AMPLIFY: Actionless Motion Priors for Robot Learning from Videos

Anonymous Author(s)
Affiliation
Address
email

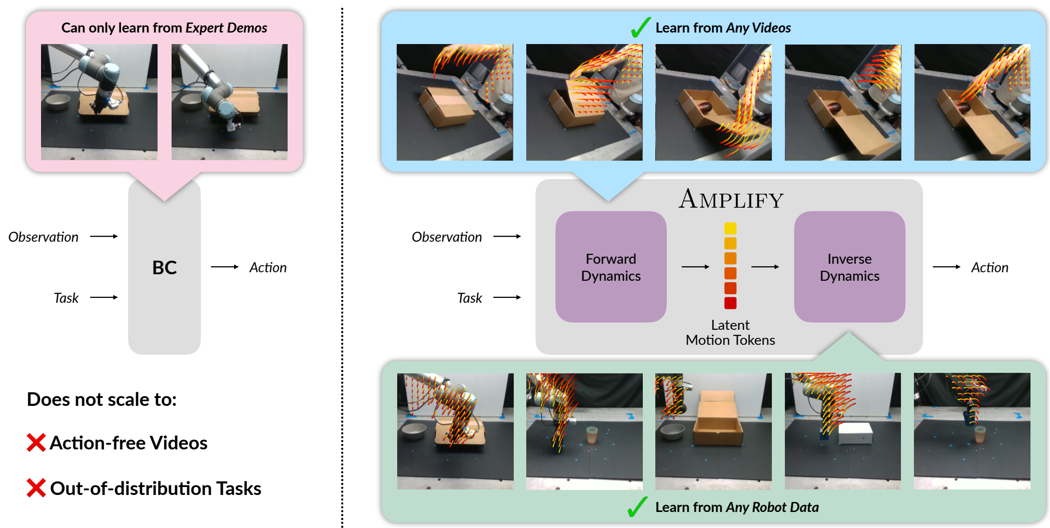


Figure 1: Overview. AMPLIFY decomposes policy learning into forward and inverse dynamics, using latent keypoint motion as an intermediate representation. The forward model can be trained on *any* video data, while the inverse model can be trained *any* interaction data. In contrast with behavior cloning (BC), AMPLIFY requires fewer demos, can generalize to tasks for which we have *zero* action data, and learn from human videos.

Abstract:

Action-labeled data for robotics is scarce and expensive, limiting the generalization of learned policies. In contrast, vast amounts of action-free video data are readily available, but translating these observations into effective policies remains a challenge. We introduce AMPLIFY, a novel framework that leverages large-scale video data by encoding visual dynamics into compact, discrete motion tokens derived from keypoint trajectories. Our modular approach separates visual motion prediction from action inference, decoupling the challenges of learning ***what*** motion defines a task from ***how*** robots can perform it. We train a forward dynamics model on abundant action-free videos and an inverse dynamics model on a limited set of action-labeled examples, allowing for independent scaling. Extensive evaluations demonstrate that the learned dynamics are both accurate—achieving up to 3.7× better MSE and over 2.5× better pixel prediction accuracy compared to prior approaches—and broadly useful. In downstream policy learning, our dynamics predictions enable a 1.2-2.2× improvement in low-data regimes, a 1.4× average improvement by learning from action-free human videos, and the first generalization to LIBERO tasks from zero in-distribution action data. Beyond robotic control, we find the dynamics learned by AMPLIFY to be a versatile latent world model, enhancing video prediction quality. Our results present a novel paradigm leveraging heterogeneous data sources to build efficient, generalizable world models. More information can be found at amplify-robotics.github.io.

Keywords: Behavior Cloning, Video Understanding, Dynamics Modeling

Submitted to the 9th Conference on Robot Learning (CoRL 2025). Do not distribute.

1 Introduction

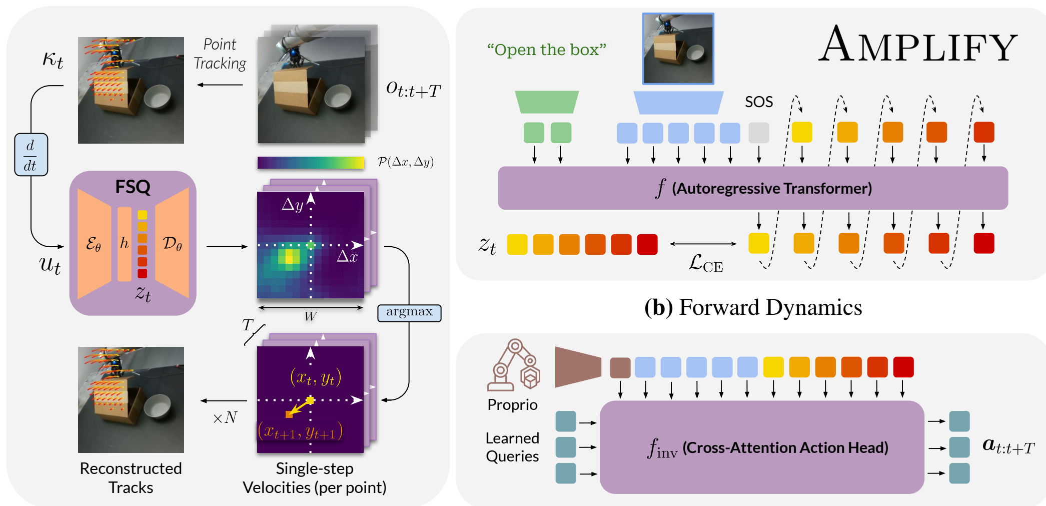
Recent successes in harnessing internet-scale data to train image and language foundation models [1, 2, 3, 4, 5, 6] have spurred an analogous push in robotics. In contrast with earlier methods that focused on achieving expert-level capabilities in narrow, controlled domains, recent efforts in robotics have aimed to generalize across tasks, object categories, object instances, environments, and the abundant variety of conditions present in the natural world [7, 8, 9, 10, 11, 12]. However, in order to train such generalist models, the typical behavior cloning (BC) approach requires prohibitively large amounts of **action-labeled expert demonstrations**. Datasets that are considered large-scale for robotics [7, 9, 13] take weeks or months to collect a few *hundred* hours of interaction data, falling far short of the roughly *one billion* hours of video data available on the internet. Therefore, methods that incorporate large-scale pre-training on these more abundant modalities tend to generalize better from limited action data [14, 15, 8]. Videos, in particular, contain rich priors on temporally-extended dynamics, behaviors, and semantics, which can be used to learn a predictive model of the world [16, 17, 18, 19, 20, 21, 22, 23]

Prior work has leveraged video pre-training to learn representations using a number of auxiliary tasks such as reward and value prediction [24, 25, 26, 27] or time-contrastive loss terms [24, 28, 29]. While useful as representations, these methods only learn an encoder for static observations and do not explicitly model sequential dynamics. In contrast, model-based approaches can improve sample efficiency by separating the challenge of policy learning from learning dynamics [30]. Since videos contain rich priors over object and agent dynamics, model-based methods offer a promising avenue for learning from limited action data. One such approach is to train a full video prediction model to capture visual dynamics, which can act as a reference generator for downstream policies [16, 31]. However, predicting in pixel space is computationally intensive and costly to run at high frequencies, forcing these methods to make compromises like open-loop control [16] or partial denoising [31]. As a result, a number of works have aimed to learn *latent action* representations from videos using next-frame prediction [32, 33, 34] or latent consistency [35], efficiently modeling features that are predictive of the future. While this avoids high inference costs, these representations are still trained on image reconstruction/prediction objectives, capturing textural details or visually salient features that may not be relevant to policy learning.

Motivated by the desire to capture motion rather than appearance, optical flow and keypoint tracking have emerged as appealing abstractions for extracting action information from videos without action labels. Recent advances in computer vision have enabled efficient and precise pixel-level point tracking, even through occlusions and limited out-of-frame tracking [36, 37, 38, 39]. As these capabilities enable fine-grained capture of motion and scene dynamics, they have found applications in robotics for visual imitation learning [40] and tool use [41]. A number of prior works predict motion from images as optical flow [42, 43, 44] or by modeling the trajectories of specified keypoints [45, 46, 47, 48, 49, 50, 51]. However, many of these works still rely on prohibitively expensive video prediction models [51, 52, 44], object-centric mask extraction [51, 49, 47, 53], calibrated cameras [50], or inefficient online planning [48], limiting their generality.

Two of the most general keypoint modeling approaches are ATM [54] and Track2Act [53], which aim to learn a universal keypoint dynamics model to predict the future trajectories of arbitrary points in an image, and condition a policy on these predictions. However, Track2Act relies on the often unrealistic assumption of a goal image and restricts its output space to single-object rigid-body transformations. ATM, while more flexible in its representation, relies on unrealistic point-sampling heuristics during training that cannot be replicated during inference. In addition, neither ATM nor Track2Act learn a latent space abstraction of keypoints, leaving them with high computational costs much like pixel-space video generation and potentially hindering generalization. Due to their high computational costs, Track2Act requires open-loop trajectory generation, and ATM only generates tracks for 32 points during policy inference, resulting in very coarse dynamics predictions. Further discussion and comparison to related work can be found in Appendix C.

In this paper, we investigate the use of *latent* keypoint motion as an abstraction for learning valuable action priors from action-free video data, combining the benefits of latent dynamics prediction with



(a) Motion Tokenization **(b)** Forward Dynamics **(c)** Inverse Dynamics

Figure 2: Architecture. AMPLIFY consists of a three-stage decomposition: (a) keypoint tracks are compressed into a discrete latent space using FSQ. For each timestep and each point, the decoder outputs a distribution in a local window centered around each point to reconstruct the instantaneous velocities, (b) a forward dynamics model is trained to predict the latent codes for the next T timesteps given an input image and task description, and (c) an inverse dynamics model decodes predicted track tokens into an action chunk.

the explicit motion information captured in keypoint trajectories. We propose AMPLIFY: Actionless Motion Priors for Learning Inverse and Forward Dynamics, a three-stage framework that flexibly decouples dynamics modeling from policy learning. First, we learn a compact latent space for modeling the motion of a dense grid of keypoints. Second, we train a latent dynamics model to predict a sequence of latent motions based on the current observation. Finally, an inverse dynamics model learns to map predicted latent motions to low-level robot actions for execution. Notably, this modular approach allows the first two stages to be trained on *any* video data, while the inverse dynamics policy can be trained on *any* interaction data (Figure 1). We show that this has profound implications for policy generalization in Section 3.2.

Through extensive real-world and simulated experiments, we evaluate both the accuracy and downstream utility of our latent dynamics model. Compared to state-of-the-art baselines, we observe that AMPLIFY leads to improved keypoint trajectory prediction, lowering mean-squared error by over $3\times$. We then demonstrate that these predictions are useful for control; conditioning the inverse dynamics policy on latent motions is a valuable prior that allows for more data-efficient learning and generalization to tasks for which we have *no action-labeled data*. Finally, we examine the versatility of our motion-based representations beyond control for tasks such as conditional video prediction.

In summary, we make the following *key contributions*:

1. We present the first *latent* keypoint dynamics model and investigate crucial design choices.
2. We demonstrate state-of-the-art keypoint prediction accuracy on three large-scale video datasets.
3. We train a data-efficient and generalizable policy that can learn from action-free human data.
4. We apply latent motions to conditional video generation, outperforming previous baselines.

