 cm for each individual action, since large actions are relatively out-of-distribution for the models.

4.2 Results

Single-goal reaching. First, we evaluate on the task of single-goal reaching, which involves moving the end-effector to a desired location in space based on a single goal image. This task measures for a basic understanding of actions as well as a 3D spatial understanding of the scene (including depth) from the monocular RGB camera.

Figure 8 shows the Euclidean distance between the end-effector and its goal position during robot execution for three different single-goal reaching tasks. In all cases, the model is able to move the end-effector within less than 4 cm of its goal position, and select actions that lead to a monotonic decrease in the error. This can be seen as a form of visual servoing (Hill, 1979), wherein visual feedback from a camera is used to control a robot's motion. However, unlike classical approaches in visual servoing, V-JEPA 2-AC achieves this by training on unlabeled, real-world video data.

In Figure 9, we visualize the V-JEPA 2-AC energy landscape from equation (5) for the Δy reaching task as a

function of a single cartesian-control action, sweeping Δx and Δy while holding $\Delta z = 0$ fixed. The energy function achieves its minimum near the ground-truth action, providing further evidence that the model has learned to reasonably infer the effect of actions without requiring precision sensing. It is also interesting to observe that the energy landscape induced by V-JEPA 2-AC is relatively smooth and locally convex, which should facilitate planning.

Prehensile manipulation. Next, we evaluate all models on more challenging prehensile object manipulation tasks, namely *grasp*, *reach with object*, and *pick-and-place*. Success rates are reported in Table 2 and Table 3, and averaged across 10 trials with various permutations to the task across trials (e.g., object location, starting pose, etc.). For the *grasp* and *reach with object* tasks the model is shown a single goal image. For the *pick-and-place* tasks we present two sub-goal images to the model in addition to the final goal. The first goal image shows the object being grasped, the second goal image shows the object in the vicinity of the goal position. The model first optimizes actions with respect to the first sub-goal for 4 time-steps before automatically switching to the second sub-goal for the next 10 time-steps, and finally the third goal for the last 4 time-steps. Examples of robot execution for the pick-and-place task are shown in Figure 10. Start and goal frames for all individual tasks in Lab 1 are shown in Appendix B.2. The *grasp* task requires precise control from visual feedback to correctly grip the object. The *reach with object* task requires the model to navigate while holding an object, which necessitates a basic understanding of intuitive physics to avoid dropping the object. Finally, the *pick-and-place* task tests for the ability to compose these atomic skills.

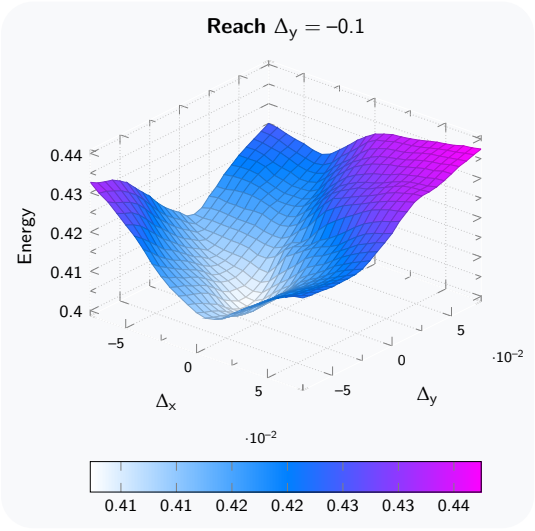


Figure 9 V-JEPA 2-AC Energy Landscape. Energy landscape for single-goal reaching task with respect to end-effector cartesian-control action (sweeping Δx and Δy while holding $\Delta z = 0$ fixed); ground truth action relating goal image to start frame is located at $(\Delta x, \Delta y) = (0, -0.1)$. We see that the energy function achieves its minimum around $(\Delta x, \Delta y) \approx (0, -0.05)$, indicating that the model has learned to reasonably infer the effect of actions without requiring precision sensing.

While all models achieve a high success-rate on *reach*, differences in performance are more apparent on tasks involving object interaction. We observe that the success-rate for all models depends on the type of object being manipulated. For instance, we find that the cup is mostly easily grasped by placing one finger inside the object and gripping around the rim, however if the control actions produced by the model are not accurate enough, the robot will miss the rim of the cup and fail to grasp the object. When manipulating the box, there are many more feasible grasping configurations, however, the model requires more precise gripper control to ensure that the fingers are open wide enough to grasp the object. We see that, for all models, the variation in success-rate with respect to the object type is due to the combination of sub-optimal actions and the unique challenges associated with manipulating each respective object. Nonetheless, we see that the V-JEPA 2-AC model achieves the highest success-rate across all tasks, highlighting the feasibility of latent planning for robot manipulation.

In Table 3, we compare planning performance when using V-JEPA 2-AC versus the Cosmos action-conditioned video generation model based on latent diffusion. In both cases we leverage the cross-entropy method (Rubinstein, 1997) for optimizing the sequence of actions using a single NVIDIA RTX 4090 GPU, and construct the energy function by encoding the goal frame in the latent space of the model, as in equation (5). With 80 samples, 10 refinement steps, and a planning horizon of 1, it takes 4 minutes to compute a single action in each planning step with Cosmos. While we achieve a high success rate of 80% on the *reach* tasks when using Cosmos, performance on object interaction tasks is weaker. Note that under a planning time of 4 minutes per action, a full pick & place trajectory requires over one hour of robot execution. By contrast, with 10× more samples in each refinement step, the V-JEPA 2-AC world model requires only 16 seconds per action and leads to higher performance across all considered robot skills. We can potentially reduce the planning time for both models in future work by leveraging additional computing resources for planning, reducing the number

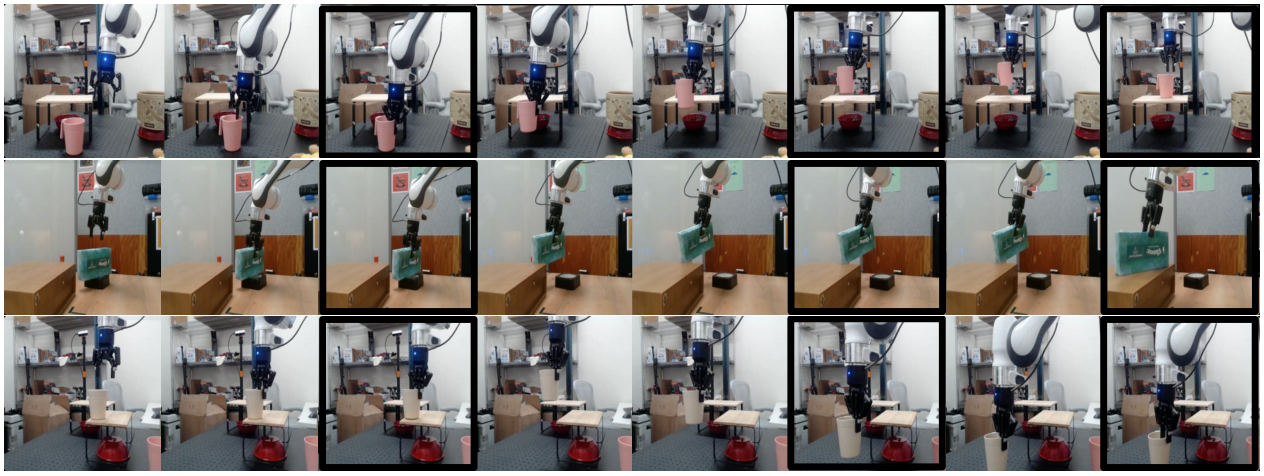


Figure 10 Pick-&-Place. Closed-loop robot execution of V-JEPA 2-AC for multi-goal pick-&-place tasks. Highlighted frames indicate when the model achieves a sub-goal and switches to the next goal. The first goal image shows the object being grasped, the second goal image shows the object in the vicinity of the desired location, and the third goal image shows the object placed in the desired position. The model first optimizes actions with respect to the first sub-goal for 4 time-steps before automatically switching to the second sub-goal for the next 10 time-steps, and finally the third goal for the last 4 time-steps. Robot actions are inferred through goal-conditioned planning. The V-JEPA 2-AC model is able to perform zero-shot pick-&-place tasks on two Franka arms in different labs, with various object configurations and cluttered environments.

Table 2 Zero-Shot Robot Manipulation. All models are deployed zero-shot on two Franka arms with RobotiQ grippers located in different labs. Given image-goals for each considered task, all models run closed loop to infer a sequence of actions to achieve the goal. Success rates are reported out of 10 trials with various permutations to the task across trials (e.g., object location, starting pose, etc.).

			Grasp		**Reach w/ Obj.**		**Pick-&-Place**	
Method		**Reach**	Cup	Box	Cup	Box	Cup	Box
Octo (Octo Model Team et al., 2024)	Lab 1	100%	20%	0%	20%	70%	20%	10%
	Lab 2	100%	10%	0%	10%	70%	10%	10%
	Avg	*100%*	*15%*	*0%*	*15%*	*70%*	*15%*	*10%*
V-JEPA 2-AC (ours)	Lab 1	100%	70%	30%	90%	80%	80%	80%
	Lab 2	100%	60%	20%	60%	70%	80%	50%
	Avg	*100%*	*65%*	*25%*	*75%*	*75%*	*80%*	*65%*

of samples and refinement steps used at each time step, training a feed-froward policy in the world-models' imagination to initialize the planning problem, or potentially leveraging gradient-based planning in the case of V-JEPA 2-AC.

4.3 Limitations

Sensitivity to camera positioning. Since the V-JEPA 2-AC model is trained to predict representations of the next video frame given an end-effector Cartesian control action, without any explicit camera calibration, it must therefore implicitly infer the action coordinate axis from the monocular RGB camera input. However, in many cases, the robot base is not visible in the camera frame, and thus the problem of inferring the action coordinate axis is not well defined, leading to errors in the world model. In practice, we manually tried different camera positions before settling on one that worked well across all of our experiments. We conduct a quantitative analysis of the V-JEPA 2-AC world model's sensitivity to camera position in Appendix B.4.

Long horizon planning. Long horizon planning with world models is limited by a number of factors. First, autoregressive prediction suffers from error accumulation: the accuracy of the representation-space

Table 3 Planning Performance. Comparing closed-loop robot manipulation using MPC with V-JEPA 2-AC world model and Cosmos world model. In both cases we leverage the cross-entropy method (Rubinstein, 1997) for optimizing the sequence of actions using a single NVIDIA RTX 4090 GPU. For each robot skill, we evaluate each model across 10 tasks and average the results. With 80 samples, 10 refinement steps, and a planning horizon of 1, it takes 4 minutes to compute a single action in each planning step with Cosmos, which is an action-conditioned video generation model based on latent diffusion. Note that under a planning time of 4 minutes per action, a full pick & place trajectory takes over one hour. By contrast, with 10× more samples in each refinement step, the V-JEPA 2-AC world model requires only 16 seconds per action and leads to higher performance across all considered robot skills.

Lab 2	Planning Details					Grasp		Pick-&-Place	
Method	#Samples	Iter.	Horizon	Time	**Reach**	Cup	Box	Cup	Box
Cosmos (Agarwal et al., 2025)	80	10	1	4 min.	80%	0%	20%	0%	0%
V-JEPA 2-AC (ours)	800	10	1	16 sec.	100%	60%	20%	80%	50%

predictions decreases with longer autoregressive rollouts, thereby making it more difficult to reliably plan over long horizons. Second, long-horizon planning increases the size of the search space: the number of possible action trajectories increases exponentially given a linear increase in the planning horizon, thereby making it computationally challenging to plan over long horizons. On the other hand, long-horizon planning is necessary for solving non-greedy prediction tasks, e.g., pick-and-place without image sub-goals. Future work exploring world models for long-horizon planning will enable the solution of many more complex and interesting tasks.

Image goals. Following many previous works in goal-conditioned robot manipulation (Finn and Levine, 2017; Lynch et al., 2020; Chebotar et al., 2021; Jang et al., 2022; Liu et al., 2022; Gupta et al., 2022), our current formulation of the optimization target assumes that we have access to visual goals. However, when deploying robots in-the-wild, it may be more natural to express goals in other forms, such as with language. Future work that aligns latent action-conditioned world models with language models will step towards more general task specification via natural language.

5 Understanding: Probe-based Classification

The capabilities of a representation-space world model, such as V-JEPA 2-AC discussed above, are inherently limited by the state information encoded in the learned representation space. In this section and subsequent sections, we probe the representations learned by V-JEPA 2 and compare the V-JEPA 2 encoder to other vision encoders on visual classification.

Visual classification tasks can focus either on *appearance understanding* or *motion understanding*. While appearance understanding tasks can generally be solved using information visible in a single frame of an input video clip (even when the classification labels describe actions), motion understanding tasks require several frames to correctly classify a video (Goyal et al., 2017). To ensure a balanced evaluation of both motion and appearance, we have selected three motion understanding tasks, namely Something-Something v2 (SSv2), Diving-48, and Jester, which require the model to understand human gestures and movements. For appearance understanding, we have chosen Kinetics400 (K400), COIN, and ImageNet (IN1K), which involve recognizing actions, scenes, and objects. Empirically, we show that V-JEPA 2 outperforms state-of-the-art visual encoders on motion understanding tasks, while being competitive on appearance understanding tasks.

Attentive Probe. We train an 4-layers attentive probe on top of the frozen encoder output using the training data from each task. Our attentive probe is composed of four transformer blocks, the last of which replaces standard self-attention with a cross-attention layer using a learnable query token. Following standard practice, several clips with a fixed number of frames are sampled from a video during inference. The classification logits are then averaged across clips. We keep the resolution similar to the one used for V-JEPA 2 pretraining. We ablate the number of layers of our attentive probe in Appendix C.2, and also provide full details on the number of clips, clip size, and other hyperparameters used in the downstream tasks.

Table 4 Action and Object Classification. We report the classification performance of V-JEPA 2 models pretrained on 64 frames at resolution 256×256 for all models, except V-JEPA 2 ViT-g_{384} which was pretrained at resolution 384×384, on action and object classification, and compare their performance with state-of-art image and video encoders. All models follow the same evaluation protocol except for V-JEPA 2 ViT-g_{384}. We use 256×256 resolution with $16 \times 2 \times 3$ inputs for SSv2 (16 frames clip, 2 temporal crops, 3 spatial crops), $16 \times 8 \times 3$ for K400, $32 \times 8 \times 3$ for COIN and $32 \times 4 \times 3$ for Diving-48 and Jester. V-JEPA 2 ViT-g_{384} uses a higher resolution of 384×384 for all six tasks, and additionally uses $64 \times 2 \times 3$ inputs for SSv2 and 32x8x3 inputs for COIN. Our V-JEPA 2 ViT-g significantly outperforms other vision encoders on motion understanding tasks and is competitive on appearance tasks. It achieves the best average performance of 87.5 across all image and videos encoders. V-JEPA 2 ViT-g_{384} further improves results consistently across tasks, reaching 88.2 average performance. $*$: **PE$_{\text{core}}$G** achieves an accuracy 89.8% on ImageNet using an attentive probe and input resoluton of 448px (Bolya et al., 2025). We use an input resolution of 256px and a different probe architecture in our case.

			Motion Understanding			*Appearance Understanding*		
Method	**Param.**	**Avg.**	**SSv2**	**Diving-48**	**Jester**	**K400**	**COIN**	**IN1K**
Results Reported in the Literature								
VideoMAEv2 (Wang et al., 2023)	1B	–	56.1	–	–	82.8	–	71.4
InternVideo2-1B (Wang et al., 2024b)	1B	–	67.3	–	–	87.9	–	–
InternVideo2-6B (Wang et al., 2024b)	6B	–	67.7	–	–	88.8	–	–
VideoPrism (Zhao et al., 2024)	1B	–	68.5	71.3	–	87.6	–	–
Image Encoders Evaluated Using the Same Protocol								
DINOv2 (Darcet et al., 2024)	1.1B	81.1	50.7	82.5	93.4	83.6	90.7	86.1
PE$_{\text{core}}$G (Bolya et al., 2025)	1.9B	82.3	55.4	76.9	90.0	88.5	**95.3**	87.6*
SigLIP2 (Tschannen et al., 2025)	1.2B	81.1	49.9	75.3	91.0	87.3	95.1	**88.0**
Video Encoders Evaluated Using the Same Protocol								
V-JEPA ViT-H (Bardes et al., 2024)	600M	85.2	74.3	87.9	97.7	84.5	87.1	80.0
InternVideo2$_{s2}$-1B (Wang et al., 2024b)	1B	87.0	69.7	86.4	97.0	**89.4**	93.8	85.8
V-JEPA 2 ViT-L	300M	86.0	73.7	89.0	97.6	85.1	86.8	83.5
V-JEPA 2 ViT-H	600M	86.4	74.0	89.8	97.7	85.3	87.9	83.8
V-JEPA 2 ViT-g	1B	87.5	75.3	90.1	97.7	86.6	90.7	84.6
V-JEPA 2 ViT-g$_{384}$	1B	**88.2**	**77.3**	**90.2**	**97.8**	87.3	91.1	85.1

Evaluation protocol. We compare the performance of V-JEPA 2 on motion and appearance tasks with several other visual encoders: DINOv2 with registers (Darcet et al., 2024) is the current state-of-the-art model for self-supervised learning with images, while SigLIP2 (Tschannen et al., 2025) and the Perception Encoder PE$_{\text{core}}$G (Bolya et al., 2025) are two state-of-the-art models for image-text contrastive pretraining. We also consider two video encoders: the self-supervised V-JEPA (Bardes et al., 2024), and InternVideo2$_{s2}$-1B (Wang et al., 2024b) which relies primarily on vision-text contrastive pretraining.