2 AMPLIFY: Method

Problem Setup – We assume access to two types of data: a video dataset $\mathcal{V} = \{(o_t, g)\}$ and a dataset of robot interaction data $\mathcal{R} = \{(o_t, q_t, a_t)\}$ where $o \in \mathcal{O}$ are RGB image observations, $g \in \mathcal{G}$ is a goal (e.g., a language description), and $a \in \mathcal{A}, q \in \mathcal{Q}$ are the action and proprioceptive state of the robot, respectively[1]. Given these datasets, our aim is to learn the parameters of a visual

[1] $\mathcal{V}$ and $\mathcal{R}$ need not be disjoint in general, and any goal-directed interaction data (demonstrations) may be included in both $\mathcal{V}$ and $\mathcal{R}$. However, $\mathcal{V}$ may additionally contain non-robot videos and $\mathcal{R}$ may contain undirected action data such as exploration or play data.

control policy $\pi : \mathcal{O} \times \mathcal{Q} \times \mathcal{G} \to \mathcal{P}(\mathcal{A}) = f_{\text{inv}}(o_t, q_t, f(o_t, g))$ composed of a forward dynamics model $f : \mathcal{O} \times \mathcal{G} \to \mathcal{Z}$ that learns a *motion prior* in a latent space $\mathcal{Z}$ and an inverse dynamics model $f_{\text{inv}} : \mathcal{O} \times \mathcal{Q} \times \mathcal{Z} \to \mathcal{A}$ that maps the latent motion to a sequence of actions. Crucially, this decomposition allows for independent scaling of f and f_{inv} by training on $\mathcal{V}$ and $\mathcal{R}$, respectively. We provide an extended discussion of the benefits of this decomposition in Appendix B. The following sections detail preprocessing (Sec. 2.1), learning the latent motion representation (Sec. 2.2), and training the forward (Sec. 2.3) and inverse (Sec. 2.4) dynamics models.

2.1 Preprocessing Keypoint Tracks

We first augment $\mathcal{V} \to \mathcal{V}' = \{(o_t, \kappa_t, g)\}$ in a preprocessing step using the off-the-shelf point tracking model from [36] to obtain a set of keypoint tracks $\kappa_t \in \mathbb{R}^{T \times N \times 2}$ for each timestep t. More precisely, we initialize a 20×20 uniform grid of $N = 400$ points in each image o_t, then track the points through the next $T = 16$ frames $o_{t:t+T}$, capturing their 2-dimensional pixel coordinates. Although extracting specific task-relevant keypoints could potentially yield more informative predictions, we favor the uniform grid for its simplicity and generality, similar to [53], and find that it works effectively to model a variety of motions. Other works have attempted to select key points according to heuristics such as movement throughout the video [54], but we found that this led the model to learn spurious correlations and relies on unrealistic assumptions at test time. By reinitializing the grid of keypoints in each frame, we ensure no points are occluded and guarantee consistent coverage throughout every frame, even with moving cameras. See Appendix D.4 for further details on preprocessing.

2.2 Motion Tokenization

Unlike prior keypoint-based methods which predict directly in pixel space [54, 53, 48, 51], we argue that learning to predict dynamics in a compressed latent space enables a more efficient and generalizable representation, similar to findings in model-based reinforcement learning [55, 56, 57]. To this end, we learn a compact discrete latent space from pre-processed keypoint trajectories using Finite Scalar Quantization (FSQ) [58], a drop-in replacement for vector-quantized variational autoencoders (VQ-VAEs) [59]. FSQ employs an implicit codebook and a single reconstruction loss term, avoiding representation collapse and resulting in better codebook utilization.

Figure 2a illustrates our tokenization scheme. We compute single-step velocities $u_t \in \mathbb{R}^{(T-1) \times N \times 2}$ from the pre-processed keypoint trajectories κ_t. Then, a keypoint encoder $\mathcal{E}_\theta : \mathbb{R}^{(T-1) \times N \times 2} \to \mathbb{R}^{b \times d}$ maps u_t to a d-length sequence $\tilde{z}_t$ of latent vectors $\tilde{z}_{t,i} \in \mathbb{R}^b$, which are quantized via FSQ to a sequence $z_t \in \mathbb{Z}^{b \times d}$ of discrete codes, and decoded by the keypoint decoder $\mathcal{D}_\theta : \mathbb{R}^{b \times d} \to \mathbb{R}^{(T-1) \times N \times W^2}$ for reconstruction. Rather than just predicting the 2-dimensional pixel coordinate of each point directly, the decoder outputs a categorical distribution over W^2 classes representing a local $W \times W$ window of motions centered at the same point in the previous timestep. This imposes an inductive bias on the model toward next-keypoint predictions that are close to locations in the current timestep, and additionally better captures multimodal distributions compared to performing regression on the coordinates. The keypoint encoder has a causally-masked transformer encoder architecture, and the keypoint decoder is an unmasked transformer decoder that cross-attends between a sequence of N learned positional encodings and the quantized codes from the encoder. The encoder and decoder are jointly trained on $\mathcal{V}$ using a cross-entropy loss:

$$\mathcal{L}_{AE}(\theta) = \text{CE}\Big(\mathcal{D}_\theta\Big(h(\mathcal{E}_\theta(u_t))\Big), \omega_t\Big) \tag{1}$$

where $\omega_t = \Omega(u_t)$, $\Omega : \mathbb{R}^{(T-1) \times N \times 2} \to \mathbb{R}^{(T-1) \times N \times W^2}$ maps ground-truth velocity vectors to their corresponding class based on the displacement in the local $W \times W$ window, and h is the FSQ discretization function. When available, multi-view inputs are tokenized together into a single sequence of codes. For simplicity, we do not include the view dimension in our notation. For ablations and an extended discussion on the effects of these design choices, we refer readers to Appendix E.

2.3 Forward Dynamics (Actionless Motion Prior)

After training the motion tokenizer, we train an autoregressive transformer $f(o_t, g)$ to predict the tokenized motion sequence z_t corresponding to the video $o_{t:t+T}$ based on the current observation and task description. Image observations are encoded and projected into the embedding space of the

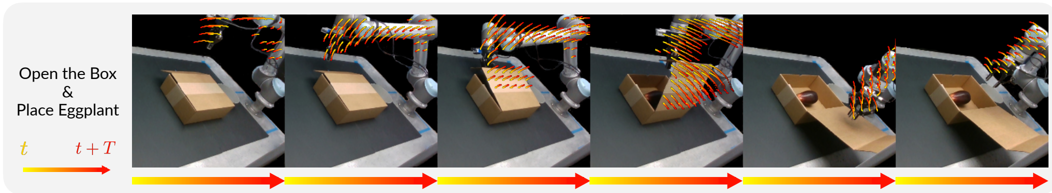


Figure 3: Decoded keypoint trajectory predictions from AMPLIFY. Zero-movement points are not shown.

transformer using the flattened feature map from a pre-trained ResNet-18 [60] to generate $7 \times 7 = 49$ vision tokens per image. The summary token from a T5 [61] text embedding of the task description is used to tokenize language inputs. These conditioning tokens are then concatenated with a start of sequence (SOS) token and the latent motion tokens to predict the next tokens in the sequence (Figure 2b). A block-causal attention mask is used, where the conditioning part of the sequence is non-causal and the motion tokens are causally masked. We use a cross-entropy loss on the predicted codes without decoding to full keypoint trajectories, and only back-propagate gradients to the dynamics model while the tokenizer remains frozen (Equation 2). $\texttt{sg}$ refers to the stop-gradient operator.

$$\mathcal{L}_{\text{forward}} = \text{CE}\Big(f(o_t, g), \texttt{sg}(\mathcal{E}_\theta(u_t))\Big) \tag{2}$$

2.4 Inverse Dynamics

Finally, we learn an inverse dynamics model $f_{\text{inv}}(o_t, q_t, z_t)$ that decodes latent motion tokens into a distribution over action chunks $\boldsymbol{a}_t = a_{t:t+T}$, as shown in Figure 2c. Importantly, this module is not conditioned on the goal and instead acts as a general reference follower trained on any interaction data $\mathcal{R}$. The model uses a transformer decoder with a sequence of learned tokens that cross-attend to image tokens, a linear projection of proprioceptive state, and codes from the motion tokenizer to produce a sequence of d action tokens. These action tokens are fed into an action head to output a distribution over length-T action chunks. Following BAKU [62], we opt for an isotropic Gaussian prior on the action distribution. In Appendix E, we discuss alternative choices for the action head. The inverse dynamics model is trained with a negative log-likelihood (NLL) loss with a temporal discount γ to reduce the impact of inaccurate predictions towards the end of the sequence.

$$\mathcal{L}_{\text{inv}} = -\sum_{\tau=t}^{t+T-1} \gamma^{\tau-t} \cdot \log p\left(a_\tau \mid \mu_{\tau-t}, \sigma_{\tau-t}\right) \tag{3}$$

where $\mu_{\tau-t} = f^{\mu}_{\text{inv}}(o_t, q_t, z_t)[\tau - t]$ and $\sigma_{\tau-t} = \exp(f^{\sigma}_{\text{inv}}(o_t, q_t, z_t)[\tau - t])$ are the predicted mean and standard deviation. The inverse dynamics model can be trained on ground truth tokens $z_t = \mathcal{E}_\theta(u_t)$, but in practice, we fine-tune the action decoder on the predicted outputs $\hat{z}_t$ of the forward dynamics model. Both the motion tokenizer and the forward dynamics model are frozen for this stage. The keypoint decoder $\mathcal{D}_\theta$ is not used, as we condition f_{inv} on latent motions rather than decoded tracks.

2.5 Inference

During inference, the forward dynamics model takes the current observation and task description at each timestep t and autoregressively predicts a sequence of latent motion tokens $\hat{z}_t = f(o_t, g)$. The inverse dynamics model then decodes these tokens, along with image and proprioception tokens, into an action chunk $\boldsymbol{a}_t = f_{\text{inv}}(o_t, q_t, \hat{z}_t)$. Following ACT [63], we use temporal ensembling to aggregate information over previously predicted action chunks using the same temporal discount γ.

3 Experiments

We evaluate AMPLIFY guided by two main axes of investigation: **quality** of dynamics prediction (Sec. 3.1) and **utility** of predictions for downstream tasks, including policy learning (Sec. 3.2) and conditional video generation (Sec. 3.3). See Appendix D for extended details on all experiments.

3.1 Quality of Forward Dynamics Prediction

We test the prediction accuracy of our forward dynamics model on a combination of three simulated and real-world video datasets, including both human and robot data: BridgeData v2 [64], a large-scale robot dataset consisting of over 60k real-world rollouts of diverse manipulation tasks in 24 different

Policy Learning Experiment	**Forward Dynamics**	**Inverse Dynamics**	**BC Baselines**
In-Distribution	$\mathcal{V}_{\text{id}}^R$	$\mathcal{R}_{\text{id}}$	$\mathcal{R}_{\text{id}}$
Few-Shot	$\mathcal{V}_{\text{id}}^R$	$\subseteq \mathcal{R}_{\text{id}}$	$\subseteq \mathcal{R}_{\text{id}}$
Cross-Embodiment	$\mathcal{V}_{\text{id}}^R \cup \mathcal{V}_{\text{id}}^H$	$\mathcal{R}_{\text{id}}$	$\mathcal{R}_{\text{id}}$
Generalization	$\mathcal{V}_{\text{id}}^R \cup \mathcal{V}_{\text{ood}}^R$	$\mathcal{R}_{\text{ood}}$	$\mathcal{R}_{\text{ood}}$

Table 1: Training dataset setup for each component by experiment. Subscript `id` and `ood` indicate in-distribution and out of distribution tasks and superscript H and R distinguish human and robot video data. $\subseteq$ indicates training on limited subsets of the data.

Table 2: Prediction. AMPLIFY achieves 3.7× better MSE and 2.5× better pixel accuracy compared to ATM, and a 4-6% improvement over Track2Act, which uses a goal image, and Seer, which requires full video prediction.

Method	LIBERO			BridgeDatav2	Something-Something v2
Metric	MSE ↓	Δ_{AUC} ↑	Pixel Acc. ↑	Δ_{AUC} ↑	Δ_{AUC} ↑
ATM [54]	0.022	0.767	0.250	–	–
Track2Act [53]	–	–	–	0.770	0.700
Seer [67]	–	–	–	0.914	–
AMPLIFY	**0.006**	**0.913**	**0.629**	**0.968**	**0.725**

Table 3: Behavior Cloning performance on LIBERO. AMPLIFY is competitive with various state-of-the-art baselines, both with and without video pretraining.