We use the same evaluation protocol for every baseline and for V-JEPA 2, learning an attentive probe on top of the frozen encoder, similar to Bardes et al. (2024). We adapt image-based models to video following the procedure used in Oquab et al. (2023), concatenating the features of each input frame. For InternVideo2$_{s2}$-1B, we use its image positional embedding for the ImageNet task, and for video tasks we interpolate its positional embedding from 4 frames to 8, producing a token count similar to V-JEPA 2. Despite using a common evaluation protocol, the baseline encoders are trained on different data (e.g., DINOv2 on LVD-142M, PE$_{\text{core}}$G on MetaCLIP) and are thus not directly comparable. We can therefore only compare different approaches at a system level; i.e., with a consistent evaluation protocol despite differences in training protocol and data. We also include existing results from the literature using a similar frozen protocol, but with potentially different attentive head architecture. In particular, we share reported results of VideoMAEv2 (Wang et al., 2023), InternVideo-1B and 6B (Wang et al., 2024b), and VideoPrism (Zhang et al., 2024c) on the classification tasks we consider, when available. We provide complete evaluation and hyperparameters in Appendix C.1.

Results. Table 4 reports the classification performance of V-JEPA 2, the other encoders we evaluated, and other notable results reported in the literature. V-JEPA 2 ViT-g (at 256 resolution) significantly outperforms other vision encoders on motion understanding tasks. It achieves a top-1 accuracy of 75.3 on SSv2 compared to

Table 5 Prediction: Human Action Anticipation. Comparison with the state-of-the-art on the EK100 Action Anticipation benchmark. We report mean-class recall-at-5 for verb, noun and action on the validation set of EK100. V-JEPA 2 performance scales linearly with model size and outperforms previous state-of-the-art across all model sizes.

Method	Param.	*Action Anticipation*		
		Verb	Noun	**Action**
InAViT (Roy et al., 2024)	160M	51.9	52.0	25.8
Video-LLaMA (Zhang et al., 2023)	7B	52.9	52.0	26.0
PlausiVL (Mittal et al., 2024)	8B	55.6	54.2	27.6
Frozen Backbone				
V-JEPA 2 ViT-L	300M	57.8	53.8	32.7
V-JEPA 2 ViT-H	600M	59.2	54.6	36.5
V-JEPA 2 ViT-g	1B	61.2	55.7	38.0
V-JEPA 2 ViT-g$_{384}$	1B	**63.6**	**57.1**	**39.7**

69.7 for InternVideo and 55.4 for PE$_{Core}$G. V-JEPA 2 is also competitive on appearance tasks, reaching 84.6 on ImageNet (a +4.6 point improvement over V-JEPA). Overall, V-JEPA 2 obtains the best average performance across all six tasks, compared to other video and image encoders. The higher-resolution, longer-duration V-JEPA 2 ViT-g$_{384}$ shows further improvement across all tasks, reaching 88.2 average performance.

6 Prediction: Probe-based Action Anticipation

Action anticipation consists in predicting the future action given a contextual video clip leading up to some time before the action. Using the Epic-Kitchens-100 (EK100) benchmark (Damen et al., 2022), we demonstrate that V-JEPA 2 action anticipation performance increases consistently with model size. Furthermore, despite only using an attentive probe trained on top of V-JEPA 2 representations, we show that V-JEPA 2 significantly outperforms prior state-of-the-art approaches that were specifically designed for this task.

Task. The EK100 dataset is comprised of 100 hours of cooking activities recorded from an egocentric perspective across 45 kitchen environments. Each video in EK100 is annotated with action segments, which include a start timestamp, an end timestamp, and an action label. There are 3,568 unique action labels, each consisting of a verb and a noun category, with a total of 97 verb categories and 300 noun categories. The EK100 action anticipation task involves predicting noun, verb, and action (i.e., predicting verb and noun jointly) from a video clip, referred to as context, that occurs before the start timestamp of an action segment. The interval between the end of the context and the beginning of the action segment is the anticipation time, which is set to 1 second by default. Given that different future actions may be possible from a given context, mean-class recall-at-5 is used as the metric to measure performance (Damen et al., 2022).

Anticipation Probe. An attentive probe is trained on top of the frozen V-JEPA 2 encoder and predictor to anticipate future actions. Specifically, we sample a video clip that ends 1 second before an action starts. This video context is fed to the V-JEPA 2 encoder. The predictor takes the encoder representation, along with the mask tokens corresponding to the frame 1 second into the future, and predicts the representation of the future video frame. The outputs of the predictor and encoder are concatenated along the token dimension and fed to an attentive probe with a similar architecture to those used in Section 5, with the difference being that the anticipation probe's final cross-attention layer learns three query tokens (as opposed to one), and each query output is fed to a different linear classifier to predict the action category, the verb category, and the noun category respectively. A focal loss (Lin et al., 2017) is applied to each classifier independently and then summed before backpropagating through the shared attention blocks of the probe. We provide additional details and evaluation hyperparameters in Appendix D.1.

Baselines. We compare our model with three baselines that are trained specifically for action anticipation: InAViT (Roy et al., 2024) is a a supervised approach that leverages explicit hand-object interaction modeling,

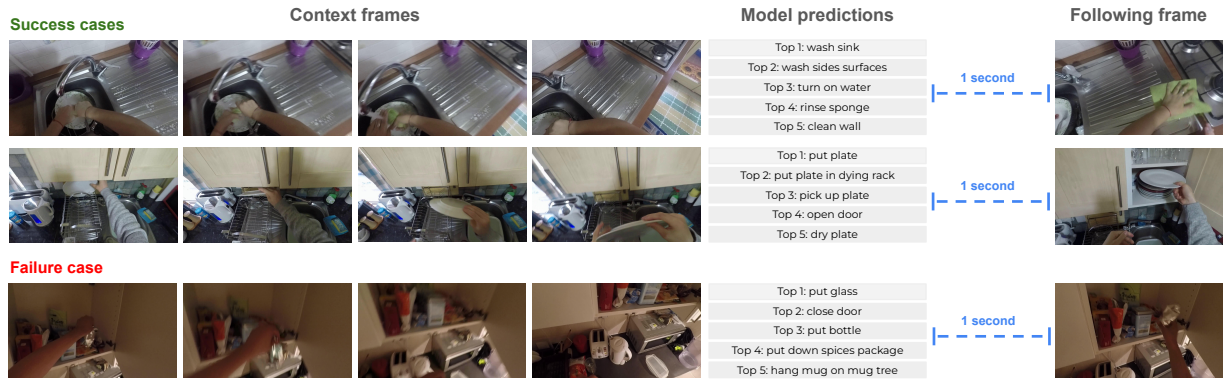


Figure 11 Visualization of EK100 prediction. (Left): four selected frames from the context frames. (Middle): model predictions, ordered by likelihood. (Right): following frame after the 1 second anticipation time. We show two examples where the model is successful and one example where the model fails.

and Video-LLaMA (Zhang et al., 2023) and PlausiVL (Mittal et al., 2024) are both approaches that leverage a large language model, with up to 7 billion parameters.

Results. Table 5 summarizes the results on the EK100 action anticipation benchmark. We compare V-JEPA 2 ViT-L, ViT-H and ViT-g encoders, increasing parameter count from 300 millions to 1 billion. All three leverage 32 frames with 8 frames per second at resolution 256×256 as video context. We also report results of ViT-g_{384} which uses a resolution of 384×384. V-JEPA 2 demonstrates a linear scaling behavior with respect to model size, in terms of action prediction recall-at-5. V-JEPA 2 ViT-L with 300 million parameters achieves 32.7 recall-at-5. Increasing the size of the model to 1 billion parameters leads to a +5.3 point improvement with an action recall-at-5 of 38.0. Furthermore, V-JEPA 2 benefits from using a context with higher resolution, and V-JEPA 2 ViT-g_{384} at resolution 384×384 improves recall-at-5 by an additional +1.7 points over the other models using 256×256 resolution.

V-JEPA 2 outperforms the previous state-of-the-art model PlausiVL by a significant margin, even with its 300 million parameters compared to the 8 billion parameters used in PlausiVL. In particular, V-JEPA 2 ViT-g_{384} demonstrates a +12.1 points improvement over PlausiVL on action recall-at-5, corresponding to a 44% relative improvement.

In Figure 11 we visualize V-JEPA 2 predictions on three samples from the EK100 validation set, two where the model is successful and one where the model fails. For both successful examples, V-JEPA 2 not only retrieves the correct action correctly with top 1 confidence, but also proposes coherent top 2 to 5 actions, based on the given context. For example, in the top row, the correct action is "wash sink", but "turn on water" or "clean wall" would both have been valid actions given the presence of a tap and a wall. The model also predicts "rinse sponge", which is the current action being performed, probably assuming that this action could still be going on after 1 second. For the failure case, V-JEPA 2 still proposes coherent actions such as "close door" and "put down spices package", but misses the exact nature of the object: "tea package".

Limitations. V-JEPA 2 and the EK100 benchmark have several limitations. First, V-JEPA 2 does not fully solve EK100, there are failure cases where the model either gets the verb, the noun, or both wrong. We study the distribution of these failures in Appendix D.2. Second, we focus here on predicting actions with a 1 second anticipation time. The accuracy of V-JEPA 2 degrades when predicting at longer time horizons, see Appendix D.2. Third, the EK100 benchmark is limited to kitchen environments, with a closed well-defined vocabulary, and we do not know how well V-JEPA 2 generalizes to other environments. This limits the utility and applicability of models trained on EK100. Lastly, actions in EK100 are chosen from a fixed set of categories, making it impossible to generalize to action categories not present in the training set.

7 Understanding : Video Question Answering

In this section, we explore V-JEPA 2's ability to perform open-language video question answering (VidQA). To enable language capabilities, we train a Multimodal Large Language Model (MLLM) using V-JEPA 2 as the visual encoder in the *non-tokenized early fusion* (Wadekar et al., 2024) setup popularized by the LLaVA family of models (Li et al., 2024b). In this family of MLLMs, a visual encoder is *aligned* with a large language model by projecting the output patch embeddings of the vision encoder to the input embedding space of the LLM. The MLLM is then trained either end-to-end, or with a frozen vision encoder. The majority of the encoders used in MLLMs for VidQA are typically image encoders, which are applied independently per-frame for video inputs (Qwen Team et al., 2025; Zhang et al., 2024b). Popular instances of such encoders are CLIP (Radford et al., 2021), SigLIP (Tschannen et al., 2025), and Perception Encoder (Bolya et al., 2025), which are chosen primarily due to their semantic alignment with language, obtained by pretraining with image-caption pairs. To the best of our knowledge, our work is the first to use a video encoder that is pretrained *without* any language supervision, to train an MLLM for VidQA.

MLLM performance on downstream tasks is also highly dependent on the alignment data. In these experiments we use a dataset of 88.5 million image- and video-text pairs, similar to what was used to train PerceptionLM (Cho et al., 2025). To demonstrate the effectiveness of the V-JEPA 2 encoder, first we compare V-JEPA 2 with other state-of-the-art vision encoders in a *controlled* data setup in Section 7.2, using a subset of 18 million samples. Then, in the same controlled setup, we show that scaling the vision encoder and input resolution size both consistently improve VidQA performance in Section 7.3. Finally, we scale the alignment data, using the full 88.5 million samples to test the limits of language alignment with V-JEPA 2 in Section 7.4. Our results demonstrate that in a controlled data setup, V-JEPA 2 obtains competitive performance on open-ended VidQA tasks compared to other vision encoders. Upon scaling the alignment data, V-JEPA 2 achieves state-of-the-art performance on several VidQA benchmarks.

7.1 Experiment Setup

Video Question Answering Tasks. We evaluate on PerceptionTest (Pătrăucean et al., 2023), which assesses model performance across different skills such as memory, abstraction, physics, and semantics. Additionally, we evaluate on the MVP dataset (Krojer et al., 2024) for physical world understanding, which utilizes a minimal-video pair evaluation framework to mitigate text and appearance biases. We also evaluate on TempCompass, TemporalBench and TOMATO (Liu et al., 2024c; Cai et al., 2024; Shangguan et al., 2024) to investigate temporal understanding, and memory capabilities of models. Finally, we report results on general understanding ability using MVBench (Li et al., 2024c), which has a bias towards single-frame appearance features (Krojer et al., 2024; Cores et al., 2024), and TVBench (Cores et al., 2024), which is proposed in the literature as an alternative for general and temporal understanding, mitigating those biases.

Visual Instruction Tuning. To evaluate the V-JEPA 2 representations on visual-question answering tasks, we align V-JEPA 2 with an LLM using the visual instruction tuning procedure from the LLaVA framework (Liu et al., 2024a). This process involves converting the visual encoder outputs (or visual tokens) into LLM inputs using a learnable *projector* module, which is typically an MLP. We train MLLMs through a progressive three-stage process following Liu et al. (2024b): Stage 1, where we train the projector solely on image captioning data; Stage 2, where we train the full model on large-scale image question answering, and Stage 3, where we further train the model on large-scale video captioning and question answering. Through this staged training approach, the LLM incrementally improves its understanding of visual tokens. The vision encoder can either be frozen or finetuned along with the rest of the MLLM. We explore both settings as freezing the vision encoder gives a cleaner signal about the quality of the visual features, while finetuning the vision encoder yields better overall performance. Further details of the visual instruction training are described in Appendix E.

7.2 Comparing with Image Encoders

To isolate the contribution of vision encoders to MLLM performance and compare with V-JEPA 2, we introduce a *controlled* setup: we train individual MLLMs with different state-of-the-art encoders using the

Table 6 Comparison between off-the-shelf image encoders and V-JEPA 2 in frozen encoder setting. All experiments use the same LLM backbone (Qwen2-7B-Instruct), data, and training setup with a **frozen** vision encoder. PerceptionTest accuracy is reported on the validation set post SFT.

Method	Params Enc / LLM	Avg.	PerceptionTest SFT / Acc	MVP Paired-Acc	TempCompass multi-choice	TemporalBench (MBA-short QA)	TVBench Acc	TOMATO Acc	MVBench Acc
Off-the-shelf image encoders									
DINOv2 ViT-g$_{518}$	1.1B/7B	45.7	67.1	22.4	62.3	26.8	47.6	32.0	61.8
SigLIP2 ViT-g$_{384}$	1.1B/7B	48.1	**72.4**	26.2	66.8	25.7	48.7	33.2	64.0
PE ViT-G/14$_{448}$	1.9B/7B	49.1	72.3	26.7	67.0	27.5	51.6	34.0	64.7
V-JEPA 2 ViT-g$_{512}$	1B/7B	**52.3**	72.0	**31.1**	**69.2**	**33.3**	**55.9**	**37.0**	**67.7**

Table 7 Scaling Vision Encoder Size and Resolution. We scale the vision encoder from 300 million to 1 billion parameters and input resolution from 256 pixels to 512 pixels. All experiments use the same LLM backbone (Qwen2-7B-Instruct), data, and end-to-end training (unfrozen vision encoder) setup. PerceptionTest accuracy is reported on the validation set post SFT. Increasing V-JEPA 2 encoder scale and resolution improve average performances on VidQA tasks.

Method	Params Enc / LLM	Avg.	PerceptionTest SFT / Acc	MVP Paired-Acc	TempCompass multi-choice	TemporalBench (MBA-short QA)	TVBench Acc	TOMATO Acc	MVBench Acc
End-to-end Evaluation									
V-JEPA 2 ViT-L$_{256}$	300M/7B	51.7	74.6	32.3	70.1	30.2	50.9	36.5	67.1
V-JEPA 2 ViT-H$_{256}$	600M/7B	52.0	74.7	30.6	70.9	29.8	54.6	35.1	68.0
V-JEPA 2 ViT-g$_{256}$	1B/7B	52.3	75.5	31.9	70.7	28.3	54.2	37.3	68.3
V-JEPA 2 ViT-g$_{384}$	1B/7B	54.0	76.5	33.0	**71.7**	**33.1**	56.5	**39.0**	68.5
V-JEPA 2 ViT-g$_{512}$	1B/7B	**54.4**	**77.7**	**33.7**	71.6	32.3	**57.5**	38.5	**69.5**

same LLM backbone and training setup. In this controlled setup, we use Qwen2-7B-Instruct (Yang et al., 2024a) and freeze the vision encoder. We use 18 million image and video-text aligned samples. We first compare V-JEPA 2, pretrained at resolution 512×512 with DINOv2 (Oquab et al., 2023), SigLIP-2 (Tschannen et al., 2025), and Perception Encoder (Bolya et al., 2025).

We observe that V-JEPA 2 exhibits competitive performance in the frozen setup, outperforming DINOv2, SigLIP, and Perception Encoder (PE) in all of the tested benchmarks (Table 6) except PerceptionTest where V-JEPA 2 slightly underperforms SigLIP and PE. The improvement is especially noticeable on MVP, TemporalBench, and TVBench — benchmarks that are primarily focused on temporal understanding. Additionally, since we only change the vision encoder, we provide evidence that a video encoder trained without language supervision can outperform encoders trained with language supervision, in contrast to conventional wisdom (Tong et al., 2024; Li et al., 2024b; Liu et al., 2024d; Yuan et al., 2025). The results also indicate that using a video encoder instead of an image encoder for VidQA improves spatiotemporal understanding, highlighting the need to develop better video encoders.

7.3 Scaling Vision Encoder Size and Input Resolution

Prior work (Fan et al., 2025) suggests that scaling the vision encoder and input resolution significantly improves VQA performance for self-supervised image encoders. Thus, we scale V-JEPA 2 from 300M to 1B parameters and the input resolution from 256 to 512 pixels, and show the results in Table 7. When increasing vision encoder capacity from 300M to 1B parameters for a fixed input resolution of 256 pixels, we observe improvements of 0.9 points on PerceptionTest, 3.3 points on TVBench, and 1.2 points on MVBench. Additionally, increasing the input resolution to 512 pixels yields further improvements across all downstream tasks, such as an improvement of 2.2 points on PerceptionTest, 4.0 points on TemporalBench, and 3.3 points on TVBench. These results suggest that further scaling the vision encoder and input resolution is a promising

Table 8 Comparison with state-of-the-art. We use the full 88.5M-sample alignment dataset and train using the same methodology as PLM 8B Cho et al. (2025), using a Llama 3.1 backbone. We observe significant improvements in downstream evaluations, obtaining state-of-the-art results in the 8B model class. PerceptionTest accuracy is reported on the test set with SFT for V-JEPA 2; all other results are zero-shot.

Method	Params Enc / LLM	Avg.	PerceptionTest Test Acc	MVP Paired-Acc	TempCompass multi-choice	TemporalBench (MBA-short QA)	TOMATO Acc	TVBench Acc	MVBench Acc
≤ 8B Video Language Models Results Reported in the Literature									
InternVL-2.5 (Chen et al., 2024)	300M/7B	52.1	68.9	39.9	68.3	24.3	29.4	61.6	72.6
Qwen2VL (Wang et al., 2024a)	675M/7B	47.0	66.9	29.2	67.9	20.4	31.5	46.0	67.0
Qwen2.5VL (Qwen Team et al., 2025)	1B/7B	49.7	70.5	36.7	71.7	24.5	24.6	50.5	69.6
PLM 8B (Cho et al., 2025)	1B/8B	56.7	82.7	39.7	72.7	28.3	33.2	**63.5**	**77.1**
V-JEPA 2 ViT-g$_{384}$ LLama 3.1 8B	1B/8B	**59.5**	**84.0**	**44.5**	**76.9**	**36.7**	**40.3**	60.6	73.5

direction for improving VidQA performance.

7.4 Improving the State-of-the-art by Scaling Data

After developing a better understanding of the capabilities of V-JEPA 2 for training an MLLM in the controlled setup, we study the effect of increasing alignment dataset size to improve the state-of-the-art of VidQA. Step changes on downstream task performance are often achieved by increasing the scale of the training data, as observed by Cho et al. (2025). To that end, we increase the scale of MLLM training data from 18 million to the full 88.5 million (4.7×). While increasing the model resolution helps in downstream performance, it comes with the challenge of accommodating a large number of visual tokens in the LLM input. We therefore choose V-JEPA 2 ViT-g$_{384}$, leading to 288 visual tokens per frame. We follow the same recipe as Cho et al. (2025) to train V-JEPA 2 ViT-g$_{384}$, using Llama 3.1 as the backbone. To simplify the training process, we use an MLP projector without pooling. Details on the scaling training setup are described in Appendix E.

Scaling the data uniformly improves the downstream benchmark performance, resulting in state-of-the-art results (Table 8) on multiple benchmarks — PerceptionTest, MVP, TempCompass, TemporalBench and TOMATO. Compared to the current state-of-the-art PerceptionLM 8B (Cho et al., 2025), we observe an increase of 1.3 points on accuracy for PerceptionTest test set, 4.8 points on paired accuracy for MVP, 4.2 points on accuracy for TempCompass, 8.4 points on Multi-binary accuracy for short-QA segment for TemporalBench and 7.1 points on accuracy for TOMATO. V-JEPA 2 does not outperform PerceptionLM on TVBench and MVBench, however it still significantly outperforms other related baselines (InternVL 2.5, Qwen2VL and Qwen2.5VL). These results underscore the need to scale training data for vision-language alignment and provide evidence that an encoder pretrained without language supervision, such as V-JEPA 2, can achieve state-of-the-art results with sufficient scale.

8 Related Work

World models and planning. As early as the work of Sutton and Barto (1981) and Chatila and Laumond (1985), AI researchers have sought to build agents that use internal models of the world — modeling both dynamics of the world, as well as mapping the static environment — to enable efficient planning and control. Previous work has investigated world models in simulated tasks (Fragkiadaki et al., 2015; Ha and Schmidhuber, 2018; Hafner et al., 2019b,a; Hansen et al., 2022, 2023; Hafner et al., 2023; Schrittwieser et al., 2020; Samsami et al., 2024), as well as real-world locomotion and manipulation tasks (Lee et al., 2020; Nagabandi et al., 2020; Finn et al., 2016; Ebert et al., 2017, 2018; Yen-Chen et al., 2020). World model approaches either learn predictive models directly in pixel-space (Finn et al., 2016; Ebert et al., 2017, 2018; Yen-Chen et al., 2020), in a learned representation space (Watter et al., 2015; Agrawal et al., 2016; Ha and Schmidhuber, 2018; Hafner et al., 2019b; Nair et al., 2022; Wu et al., 2023b; Tomar et al., 2024; Hu et al., 2024; Lancaster et al., 2024), or utilizing more structured representation spaces such as keypoint representations (Manuelli et al., 2020; Das

et al., 2020). Previous approaches that have demonstrated real world performance on robotics tasks have trained task-specific world models, and they rely on interaction data from the environment in which the robot is deployed. Evaluation is focused on demonstrating performance of world modeling approaches within the explored task space, instead of generalization to new environments or unseen objects. In this work we train a task-agnostic world model, and demonstrate generalization to new environments and objects.

Some recent works leverage both internet-scale video and interaction data towards training general purpose (task-agnostic) action-conditioned video generation models for autonomous robots (Bruce et al., 2024; Agarwal et al., 2025; Russell et al., 2025). However, thus far these approaches only demonstrate the ability to generate visually valid-looking plans given actions of the robot, but they have not demonstrated the ability to use those models to actually control the robot.

Other works have explored the integration of generative modeling into policy learning (Du et al., 2024; Wu et al., 2023a; Zhao et al., 2025; Zhu et al., 2025; Du et al., 2023; Zheng et al., 2025; Rajasegaran et al., 2025). Differently from this line of work, our goal is to leverage a world model through model-predictive control instead of policy learning to avoid the imitation learning phase that requires expert trajectories. Both approaches are orthogonal and could be combined in future works. Closest to our work, Zhou et al. (2024); Sobal et al. (2025) show that you can learn a world model stage-wise or end-to-end and use it to solve planning tasks zero-shot. While those previous works focus on small-scale planning evaluation, we show that similar principles can be scaled and used to solve real-world robotic tasks.

Vision-Language-Action models for Robotic Control. Recent imitation learning approaches in real-world robotic control have made significant progress towards learning policies that show increasingly good generalization capabilities. This is achieved by leveraging video-languange models that have been pre-trained on internet scale video and text data, which are then fine-tuned (or adapted) to also predict actions by using behavior cloning from expert demonstrations (Driess et al., 2023; Brohan et al., 2023; Black et al., 2024; Kim et al., 2024; Bjorck et al., 2025; Black et al., 2025). Although these approaches show promising generalization results, it is unclear whether they will be able to learn to predict behaviors that were not demonstrated in the training data since they lack an explicit predictive model of the world and do leverage inference-time computation for planning. They require high-quality large scale teleoperation data, and can only utilize successful trajectories. In contrast, we focus on leveraging any interaction data whether it comes from a successful or failed interaction with the environment.

Vision Foundation Models. Video foundation models in computer vision have shown that large-scale observation datasets comprised of images and/or videos can be leveraged to learn generalist vision encoders that perform well along a wide range of downstream tasks using self-supervised learning approaches from images (Grill et al., 2020; Assran et al., 2023; Oquab et al., 2023; Fan et al., 2025), videos (Bardes et al., 2024; Carreira et al., 2024; Wang et al., 2023; Rajasegaran et al., 2025), with weak language supervision (Wang et al., 2024b; Bolya et al., 2025), or a combination thereof (Tschannen et al., 2025; Fini et al., 2024). Previous works, however, tend to focus on understanding evaluation using probe-based evaluation or visual question answering tasks after aligning with a large-language model. While such tasks have served to drive progress, it remains an important goal of a visual system to enable an agent to interact with the physical world (Gibson, 1979). Beyond results on visual understanding tasks, we investigate how large-scale self-supervised learning from video can enable solving planning tasks in new environments in a zero-shot manner.

9 Conclusion

This study demonstrates how joint-embedding predictive architectures, learning in a self-supervised manner from web-scale data and a small amount of robot interaction data, can yield a world model capable of understanding, predicting, and planning in the physical world. V-JEPA 2 achieves state-of-art performances on action classification requiring motion understanding and human action anticipation. V-JEPA 2 also outperforms previous vision encoders on video questions-answering tasks when aligned with a large-language model. Additionally, post-training an action-conditioned world model, V-JEPA 2-AC, using V-JEPA 2's representation, enables successful zero-shot prehensile manipulation tasks, such as Pick-and-Place, with real-world robots. These findings indicate V-JEPA 2 is a step towards developing advanced AI systems that can effectively perceive and act in their environment.

Future work. There are several important avenues for future work to address limitations of V-JEPA 2. First, in this work we have focused on tasks requiring predictions up to roughly 16 seconds into the future. This enables planning for simpler manipulation tasks, like grasp and reach-with-object, from a single goal image. However, to extend this to longer-horizon tasks such as pick-and-place or even more complex tasks, without requiring sub-goals will require further innovations in modeling. Developing approaches for hierarchical models capable of making predictions across multiple spatial and temporal scales, at different levels of abstraction, is a promising direction.

Second, as mentioned in Section 4, V-JEPA 2-AC currently relies upon tasks specified as image goals. Although this may be natural for some tasks, there are other situations where language-based goal specification may be preferable. Extending the V-JEPA 2-AC to accept language-based goals, e.g., by having a model that can embed language-based goals into the V-JEPA 2-AC representation space, is another important direction for future work. The results described in Section 7, aligning V-JEPA 2 with a language model, may serve as a starting point.

Finally, in this work we scaled V-JEPA 2 models up to a modest 1B parameters. The results in Section 2 demonstrated consistent performance improvements while scaling to this level. Previous work has investigated scaling vision encoders to as large as 20B parameters (Zhai et al., 2022; Carreira et al., 2024). Additional work is needed in this direction to develop scalable pre-training recipes that lead to sustained performance improvements with scale.

Acknowledgements

We thank Rob Fergus, Joelle Pineau, Stephane Kasriel, Naila Murray, Mrinal Kalakrishnan, Jitendra Malik, Randall Balestriero, Julen Urain, Gabriel Synnaeve, Michel Meyer, Pascale Fung, Justine Kao, Florian Bordes, Aaron Foss, Nikhil Gupta, Cody Ohlsen, Kalyan Saladi, Ananya Saxena, Mack Ward, Parth Malani, Shubho Sengupta, Leo Huang, Kamila Benzina, Rachel Kim, Ana Paula Kirschner Mofarrej, Alyssa Newcomb, Nisha Deo, Yael Yungster, Kenny Lehmann, Karla Martucci, and the PerceptionLM team, including Christoph Feichtenhofer, Andrea Madotto, Tushar Nagarajan, and Piotr Dollar for their feedback and support of this project.

References

Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, Daniel Dworakowski, Jiaojiao Fan, Michele Fenzi, Francesco Ferroni, Sanja Fidler, Dieter Fox, Songwei Ge, Yunhao Ge, Jinwei Gu, Siddharth Gururani, Ethan He, Jiahui Huang, Jacob Huffman, Pooya Jannaty, Jingyi Jin, Seung Wook Kim, Gergely Klár, Grace Lam, Shiyi Lan, Laura Leal-Taixe, Anqi Li, Zhaoshuo Li, Chen-Hsuan Lin, Tsung-Yi Lin, Huan Ling, Ming-Yu Liu, Xian Liu, Alice Luo, Qianli Ma, Hanzi Mao, Kaichun Mo, Arsalan Mousavian, Seungjun Nah, Sriharsha Niverty, David Page, Despoina Paschalidou, Zeeshan Patel, Lindsey Pavao, Morteza Ramezanali, Fitsum Reda, Xiaowei Ren, Vasanth Rao Naik Sabavat, Ed Schmerling, Stella Shi, Bartosz Stefaniak, Shitao Tang, Lyne Tchapmi, Przemek Tredak, Wei-Cheng Tseng, Jibin Varghese, Hao Wang, Haoxiang Wang, Heng Wang, Ting-Chun Wang, Fangyin Wei, Xinyue Wei, Jay Zhangjie Wu, Jiashu Xu, Wei Yang, Lin Yen-Chen, Xiaohui Zeng, Yu Zeng, Jing Zhang, Qinsheng Zhang, Yuxuan Zhang, Qingqing Zhao, and Artur Zolkowski. Cosmos world foundation model platform for physical AI. *arXiv preprint arXiv:2501.03575*, 2025.

Pulkit Agrawal, Ashvin V Nair, Pieter Abbeel, Jitendra Malik, and Sergey Levine. Learning to poke by poking: Experiential learning of intuitive physics. *Advances in Neural Information Processing Systems*, 29, 2016.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. *Advances in Neural Information Processing Systems*, 30, 2017.

Mahmoud Assran, Randall Balestriero, Quentin Duval, Florian Bordes, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, and Nicolas Ballas. The hidden uniform cluster prior in self-supervised learning. *arXiv preprint arXiv:2210.07277*, 2022.

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15619–15629, 2023.

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mahmoud Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *arXiv preprint arXiv:2404.08471*, 2024.

Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi "Jim" Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π0: A vision-language-action flow model for general robot control, 2024. *arXiv preprint arXiv:2410.24164*, 2024.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Daniel Bolya, Po-Yao Huang, Peize Sun, Jang Hyun Cho, Andrea Madotto, Chen Wei, Tengyu Ma, Jiale Zhi, Jathushan Rajasegaran, Hanoona Rasheed, Junke Wang, Marco Monteiro, Hu Xu, Shiyu Dong, Nikhila Ravi, Daniel Li, Piotr Dollár, and Christoph Feichtenhofer. Perception encoder: The best visual embeddings are not at the output of the network. *arXiv preprint arXiv:2504.13181*, 2025.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alexander Herzog, Jasmine Hsu, Brian Ichter, Alex Irpan, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Lisa Lee, Tsang-Wei Edward Lee, Sergey Levine, Yao Lu, Henryk Michalewski, Igor Mordatch, Karl Pertsch, Kanishka Rao, Krista Reymann, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Pierre Sermanet, Jaspiar Singh, Anikait Singh, Radu Soricut, Huong Tran, Vincent

Vanhoucke, Quan Vuong, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Jialin Wu, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

Jake Bruce, Michael Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, Yusuf Aytar, Sarah Bechtle, Feryal Behbahani, Stephanie Chan, Nicolas Heess, Lucy Gonzalez, Simon Osindero, Sherjil Ozair, Scott Reed, Jingwei Zhang, Konrad Zolna, Jeff Clune, Nando de Freitas, Satinder Singh, and Tim Rocktäschel. Genie: Generative interactive environments. In *International Conference on Machine Learning*, 2024.