Method	**Video Pre-training**	**LIBERO Long**	**LIBERO 90**	**LIBERO Object**	**LIBERO Spatial**	**LIBERO Goal**
Diffusion Policy [68]	✗	0.73	0.67	**0.70**	0.79	0.83
QueST [69]	✗	0.67	0.89	–	–	–
BAKU [62]	✗	**0.86**	**0.90**	–	–	–
AMPLIFY (Inverse only)	✗	0.76	0.83	0.64	**0.83**	**0.92**
UniPi [16]	✓	0.06	–	0.60	0.69	0.12
ATM [54]	✓	0.44	0.63	0.81	**0.79**	0.59
AMPLIFY (Full)	✓	**0.75**	**0.88**	**0.93**	0.73	**0.92**

environments; Something-Something v2 [65], a video dataset consisting of over 220,000 videos of humans performing everyday manipulation tasks with a variety of objects and primitive motion categories; and LIBERO [66], a benchmark of 130 diverse simulated robotic manipulation tasks, from which we use the observations from 6500 demonstration rollouts as a video dataset.

We compare to ATM [54] and Track2Act [53], two state-of-the-art keypoint trajectory prediction approaches. In addition, on BridgeData v2 we compare track prediction accuracy to a baseline of first predicting videos with Seer [67], then applying CoTracker [36] to the initial set of points and tracking through the generated videos. Since our forward dynamics model predicts in latent space, we use the decoder from the Motion Tokenization stage for fair comparison in pixel space. We measure performance on normalized tracks ($\kappa \in [-1, 1]$) using Mean Squared Error (MSE), Pixel-Wise Accuracy (Pixel Acc.), and a metric Δ_{AUC} originally used by point tracking methods [38, 36], and later used for track point prediction by Track2Act. See Appendix D.3 for further details on metrics.

Results are summarized in Table 2, demonstrating that AMPLIFY consistently leads to more accurate predictions, even though the forward dynamics model is only trained on a latent consistency loss rather than pixel-space prediction objectives. On the LIBERO dataset, we achieve over twice the pixel-wise accuracy of ATM, and we outperform Track2Act (which, unlike our method, has access to goal images) on their chosen Δ_{AUC} metric across BridgeData v2 and Something-Something v2. We attribute this success to several design choices, including the compression of motion into a compact latent space, thus improving efficiency and generalization; the prediction of discrete tokens to leverage the expressive power of autoregressive transformers; and the use of local-window pixel space classification, which gives our forward dynamics model the ability to model rich multi-modal distributions of motion and capture fine-grained dynamics. Further investigation into design choices (E), detailed results (F.2), and qualitative visualizations (F.3) can be found in the Appendix.

3.2 Utility of Predicted Latent Motions for Policy Learning

Beyond prediction accuracy, we examine whether video pre-training using AMPLIFY can provide a useful prior for policy learning in both real-world and simulated experiments. Specifically, we

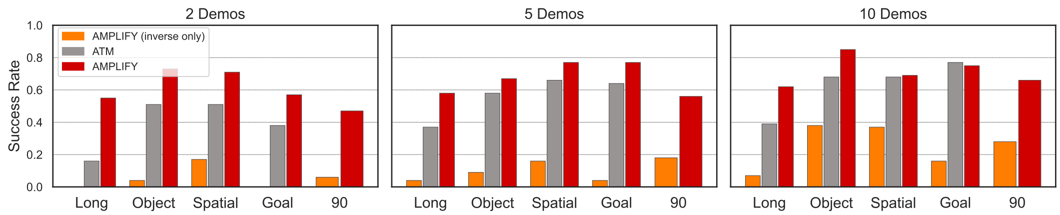


Figure 4: LIBERO few-shot. Comparison of AMPLIFY against ATM [54] and a no-video-pre-training baseline. Our forward model is trained on all videos, and the inverse model is only trained on a limited number of demos.

evaluate AMPLIFY along four dimensions measuring (1) in-distribution performance, (2) few-shot learning, (3) cross-embodiment transfer, and (4) generalization. Table 1 summarizes the training datasets for different stages under each experimental setup. We evaluate performance using success rates on all five subsets of LIBERO, as well as a set of 3 real-world tasks: "Put the Rubik's Cube on the Box" (`Place Cube`), "Stack the Green and Blue Cups in the Orange Cup" (`Stack Cups`), and "Open the Box and Move the Eggplant into the Bowl" (`Open Box & Place Eggplant`).

In-Distribution Performance – We first evaluate AMPLIFY in a standard behavior cloning setup, training both the forward and inverse dynamics models on only the demonstration data. We compare to state-of-the-art approaches with and without video pre-training. Results in Table 3 indicate that AMPLIFY, even without additional data, is competitive with SOTA behavior cloning methods and outperforms video pre-training methods trained with (ATM) and without (UniPi) keypoint tracks. In this setting, we observe that since there is sufficient information to learn tasks to a high degree *without* video pre-training, standard BC methods tend to match or outperform approaches using pre-training. However, in subsequent sections, we demonstrate that these approaches under-perform in limited data regimes and do not generalize effectively to new tasks.

Few-Shot Learning – We study whether AMPLIFY can learn from fewer action-labeled demonstrations by training the forward model on all videos, while the inverse model is only trained on 4%, 10%, or 20% of the 50 demonstrations available for each of the subsets of LIBERO. In Figure 4, we compare AMPLIFY with ATM, trained on all videos and the same subsets of action data, as well as a variant of AMPLIFY that does not condition on motion tokens to predict actions. Both AMPLIFY and ATM consistently outperform the no-pre-training variant, indicating that in low-data regimes, video pre-training on keypoint dynamics provides a strong prior for data-efficient policy learning. In addition, AMPLIFY achieves stronger performance than ATM on nearly every subset, suggesting that a latent motion representation has higher utility for action prediction than conditioning the policy directly on pixel-space track predictions. This seems to be especially true at the extreme low end–when provided with only 2 demonstrations per task, AMPLIFY achieves an average 1.94$\times$ improvement over ATM. Full numerical results are included in Table 16.

Cross-Embodiment Transfer – Since the forward dynamics model can be trained on any observation data, we study whether videos of humans demonstrating a task can be used to improve policy learning. We train the forward dynamics model on both human and robot video data, while the inverse dynamics model is trained only on the action-labeled robot data. This setup highlights how the two stages can be decoupled to scale independently, unlike BC methods that cannot effectively harness action-free data. We evaluate success rates on three real-world tasks of varying difficulty, using Diffusion Policy as the BC baseline. For fair comparison, we replace the Gaussian head used in other experiments with a Diffusion Policy head in the inverse dynamics model. This ensures that the only difference between the two approaches is whether the predictions from our forward dynamics model are used to condition the policy. Similarly to the previous section, we evaluate AMPLIFY in both the few-shot setting and the full demonstration setting. Results in Table 4 demonstrate that AMPLIFY can effectively leverage

Method	Place Cube			Stack Cups			Box/Eggplant			Avg.
# Demos	5	10	All	5	10	All	5	10	All	
Diffusion Policy [68]	0.6	0.5	**0.9**	**0.3**	0.5	0.5	**0.1**	0.2	0.2	0.42
AMPLIFY (DP head)	**0.7**	**0.9**	**0.9**	**0.3**	**0.6**	**1.0**	**0.1**	**0.3**	**0.4**	**0.58**

Table 4: Cross-Embodiment Transfer. By leveraging human video demonstrations to train the forward dynamics model, AMPLIFY outperforms Diffusion Policy on real-world tasks.

Method	LIBERO Long	LIBERO Object	LIBERO Spatial	LIBERO Goal
Diffusion Policy [68]	0.00	0.00	0.00	0.00
QueST [69]	0.07	0.00	0.01	0.01
BAKU [62]	0.06	0.00	0.00	0.00
AMPLIFY (w/o tracks)	0.00	0.00	0.00	0.02
AMPLIFY	**0.52**	**0.80**	**0.69**	**0.41**

Table 5: Zero-shot task generalization from LIBERO 90 to unseen LIBERO subsets. We are the first to report non-trivial success on LIBERO without using *any* action data from the target tasks. Compared to the best BC baseline, AMPLIFY provides a $27\times$ average improvement.

additional human data to learn common dynamics between human and robot motions, and use the predicted latent motions to improve policy learning. The average improvements of $1.32\times$, $1.4\times$, and $1.5\times$ indicate a more prominent gap as task complexity increases. See Table 17 for complete results.

Generalization – Observing that AMPLIFY excels in learning from *limited* action data, we now turn to a setting where *no* action data is available for target tasks. Given *only observations* of target tasks, as well as a dataset of out-of-distribution interaction data, we evaluate how well AMPLIFY can solve the target tasks zero-shot. This challenging setting requires methods to both learn a good abstraction of the mapping from observations to actions, and also generalize that abstraction to predict correct actions on new tasks. To test this setting, we train the forward dynamics model on observations from all subsets of LIBERO, and train the inverse dynamics model and BC baselines on actions from *only* LIBERO 90. We then evaluate on four LIBERO target suites (Long, Object, Spatial, Goal), specifically designed to test different categories of generalization [66]. We find that BC methods completely fail in this scenario, achieving near-zero success rates (Table 5). We attribute this failure to two main shortcomings of BC: (1) the supervised imitation objective has no incentive to learn a generalizable abstraction, and (2) BC has no mechanism for harnessing additional data that may be informative, such as videos. In contrast, AMPLIFY attains an average 60.5% success rate on target tasks, approaching the success rates of models that were directly trained on the target tasks. This success highlights the value of latent dynamics prediction as a versatile interface for learning general priors from action-free videos. In addition, it suggests that training a general reference following inverse dynamics model may be a more generalizable objective compared to imitation learning.

3.3 Utility of Predicted Latent Motions for Conditional Video Generation

To demonstrate the utility of predicting keypoint trajectories beyond robotic control, we condition a video prediction model [44] on the latent motion tokens predicted by our forward dynamics model. We find that conditioning a video prediction model on our latent motion tokens leads to improved generation quality (Table 6). Compared to a baseline model that does not use track inputs, our approach yields better performance on all metrics (details on metrics in Appendix D.3). This improvement suggests that our latent motion representation captures rich, structured dynamics that improve not only control tasks but also the fidelity of generated video content. Further details on training and generation are provided in Appendix D.5 and qualitative results in Appendix F.3.

Method	PSNR ↑	LPIPS ↑	SSIM ↑
AVDC [44]	15.93	0.16	0.56
AVDC + AMPLIFY	**16.40**	**0.19**	**0.59**

Table 6: Video Prediction. Conditioning AVDC on predicted motion tokens from our dynamics model improves generated video quality on BridgeData v2.

4 Conclusion

In this work, we introduced AMPLIFY, a framework that leverages large-scale action-free video data and a small amount of interaction data to significantly enhance robotic policy performance. By decoupling the learning of *what* constitutes a task from *how* to execute it, our approach efficiently utilizes heterogeneous data sources. Our key insight lies in representing scene dynamics through compact latent motion tokens derived from keypoint trajectories, which enables higher efficiency and improved performance compared to pixel-level reconstruction methods. Experimental results show that AMPLIFY consistently outperforms baseline methods, particularly in the limited action data regime and in zero-shot generalization settings. Moreover, the versatility of our latent representation extends beyond control, proving useful in tasks such as conditional video prediction. Our findings demonstrate the promise of harnessing large-scale human video data to inform robotic control policies and pave the way for more scalable, generalizable, and efficient robot learning.