Mu Cai, Reuben Tan, Jianrui Zhang, Bocheng Zou, Kai Zhang, Feng Yao, Fangrui Zhu, Jing Gu, Yiwu Zhong, Yuzhang Shang, Yao Dou, Jaden Park, Jianfeng Gao, Yong Jae Lee, and Jianwei Yang. Temporalbench: Benchmarking fine-grained temporal understanding for multimodal video models. *arXiv preprint arXiv:2410.10818*, 2024.

Joao Carreira, Eric Noland, Andras Banki-Horvath, Chloe Hillier, and Andrew Zisserman. A short note about kinetics-600. *arXiv preprint arXiv:1808.01340*, 2018.

Joao Carreira, Eric Noland, Chloe Hillier, and Andrew Zisserman. A short note on the kinetics-700 human action dataset. *arXiv preprint arXiv:1907.06987*, 2019.

João Carreira, Dilara Gokay, Michael King, Chuhan Zhang, Ignacio Rocco, Aravindh Mahendran, Thomas Albert Keck, Joseph Heyward, Skanda Koppula, Etienne Pot, Goker Erdogan, Yana Hasson, Yi Yang, Klaus Greff, Guillaume Le Moing, Sjoerd van Steenkiste, Daniel Zoran, Drew A. Hudson, Pedro Vélez, Luisa Polanía, Luke Friedman, Chris Duvarney, Ross Goroshin, Kelsey Allen, Jacob Walker, Rishabh Kabra, Eric Aboussouan, Jennifer Sun, Thomas Kipf, Carl Doersch, Viorica Pătrăucean, Dima Damen, Pauline Luc, Mehdi S. M. Sajjadi, and Andrew Zisserman. Scaling 4d representations. *arXiv preprint arXiv:2412.15212*, 2024.

Raja Chatila and Jean-Paul Laumond. Position referencing and consistent world modeling for mobile robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 2, pages 138–145, 1985.

Yevgen Chebotar, Karol Hausman, Yao Lu, Ted Xiao, Dmitry Kalashnikov, Jake Varley, Alex Irpan, Benjamin Eysenbach, Ryan Julian, Chelsea Finn, and Sergey Levine. Actionable models: Unsupervised offline reinforcement learning of robotic skills. *arXiv preprint arXiv:2104.07749*, 2021.

Zhe Chen, Weiyun Wang, Yue Cao, Yangzhou Liu, Zhangwei Gao, Erfei Cui, Jinguo Zhu, Shenglong Ye, Hao Tian, Zhaoyang Liu, Lixin Gu, Xuehui Wang, Qingyun Li, Yimin Ren, Zixuan Chen, Jiapeng Luo, Jiahao Wang, Tan Jiang, Bo Wang, Conghui He, Botian Shi, Xingcheng Zhang, Han Lv, Yi Wang, Wenqi Shao, Pei Chu, Zhongying Tu, Tong He, Zhiyong Wu, Huipeng Deng, Jiaye Ge, Kai Chen, Kaipeng Zhang, Limin Wang, Min Dou, Lewei Lu, Xizhou Zhu, Tong Lu, Dahua Lin, Yu Qiao, Jifeng Dai, and Wenhai Wang. Expanding performance boundaries of open-source multimodal models with model, data, and test-time scaling. *arXiv preprint arXiv:2412.05271*, 2024.

Jang Hyun Cho, Andrea Madotto, Effrosyni Mavroudi, Triantafyllos Afouras, Tushar Nagarajan, Muhammad Maaz, Yale Song, Tengyu Ma, Shuming Hu, Suyog Jain, Miguel Martin, Huiyu Wang, Hanoona Rasheed, Peize Sun, Po-Yao Huang, Daniel Bolya, Nikhila Ravi, Shashank Jain, Tammy Stark, Shane Moon, Babak Damavandi, Vivian Lee, Andrew Westbury, Salman Khan, Philipp Krähenbühl, Piotr Dollár, Lorenzo Torresani, Kristen Grauman, and Christoph Feichtenhofer. Perceptionlm: Open-access data and models for detailed visual understanding. *arXiv preprint arXiv:2504.13180*, 2025.

Andy Clark. Whatever next? predictive brains, situated agents, and the future of cognitive science. *Behavioral and brain sciences*, 36(3):181–204, 2013.

Daniel Cores, Michael Dorkenwald, Manuel Mucientes, Cees GM Snoek, and Yuki M Asano. Tvbench: Redesigning video-language evaluation. *arXiv preprint arXiv:2410.07752*, 2024.

Kenneth James Williams Craik. *The Nature of Explanation*, volume 445. CUP Archive, 1967.

Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Antonino Furnari, Jian Ma, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. Rescaling egocentric vision: Collection, pipeline and challenges for epic-kitchens-100. *International Journal of Computer Vision (IJCV)*, 130:33–55, 2022. https://doi.org/10.1007/s11263-021-01531-2.

Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. In *The Twelfth International Conference on Learning Representations*, 2024.

Neha Das, Sarah Bechtle, Todor Davchev, Dinesh Jayaraman, Akshara Rai, and Franziska Meier. Model-based inverse reinforcement learning from visual demonstration. In *Conference on Robot Learning (CoRL)*, 2020.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. PaLM-E: An embodied multimodal language model. *arXiv preprint arXiv:2023.03378*, 2023.

Yilun Du, Sherry Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Josh Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. *Advances in Neural Information Processing Systems*, 36:9156–9172, 2023.

Yilun Du, Sherry Yang, Pete Florence, Fei Xia, Ayzaan Wahid, Brian Ichter, Pierre Sermanet, Tianhe Yu, Pieter Abbeel, Joshua B. Tenenbaum, Leslie Pack Kaelbling, Andy Zeng, and Jonathan Tompson. Video language planning. *ICLR*, 2024.

Frederik Ebert, Chelsea Finn, Alex X Lee, and Sergey Levine. Self-supervised visual planning with temporal skip connections. *CoRL*, 12(16):23, 2017.

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

David Fan, Shengbang Tong, Jiachen Zhu, Koustuv Sinha, Zhuang Liu, Xinlei Chen, Michael Rabbat, Nicolas Ballas, Yann LeCun, Amir Bar, and Saining Xie. Scaling language-free visual representation learning. *arXiv preprint arXiv:2504.01017*, 2025.

Enrico Fini, Mustafa Shukor, Xiujun Li, Philipp Dufter, Michal Klein, David Haldimann, Sai Aitharaju, Victor Guilherme Turrisi da Costa, Louis Béthune, Zhe Gan, Alexander T Toshev, Marcin Eichner, Moin Nabi, Yinfei Yang, Joshua M. Susskind, and Alaaeldin El-Nouby. Multimodal autoregressive pre-training of large vision encoders. *arXiv preprint arXiv:2411.14402*, 2024.

Chelsea Finn and Sergey Levine. Deep visual foresight for planning robot motion. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 2786–2793. IEEE, 2017.

Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. *Advances in Neural Information Processing Systems*, 29, 2016.

Katerina Fragkiadaki, Pulkit Agrawal, Sergey Levine, and Jitendra Malik. Learning visual predictive models of physics for playing billiards. *arXiv preprint arXiv:1511.07404*, 2015.

Karl Friston. The free-energy principle: a unified brain theory? *Nature Reviews Neuroscience*, 11(2):127–138, 2010.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, 07 2024. https://zenodo.org/records/12608602.

Dibya Ghosh, Abhishek Gupta, Ashwin Reddy, Justin Fu, Coline Devin, Benjamin Eysenbach, and Sergey Levine. Learning to reach goals via iterated supervised learning. *arXiv preprint arXiv:1912.06088*, 2019.

James J Gibson. *The ecological approach to visual perception: classic edition.* Psychology press, 1979.

Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzyńska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Fruend, Peter 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 *Proceedings of the IEEE international conference on computer vision*, pages 5842–5850, 2017.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu,

Rémi Munos, and Michal Valko. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in Neural Information Processing Systems*, 33:21271–21284, 2020.

Agrim Gupta, Stephen Tian, Yunzhi Zhang, Jiajun Wu, Roberto Martín-Martín, and Li Fei-Fei. Maskvit: Masked visual pre-training for video prediction. *arXiv preprint arXiv:2206.11894*, 2022.

David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019a.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pages 2555–2565. PMLR, 2019b.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

Alex Hägele, Elie Bakouch, Atli Kosson, Loubna Ben Allal, Leandro Von Werra, and Martin Jaggi. Scaling laws and compute-optimal training beyond fixed training durations. *Advances in Neural Information Processing Systems*, 37: 76232–76264, 2024.

Nicklas Hansen, Xiaolong Wang, and Hao Su. Temporal difference learning for model predictive control. *arXiv preprint arXiv:2203.04955*, 2022.

Nicklas Hansen, Hao Su, and Xiaolong Wang. Td-mpc2: Scalable, robust world models for continuous control. *arXiv preprint arXiv:2310.16828*, 2023.

John Hill. Real time control of a robot with a mobile camera. In *Proc. 9th Int. Symp. on Industrial Robots*, pages 233–245, 1979.

Anthony Hu, Lloyd Russell, Hudson Yeo, Zak Murez, George Fedoseev, Alex Kendall, Jamie Shotton, and Gianluca Corrado. Gaia-1: A generative world model for autonomous driving. *arXiv preprint arXiv:2309.17080*, 2023.

Edward S Hu, Kwangjun Ahn, Qinghua Liu, Haoran Xu, Manan Tomar, Ada Langford, Dinesh Jayaraman, Alex Lamb, and John Langford. Learning to achieve goals with belief state transformers. *arXiv preprint arXiv:2410.23506*, 2024.

Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, Olivier Hénaff, Matthew M. Botvinick, Andrew Zisserman, Oriol Vinyals, and João Carreira. Perceiver io: A general architecture for structured inputs & outputs. *arXiv preprint arXiv:2107.14795*, 2021.

Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, pages 991–1002. PMLR, 2022.

Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, Peter David Fagan, Joey Hejna, Masha Itkina, Marion Lepert, Yecheng Jason Ma, Patrick Tree Miller, Jimmy Wu, Suneel Belkhale, Shivin Dass, Huy Ha, Arhan Jain, Abraham Lee, Youngwoon Lee, Marius Memmel, Sungjae Park, Ilija Radosavovic, Kaiyuan Wang, Albert Zhan, Kevin Black, Cheng Chi, Kyle Beltran Hatch, Shan Lin, Jingpei Lu, Jean Mercat, Abdul Rehman, Pannag R Sanketi, Archit Sharma, Cody Simpson, Quan Vuong, Homer Rich Walke, Blake Wulfe, Ted Xiao, Jonathan Heewon Yang, Arefeh Yavary, Tony Z. Zhao, Christopher Agia, Rohan Baijal, Mateo Guaman Castro, Daphne Chen, Qiuyu Chen, Trinity Chung, Jaimyn Drake, Ethan Paul Foster, Jensen Gao, Vitor Guizilini, David Antonio Herrera, Minho Heo, Kyle Hsu, Jiaheng Hu, Muhammad Zubair Irshad, Donovon Jackson, Charlotte Le, Yunshuang Li, Kevin Lin, Roy Lin, Zehan Ma, Abhiram Maddukuri, Suvir Mirchandani, Daniel Morton, Tony Nguyen, Abigail O'Neill, Rosario Scalise, Derick Seale, Victor Son, Stephen Tian, Emi Tran, Andrew E. Wang, Yilin Wu, Annie Xie, Jingyun Yang, Patrick Yin, Yunchu Zhang, Osbert Bastani, Glen Berseth, Jeannette Bohg, Ken Goldberg, Abhinav Gupta, Abhishek Gupta, Dinesh Jayaraman, Joseph J Lim, Jitendra Malik, Roberto Martín-Martín, Subramanian Ramamoorthy, Dorsa Sadigh, Shuran Song, Jiajun Wu, Michael C. Yip, Yuke Zhu, Thomas Kollar, Sergey Levine, and Chelsea Finn. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Benno Krojer, Mojtaba Komeili, Candace Ross, Quentin Garrido, Koustuv Sinha, Nicolas Ballas, and Mido Assran. A shortcut-aware video-qa benchmark for physical understanding via minimal video pairs. *preprint*, 2024.

Patrick Lancaster, Nicklas Hansen, Aravind Rajeswaran, and Vikash Kumar. Modem-v2: Visuo-motor world models for real-world robot manipulation. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 7530–7537, 2024.

Yann LeCun. A path towards autonomous machine intelligence version 0.9.2, 2022-06-27. *Open Review*, 62(1):1–62, 2022.

Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning quadrupedal locomotion over challenging terrain. *Science Robotics*, 5(47):eabc5986, 2020.

Bo Li, Peiyuan Zhang, Kaichen Zhang, Fanyi Pu, Xinrun Du, Yuhao Dong, Haotian Liu, Yuanhan Zhang, Ge Zhang, Chunyuan Li, and Ziwei Liu. Lmms-eval: Accelerating the development of large multimoal models, March 2024a. https://github.com/EvolvingLMMs-Lab/lmms-eval.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*, 2024b.

Kunchang Li, Yali Wang, Yinan He, Yizhuo Li, Yi Wang, Yi Liu, Zun Wang, Jilan Xu, Guo Chen, Ping Luo, Limin Wang, and Yu Qiao. Mvbench: A comprehensive multi-modal video understanding benchmark. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22195–22206, 2024c.

Yingwei Li, Yi Li, and Nuno Vasconcelos. Resound: Towards action recognition without representation bias. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 513–528, 2018.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017.

Fangchen Liu, Hao Liu, Aditya Grover, and Pieter Abbeel. Masked autoencoding for scalable and generalizable decision making. *Advances in Neural Information Processing Systems*, 35:12608–12618, 2022.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26296–26306, 2024a.

Haotian Liu, Chunyuan Li, Yuheng Li, Bo Li, Yuanhan Zhang, Sheng Shen, and Yong Jae Lee. Llava-next: Improved reasoning, ocr, and world knowledge, January 2024b. https://llava-vl.github.io/blog/2024-01-30-llava-next/.

Yuanxin Liu, Shicheng Li, Yi Liu, Yuxiang Wang, Shuhuai Ren, Lei Li, Sishuo Chen, Xu Sun, and Lu Hou. TempCompass: Do video LLMs really understand videos? *arXiv preprint arXiv:2403.00476*, 2024c.

Zhijian Liu, Ligeng Zhu, Baifeng Shi, Zhuoyang Zhang, Yuming Lou, Shang Yang, Haocheng Xi, Shiyi Cao, Yuxian Gu, Dacheng Li, Xiuyu Li, Yunhao Fang, Yukang Chen, Cheng-Yu Hsieh, De-An Huang, An-Chieh Cheng, Vishwesh Nath, Jinyi Hu, Sifei Liu, Ranjay Krishna, Daguang Xu, Xiaolong Wang, Pavlo Molchanov, Jan Kautz, Hongxu Yin, Song Han, and Yao Lu. NVILA: Efficient frontier visual language models. *arXiv preprint arXiv:2412.04468*, 2024d.

Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

Corey Lynch, Mohi Khansari, Ted Xiao, Vikash Kumar, Jonathan Tompson, Sergey Levine, and Pierre Sermanet. Learning latent plans from play. In *Conference on Robot Learning*, pages 1113–1132. PMLR, 2020.

Lucas Manuelli, Yunzhu Li, Pete Florence, and Russ Tedrake. Keypoints into the future: Self-supervised correspondence in model-based reinforcement learning. *arXiv preprint arXiv:2009.05085*, 2020.

Joanna Materzynska, Guillaume Berger, Ingo Bax, and Roland Memisevic. The jester dataset: A large-scale video dataset of human gestures. In *Proceedings of the IEEE/CVF international conference on computer vision workshops*, pages 0–0, 2019.

Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. HowTo100m: Learning a text-video embedding by watching hundred million narrated video clips. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2630–2640, 2019.

Himangi Mittal, Nakul Agarwal, Shao-Yuan Lo, and Kwonjoon Lee. Can't make an omelette without breaking some eggs: Plausible action anticipation using large video-language models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18580–18590, 2024.

Anusha Nagabandi, Kurt Konolige, Sergey Levine, and Vikash Kumar. Deep dynamics models for learning dexterous manipulation. In *Conference on robot learning*, pages 1101–1112. PMLR, 2020.

Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. In *Conference on Robot Learning (CoRL)*, 2022.