5 Limitations

While AMPLIFY is an exciting step towards robot learning from broad data sources, we recognize a number of limitations that could serve as promising directions for future research. First, by modeling tracks in 2D images, we are potentially leaving ambiguity in the inverse dynamics model if multiple actions could correspond to the same tracks. An explicitly 3D approach, predicting latent motions that correspond to 3D tracks [39, 70] could yield more robust representations that do not depend on fixed or known camera views. In addition, AMPLIFY currently only considers deterministic environment dynamics, since in stochastic settings additional information is required to separate agent actions from exogenous noise in purely state-to-state data [71, 72, 73]. Since AMPLIFY has demonstrated the ability to learn from off-task data, it would also be interesting to explore whether the inverse dynamics model could be trained on data collected online by an exploration policy. Finally, future research could scale the prediction backbone to a general VLM or video prediction model to enhance video and language generalization.

References

[1] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners. 2019.

[2] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.

[3] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[4] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever. Learning transferable visual models from natural language supervision, 2021. URL https://arxiv.org/abs/2103.00020.

[5] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen. Hierarchical text-conditional image generation with clip latents, 2022. URL https://arxiv.org/abs/2204.06125.

[6] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models, 2022. URL https://arxiv.org/abs/2112.10752.

[7] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, T. Jackson, S. Jesmonth, N. J. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, K.-H. Lee, S. Levine, Y. Lu, U. Malla, D. Manjunath, I. Mordatch, O. Nachum, C. Parada, J. Peralta, E. Perez, K. Pertsch, J. Quiambao, K. Rao, M. Ryoo, G. Salazar, P. Sanketi, K. Sayed, J. Singh, S. Sontakke, A. Stone, C. Tan, H. Tran, V. Vanhoucke, S. Vega, Q. Vuong, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, and B. Zitkovich. RT-1: ROBOTICS TRANSFORMER FOR REAL-WORLD CONTROL AT SCALE.

[8] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

[9] A. Padalkar, A. Pooley, A. Jain, A. Bewley, A. Herzog, A. Irpan, A. Khazatsky, A. Rai, A. Singh, A. Brohan, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023.

[10] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, et al. pi0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[11] K. Pertsch, K. Stachowicz, B. Ichter, D. Driess, S. Nair, Q. Vuong, O. Mees, C. Finn, and S. Levine. Fast: Efficient action tokenization for vision-language-action models. *arXiv preprint arXiv:2501.09747*, 2025.

[12] L. X. Shi, B. Ichter, M. Equi, L. Ke, K. Pertsch, Q. Vuong, J. Tanner, A. Walling, H. Wang, N. Fusai, et al. Hi robot: Open-ended instruction following with hierarchical vision-language-action models. *arXiv preprint arXiv:2502.19417*, 2025.

[13] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen, K. Ellis, P. D. Fagan, J. Hejna, M. Itkina, M. Lepert, Y. J. Ma, P. T. Miller, J. Wu, S. Belkhale, S. Dass, H. Ha, A. Jain, A. Lee, Y. Lee, M. Memmel, S. Park, I. Radosavovic, K. Wang, A. Zhan, K. Black, C. Chi, K. B. Hatch, S. Lin, J. Lu, J. Mercat, A. Rehman, P. R. Sanketi, A. Sharma, C. Simpson, Q. Vuong, H. R. Walke, B. Wulfe, T. Xiao, J. H. Yang, A. Yavary, T. Z. Zhao, C. Agia, R. Baijal, M. G. Castro, D. Chen, Q. Chen, T. Chung, J. Drake, E. P. Foster, J. Gao, V. Guizilini, D. A. Herrera, M. Heo, K. Hsu, J. Hu, M. Z. Irshad, D. Jackson, C. Le, Y. Li, K. Lin, R. Lin, Z. Ma, A. Maddukuri, S. Mirchandani, D. Morton, T. Nguyen, A. O'Neill, R. Scalise, D. Seale, V. Son, S. Tian, E. Tran, A. E. Wang, Y. Wu, A. Xie, J. Yang, P. Yin, Y. Zhang, O. Bastani, G. Berseth, J. Bohg, K. Goldberg, A. Gupta, A. Gupta, D. Jayaraman, J. J. Lim, J. Malik, R. Martín-Martín, S. Ramamoorthy, D. Sadigh, S. Song, J. Wu, M. C. Yip, Y. Zhu, T. Kollar, S. Levine, and C. Finn. Droid: A large-scale in-the-wild robot manipulation dataset. 2024.

[14] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, Q. Vuong, T. Kollar, B. Burchfiel, R. Tedrake, D. Sadigh, S. Levine, P. Liang, and C. Finn. OpenVLA: An Open-Source Vision-Language-Action Model, June 2024. URL http://arxiv.org/abs/2406.09246. arXiv:2406.09246 [cs].

[15] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, L. X. Shi, J. Tanner, Q. Vuong, A. Walling, H. Wang, and U. Zhilinsky. $_0$: A Vision-Language-Action Flow Model for General Robot Control, Nov. 2024. URL http://arxiv.org/abs/2410.24164. arXiv:2410.24164.

[16] Y. Du, M. Yang, B. Dai, H. Dai, O. Nachum, J. B. Tenenbaum, D. Schuurmans, and P. Abbeel. Learning Universal Policies via Text-Guided Video Generation, Nov. 2023. URL http://arxiv.org/abs/2302.00111. arXiv:2302.00111 [cs].

[17] R. McCarthy, D. C. H. Tan, D. Schmidt, F. Acero, N. Herr, Y. Du, T. G. Thuruthel, and Z. Li. Towards Generalist Robot Learning from Internet Video: A Survey, June 2024. URL http://arxiv.org/abs/2404.19664. arXiv:2404.19664 [cs].

[18] O. Rybkin, K. Pertsch, K. G. Derpanis, K. Daniilidis, and A. Jaegle. Learning what you can do before doing anything, Feb. 2019. URL http://arxiv.org/abs/1806.09655. arXiv:1806.09655 [cs, stat].

[19] NVIDIA, N. Agarwal, A. Ali, M. Bala, Y. Balaji, E. Barker, T. Cai, P. Chattopadhyay, Y. Chen, Y. Cui, Y. Ding, D. Dworakowski, J. Fan, M. Fenzi, F. Ferroni, S. Fidler, D. Fox, S. Ge, Y. Ge, J. Gu, S. Gururani, E. He, J. Huang, J. Huffman, P. Jannaty, J. Jin, S. W. Kim, G. Klár, G. Lam, S. Lan, L. Leal-Taixe, A. Li, Z. Li, C.-H. Lin, T.-Y. Lin, H. Ling, M.-Y. Liu, X. Liu, A. Luo, Q. Ma, H. Mao, K. Mo, A. Mousavian, S. Nah, S. Niverty, D. Page, D. Paschalidou,

Z. Patel, L. Pavao, M. Ramezanali, F. Reda, X. Ren, V. R. N. Sabavat, E. Schmerling, S. Shi, B. Stefaniak, S. Tang, L. Tchapmi, P. Tredak, W.-C. Tseng, J. Varghese, H. Wang, H. Wang, H. Wang, T.-C. Wang, F. Wei, X. Wei, J. Z. Wu, J. Xu, W. Yang, L. Yen-Chen, X. Zeng, Y. Zeng, J. Zhang, Q. Zhang, Y. Zhang, Q. Zhao, and A. Zolkowski. Cosmos World Foundation Model Platform for Physical AI, Mar. 2025. URL http://arxiv.org/abs/2501.03575. arXiv:2501.03575 [cs].

[20] J. Ho, W. Chan, C. Saharia, J. Whang, R. Gao, A. Gritsenko, D. P. Kingma, B. Poole, M. Norouzi, D. J. Fleet, and T. Salimans. Imagen video: High definition video generation with diffusion models, 2022. URL https://arxiv.org/abs/2210.02303.

[21] W. Kong, Q. Tian, Z. Zhang, R. Min, Z. Dai, J. Zhou, J. Xiong, X. Li, B. Wu, J. Zhang, et al. Hunyuanvideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024.

[22] Veo-Team, :, A. Gupta, A. Razavi, A. Toor, A. Gupta, D. Erhan, E. Shaw, E. Lau, F. Belletti, G. Barth-Maron, G. Shaw, H. Erdogan, H. Sidahmed, H. Nandwani, H. Moraldo, H. Kim, I. Blok, J. Donahue, J. Lezama, K. Mathewson, K. David, M. K. Lorrain, M. van Zee, M. Narasimhan, M. Wang, M. Babaeizadeh, N. Papalampidi, N. Pezzotti, N. Jha, P. Barnes, P.-J. Kindermans, R. Hornung, R. Villegas, R. Poplin, S. Zaiem, S. Dieleman, S. Ebrahimi, S. Wisdom, S. Zhang, S. Fruchter, S. Nørly, W. Hua, X. Yan, Y. Du, and Y. Chen. Veo 2. 2024. URL https://deepmind.google/technologies/veo/veo-2/.

[23] O. Bar-Tal, H. Chefer, O. Tov, C. Herrmann, R. Paiss, S. Zada, A. Ephrat, J. Hur, G. Liu, A. Raj, et al. Lumiere: A space-time diffusion model for video generation. In *SIGGRAPH Asia 2024 Conference Papers*, pages 1–11, 2024.

[24] Y. J. Ma, S. Sodhani, D. Jayaraman, O. Bastani, V. Kumar, and A. Zhang. Vip: Towards universal visual reward and representation via value-implicit pre-training. *arXiv preprint arXiv:2210.00030*, 2022.

[25] D. Ghosh, C. Bhateja, and S. Levine. Reinforcement Learning from Passive Data via Latent Intentions, Apr. 2023. URL http://arxiv.org/abs/2304.04782. arXiv:2304.04782 [cs, stat].

[26] C. Bhateja, D. Guo, D. Ghosh, A. Singh, M. Tomar, Q. Vuong, Y. Chebotar, S. Levine, and A. Kumar. Robotic Offline RL from Internet Videos via Value-Function Pre-Training, Sept. 2023. URL http://arxiv.org/abs/2309.13041. arXiv:2309.13041 [cs].

[27] N. Dashora, D. Ghosh, and S. Levine. Viva: Video-trained value functions for guiding online rl from diverse data. *arXiv preprint arXiv:2503.18210*, 2025.

[28] P. Sermanet, C. Lynch, Y. Chebotar, J. Hsu, E. Jang, S. Schaal, S. Levine, and G. Brain. Time-contrastive networks: Self-supervised learning from video. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 1134–1141. IEEE, 2018.

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

[30] T. M. Moerland, J. Broekens, A. Plaat, C. M. Jonker, et al. Model-based reinforcement learning: A survey. *Foundations and Trends® in Machine Learning*, 16(1):1–118, 2023.

[31] Y. Hu, Y. Guo, P. Wang, X. Chen, Y.-J. Wang, J. Zhang, K. Sreenath, C. Lu, and J. Chen. Video prediction policy: A generalist robot policy with predictive visual representations. *arXiv preprint arXiv:2412.14803*, 2024.

[32] J. Bruce, M. D. Dennis, A. Edwards, J. Parker-Holder, Y. Shi, E. Hughes, M. Lai, A. Mavalankar, R. Steigerwald, C. Apps, et al. Genie: Generative interactive environments. In *Forty-first International Conference on Machine Learning*, 2024.

[33] S. Ye, J. Jang, B. Jeon, S. Joo, J. Yang, B. Peng, A. Mandlekar, R. Tan, Y.-W. Chao, B. Y. Lin, et al. Latent action pretraining from videos. *arXiv preprint arXiv:2410.11758*, 2024.

[34] Y. Chen, Y. Ge, Y. Li, Y. Ge, M. Ding, Y. Shan, and X. Liu. Moto: Latent Motion Token as the Bridging Language for Robot Manipulation, Dec. 2024. URL http://arxiv.org/abs/2412.04445. arXiv:2412.04445 [cs].

[35] Z. Cui, H. Pan, A. Iyer, S. Haldar, and L. Pinto. Dynamo: In-domain dynamics pretraining for visuo-motor control. *Advances in Neural Information Processing Systems*, 37:33933–33961, 2024.