Nora Nortmann, Sascha Rekauzke, Selim Onat, Peter König, and Dirk Jancke. Primary visual cortex represents the difference between past and present. *Cerebral Cortex*, 25(6):1427–1440, 2015.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Viorica Pătrăucean, Lucas Smaira, Ankush Gupta, Adrià Recasens Continente, Larisa Markeeva, Dylan Banarse, Skanda Koppula, Joseph Heyward, Mateusz Malinowski, Yi Yang, Carl Doersch, Tatiana Matejovicova, Yury Sulsky, Antoine Miech, Alex Frechette, Hanna Klimczak, Raphael Koster, Junlin Zhang, Stephanie Winkler, Yusuf Aytar, Simon Osindero, Dima Damen, Andrew Zisserman, and João Carreira. Perception test: A diagnostic benchmark for multimodal video models. *Advances in Neural Information Processing Systems*, 36:42748–42761, 2023.

Qwen Team, Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, Junyang Lin, An Yang, Binyuan Hui, Bowen Yu, Chen Cheng, Dayiheng Liu, Fan Hong, Fei Huang, Jiawei Liu, Jin Xu, Jianhong Tu, Jianyuan Zeng, Jie Zhang, Jinkai Wang, Jianwei Zhang, Jingren Zhou, Kexin Yang, Mei Li, Ming Yan, Na Ni, Rui Men, Songtao Jiang, Xiaodong Deng, Xiaoming Huang, Ximing Zhou, Xingzhang Ren, Yang Fan, Yichang Zhang, Yikai Zhu, Yuqiong Liu, and Zhifang Guo. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

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*, pages 8748–8763, 2021.

Jathushan Rajasegaran, Ilija Radosavovic, Rahul Ravishankar, Yossi Gandelsman, Christoph Feichtenhofer, and Jitendra Malik. An empirical study of autoregressive pre-training from videos. *arXiv preprint arXiv:2501.05453*, 2025.

Rajesh PN Rao and Dana H Ballard. Predictive coding in the visual cortex: a functional interpretation of some extra-classical receptive-field effects. *Nature Neuroscience*, 2(1):79–87, 1999.

Debaditya Roy, Ramanathan Rajendiran, and Basura Fernando. Interaction region visual transformer for egocentric action anticipation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 6740–6750, 2024.

Reuven Y. Rubinstein. Optimization of computer simulation models with rare events. *European Journal of Operations Research*, 99:89–112, 1997.

Lloyd Russell, Anthony Hu, Lorenzo Bertoni, George Fedoseev, Jamie Shotton, Elahe Arani, and Gianluca Corrado. Gaia-2: A controllable multi-view generative world model for autonomous driving. *arXiv preprint arXiv:2503.20523*, 2025.

Mohammad Reza Samsami, Artem Zholus, Janarthanan Rajendran, and Sarath Chandar. Mastering memory tasks with world models. In *International Conference on Learning Representations*, 2024.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

Ziyao Shangguan, Chuhan Li, Yuxuan Ding, Yanan Zheng, Yilun Zhao, Tesca Fitzgerald, and Arman Cohan. Tomato: Assessing visual temporal reasoning capabilities in multimodal foundation models. *arXiv preprint arXiv:2410.23266*, 2024.

Vlad Sobal, Wancong Zhang, Kynghyun Cho, Randall Balestriero, Tim GJ Rudner, and Yann LeCun. Learning from reward-free offline data: A case for planning with latent dynamics models. *arXiv preprint arXiv:2502.14819*, 2025.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

Richard S Sutton and Andrew G. Barto. An adaptive network that constructs and uses and internal model of its world. *Cognition and Brain Theory*, 4(3):217–246, 1981.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT Press, Cambridge, USA, 1998.

Yansong Tang, Dajun Ding, Yongming Rao, Yu Zheng, Danyang Zhang, Lili Zhao, Jiwen Lu, and Jie Zhou. CoIN: A large-scale dataset for comprehensive instructional video analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1207–1216, 2019.

Manan Tomar, Philippe Hansen-Estruch, Philip Bachman, Alex Lamb, John Langford, Matthew E Taylor, and Sergey Levine. Video occupancy models. *arXiv preprint arXiv:2407.09533*, 2024.

Shengbang Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Manoj Middepogu, Sai Charitha Akula, Jihan Yang, Shusheng Yang, Adithya Iyer, Xichen Pan, Ziteng Wang, Rob Fergus, Yann LeCun, and Saining Xie. Cambrian-1: A fully open, vision-centric exploration of multimodal llms. *Advances in Neural Information Processing Systems*, 37: 87310–87356, 2024.

Hugo Touvron, Andrea Vedaldi, Matthijs Douze, and Hervé Jégou. Fixing the train-test resolution discrepancy. *Advances in Neural Information Processing Systems*, 32, 2019.

Michael Tschannen, Alexey Gritsenko, Xiao Wang, Muhammad Ferjad Naeem, Ibrahim Alabdulmohsin, Nikhil Parthasarathy, Talfan Evans, Lucas Beyer, Ye Xia, Basil Mustafa, Olivier Hénaff, Jeremiah Harmsen, Andreas Steiner, and Xiaohua Zhai. SigLIP 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features. *arXiv preprint arXiv:2502.14786*, 2025.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

Mathurin Videau, Badr Youbi Idrissi, Daniel Haziza, Luca Wehrstedt, Jade Copet, Olivier Teytaud, and David Lopez-Paz. Meta Lingua: A minimal PyTorch LLM training library, 2024. https://github.com/facebookresearch/lingua.

Shakti N Wadekar, Abhishek Chaurasia, Aman Chadha, and Eugenio Culurciello. The evolution of multimodal model architectures. *arXiv preprint arXiv:2405.17927*, 2024.

Limin Wang, Bingkun Huang, Zhiyu Zhao, Zhan Tong, Yinan He, Yi Wang, Yali Wang, and Yu Qiao. Videomae v2: Scaling video masked autoencoders with dual masking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14549–14560, 2023.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024a.

Yi Wang, Kunchang Li, Xinhao Li, Jiashuo Yu, Yinan He, Chenting Wang, Guo Chen, Baoqi Pei, Ziang Yan, Rongkun Zheng, Jilan Xu, Zun Wang, Yansong Shi, Tianxiang Jiang, Songze Li, Hongjie Zhang, Yifei Huang, Yu Qiao, Yali Wang, and Limin Wang. Internvideo2: Scaling foundation models for multimodal video understanding. In *European Conference on Computer Vision*, pages 396–416. Springer, 2024b.

Manuel Watter, Jost Springenberg, Joschka Boedecker, and Martin Riedmiller. Embed to control: A locally linear latent dynamics model for control from raw images. *Advances in Neural Information Processing Systems*, 28, 2015.

Daniel M Wolpert and Zoubin Ghahramani. Computational principles of movement neuroscience. *Nature neuroscience*, 3(11):1212–1217, 2000.

Hongtao Wu, Ya Jing, Chilam Cheang, Guangzeng Chen, Jiafeng Xu, Xinghang Li, Minghuan Liu, Hang Li, and Tao Kong. Unleashing large-scale video generative pre-training for visual robot manipulation. *arXiv preprint arXiv:2312.13139*, 2023a.

Philipp Wu, Alejandro Escontrela, Danijar Hafner, Pieter Abbeel, and Ken Goldberg. Daydreamer: World models for physical robot learning. In *Conference on robot learning*, pages 2226–2240. PMLR, 2023b.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, Jianxin Yang, Jin Xu, Jingren Zhou, Jinze Bai, Jinzheng He, Junyang Lin, Kai Dang, Keming Lu, Keqin Chen, Kexin Yang, Mei Li, Mingfeng Xue, Na Ni, Pei Zhang, Peng Wang, Ru Peng, Rui Men, Ruize Gao, Runji Lin, Shijie Wang, Shuai Bai, Sinan Tan, Tianhang Zhu, Tianhao Li, Tianyu Liu, Wenbin Ge, Xiaodong Deng, Xiaohuan Zhou, Xingzhang Ren, Xinyu Zhang, Xipin Wei, Xuancheng Ren, Xuejing Liu, Yang Fan, Yang Yao, Yichang Zhang, Yu Wan, Yunfei Chu, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, Zhifang Guo, and Zhihao Fan. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*, 2024a.

Mengjiao Yang, Yilun Du, Kamyar Ghasemipour, Jonathan Tompson, Dale Schuurmans, and Pieter Abbeel. Learning interactive real-world simulators. In *International Conference on Learning Representations*, 2024b.

Lin Yen-Chen, Maria Bauza, and Phillip Isola. Experience-embedded visual foresight. In *Conference on Robot Learning*, pages 1015–1024. PMLR, 2020.

Liping Yuan, Jiawei Wang, Haomiao Sun, Yuchen Zhang, and Yuan Lin. Tarsier2: Advancing large vision-language models from detailed video description to comprehensive video understanding. *arXiv preprint arXiv:2501.07888*, 2025.

Rowan Zellers, Jiasen Lu, Ximing Lu, Youngjae Yu, Yanpeng Zhao, Mohammadreza Salehi, Aditya Kusupati, Jack Hessel, Ali Farhadi, and Yejin Choi. Merlot reserve: Neural script knowledge through vision and language and sound. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16375–16387, 2022.

Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12104–12113, 2022.

Hang Zhang, Xin Li, and Lidong Bing. Video-llama: An instruction-tuned audio-visual language model for video understanding. *arXiv preprint arXiv:2306.02858*, 2023.

Kaichen Zhang, Bo Li, Peiyuan Zhang, Fanyi Pu, Joshua Adrian Cahyono, Kairui Hu, Shuai Liu, Yuanhan Zhang, Jingkang Yang, Chunyuan Li, and Ziwei Liu. Lmms-eval: Reality check on the evaluation of large multimodal models. *arXiv preprint arXiv:2407.12772*, 2024a.

Yuanhan Zhang, Bo Li, Haotian Liu, Yong jae Lee, Liangke Gui, Di Fu, Jiashi Feng, Ziwei Liu, and Chunyuan Li. LLaVA-NeXT: A strong zero-shot video understanding model, April 2024b. https://llava-vl.github.io/blog/2024-04-30-llava-next-video/.

Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. Video instruction tuning with synthetic data. *arXiv preprint arXiv:2410.02713*, 2024c.

Long Zhao, Nitesh B. Gundavarapu, Liangzhe Yuan, Hao Zhou, Shen Yan, Jennifer J. Sun, Luke Friedman, Rui Qian, Tobias Weyand, Yue Zhao, Rachel Hornung, Florian Schroff, Ming-Hsuan Yang, David A. Ross, Huisheng Wang, Hartwig Adam, Mikhail Sirotenko, Ting Liu, and Boqing Gong. VideoPrism: A foundational visual encoder for video understanding. *arXiv preprint arXiv:2402.13217*, 2024.

Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, Ankur Handa, Ming-Yu Liu, Donglai Xiang, Gordon Wetzstein, and Tsung-Yi Lin. CoT-VLA: Visual chain-of-thought reasoning for vision-language-action models. *arXiv preprint arXiv:2503.22020*, 2025.

Ruijie Zheng, Jing Wang, Scott Reed, Johan Bjorck, Yu Fang, Fengyuan Hu, Joel Jang, Kaushil Kundalia, Zongyu Lin, Loic Magne, Avnish Narayan, You Liang Tan, Guanzhi Wang, Qi Wang, Jiannan Xiang, Yinzhen Xu, Seonghyeon Ye, Jan Kautz, Furong Huang, Yuke Zhu, and Linxi Fan. FLARE: Robot learning with implicit world modeling. *arXiv preprint arXiv:2505.15659*, 2025.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: World models on pre-trained visual features enable zero-shot planning. *arXiv preprint arXiv:2411.04983*, 2024.

Chuning Zhu, Raymond Yu, Siyuan Feng, Benjamin Burchfiel, Paarth Shah, and Abhishek Gupta. Unified world models: Coupling video and action diffusion for pretraining on large robotic datasets. *arXiv preprint arXiv:2504.02792*, 2025.

Appendix

A V-JEPA 2 Pretraining

A.1 Pretraining Hyperparameters

As detailed in Section 2.4, our training pipeline consisted of two phases: 1) a constant learning rate phase and 2) a cooldown phase. For all models, we trained in the first phase until we observed plateauing or diminishing performance on the IN1K, COIN, and SSv2 tasks. At this point, we initiated the cooldown phase.

Table 9 Pretraining Hyperparameters. Common parameters for pretraining large computer vision models. We report these parameters for both the primary training phase and the cooldown phase.

Parameter	Primary Phase	Cooldown Phase
Number of frames	16	64
Frames per Second	4.0	4.0
Crop Size	256	[256, 384, 512]
Random Resize Aspect Ratio	[0.75 1.35]	[0.75, 1.35]
Random Resize Scale	[0.3, 1.0]	[0.3, 1.0]
Steps	Variable	12000
Warmup Steps	12000	N/A
Batch Size (global)	3072	3072
Starting Learning Rate	1e-4	5.25e-4
Final Learning Rate	5.25e-4	1e-6
Weight Decay	0.04	0.04
EMA	0.99925	0.99925
Spatial Mask Scale	[0.15, 0.7]	[0.15, 0.7]
Temporal Mask Scale	[1.0, 1.0]	[1.0, 1.0]
Mask Aspect Ratio	[0.75 1.5]	[0.75, 1.5]
Tubelet Size	2	2
Patch Size	16	16

Training in the first phase began with a learning rate warmup for 12,000 steps followed by a constant learning rate for the rest of the phase. We checked evaluations every 60,000 steps. The cooldown phase began with a learning rate at 5.25e-4, which was linearly ramped down to the final learning rate. Throughout both phases, all other hyperparameters were kept constant.

In the cooldown phase we increased the number of frames per clip while keeping the frames-per-second constant, as we saw a substantial benefit from feeding the model more frames (see Figure 5). In addition, we also increased the crop size of the model in this phase, which gave a substantial benefit to tasks like IN1K, which goes from 84.6 at a 256 crop to 85.1 at a 384 crop. Hyperparameters for both phases are summarized in Table 9.

Throughout the Appendix, we refer to a "abbreviated" training recipe that corresponds to a 90,000-step training following the procedure of Bardes et al. (2024). There are a few key differences with the abbreviated recipe. The first is the learning rate: the abbreviated recipe begins with a linear warmup followed by a cosine decay. The second are the schedules for weight decay and EMA, which are linearly ramped from a starting to a final value. The last is the total number of steps, which is restricted to 90,000. We use the abbreviated schedule for several of our ablations on data mixtures, as this allows us to interrogate the effects of data curation on a shorter compute budget.

A.2 Pretraining data

We began curation of YT1B by applying scene extraction via the PySceneDetect library,[3] which splits videos into clips at scene transitions. We discard scenes shorter than 4 seconds, retaining 316 million scenes. The

[3] https://github.com/Breakthrough/PySceneDetect

Table 10 Abbreviated Pretraining Hyperparameters. Common parameters for pretraining large computer vision models, targeting our abbreviated recipe.

Parameter	Abbreviated Recipe
Number of frames	16
Frames per Second	4.0
Crop Size	256
Random Resize Aspect Ratio	[0.75 1.35]
Random Resize Scale	[0.3, 1.0]
Steps	90000
Warmup Steps	12000
Batch Size (global)	3072
Starting Learning Rate	2e-4
Larning Rate	6.25e-4
Final Learning Rate	1e-6
Starting Weight Decay	0.04
Final Weight Decay	0.4
Starting EMA	0.999
Final EMA	1.0
Spatial Mask Scale	[0.15, 0.7]
Temporal Mask Scale	[1.0, 1.0]
Mask Aspect Ratio	[0.75 1.5]
Tubelet Size	2
Patch Size	16

DINOv2 ViT-L model is then applied on the middle frame of each clip to extract scene embeddings. YT1B embeddings are then clustered into 1.5 million clusters, using the same clustering strategy as Oquab et al. (2023). Embeddings are also extracted in the same manner for all videos in the target distribution, then assigned to the closest YT1B cluster. We only keep those clusters to which at least one target video was assigned—about 210k clusters out of the original 1.5 million. The retained clusters contain 115 million scenes.

Cluster-based retrieval matches the support, but not the weighting, of the target distribution. We use a weighted sampling scheme to rebalance the data to better match the target distribution. W sample from the clusters using a weighted sampling strategy: $w_c = \sum_{d=1}^{D} w_d \times \frac{N_{d,c}}{N_d}$ where w_c is the weighting coefficient for the cth cluster, w_d is the weighting coefficient for the dth target dataset (from Table 11, $N_{d,c}$ is the number of samples from the dth dataset in the cth cluster, N_d is the total number of samples in the dth dataset, and D is the total count of target datasets. We assigned the retrieval weights approximately based on how many scenes were retrieved by each target dataset, with some extra weighting assigned to EpicKitchen. This gave a final curated dataset with statistics more closely matching those of handcrafted datasets from the literature. We found that in isolation, using curated YT1B in place of its uncurated counterpart gave much better results on downstream understanding tasks (see Figure 4).