[36] N. Karaev, I. Rocco, B. Graham, N. Neverova, A. Vedaldi, and C. Rupprecht. CoTracker: It is better to track together. 2023.

[37] Q. Wang, Y.-Y. Chang, R. Cai, Z. Li, B. Hariharan, A. Holynski, and N. Snavely. Tracking everything everywhere all at once. In *International Conference on Computer Vision*, 2023.

[38] C. Doersch, Y. Yang, M. Vecerik, D. Gokay, A. Gupta, Y. Aytar, J. Carreira, and A. Zisserman. Tapir: Tracking any point with per-frame initialization and temporal refinement. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10061–10072, 2023.

[39] Y. Xiao, Q. Wang, S. Zhang, N. Xue, S. Peng, Y. Shen, and X. Zhou. Spatialtracker: Tracking any 2d pixels in 3d space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[40] M. Vecerik, C. Doersch, Y. Yang, T. Davchev, Y. Aytar, G. Zhou, R. Hadsell, L. Agapito, and J. Scholz. RoboTAP: Tracking Arbitrary Points for Few-Shot Visual Imitation, Aug. 2023. URL http://arxiv.org/abs/2308.15975. arXiv:2308.15975 [cs].

[41] Z. Qin, K. Fang, Y. Zhu, L. Fei-Fei, and S. Savarese. Keto: Learning keypoint representations for tool manipulation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7278–7285. IEEE, 2020.

[42] C. Yuan, C. Wen, T. Zhang, and Y. Gao. General flow as foundation affordance for scalable robot learning. *arXiv preprint arXiv:2401.11439*, 2024.

[43] L.-H. Lin, Y. Cui, A. Xie, T. Hua, and D. Sadigh. FlowRetrieval: Flow-Guided Data Retrieval for Few-Shot Imitation Learning, Oct. 2024. URL http://arxiv.org/abs/2408.16944. arXiv:2408.16944.

[44] P.-C. Ko, J. Mao, Y. Du, S.-H. Sun, and J. B. Tenenbaum. Learning to Act from Actionless Videos through Dense Correspondences. *arXiv:2310.08576*, 2023.

[45] S. Kareer, D. Patel, R. Punamiya, P. Mathur, S. Cheng, C. Wang, J. Hoffman, and D. Xu. Egomimic: Scaling imitation learning via egocentric video, 2024. URL https://arxiv.org/abs/2410.24221.

[46] J. Gao, Z. Tao, N. Jaquier, and T. Asfour. K-VIL: Keypoints-Based Visual Imitation Learning. *IEEE Transactions on Robotics*, 39(5):3888–3908, Oct. 2023. ISSN 1941-0468. doi: 10.1109/TRO.2023.3286074. URL https://ieeexplore.ieee.org/abstract/document/10189175. Conference Name: IEEE Transactions on Robotics.

[47] X. Fang, B.-R. Huang, J. Mao, J. Shone, J. B. Tenenbaum, T. Lozano-Pérez, and L. P. Kaelbling. Keypoint Abstraction using Large Models for Object-Relative Imitation Learning, Oct. 2024. URL http://arxiv.org/abs/2410.23254. arXiv:2410.23254.

[48] C. Gao, H. Zhang, Z. Xu, C. Zhehao, and L. Shao. Flip: Flow-centric generative planning as general-purpose manipulation world model. In *The Thirteenth International Conference on Learning Representations*, 2025.

[49] L. Manuelli, W. Gao, P. Florence, and R. Tedrake. kpam: Keypoint affordances for category-level robotic manipulation. In *The International Symposium of Robotics Research*, pages 132–157. Springer, 2019.

[50] I. Guzey, Y. Dai, G. Savva, R. Bhirangi, and L. Pinto. Bridging the Human to Robot Dexterity Gap through Object-Oriented Rewards, Oct. 2024. URL http://arxiv.org/abs/2410.23289. arXiv:2410.23289.

[51] M. Xu, Z. Xu, Y. Xu, C. Chi, G. Wetzstein, M. Veloso, and S. Song. Flow as the Cross-Domain Manipulation Interface, July 2024. URL http://arxiv.org/abs/2407.15208. arXiv:2407.15208 [cs].

[52] H. Bharadhwaj, D. Dwibedi, A. Gupta, S. Tulsiani, C. Doersch, T. Xiao, D. Shah, F. Xia, D. Sadigh, and S. Kirmani. Gen2Act: Human Video Generation in Novel Scenarios enables Generalizable Robot Manipulation, Sept. 2024. URL https://arxiv.org/abs/2409.16283v1.

[53] H. Bharadhwaj, R. Mottaghi, A. Gupta, and S. Tulsiani. Track2act: Predicting point tracks from internet videos enables diverse zero-shot robot manipulation. *arXiv preprint arXiv:2405.01527*, 2024.

[54] C. Wen, X. Lin, J. So, K. Chen, Q. Dou, Y. Gao, and P. Abbeel. Any-point Trajectory Modeling for Policy Learning, Feb. 2024. URL http://arxiv.org/abs/2401.00025. arXiv:2401.00025 [cs].

[55] N. Hansen, X. Wang, and H. Su. Temporal difference learning for model predictive control. In *ICML*, 2022.

[56] N. Hansen, H. Su, and X. Wang. TD-MPC2: Scalable, Robust World Models for Continuous Control, Mar. 2024. URL http://arxiv.org/abs/2310.16828. arXiv:2310.16828 [cs].

[57] A. Scannell, M. Nakhaei, K. Kujanpää, Y. Zhao, K. S. Luck, A. Solin, and J. Pajarinen. Discrete codebook world models for continuous control. *arXiv preprint arXiv:2503.00653*, 2025.

[58] F. Mentzer, D. Minnen, E. Agustsson, and M. Tschannen. Finite Scalar Quantization: VQ-VAE Made Simple, Oct. 2023. URL http://arxiv.org/abs/2309.15505. arXiv:2309.15505 [cs].

[59] A. v. d. Oord, O. Vinyals, and K. Kavukcuoglu. Neural Discrete Representation Learning, May 2018. URL http://arxiv.org/abs/1711.00937. arXiv:1711.00937 [cs].

[60] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, June 2016. doi:10.1109/CVPR.2016.90. ISSN: 1063-6919.

[61] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

[62] S. Haldar, Z. Peng, and L. Pinto. Baku: An efficient transformer for multi-task policy learning. *arXiv preprint arXiv:2406.07539*, 2024.

[63] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware.

[64] H. R. Walke, K. Black, T. Z. Zhao, Q. Vuong, C. Zheng, P. Hansen-Estruch, A. W. He, V. Myers, M. J. Kim, M. Du, et al. Bridgedata v2: A dataset for robot learning at scale. In *Conference on Robot Learning*, pages 1723–1736. PMLR, 2023.

[65] R. Goyal, S. E. Kahou, V. Michalski, J. Materzyńska, S. Westphal, H. Kim, V. Haenel, I. Fruend, P. Yianilos, M. Mueller-Freitag, F. Hoppe, C. Thurau, I. Bax, and R. Memisevic. The "something something" video database for learning and evaluating visual common sense, 2017. URL https://arxiv.org/abs/1706.04261.

[66] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *arXiv preprint arXiv:2306.03310*, 2023.

[67] X. Gu, C. Wen, W. Ye, J. Song, and Y. Gao. Seer: Language instructed video prediction with latent diffusion models. *arXiv preprint arXiv:2303.14897*, 2023.

[68] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023.

[69] A. Mete, H. Xue, A. Wilcox, Y. Chen, and A. Garg. QueST: Self-Supervised Skill Abstractions for Learning Continuous Control, Sept. 2024. URL http://arxiv.org/abs/2407.15840. arXiv:2407.15840 [cs].

[70] T. D. Ngo, P. Zhuang, C. Gan, E. Kalogerakis, S. Tulyakov, H.-Y. Lee, and C. Wang. Delta: Dense efficient long-range 3d tracking for any video. *arXiv preprint arXiv:2410.24211*, 2024.

[71] D. Misra, A. Saran, T. Xie, A. Lamb, and J. Langford. Towards principled representation learning from videos for reinforcement learning. *arXiv preprint arXiv:2403.13765*, 2024.

[72] M. Yang, D. Schuurmans, P. Abbeel, and O. Nachum. Dichotomy of control: Separating what you can control from what you cannot. *arXiv preprint arXiv:2210.13435*, 2022.

[73] S. Park, D. Ghosh, B. Eysenbach, and S. Levine. Hiql: Offline goal-conditioned rl with latent states as actions. *Advances in Neural Information Processing Systems*, 36:34866–34891, 2023.

[74] C. Wang, L. Fan, J. Sun, R. Zhang, L. Fei-Fei, D. Xu, Y. Zhu, and A. Anandkumar. Mimicplay: Long-horizon imitation learning by watching human play. *arXiv preprint arXiv:2302.12422*, 2023.

[75] L. Chen, S. Bahl, and D. Pathak. PlayFusion: Skill Acquisition via Diffusion from Language-Annotated Play. In *Proceedings of The 7th Conference on Robot Learning*, pages 2012–2029. PMLR, Dec. 2023. URL https://proceedings.mlr.press/v229/chen23c.html. ISSN: 2640-3498.

[76] C. Lynch, M. Khansari, T. Xiao, V. Kumar, J. Tompson, S. Levine, and P. Sermanet. Learning Latent Plans from Play. In *Proceedings of the Conference on Robot Learning*, pages 1113–1132. PMLR, May 2020. URL https://proceedings.mlr.press/v100/lynch20a.html. ISSN: 2640-3498.

[77] H. Bharadhwaj, A. Gupta, S. Tulsiani, and V. Kumar. Zero-shot robot manipulation from passive human videos. *arXiv preprint arXiv:2302.02011*, 2023.

[78] H. Xiong, Q. Li, Y.-C. Chen, H. Bharadhwaj, S. Sinha, and A. Garg. Learning by watching: Physical imitation of manipulation skills from human videos. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7827–7834. IEEE, 2021.

[79] K. Shaw, S. Bahl, and D. Pathak. Videodex: Learning dexterity from internet videos. In *Conference on Robot Learning*, pages 654–665. PMLR, 2023.

[80] P. Mandikal and K. Grauman. Dexvip: Learning dexterous grasping with human hand pose priors from video. In *Conference on Robot Learning*, pages 651–661. PMLR, 2022.

[81] S. Bahl, A. Gupta, and D. Pathak. Human-to-robot imitation in the wild. *arXiv preprint arXiv:2207.09450*, 2022.

[82] S. Bahl, R. Mendonca, L. Chen, U. Jain, and D. Pathak. Affordances from human videos as a versatile representation for robotics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13778–13790, 2023.

[83] R. Mendonca, S. Bahl, and D. Pathak. Structured world models from human videos. *arXiv preprint arXiv:2308.10901*, 2023.

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

[85] K. Schmeckpeper, O. Rybkin, K. Daniilidis, S. Levine, and C. Finn. Reinforcement learning with videos: Combining offline observations with interaction. *arXiv preprint arXiv:2011.06507*, 2020.

[86] C. Bhateja, D. Guo, D. Ghosh, A. Singh, M. Tomar, Q. Vuong, Y. Chebotar, S. Levine, and A. Kumar. Robotic offline rl from internet videos via value-function pre-training. *arXiv preprint arXiv:2309.13041*, 2023.

[87] D. Ghosh, C. A. Bhateja, and S. Levine. Reinforcement learning from passive data via latent intentions. In *International Conference on Machine Learning*, pages 11321–11339. PMLR, 2023.

[88] J. Bruce, M. Dennis, A. Edwards, J. Parker-Holder, Y. Shi, E. Hughes, M. Lai, A. Mavalankar, R. Steigerwald, C. Apps, Y. Aytar, S. Bechtle, F. Behbahani, S. Chan, N. Heess, L. Gonzalez, S. Osindero, S. Ozair, S. Reed, J. Zhang, K. Zolna, J. Clune, N. de Freitas, S. Singh, and T. Rocktäschel. Genie: Generative Interactive Environments, Feb. 2024. URL http://arxiv.org/abs/2402.15391. arXiv:2402.15391 [cs].

[89] A. Escontrela, A. Adeniji, W. Yan, A. Jain, X. B. Peng, K. Goldberg, Y. Lee, D. Hafner, and P. Abbeel. Video prediction models as rewards for reinforcement learning. *arXiv preprint arXiv:2305.14343*, 2023.

[90] T. Huang, G. Jiang, Y. Ze, and H. Xu. Diffusion reward: Learning rewards via conditional video diffusion. *arXiv preprint arXiv:2312.14134*, 2023.

[91] D. Schmidt and M. Jiang. Learning to act without actions. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.

[92] Y. Wen, J. Lin, Y. Zhu, J. Han, H. Xu, S. Zhao, and X. Liang. Vidman: Exploiting implicit dynamics from video diffusion model for effective robot manipulation, 2024. URL https://arxiv.org/abs/2411.09153.

[93] J. Liang, R. Liu, E. Ozguroglu, S. Sudhakar, A. Dave, P. Tokmakov, S. Song, and C. Vondrick. Dreamitate: Real-world visuomotor policy learning via video generation, 2024. URL https://arxiv.org/abs/2406.16862.

[94] A. Jaegle, F. Gimeno, A. Brock, A. Zisserman, O. Vinyals, and J. Carreira. Perceiver: General Perception with Iterative Attention, June 2021. URL http://arxiv.org/abs/2103.03206. arXiv:2103.03206 [cs, eess].

[95] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

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

[97] A. Veit, M. Wilber, and S. Belongie. Residual Networks Behave Like Ensembles of Relatively Shallow Networks, Oct. 2016. URL http://arxiv.org/abs/1605.06431. arXiv:1605.06431 [cs].

[98] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le. Flow matching for generative modeling, 2023. URL https://arxiv.org/abs/2210.02747.

[99] NVIDIA, :, J. Bjorck, F. Castañeda, N. Cherniadev, X. Da, R. Ding, L. J. Fan, Y. Fang, D. Fox, F. Hu, S. Huang, J. Jang, Z. Jiang, J. Kautz, K. Kundalia, L. Lao, Z. Li, Z. Lin, K. Lin, G. Liu, E. Llontop, L. Magne, A. Mandlekar, A. Narayan, S. Nasiriany, S. Reed, Y. L. Tan, G. Wang, Z. Wang, J. Wang, Q. Wang, J. Xiang, Y. Xie, Y. Xu, Z. Xu, S. Ye, Z. Yu, A. Zhang, H. Zhang, Y. Zhao, R. Zheng, and Y. Zhu. Gr00t n1: An open foundation model for generalist humanoid robots, 2025. URL https://arxiv.org/abs/2503.14734.

[100] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion, Mar. 2023. URL http://arxiv.org/abs/2303.04137. arXiv:2303.04137 [cs].

Appendix

In this document, we provide detailed supplementary material including a table summarizing notation (A), a discussion on the three-stage decomposition (B), extended related work (C), detailed experimental and training details (D), ablation studies (E), and additional quantitative (F) and qualitative (F.3) results.

A Notation

Table 7: Description of the Notation and Acronyms used in this manuscript

Symbol	Meaning
o_t	Image (visual) observation at time t.
q_t	Proprioceptive state of the robot at time t (e.g., joint angles).
a_t	Action at time t.
g	Goal specification (e.g., language instruction or task label).
f	Forward dynamics model that autoregressively predicts latent motion tokens from o_t and g.
f_{inv}	Inverse dynamics model mapping latent motion tokens and current state (o_t, q_t) to a sequence of actions.
π	Our policy, defined as $f_{\mathrm{inv}}(o_t, q_t, f(o_t, g))$.
$\mathcal{V}$	Video dataset $\{(o_t, g)\}$
$\mathcal{R}$	Action-labeled robot interaction dataset $\{(o_t, q_t, a_t)\}$
κ_t	Raw keypoint trajectories over a horizon from time t, with dimensions $T \times N \times 2$.
u_t	Single-timestep velocities computed from κ_t.
$\tilde{z}_t$	Continuous latent vectors produced by the keypoint encoder.
z_t	Discrete latent codes (tokens) representing keypoint motion, obtained via FSQ.
$\mathcal{E}_\theta$	Keypoint encoder that maps velocities u_t to latent representations.
$\mathcal{D}_\theta$	Keypoint decoder that reconstructs velocity distributions from latent codes.
ω_t	Ground-truth discretized labels for velocities, computed as $\Omega(u_t)$.
T	Prediction horizon (number of timesteps over which motion is predicted).
N	Number of keypoints in the grid.
W	Local window size for pixel classification in the decoder.
$\boldsymbol{a}_t$	Action chunk (sequence of actions over the horizon), i.e., $a_{t:t+T}$.
γ	Temporal discount factor used in the inverse dynamics loss.

B Discussion on the Three-Stage Decomposition

One fundamental limitation of Behavior Cloning is that it is a monolithic architecture that requires paired (action, observation) data to learn a policy $\pi(o_t) = a_t$, which is not readily available at scale. Moreover, the data is assumed to be a goal-directed sequence of expert actions–standard imitation learning has no mechanism for harnessing interaction data that is suboptimal or not directed towards solving the tasks in the test set, even though such data (1) contains rich information about the relationship between visual observations, environment dynamics, and agent actions, and (2) may be easier to collect through exploration or play, compared to expert demonstrations [74, 75, 76]. In Section 3.2, we demonstrate that these limitations prevent BC approaches from learning reusable abstractions. Based on these observations, we argue for a decoupled multi-stage approach that can learn from heterogeneous data sources. We classify data sources into three distinct categories based on their modality composition:

Action-Free Videos: observations of goal-directed behavior, but no action labels $\{(o_t, g)\}$

Undirected Interaction Data: observations and robot actions, but not in a goal-directed manner $\{(o_t, q_t, a_t)\}$

Expert Robot Demonstrations: goal-directed action-labeled rollouts $\{(o_t, q_t, a_t, g)\}$

Note that Expert Demonstrations can be treated as both Action-Free Videos and Interaction Data, since the modalities are a strict superset of the modalities in the other two. For this reason, in the main body of the paper we simply refer to $\mathcal{V}$ and $\mathcal{R}$, with any available demo data included in both sets by default. This taxonomy points towards one possible natural decomposition:

1. Use Action-Free Videos (and Expert Demonstrations) to learn how observations evolve with respect to a goal

Algorithm 1 AMPLIFY Training.

Require: Datasets $\mathcal{V}, \mathcal{R}$
1: Preprocess keypoint tracks κ_t in $\mathcal{V}$
2: Learn latent motion encoding to compress κ_t into discrete tokens z_t using Eq. 1
3: Train forward dynamics model $f(o_t, g) = z_t$ on $\mathcal{V}$ using Eq. 2
4: Train inverse dynamics model $f_{\text{inv}}(o_t, q_t, z_t) = a_{t:t+T}$ on $\mathcal{R}$ using Eq. 3
5: **return** $\pi(o_t, q_t, g) = f_{\text{inv}}(o_t, q_t, f(o_t, g))$

2. Use Interaction Data (Undirected and Demonstrations) to learn how a sequence of observations maps to a sequence of actions

This decomposition effectively decouples *task understanding* (the sequence of observations that correspond to a goal) and *task execution* (translating a reference sequence of states into low-level actions). Observations, the only shared modality, operate as the interface bridging the gap, serving as prediction targets for the first stage and input references for the second. Seeking a compact, action-informative representation of these observations, AMPLIFY employs latent keypoint tokens, providing the third component of the three-stage decomposition. However, plenty of other representations are possible, including uncompressed images [16] and pixel-space tracks [54, 53]. In Algorithm 1 we summarize the training procedure for the three-stage approach, and in Table 8 we highlight the different data sources used for each component of AMPLIFY in comparison to BC.

Table 8: Compared to BC, AMPLIFY can leverage video and off-task data by decoupling forward and inverse dynamics.

Data type	BC	Forward Dynamics	Inverse Dynamics	AMPLIFY
Action-Free Videos		✓		✓
Expert Robot Demonstrations	✓	✓	✓	✓
Undirected Interaction Data			✓	✓

C Extended Related Work

In this section we provide context on additional related works and alternative approaches for robot learning from videos.

C.1 Learning from Hand Pose

Videos have shown to be an effective source of data for learning robotic policies from human demonstrations. One method for attaining action labels for unannotated videos is to estimate hand pose to gain information about human action. [74, 77, 45] estimate the trajectory of the hand position or pose, and then train a policy to replicate these trajectories with a robotic arm. While this representation reduces the domain gap, it lacks granularity, as the degrees of freedom represented (either 3 or 6 DoF) fall short of capturing the full complexity of the human hand, which possesses 20-30 DoF.

The retargeting of human actions to the robot's action space is another prevalent strategy. This has been achieved through various means, including image translation architectures [78] analytical remappings to optimize cost functions [79, 80], and masking the agent from the scene [81].

C.2 Learning from Affordances

Affordances, or the set of ways in which a given object or environment may be manipulated, are a common abstraction between human and robot data that lends itself well to learning from videos. The estimation of contact locations, trajectories, and future states are prevalent strategies for interpreting and acting upon environmental cues [80, 82, 83]. These methods aim to deduce actionable information from video data, and attempt to learn how to interact with various objects and environments based on observed human actions.

C.3 Reward/Representation Learning from Videos

A common method of extracting action-relevant information from videos is via self-supervised representation learning. Some works align video data with language descriptors via contrastive learning [29, 84] or minimize the distance from a specified goal representation [24]. Some works extract latent action representations from videos such as [85, 86, 87]. More recently, works such as [88, 34, 33] extract latent actions via pixel-level reconstruction and use them to learn from action-free videos.

Representations learned from video and language often serve as the basis for reward or value functions in deep reinforcement learning settings, predominantly within goal-conditioned RL frameworks [24, 85, 87], where the aim is to produce actions that minimize the distance to the desired outcome. Other works such as [89, 90] utilize models trained on objectives such as video prediction to estimate values.

C.4 Forward and Inverse Dynamics for Robot Learning

AMPLIFY benefits from multiple sources of data by decoupling the problem of policy learning into forward and inverse dynamics, where the forward dynamics model predicts future states or latent representations, and the inverse dynamics model maps these predictions to actions. [35, 91] focus on recovering latent actions from video data using self-supervised pretraining, enabling control with minimal action labels. Methods such as [92, 93, 44, 16] use text-guided video generation to predict future visual trajectories, from which actions can be inferred. These works present a promising direction for transferring information from large-scale video data into visuomotor policies using pre-trained foundation and frontier models, thus improving generalization to new tasks and environments.

D Experimental and Training Details

In this section we provide extensive details on the LIBERO benchmark (D.1), our real-world setup (D.2), metrics used for evaluation (D.3), preprocessing (D.4), and training details for each stage (D.5).

D.1 LIBERO Benchmark

We evaluate on the LIBERO [66] benchmark, which consists of 130 manipulation tasks. The LIBERO benchmark is categorized into distinct subsets:

- **LIBERO-Long**: A subset of 10 long-horizon manipulation tasks.
- **LIBERO-90**: A broad set of 90 tasks with diverse objects, layouts, and backgrounds.
- **LIBERO-Object**: Tasks that evaluate generalization to novel object categories.
- **LIBERO-Spatial**: Tasks that test generalization across varied spatial arrangements.
- **LIBERO-Goal**: Tasks with the same starting scene but different goals to assess goal conditioning.