Table 11 Data Curation Statistics. We summarized the number of extracted scenes and hours of videos across clusters extracted from YT1B. The final line includes duplicates among retrievals of K710, SSv2, COIN, and EpicKitchen.

Retrieval Target	Cluster Count	Number of Scenes	Retrieval Weight
Uncurated YT1B	1.5M	316M	
K710	170k	100M	0.7
SSv2	41k	19M	0.125
COIN	37k	21M	0.125
EpicKitchen	4k	13k	0.05
Final Curated (includes duplicates)	210k	115M	

The overall statistics from how many clusters and scenes were retrieved with this strategy are summarized in Table 11. The overall dataset is weighted towards clusters retrieved with K710. This, combined with its retrieval weight of 0.7, gives the overall curated dataset a heavy Kinetics weighting that we saw reflected in

K400 performance for our ablation experiments (see Appendix A.4.1). As shown in Table 1 in the main body, we combined this Curated YT1B with SSv2, Kinetics, HowTo100M, and ImageNet to create our final VM22M dataset.

A.3 Scaling Model Size

Details of the model architecture are shown in Table 12. All models are parameterized as vision transformers Dosovitskiy et al. (2020), using the standard 16×16 patch size. When scaling model size, we increase the encoder from a ViT-L (300M parameters) to a ViT-g (1B parameters), while the predictor size is kept fixed across all pre-training experiments.

Table 12 Model architecture details. Family of encoders and predictor architectures used during V-JEPA 2 pretraining, with some of the major parameters.

Model	Params	Width	Depth	Heads	MLP	Embedder
Encoders: $E_\theta(\cdot)$						
ViT-L	300M	1024	24	16	4096	$2 \times 16 \times 16$ strided conv
ViT-H	600M	1280	32	16	5120	$2 \times 16 \times 16$ strided conv
ViT-g	1B	1408	40	22	6144	$2 \times 16 \times 16$ strided conv
Predictor: $P_\phi(\cdot)$						
ViT-s	22M	384	12	12	1536	N.A.

A.4 Additional Results

A.4.1 Effect of Data Curation

Table 13 shows the results of data curation on a subset of downstream classification tasks. For this table, we trained models at the ViT-L and ViT-g scale using the abbreviated training recipe of the original V-JEPA (Bardes et al., 2024). When training smaller scale models (ViT-L), training a model on the curated variant of YT1B leads to across-the-board improvements over the uncurated variant. However, when moving to a mixed data setting (i.e., adding images and hand-selected videos), performance actually drops for a subset of tasks when using curated data, with performance on SSv2 at 72.8 for VM22M (Mixed+Curated YT1B) vs. 73.3 for Mixed+Uncurated YT1B. In some cases, the model trained with Curated YT1B alone is better than the one with mixed data, such as on the COIN (86.5 vs. 86.25) and K400 (84.6 vs. 83.7) evaluation tasks. This result is somewhat surprising, as despite including the K710 training data in the Mixed setting, we find that it does not improve performance over Curated YT1B for the K400 evaluation task.

Table 13 Effects of Data Curation on Video Understanding. Results are reported at both the ViT-L and ViT-g model scales. Models at both scales were pretrained using the abbreviated schedule of Bardes et al. (2024).

Training Data	IN1K	COIN	SSv2	K400
ViT-L				
Uncurated YT1B	80.6	83.2	70.9	82.9
Curated YT1B	80.8	**86.5**	73.1	**84.6**
Mixed+Uncurated YT1B	**82.9**	86.25	**73.3**	83.0
VM22M	**82.9**	86.0	72.8	83.7
ViT-g				
Uncurated YT1B	81.8	86.4	73.6	85.1
Curated YT1B	81.7	88.4	74.8	**86.5**
Mixed+Uncurated YT1B	83.7	88.5	75.5	85.9
VM22M	**83.9**	**89.2**	**75.6**	86.2

However, this behavior is not constant across scales. At the ViT-g scale, VM22M (Mixed+Curated YT1B) outperforms Mixed+Uncurated YT1B on all tasks.

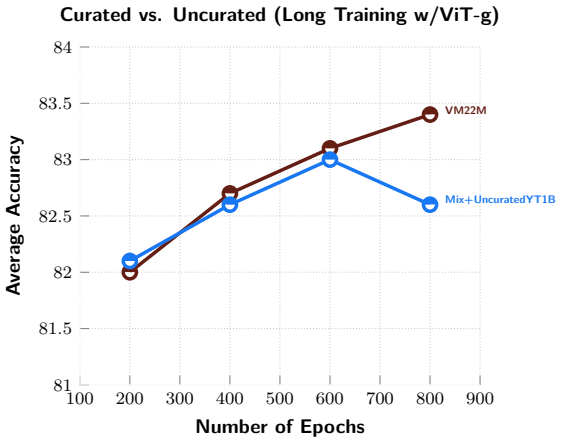


Figure 12 Effect of data curation for V-JEPA 2 pre-training. We show model performance averaged across the IN1K, COIN, SSv2, and K400 tasks as a function of pre-training "epochs" (equivalent to 300 optimization steps). Models trained with and without uncurated data achieve similar performance until epoch 600, at which point the performance of the model trained with uncurated YT1B beings degrading.

When following these tasks with the long training schedule, we continue to see differences between VM22M and Mixed+Uncurated YT1B at the ViT-g model scale, as shown in Figure 12, which compares the performance of the models while averaging across the IN1K, COIN, SSv2, and K400 image understanding tasks. Initially, the two models improve at roughly the same rate, but their performance diverges after epoch 600 where the model using uncurated data fails to continue improving.

A.4.2 Effect of Long Training Schedule and cooldown

In Table 14 we demonstrate the effects of the two-stage training process. When comparing to the ViT-g results in Table 13, we see that the abbreviated schedule is superior to the constant learning rate schedule prior to the cooldown phase. The primary benefits are achieved during the cooldown phase, which uses 64 frames for pretraining in combination with a ramped down learning rate. This leads to a large benefit of over a full point across all evaluations.

Table 14 Effects of Long Training and cooldown. Results are reported at ViT-g model with cooldown at different resolutions.

Training Stage	IN1K	COIN	SSv2	K400
Phase 1 (epoch 800, no cooldown)	83.8	89.1	75.1	85.8
Phase 2 (annealed, 256 × 256 resolution)	84.6	**90.7**	75.3	86.6
Phase 2 (annealed, 384 × 384 resolution)	**85.1**	90.2	**76.5**	**87.3**

A.4.3 Effect of Video Length at Evaluation.

Figure 13 examines how input video duration affects downstream task performance during evaluation. Using a model pretrained on 64-frame clips, we observe a +9.7 percentage point average improvement when increasing the video duration from 16 to 64 frames during evaluation. Note that this ablation uses a single clip evaluation protocol (i.e. we sample only one clip per video) instead of the standard multiclip evaluations due to memory constraints.

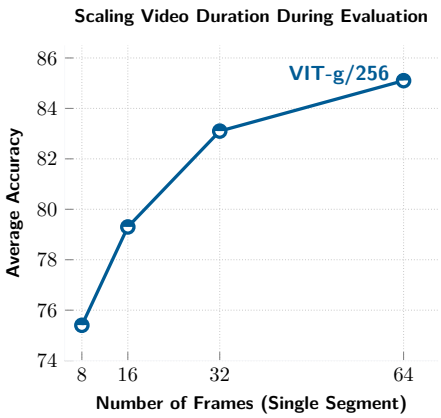


Figure 13 Effect of video duration during evaluation. Task performance further improves by running inference on longer video clips. All evaluations use ViT-g models that were annealed with 64 frames at resolution 256×256. Due to memory constraints, results are reported using a single clip evaluation protocol. Increasing the number of frames processed at inference time boosts average performance by up to +9.7 points.

B V-JEPA 2-AC Post-training

B.1 Post-Training Hyperparameters

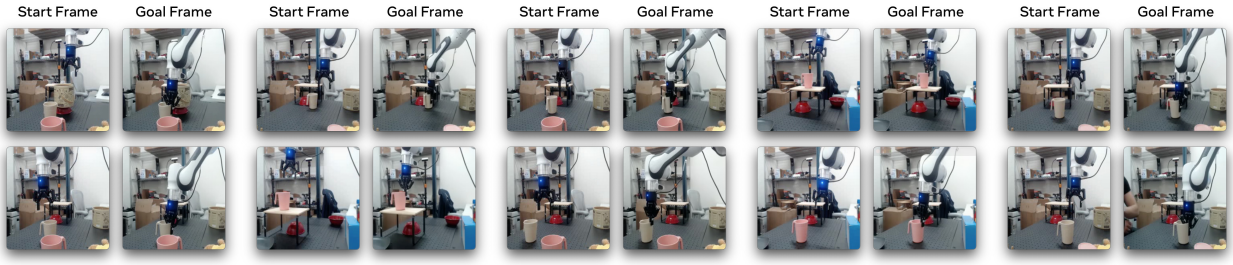
The V-JEPA 2-AC model is trained with the AdamW (Loshchilov and Hutter, 2017) optimizer using a warmup-constant-decay learning-rate schedule, and a constant weight-decay of 0.04. We linearly warmup the learning rate from 7.5×10^{-5} to 4.25×10^{-4} over 4500 iterations, then hold it constant for 85500 iterations, and finally decay it to 0 over 4500 iterations. We use a batch size of 256 comprising 4 second video clips sampled randomly from trajectories in the Droid raw dataset at a frame rate of 4 fps. We train on the left extrinsic camera views from Droid — one could also train on videos from right camera views, however we found that training on both left and right camera views, without additionally conditioning on the camera position, degraded performance. For simplicity, we discard any videos shorter than 4 seconds, leaving us with less than 62 hours of video for training. We apply random-resize-crop augmentations to the sampled video clips with the aspect-ratio sampled in the range (0.75, 1.35).

B.2 Robot Task Definitions

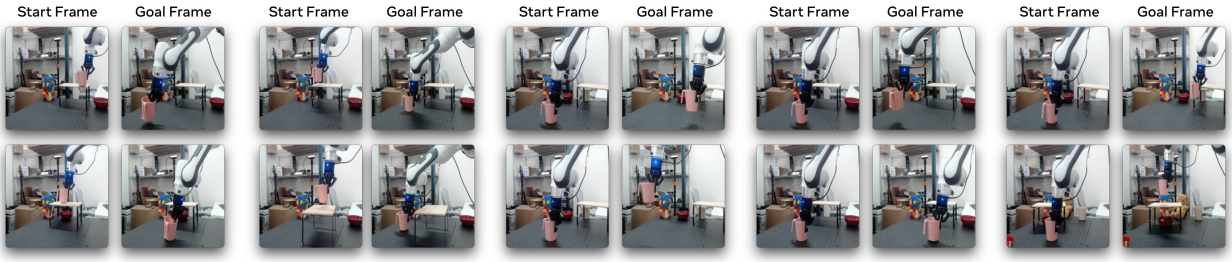
Figure 14 shows examples of start and goal frames for prehensile manipulation task with a cup in Lab 1. For the *grasp* and *reach with object* tasks the model is shown a single goal image. For the *pick-and-place* tasks we present two sub-goal images to the model in addition to the final goal. The first goal image shows the object being grasped, the second goal image shows the object in the vicinity of the goal position. The model first optimizes actions with respect to the first sub-goal for 4 time-steps before automatically switching to the second sub-goal for the next 10 time-steps, and finally the third goal for the last 4 time-steps. When planning with V-JEPA 2-AC, we use 800 samples, 10 refinement steps based on the top 10 samples from the previous iteration, and a planning horizon of 1. Since all considered tasks are relatively greedy, we found a short planning horizon to be sufficient for our setup. While longer planning horizons also worked reasonably well, they require more planning time.

B.3 Visualizing World Model Predictions

To visualize the model's predictions, we train a frame decoder on the Droid dataset that maps the V-JEPA 2 representations to human-interpretable pixels. Specifically, we process 4 frame clips with the frozen V-JEPA 2 video encoder, decode each frame separately using our decoder network, and then update the weights of the decoder using a mean-squared error (L2) pixel reconstruction loss. The decoder is a feedforward network (fully deterministic regression model that does not use any sampling internally) with output dimension $256 \times 256 \times 3$, parameterized as a ViT-L. We train the decoder for 150000 optimization steps using AdamW with a fixed



(a) Grasp Cup

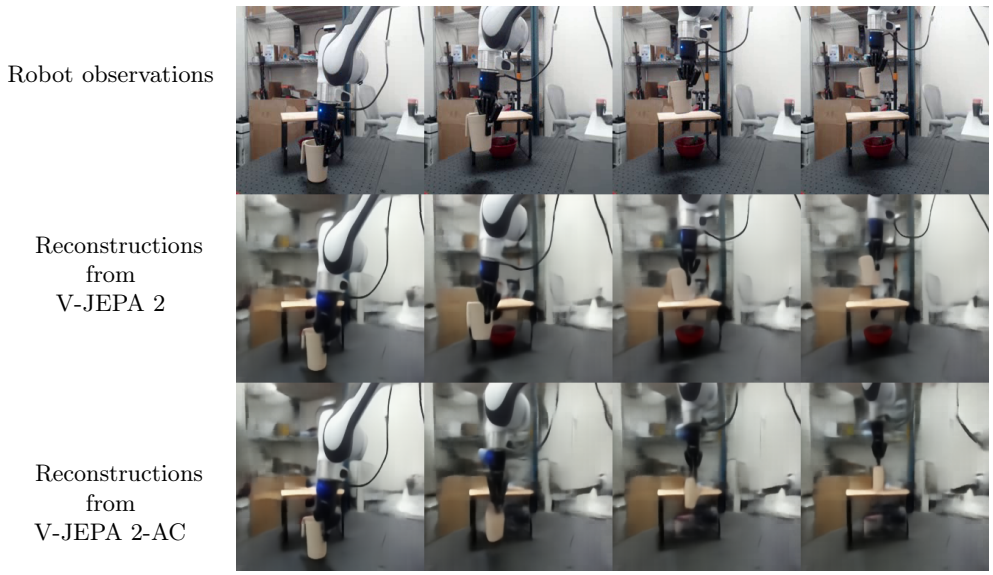


(b) Reach with Cup

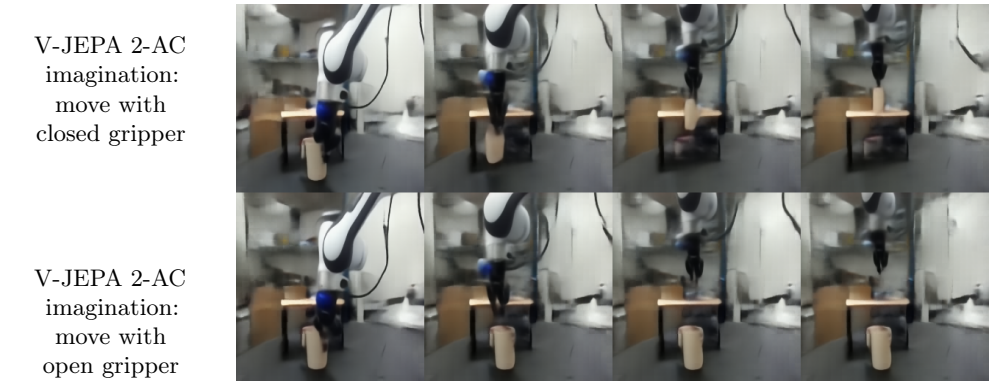
(c) Pick and Place Cup

Figure 14 Prehensile Manipulation Task Definition. Start and goal frames for prehensile manipulation tasks with a cup in Lab 1. For the *grasp* and *reach with object* tasks the model is shown a single goal image. For the *pick-and-place* tasks we present two sub-goal images to the model in addition to the final goal. The first goal image shows the object being grasped, the second goal image shows the object in the vicinity of the goal position. The model first optimizes actions with respect to the first sub-goal for 4 time-steps before automatically switching to the second sub-goal for the next 10 time-steps, and finally the third goal for the last 4 time-steps.

weight decay of 0.1, gradient clipping of 1.0, and a batch size of 1024 frames. We linearly warmup the learning rate for 2000 steps to a peak value of 5×10^{-4} and then decay it following a cosine schedule. For inference, we take the decoder trained on the V-JEPA 2 encoder and apply it off-the-shelf to the representations produced by the V-JEPA 2-AC predictor. The decision to only use use a simple feedforward architecture and



(a) Comparing accuracy of predictions to ground truth trajectory. (Top Row) Video frames of a ground-truth trajectory from a robot in our lab. **(Middle row)** Each frame is encoded by V-JEPA 2 encoder, and then decoded using the feedforward frame decoder. Reconstructions of the V-JEPA 2 representations show that the encoder captures the salient parts of the scene necessary for vision-based control; blurry background generation can be partially attributed to the low-capacity of our feedforward frame decoder. **(Bottom Row)** Autoregressive rollout produced by V-JEPA 2-AC world model using the ground-truth action sequence given first frame as context, and then decoded using feedforward frame decoder. Reconstructions of the V-JEPA 2-AC rollout show that the action-conditioned world model successfully animates the robot while keeping the background and non-interactated objects (e.g., the shelf) unaffected. However, we do observe error accumulation as the world model predicts the location of the cup to be slightly lower than that of the real trajectory in the final frame.