Figure 5 shows a sample of the tasks and environments in the collection. The benchmark comes with a dataset of 50 expert demonstrations per task, obtained through VR teleoperation [66]. When evaluating on LIBERO, AMPLIFY takes in the standard 128×128 RGB images as observations and produces normalized axis-angle actions. We execute actions in the environment at 20 Hz, and give our model a maximum of 500 environment steps to solve each task. We perform rollouts on 10 random seeds per task for all subsets of LIBERO except for LIBERO-90, for which we only perform one rollout for each of the 90 tasks to produce our results.

D.2 Real-World Setup

Robot and Camera Setup – In our real-world experiments, we evaluate the performance of our method on three distinct manipulation tasks using a UR5 robotic arm equipped with three synchronized static RGB camera views positioned on three sides of the workspace (Pictured in Figure 6). This multi-view configuration ensures robust perception of the scene. We do not use a wrist camera. Camera observations are captured at 60Hz but downsampled to 20Hz to match action inputs on the

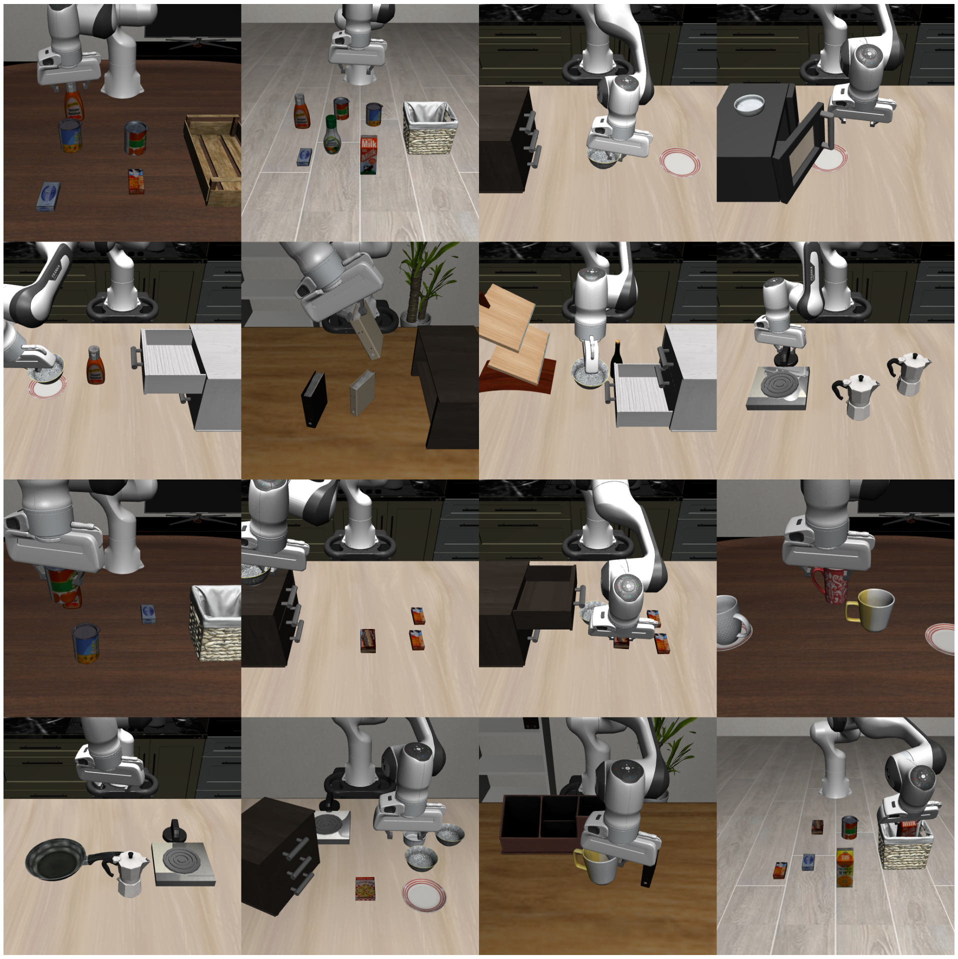


Figure 5: A sample of the 130 diverse tasks and environment configurations in LIBERO.

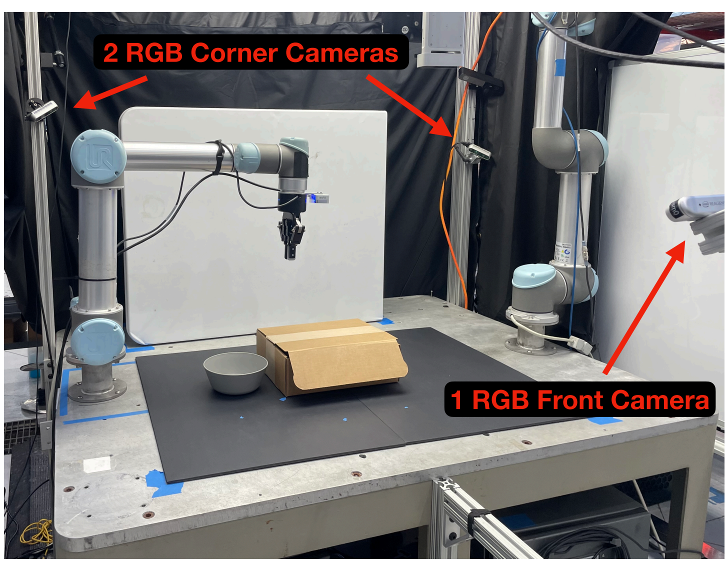


Figure 6: We use three static RGB cameras as input observations for both human and robot (UR5) data.

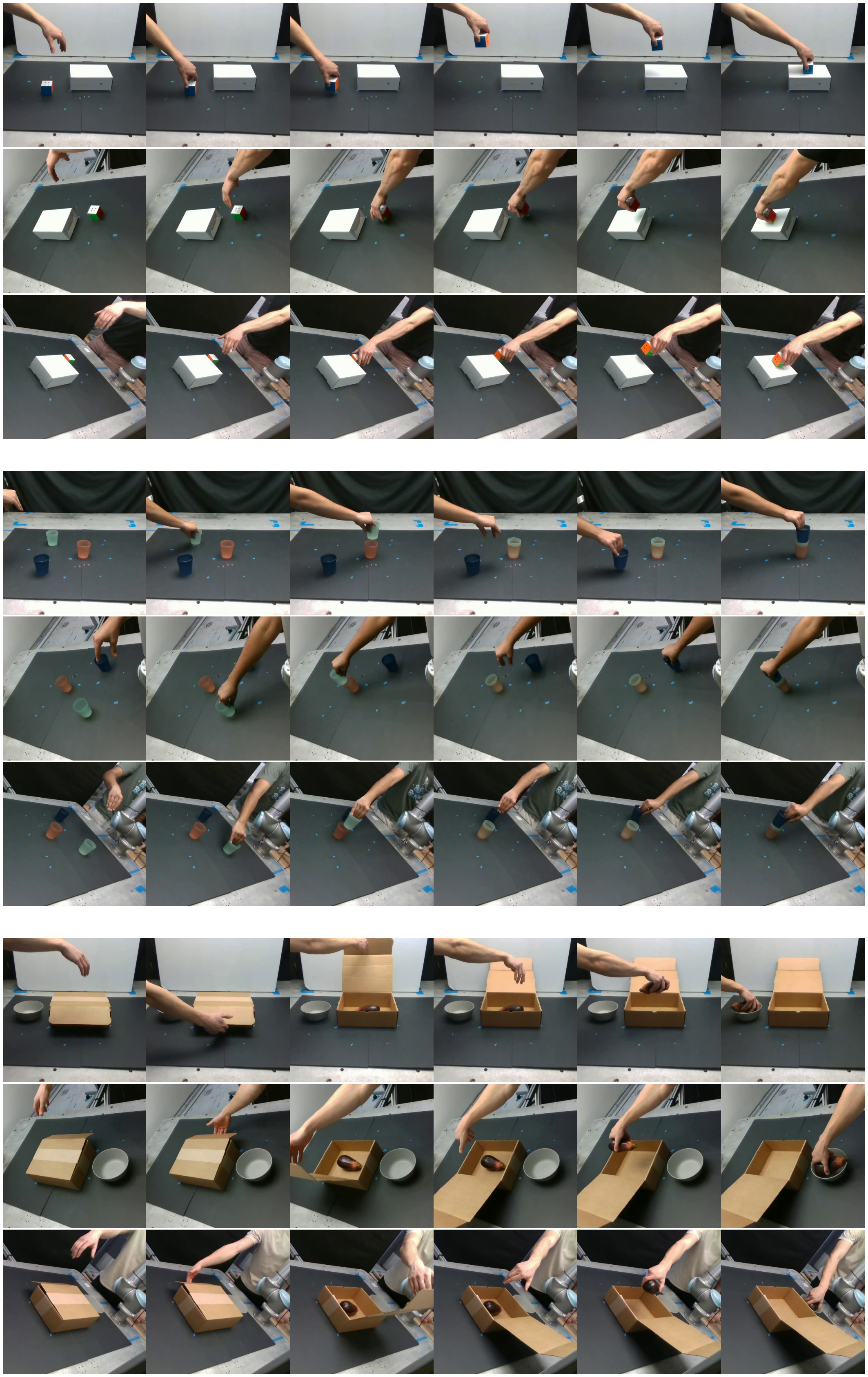


Figure 7: Real-World Tasks and Cameraviews. Each row shows front and corner camera views for three different tasks: *Task 1:* “Put the Rubik’s Cube on the Box”, *Task 2:* “Stack the Green and Blue Cups in the Orange Cup”, and *Task 3:* “Open the Box and Move the Eggplant into the Bowl”.

robot. Images are cropped and resized to 224×224 for all views. Actions are output as absolute normalized end-effector positions, which we found to work better than delta pose.

Task Descriptions and Demonstration Details – To test performance on our real-world setup, we evaluate AMPLIFY across three tasks of varying difficulty. Fig 7 shows each task from the 3 cameraviews, and Table 9 records the number of human/robot demonstrations we collect for each task.

- **Task 1: "Put the Rubik's Cube on the Box"** - This is the easiest task, consisting of a single pick-and-place movement. We slightly vary the initial placement and orientation of the cube during both demonstrations and evaluation.
- **Task 2: "Stack the Green and Blue Cups in the Orange Cup"** - This tasks consists of two cup-stacking motions, requiring more precise grasping and a longer-horizon rollout. We measure both partial and full success rate (stacking one or both cups correctly, invariant of order) and report full results in Table 17. Cup placements are also varied slightly during demonstrations and evaluations.
- **Task 3: "Open the Box and Move the Eggplant into the Bowl"** - This task requires precise object articulation (the clearance between the box lid and the table is only a few centimeters), continued contact, and then a pick-and-place movement with partial observability. Like Task 2, we measure both partial and full success rates (partial for opening the box, full for also placing the eggplant in the bowl correctly), also shown in Table 17. We vary the orientation of the eggplant and position of the box flap.

Table 9: Demonstration Counts per Task. Number of robot and human demonstrations collected for each real-world task.

Task	Robot Demos	Human Demos
`Place Cube`	24	48
`Stack Cups`	13	59
`Open Box & Place Eggplant`	15	30

Evaluation Criteria – We measure success rates from 10 rollouts on each task for both AMPLIFY and the baseline Diffusion Policy, with a time limit of 90 seconds. Partial and full success rates are measured as follows:

`Place Cube`
- *Partial Success:* N/A.
- *Full Success:* The Rubik's cube is safely placed on the box.

`Stack Cups`
- *Partial Success:* Two out of the three cups are successfully stacked.
- *Full Success:* All three cups are correctly stacked.

`Open Box & Place Eggplant`
- *Partial Success:* The box is opened, regardless of whether the eggplant is picked and placed.
- *Full Success:* The robot opens the box and successfully places the eggplant in the bowl.