(b) Ablating predictions with open versus closed gripper. We explore how the V-JEPA 2-AC predictions change when driving the model with identical action sequences, but in one cause using a closed gripper **(top row)** and in the other with an open gripper **(bottom row)**. The world model predicts the location of the cup to be unchanged across time steps when using an open gripper action sequence, suggesting a reasonable understanding of intuitive physics (e.g., object constancy, shape constancy, and gravity).

Figure 15 Decoding representations. To visualize the model's predictions, we train a frame decoder on the Droid dataset that maps the V-JEPA 2 representations to human-interpretable pixels. The decoder is a feedforward network (fully deterministic regression model that does not use any sampling internally) trained with a mean-squared error pixel reconstruction loss. By applying the frame decoder, trained on the V-JEPA 2 encoder, to the representations produced by the V-JEPA 2-AC predictor, we can visualize world model rollouts for various action sequences.

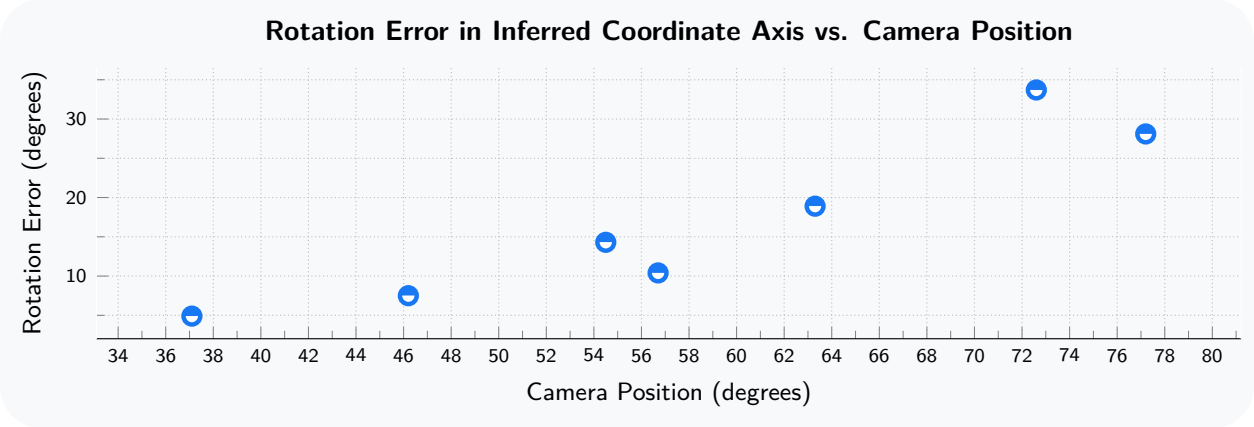


Figure 16 Sensitivity to camera position. Rotation error (in the x-y plane) of the action coordinate axis inferred by V-JEPA 2-AC as a function of camera position, with 0 degrees corresponding to a camera located at the robot base, and 90 degrees corresponding to a camera located left of the robot base. While ideally, the model's inferred coordinate axis would be invariant to camera position, here we observe that the model's inferred coordinate axis is sensitive to the camera position.

decode representations at the frame level (as opposed to video level), is to better leverage the decoder as an interpretability tool to analyze the V-JEPA 2-AC rollouts for a set of robot action sequences.

In Figure 15a, we show the video frames of a ground-truth trajectory from a robot in our lab (top row), the decoded V-JEPA 2 encoder representations of each frame (middle row), and the decoded V-JEPA 2-AC world model rollout using the ground-truth action sequence and a single starting frame as context (bottom row). Reconstructions of the V-JEPA 2 representations (middle row) show that the encoder captures the salient parts of the scene necessary for vision-based control; blurry background generation can be partially attributed to the low-capacity of our feedforward frame decoder. Reconstructions of the V-JEPA 2-AC rollout show that the action-conditioned world model successfully animates the robot while keeping the background and non-interactated objects (e.g., the shelf) unaffected. We also see that, with a closed gripper, the model correctly predicts the movement of the cup with the arm, suggesting a reasonable understanding of intuitive physics (e.g., object constancy, shape constancy, and gravity), but we do observe error accumulation as the world model predicts the location of the cup to be slightly lower than that of the real trajectory in the final frame. In Figure 15b we explore how the V-JEPA 2-AC predictions change when driving the model with identical action sequences, but in one cause using a closed gripper (top row) and in the other with an open gripper (bottom row). The world model predicts the location of the cup to be unchanged across time steps when using an open gripper action sequence.

B.4 Assessing Sensitivity to Camera Position

In practice, we manually tried different camera positions before settling on one that worked best for our experiments; then the camera is kept in the same location for all experiments, across all tasks. In this section, we conduct a quantitative analysis of the V-JEPA 2-AC world model's sensitivity to camera position. While ideally, the model's inferred coordinate axis would be invariant to camera position, here we observe that the model's inferred coordinate axis is sensitive to the camera position; this is problematic as large errors in the inferred coordinate axis can degrade success rate on downstream tasks.

We sweep several camera positions around the robot base, which we describe as a clockwise angular position around the center of the table, with 0 degrees being located at the robot base, and 90 degrees being left of the robot base. Since we train on the left exocentric camera views from the Droid dataset, we sweep camera positions between roughly 35 degrees and 85 degrees. Next, for each camera position, we collect a 201 step trajectory of random robot movements within the horizontal x-y plane. For each pair of adjacent frames in this 201 step trajectory, we compute the optimal action inferred by V-JEPA 2-AC, i.e., the action that minimizes the energy function in eq. (5) given a 1-step rollout. This allows us to construct a dataset for

each camera position consisting of *real action* versus *inferred action* pairs. We only focus on the Δx and Δy cartesian control actions (first two dimensions of the action vector) for our analysis. Let $A \in \mathbb{R}^{200\times 2}$ denote the inferred actions and $B \in \mathbb{R}^{200\times 2}$ denote the ground truth actions. Based on this, we can solve a linear least squares problem to identify the linear transformation $W^\star \in \mathbb{R}^{2\times 2}$ that maps inferred actions A to real actions B,

$$W^\star = \underset{W\in\mathbb{R}^{2\times 2}}{\operatorname{argmin}} \|AW - B\|_2.$$

The mean absolute prediction error for all camera position is roughly 1.6cm (compared to a ground truth delta pose of roughly 5cm), suggesting that the error is systematic. In addition, we observe that for each camera position, the matrix $W^\star$ has condition number ≈ 1.5, i.e., modulo a fixed scalar coefficient, $W^\star$ is approximately a rotation matrix, and thus we can compute the rotation error in the inferred coordinate axis by using

$$W^\star \approx \overline{W}^\star = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix},$$

with $\overline{W}^\star := UV^\top$ where U and V are the left and right singular vectors of $W^\star$, respectively.

Figure 16 shows the camera position plotted against the rotation error in the V-JEPA 2-AC inferred coordinate axis. We observe that the rotation error in the inferred coordinate axis is almost a linear function of the camera position. We can most clearly see the effects of rotation errors in the inferred coordinate axis in our single-goal reaching experiments in Figure 8. While the model is always able to move the arm within 4 cm of the goal based on visual feedback from the monocular RGB camera, rotation errors in the inferred coordinate axis result in relatively suboptimal actions at each planning step, yielding a non-maximal, albeit monotonic, decrease in the distance to goal at each step.

Interestingly, since errors in the inferred coordinate axis are primarily rotation-based, one can use this approach to "calibrate" their world model by simply rotating all inferred actions by $W^\star$, and thereby introduce the desired invariance to camera position. Such an unsupervised calibration phase would involve the robot performing random actions, solving a linear least squares problem by comparing its inferred optimal actions to the actual actions it executed, and then multiplying its inferred actions by the rotation matrix before sending them to the controller during task execution. While such an approach is interesting, we emphasize that we *do no such calibration* in our experiments.

C Visual Classification

We describe in more detail the evaluation procedure used for the classification tasks described in Section 5.

C.1 Hyperparameters

Probe Architecture. We train an attentive probe on top of the frozen encoder output using the training data from each downstream task. Our attentive probe is composed of four transformer blocks, each using 16 heads in the attention layer. The first three blocks use standard self-attention; the final block uses a cross-attention layer with a learnable query token. The output of the cross-attention layer in the final block is added back to the query token as a residual connection before applying the rest of the block (LayerNorm, followed by MLP with a single GeLU activation). The transformer blocks are followed by a final linear classifier layer.

Evaluation setup parameters. All models follow the same evaluation protocol and use a resolution of 256×256, except our V-JEPA 2 ViT-g_{384}. For video evaluations, we sampled multiple clip segments from each input video. During validation, we also extracted three spatial views from each segment (instead of one view during training). The number of clip segments, frame step parameter, and global batch size vary for each eval; parameters used for each evaluation can be found in Table 15. By default, we use $16 \times 2 \times 3$ inputs for SSv2 (16 frames clip, 2 temporal crops, 3 spatial crops), $16 \times 8 \times 3$ for K400, $32 \times 8 \times 3$ for COIN, and $32 \times 4 \times 3$ for Diving-48 and Jester. V-JEPA 2 ViT-g_{384} uses a higher resolution of 384×384 for K400, COIN, Diving-48, and Jester, 512×512 for ImageNet and 384×384 with $64 \times 2 \times 3$ inputs for SSv2.

Table 15 Visual Classification eval params. Default parameters used for the visual classification evaluations, with non-default values for each eval (* denotes default). All attentive probes use 4 transformer blocks with 16 heads.

Parameter	Default (K400)	ImageNet	SSv2	COIN	Jester/Diving-48
Number of frames	16	16	16	32	32
Segments / Clip	8	1	2	8	4
Views / Segment	3	1	*	*	*
Frame Step	4	n/a	*	*	2
Epochs	20	*	*	*	100
Batch Size (global)	256	1024	*	128	128
Resolution	256×256	*	*	*	*
Classifier Heads	20 (4x5)	*	*	*	3 (3x1)
Classifier Learning Rates	[5e-3 3e-3 1e-3 3e-4 1e-4]	*	*	*	[1e-3 3e-4 1e-4]
Classifier Weight Decay	[.8 .4 .1 .01]	*	*	*	[.8]

Table 16 Input layers for Jester/Diving-48. For each encoder size, indices of the four encoder layers whose tokens are used as input to the linear classifier in the Jester and Diving-48 evaluations.

Encoder	# Layers	Attended Layers
ViT-L	24	17, 19, 21, 23
ViT-H	32	25, 27, 29, 31
ViT-g	40	24, 29, 34, 39

ImageNet evaluation. For ImageNet, we repeat each input image to produce a 16-frame video clip. We also use a larger global batch size (1024 instead of 256 or 128), and do not use multiple clips or views per sample.

Jester and Diving-48 evaluation. Our Jester and Diving-48 action classification evaluation tasks differ from the other understanding evaluations in several ways, primarily in that we employ a multilayer strategy. Instead of attending to the tokens from only the last layer of the encoder, we extract tokens from four encoder layers (the last layer and three intermediate layers) and attend to all of them. (Table 16 shows the layers we used for each encoder size.) We also train the probes for these two evaluations with only three classification heads (instead of 20 for the other evaluations), but train for 100 epochs (instead of 20) as these evaluations benefit from longer training. We use a global batch size of 128 for both evaluations.

Optimization. For each evaluation, we simultaneously train multiple classifier heads with different hyperparameters (learning rate and weight decay), reporting the accuracy of the best-performing classifier. For most of our evaluations (Kinetics, SSv2, COIN, and ImageNet), we train for 20 epochs and use 20 heads, each using one of five learning rate values and four weight decay values, and the learning rate decays according to a cosine schedule. We provide a summary of all hyperparameters in Table 15.

C.2 Additional Results

Probe Size. Since we use a four-layer attentive probe for these evaluations, we investigate whether using a smaller probe impacts evaluation performance. We re-run our six understanding evaluations (for two model sizes, ViT-L and ViT-g) with a smaller probe consisting of a single cross-attention block using 16 attention heads. Unlike in Section 5, we use 16 frames for all evaluations, including Diving-48 and Jester. See Table 18 for classification performance—we confirm that our four-layer probe outperforms a single-layer attentive probe across all understanding evaluations (except for Jester), by an average of +1.4 points accuracy for ViT-L and +1.0 points for ViT-g.

Impact of encoder multilayer. We study the impact of feeding tokens from multiple layers from the encoder to the attentive probe during evaluation. Table 17 shows that Diving-48 and Jester strongly benefit from information from deeper layers of the encoder.

Table 17 Encoder Multilayer Ablation. We vary the number of encoder layers fed to the attentive probe. We report the classification performances of attentive probes trained on top of V-JEPA 2 with 16 frames at 256×256 resolution.

Model	Encoder Layers	Diving-48	Jester
ViT-g	1	82.9	96.1
ViT-g	4	86.7	97.6

Table 18 Probe Size Ablation. We vary the number of layers in the attentive probe. We report the classification performances of attentive probes trained on top of V-JEPA 2 with 16 frames at 256×256 resolution.

			Motion Understanding			*Appearance Understanding*		
Model	**Probe Layers**	**Avg.**	**SSv2**	**Diving-48**	**Jester**	**K400**	**COIN**	**IN1K**
ViT-L	1	84.0	72.0	83.2	97.7	83.3	85.9	81.8
ViT-L	4	85.6	73.6	87.1	97.7	85.1	86.8	83.5
ViT-g	1	86.0	74.8	85.3	97.8	85.6	88.9	83.5
ViT-g	4	87.0	75.6	86.7	97.6	86.6	90.7	84.6

D Action Anticipation

We provide additional details, results, and ablations related to the Epic-Kitchen 100 action anticipation evaluation of Section 6.

D.1 Hyperparameters

Probe Architecture. Our probe architecture for action anticipation follows the architecture of our classification probe described in Appendix C.1, consisting of four transformer blocks, including a last cross-attention layer with a set of learnable query tokens, followed by a final linear classifier layer for each query token.

Evaluation setup parameters. We use a focal loss (Lin et al., 2017) with a $\alpha = 0.25$ and $\gamma = 2.0$ when training the probe; this loss is more suited for training with long-tailed imbalanced class distributions. We use a context of 32 frames with a frame-rate of 8 frames per second at resolution 256×256 for V-JEPA 2 ViT-L, ViT-H, and ViT-g; and resolution 384×384 for V-JEPA 2 ViT-g_{384}. During probe training, we randomly sample an anticipation time between 0.25 and 1.75 seconds, and an anticipation point between 0.0 and 0.25. The *anticipation point* identifies a point in the action segment from which to perform anticipation; i.e., an anticipation point of 0 means that we predict the representation of the first frame in the action segment using our V-JEPA 2 predictor before feeding it to the probe, whereas an anticipation point of 1 means that we predict the representation of the last frame in the action segment before feeding it to the probe. The validation anticipation time is set to 1 second and the validation anticipation point is set to 0. We provide a summary of the hyperparameters, including the optimization parameters in Table 19.

D.2 Additional results

Impact of Architecture. Table 20 investigates the impact of providing the output of the V-JEPA 2 encoder, predictor, or both, to the action anticipation probe. Using encoder outputs already leads to competitive performance on the EK100 task. Adding the predictor provides a small but consistent improvement across action, verb, and object categories. In addition, using predictor outputs yields a non-trivial performance, but still at a much lower point compared to using the encoder, showing that the EK100 task mostly requires strong semantic understanding, as opposed to forecasting capabilities.

Impact of Input Resolution. We report in Figure 17, the impact of input resolution and frame sampling parameters. In summary, V-JEPA 2 benefits from a longer context, a higher frame rate, and higher resolution,

Table 19 Action Anticipation Evaluation params. Default parameters used for the EK100 Action Anticipation evaluation.

Parameter	EK100
Train Anticipation time	0.25s – 1.75s
Train Anticipation point	0.0 – 0.25
Val Anticipation time	1s
Val Anticipation point	0.0
Number of frames	32
Frames per second	8
Epochs	20
Warmup epochs	0
Batch Size (global)	128
Classifier Heads	20 (4x5)
Classifier Learning Rates	[5e-3 3e-3 1e-3 3e-4 1e-4]
Classifier Weight Decay	[1e-4 1e-3 1e-2 1e-1]

Table 20 EK100: Impact of Anticipation Probe Inputs. We investigate the impact of providing the outputs of the V-JEPA 2 encoder, predictor, or both, to the action anticipation probe. Using encoder outputs already leads to competitive performance on the EK100 task. Adding the predictor provides a small but consistent improvement across action, verb and object categories.

		Action Anticipation		
Encoder	**Predictor**	Verb	Noun	Action
✓		61.3	57.0	39.1
	✓	48.7	34.7	20.2
✓	✓	63.6	57.1	39.7

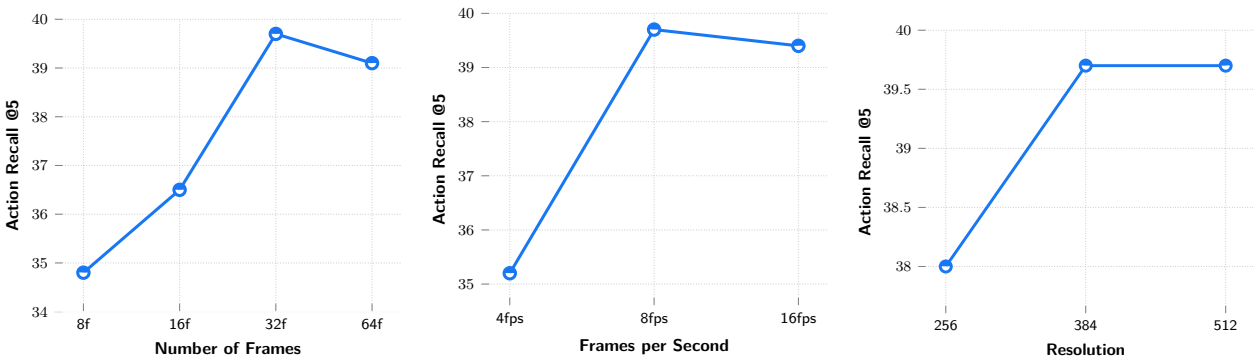


Figure 17 Protocol ablation for action anticipation on EK100. (Left) Performance with respect to the number of context frames used for action anticipation. **(Middle)** Performance with respect to the frame rate (fps) used for inference; number of context frames fixed at 32. **(Right)** Performance with respect to the spatial resolution (height and width) of the context frames used for action anticipation.

up to a point where the performance saturates or slightly decreases. The optimal performance is obtained by training with a 32-frames context length, a frame rate of 8 and a resolution of 384×384.

Longer-term Prediction. We report in Figure 18 (Left), the impact of predicting at a longer horizon, by varying the anticipation time between (1s, 2s, 4s, 10s). For each anticipation time, we report recall at several values (1, 5, 10, 20). The results show that the performance sharply decreases as the anticipation time increases, which is expected since forecasting the future in EK100 is a non-deterministic task.

Analysis of Failure Cases. We report in Figure 18 (Right), the distribution of failure and success prediction, on the EK100 validation set, between each configuration of success/failure for verb, noun, and action. The model performs very well, and the most represented configuration is, therefore, a full success

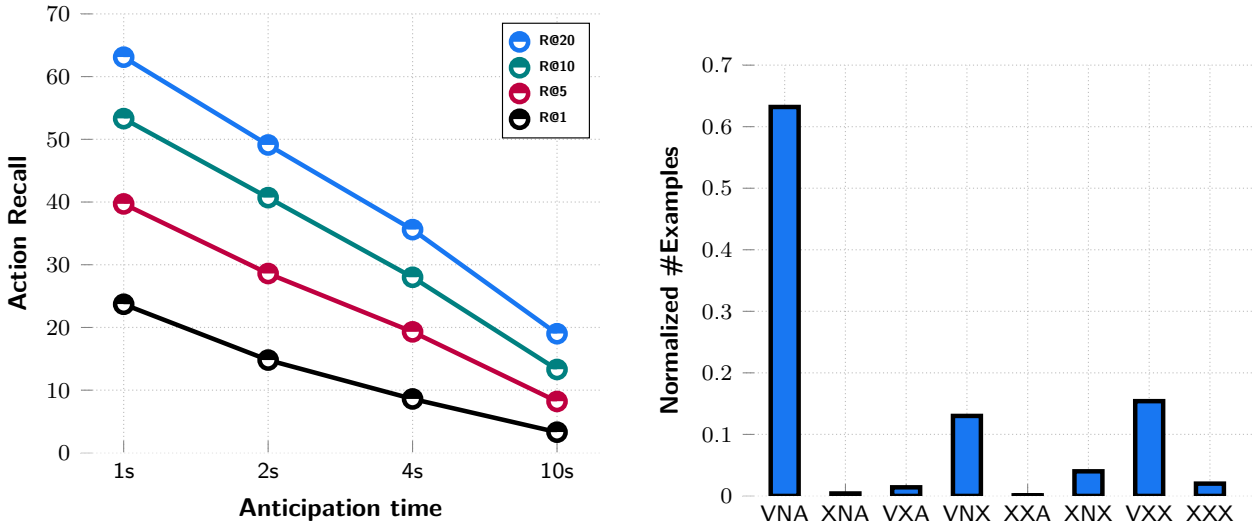


Figure 18 (Left): Impact of longer-term anticipation times. Performance on EK100 action anticipation, at several recall values and anticipation times. **(Right): Distribution of success and failure cases of V-JEPA 2.** Calculated on the validation set of EK100. VNA means that verb, noun and action are all correctly classified by the model. An X symbol means that the corresponding attribute is not correctly classified by the model. Note: the probe is composed of 3 independent classifiers for verb, noun and action, hence why the model can have a different prediction for the action and for the verb/noun pair.

across verb noun and action. The most represented failure configurations all include a failure to find the action.

E Video Question Answering

In this section, we provide details on training a V-JEPA 2 Multi-modal Large Language Model (MLLM). We follow the LLaVA framework (Liu et al., 2024a) to train the MLLM, where the vision backbone is using V-JEPA 2, and the LLM backbone can be any off-the-shelf pretrained LLM, akin to the *non-tokenized early fusion* (Wadekar et al., 2024) setup. The MLLM ingests the output embeddings of the vision encoder, which are projected to the hidden dimension of the LLM backbone using a *projector* module. The *projector* is typically a 2-layer MLP. The MLLM is trained using a mix of image-text and video-text paired data, in a series of progressive training steps.

To understand the impact of data scale, we use a dataset of 88.5 million image-text and video-text pairs, similar to what was used for training PerceptionLM (Cho et al., 2025). As mentioned in Section 7, we investigate two setups: (a) *controlled*, where we train with 18M image and video-text pairs, and we evaluate V-JEPA 2 and other encoders on the exact same MLLM training setup, and (b) *scaling*, where we take V-JEPA 2 VITg_{384} and use the full aligment dataset. To further test the versatility of V-JEPA 2, we use Qwen2-7B-Instruct (Yang et al., 2024a) as the language backbone for the controlled experiments, and Llama 3.1 8B Instruct (Grattafiori et al., 2024) for the scaling experiments. We describe the training details in the following sections.

E.1 Processing Images and Videos as Input

Since video question answering uses video instead of image inputs, the number of output visual tokens increases significantly compared to image question answering. If required, we can use pooling methods to reduce the number of visual tokens. Popular pooling methods involve adaptive 2x2 pooling (Cho et al., 2025), Perceiver Sampler (Jaegle et al., 2021), Attentive Pooling (Bardes et al., 2024), etc.

Additionally, we observed that learning from image-text pairs is crucial for high performance in downstream

benchmarks. In order to train with images, a simple approach is to repeat the given image for k frames, where k is the maximum amount of frames supported by V-JEPA 2. However, during our initial experiments we find this strategy is ineffective at improving downstream performance, as it does not allow the model to extract fine-grained information. Therefore, we employ a modified *Dynamic* S^2 strategy introduced by Liu et al. (2024d) to provide V-JEPA 2 higher resolution granularity during training. This method adaptively processes an image at native resolution with different aspect ratios to preserve their original resolution, by creating a sequence of tiles of maximum size supported by V-JEPA 2. In case of videos, we choose to train with a fixed number of frames f_n, by balancing the number of visual tokens with compute budget.

E.2 Controlled Setup

Training details For the controlled setup, we follow the LLaVA-NEXT framework (Liu et al., 2024a; Zhang et al., 2024b), where we use Qwen2-7B-Instruct (Yang et al., 2024a) as the base LLM for all encoders. To reduce the number of visual tokens, we employ an attentive pooler with a factor of 4-16, depending on the compute budget and the number of visual patches. See Table 21 for more details.

Our training setup follows the LLaVA-NeXT pipeline (Li et al., 2024b), which consists of multiple staged training phases. Concretely, the stages consist of: a) aligning the attentive pooler with image captioning data (Stage 1), b) training the full model on high quality image captioning (Stage 1.5), and c) training the full model on large scale image question answering (Stage 2). We add an extra stage to train on large scale video captioning and question answering (Stage 3). We use 18 million image and video-text aligned data. The LLM progressively improves its understanding of the visual tokens after multiple staged training, with the biggest improvement in video question answering tasks after Stage 3.

We explore *frozen* and *finetuned* encoder alignment setups. In both setups, full parameters of the LLM and projector are trained, and in the latter, the V-JEPA 2 parameters are additionally unfrozen. To reduce the number of visual tokens and keep the MLLM context length fixed, we employ an attentive pooler as the projector to reduce the number of visual tokens by a factor of 4, unless otherwise denoted. The implementation used for this controlled study is based on the Llava-NEXT codebase,[4] and uses Pytorch 2.5.1, Transformers 4.46.0, Flash attention 2 and DeepSpeed 0.14.4 for model implementation, faster training and multi-gpu model sharding respectively. We train all models using 128 H100 GPUs with an effective batch size of 256 across all stages. We perform all optimizations using AdamW with 0 weight decay. For Stages 1 and 1.5, we use learning rate of 1e-5 with cosine decay, and for Stages 2 and 3 we use constant learning rate of 5e-6. In all stages, we use linear warmup for the first 3% of training steps. Training hyperparameters are listed in Table 21.

Baselines. To assess the ability of V-JEPA 2 to capture spatiotemporal details for VidQA, we compare to leading off-shelf image encoders. Specifically, we compare to DINOv2 (Oquab et al., 2023), SigLIP2 (Tschannen et al., 2025), and Perception Encoder (Bolya et al., 2025). DINOv2 is a self-supervised image model, while SigLIP2 and Perception Encoder are both trained with language supervision using noisy image-text captions. We apply all image encoders at their "native" pretrained resolution, which is 518px, 384px, and 448px, respectively, on each video frame independently.

We keep all training details the same, except that we increase the attentive pooling ratio to 16 to keep the number of image tokens relatively similar among models. See Table 21 for details.

Evaluation. To evaluate the capability of V-JEPA 2 to understand the world through video and language, we select popular evaluation datasets built to test spatio-temporal reasoning abilities. To ensure reproducible evaluation, we utilize the `lmms-eval` library (Li et al., 2024a; Zhang et al., 2024a) to conduct our experiments, which is a vision model enabled fork of `llm-eval-harness` (Gao et al., 2024), which is a popular evaluation library for evaluating LLMs on text-based tasks. In the controlled setup, for each model and dataset, we evaluate by using uniform frame sampling mechanism, and choosing 128 frames during inference. For PerceptionTest, we further train the model for 5 epochs on the training set.

[4] https://github.com/LLaVA-VL/LLaVA-NeXT

Table 21 Hyperparameters for controlled comparison of vision encoders. We use each vision encoder with its native pretrained input resolution.

Model	Pooling Ratio	Vision Tokens	BS	Stage 1/1.5 LR	Stage 2/3 LR	WD	Frames
V-JEPA 2 ViT-L_{256}	4	4096	256	1e-5	5e-6	0.0	128
V-JEPA 2 ViT-H_{256}	4	4096	256				
V-JEPA 2 ViT-g_{256}	4	4096	256				
V-JEPA 2 ViT-g_{384}	4	9216	256				
V-JEPA 2 ViT-g_{512}	8	8192	256				
$DINOv2_{518}$	16	10952	256	1e-5	5e-6	0.0	128
$SigLIP2_{384}$	16	5832	256				
PE_{448}	16	8192	256				

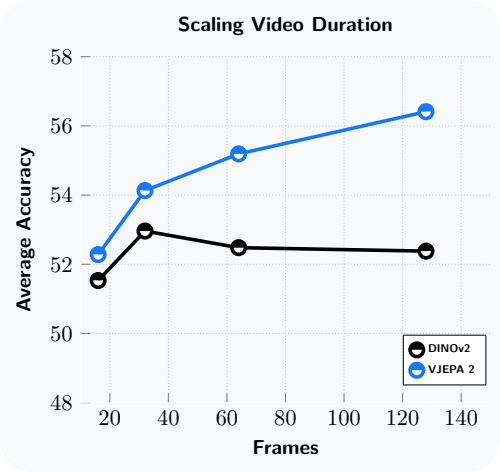


Figure 19 Impact of video duration during visual instruction tuning. We investigate the effect of increasing the number of frames during visual instruction tuning, where the encoder is frozen. We observe that with more frames, V-JEPA 2 performance linearly increases compared to DINOv2, an SSL-based image encoder, showing the potential of V-JEPA 2 to scale with more frames.

Impact of Video Duration In the controlled setup, we perform an analysis to understand V-JEPA 2's capability in long-form video understanding. We train MLLMs on V-JEPA 2 and DINOv2, keeping the encoders frozen, and by increasing the number of frames we use in training and testing. We observe as the number of frames increases, performance on downstream tasks *linearly* improves for V-JEPA 2, but decreases and remains flat in case of DINOv2 (Figure 19). This highlights the potential of video encoders such as V-JEPA 2 to understand long-form videos with natural language queries, via adapting an LLM using V-JEPA 2 as the visual encoder.

E.3 Data scaling setup

Training details In the scaling setup, we follow the framework used by Cho et al. (2025) to train Perception LM 8B. Specifically, we utilize the released codebase, which is based on Lingua (Videau et al., 2024). We modify the code to use V-JEPA 2 encoder, and we use the Llama 3.1 8B Instruct (Grattafiori et al., 2024) as the backbone LLM. Unlike Cho et al. (2025), we do not use pooling, instead we train V-JEPA 2 VIT-g_{384} using MLP projector, leading to 288 tokens per frame. The training setup also consists of three progressive stages: Stage 1: aligning the MLP pooler with image captioning data; Stage 2: training on a mix of image-text captioning and QA data; and Stage 3) training on video-text captioning and QA data. We scale up the data size to 88.5 million samples. Our setup uses Pytorch 2.5.1 and Perception LM training code,[5] modified with the V-JEPA 2 encoder. We train on 512 H100 GPUs for Stage 2 and Stage 3 with a global batch size of 2048 and 1024 respectively. Details of the training hyperparams are provided in Table 22.

[5] https://github.com/facebookresearch/perception_models

Table 22 **Data scaling training parameters.**

Parameter	Values
Common parameters	
Crop Size	384
Video Frames per Second	1
Sampling method	Uniform
Seed	777
Stage 1	
Steps	16000
Warmup Steps	96
Batch Size (global)	128
Learning Rate	1e-4
Final Learning Rate	1e-6
Weight Decay	0.05
Max sequence length	1920
Stage 2	
Steps	35000
Warmup Steps	200
Batch Size (global)	2048
Learning Rate	4e-5
Final Learning Rate	4e-7
Weight Decay	0.05
Max sequence length	6400
Image tiles	16
Video frames	16
Stage 3	
Steps	28000
Early stopping step	22000
Warmup Steps	168
Batch Size (global)	2048
Learning Rate	1e-5
Final Learning Rate	1e-7
Weight Decay	0.05
Max sequence length	12800
Image tiles	32
Video frames	32

Baselines. We compare our scaling runs with Qwen2VL (Wang et al., 2024a), Qwen2.5VL (Qwen Team et al., 2025), InternVL-2.5 (Chen et al., 2024), and PerceptionLM 8B (Cho et al., 2025). Baseline numbers are sourced directly from the papers, except for MVP which we run ourselves.

Evaluation. We follow similar evaluation pipeline as reported in the controlled setup, using `lmms-eval` library. We report our model evaluations on 32 frames.