D.3 Metrics

Track Prediction Metrics – We use three main metrics to measure performance in our keypoint trajectory (track) prediction experiments: mean squared error (MSE), pixel accuracy, and Δ_{AUC} (Area Under Curve). For each, we use tracks obtained from CoTracker [36] as "ground-truth" for predictions. All methods use a uniformly spaced 20×20 grid as initial query points to track. Track2Act [53] also used this initial grid for their predictions, so we take their reported numbers directly. In ATM [54], the model is trained to take in any point location (including the uniform grid). Seer [67] is a video prediction model, so any queries can be applied to its output and tracked by CoTracker.

1. **MSE**: We take normalized track predictions in the range $[-1, 1]$ and compute MSE between predicted (x, y) values for each point and the corresponding ground-truth point from CoTracker.
2. **Pixel Accuracy**: We measure the (normalized) percentage of predictions that are pixel-perfect compared to ground-truth.
3. $\Delta_{\textbf{AUC}}$: This metric was originally introduced by works presenting point tracking [38, 36] and later used for track prediction [53]. The metric is computed as follows. Let δ_t^x be the fraction of point predictions that are within a threshold pixel distance of x of their ground truth in a time-step $t \in [0, H]$. Following [53], we report the area under the curve Δ with δ_t^x by varying x from 1 to $N = 10$ and taking the average across the prediction horizon H i.e. $\Delta = \left(\sum_{t=1}^{H} \sum_{x=1}^{N} \delta_t^x\right)/H$. Hence, Δ is normalized to $[0, 1]$, with higher values corresponding to better predictions.

Video Prediction Metrics – We use three standard metrics for measuring generated video quality: Learned Perceptual Image Patch Similarity (LPIPS), Structural Similarity Index (SSIM), and Peak Signal-to-Noise Ratio (PSNR). These metrics are defined as follows:

$$\text{LPIPS}(x, \hat{x}) \coloneqq \frac{1}{HW} \sum_{h,w} ||\phi(x)_{hw} - \phi(\hat{x})_{hw}||_2^2$$

$$\text{SSIM}(x, \hat{x}) \coloneqq \frac{(2\mu_x \mu\hat{x} + c_1)(2\sigma_{x\hat{x}} + c_2)}{(\mu_x^2 + \mu_{\hat{x}}^2 + c_1)(\sigma_x^2 + \sigma_{\hat{x}}^2 + c_2)}$$

$$\text{PSNR}(x, \hat{x}) \coloneqq 10 \log_{10} \left(\frac{\text{MAX}_I^2}{\frac{1}{HW} \sum_{h,w} (x_{hw} - \hat{x}_{hw})^2} \right)$$

where x_{hw} and $\hat{x}_{hw}$ are ground truth and predicted images respectively, ϕ denotes a pretrained deep network, μ and σ represent mean and variance, c_1 and c_2 are constants for numerical stability, and MAX is the maximum possible pixel value. Note that Frechet Inception Distance (FID) and Frechet Video Distance (FVD) are not meaningful metrics in our scenario due to a mismatch between the input image size for the Inception network and the videos produced by AVDC.

D.4 Preprocessing

We unfold (in time) a length-τ video $o \in \mathbb{R}^{\tau \times H \times W \times 3}$ into τ length-T windows to be tracked by CoTracker [36]. For each window, we initialize a uniform $N = 20 \times 20$ grid of query points to be tracked through the T frames of the window. Thus, we obtain an output $\kappa_{0:\tau} \in \mathbb{R}^{\tau \times T \times N \times 2}$, where κ_t are the tracks corresponding to the grid initialized in o_t and tracked through $o_{t:t+T}$.

Note that tracks corresponding to nearby windows overlap in time, but correspond to *different* initial query points, since we re-initialize in every frame. This is in contrast to the way tracks are preprocessed in ATM [54], where the points are initialized *once* (in the last frame) and the *same* points are tracked through the entire video. Our preprocessing strategy is $T\times$ more compute-intensive, but prevents issues of prolonged occlusion and gracefully handles moving/panning cameras, which is important when using a wrist camera, for example.

Once tracks are obtained from CoTracker, we treat them as ground-truth targets for the rest of the pipeline. For training stability, we normalize tracks from pixel coordinates to the range $[-1, 1]$. To ensure tracks correspond to the same length of time, we interpolate tracks in time on same datasets that have a different sampling frequency. On LIBERO, Real robot tasks, and Something-Someting v2 we use a 16-frame true horizon, corresponding to 0.8 seconds (at 20Hz). For BridgeData v2, the observation frequency is 5Hz so we interpolate a 4-frame horizon to 16 using a cubic spline. For real-world human data, we use 8 frames, since human execution is roughly $2\times$ that of the robot. In Appendix E we examine the effect of changing prediction horizon on downstream policy performance and find that 16 frames (0.8s) lead to the best result. This echoes findings in [54]. Before tokenizing, we compute instantaneous velocities through simple differencing in the time dimension. Since the initial queries are always the same, velocities contain the same information and reduce the sequence length slightly.

D.5 Training Details

We present detailed hyperparameter choices in Table 10 and a high-level overview of AMPLIFY training in Algorithm 1. All three components were trained on a single GPU (either RTX 6000 or L40S, depending on availability). We report the default number of epochs used for experiments. For the LIBERO benchmark, we trained the motion tokenizer for approximately 50,000 gradient steps, the forward dynamics model for 25,000 gradient steps, and the inverse dynamics model for 75,000 gradient steps. For Something-Something v2, we trained the motion tokenizer for 32,000 gradient steps and the forward dynamics model for 22,000 gradient steps. For BridgeData, we trained the motion tokenizer for 20,000 gradient steps and the forward dynamics model for 175,000 gradient steps. Each gradient step is counted as the accumulated gradients over 4 backward passes to simulate the reported batch size.

Table 10: Hyperparameters for Motion Tokenizer, Forward Dynamics, and Inverse Dynamics model training.

Hyperparameter	Motion Tokenizer	Forward Dynamics	Inverse Dynamics
number of parameters	31M	70M	57M
epochs	100	100	250
batch size	256	256	256
gradient accumulation	4	4	4
learning rate	1e-4	1e-4	1e-4
optimizer	AdamW	AdamW	AdamW
image size	-	128×128	128×128
number of points N	-	400	-
track prediction horizon T	-	16	-
decoder local window size W	-	15	-
number of heads	8	8	8
number of layers	2	8	4
hidden dimension	768	768	768
dropout	0.1	0.1	0.1
FSQ implicit codebook size	-	2048	-
Action loss discount γ	-	-	0.99

Video Prediction – To demonstrate the utility of our method beyond robotics, we extend the AVDC [44] video generation model by conditioning it on the motion tokens produced by the forward dynamics model. These motion tokens serve as a conditioning signal that guides video prediction based on the expected dynamics. To condition the video generation model, motion tokens generated by the forward dynamics model are concatenated with text tokens along the sequence dimension before features before being pooled by a Perceiver [94] resampler in AVDC's UNet. To ensure efficient batching during training and inference, the concatenated sequences are padded to a fixed sequence length. During training, the video prediction model is conditioned on motion tokens obtained directly from the motion tokenizer. This direct conditioning allows the model to learn directly from "ground truth" motion tokens, increasing training efficiency.

At test time, a new sequence of motion tokens is sampled every T timesteps. This sampling strategy allows the model to generate a complete trajectory by sequentially updating the conditioning information over time. The model is conditioned on the outputs of the forward dynamics model during inference, similarly to AMPLIFY's inverse dynamics model. We find that track-conditioned video generation leverages the strength of our forward dynamics model to guide the video generation process, thus improving the quality and coherence of the generated video sequences.

E Ablation Studies

We conduct an ablation analysis to study the effect of various design choices in our architecture. All evaluations are performed on LIBERO-Long, which is the most challenging subset of LIBERO due to it's long horizon (up to 500+ steps) compared to the other tasks.

Motion Tokenization – We examine the effect of a number of architectural choices in the Motion Tokenizaton stage, using the Δ_{AUC} as the principal metric, summarized in Table 11.

- **Attention Masks**: We consider three attention masks in the encoder: per-timestep tokenization, where there is no information transfer across time; causal attention, where information flows one way in time, and no mask. We observe a significant benefit from attention across time, but marginal difference between causal and full attention (we opt to use causal for efficiency).
- **Decoder Output Loss**: We consider two options for decoder output loss: MSE, and Local Window Classification. Under the MSE setup, the decoder is configured to directly output (normalized) pixel coordinates, which are regressed to the targets from CoTracker. With Local Window Classification, the decoder instead outputs classification logits over $W^2 = 225$ classes for each point for each timestep. Each class corresponds to a pixel in a the local W^2 window of pixels centered around the previous point in the current timestep. For example, the class corresponding to the middle of the window predicts a velocity (0,0), whereas the class corresponding to the top-right pixel is (7,7). The size of the window can be inferred from the data and leads to a bias for local motion. We observe that Local Window Classification models tracks better and leads to more accurate predictions. Qualitatively, we notice that MSE tends to lead to blurrier predictions and tends towards 0-movement more. We suspect this is because the classification objective can model multi-modal distributions, whereas MSE simply regresses to the mean.
- **Joint Tokenization**: We study whether conditioning track tokenization on the image helps reconstruction by conditioning the encoder on both image tokens (same as forward dynamics) and the input velocities. Performance dropped slightly, though not much difference was apparent.
- **Prediction Horizon**: We vary the prediction horizon from 4 to 16 timesteps and find that reconstructing tracks over a shorter horizon is easier. This is expected, as there is less uncertainty with shorter horizons. We observe a similar case when trainig forward dynamics. However, the inverse dynamics model has the opposite trend, since longer horizon predictions are ultimately more useful for action inference. As a result, we choose to use a horizon of 16 in our final model, as we are ultimately interested in downstream policy performance.
- **Model Dimensions**: Finally, we examine FSQ (effective) codebook size, code sequence length, and hidden dimension of the transformers. We find that an effective codebook size of 2048, a latent sequence length of 16, and a hidden dimension of 768 perform marginally better.

Forward Dynamics – We study design choices in the forward dynamics model, which predicts tokenized trajectories from observations. We evaluate models based on Δ_{AUC} and pixel accuracy, summarized in Table 12.

- **Prediction Horizon**: Similar to motion tokenization, we observe that shorter prediction horizons improve accuracy. Predicting only 4 steps ahead achieves the highest accuracy, while 16-step prediction is substantially harder. We nevertheless use 16 in the final model to match the inverse dynamics setup.
- **Vision Encoder Architecture**: We evaluate multiple vision encoders. Interestingly, despite the popularity of larger and pretrained architectures (e.g., DINOv2 [95], [96]), ResNet-18 [97] performs competitively, with minimal drop in performance, making it a computationally efficient default choice.
- **Token Pooling Strategy**: We compare using CLS tokens vs. patch tokens from ResNet-18 as inputs to the transformer. Patch tokens (i.e., per-patch embeddings) slightly underperform CLS pooling, but are preferable due to their richer spatial structure and compatibility with other modules.
- **Transformer Depth**: We evaluate transformer depth and find that using 4 layers performs slightly better than 8. This may be due to overfitting or optimization instability with deeper models on limited data.
- **Frozen vs. Fine-tuned Vision Encoder**: We find no benefit to fine-tuning the ResNet encoder during forward dynamics training, so we freeze it to save compute and stabilize training.

Inverse Dynamics – Finally, we investigate the impact of the action prediction horizon and choice of output head in the inverse dynamics module, using downstream task success rate as the evaluation metric. Results are shown in Table 13.

Table 11: Motion Tokenizer Ablations.

Ablation Factor	Configuration	Metric	Performance
Attention Mask	Per-Timestep	Δ_{AUC}	0.877
	Causal	Δ_{AUC}	**0.919**
	Full	Δ_{AUC}	0.918
Decoder Output Loss	MSE Loss	Δ_{AUC}	0.883
	Local Window Classification Loss	Δ_{AUC}	**0.919**
Joint Tokenization	**Tracks Only**	Δ_{AUC}	**0.929**
	Tracks + Image	Δ_{AUC}	0.926
Prediction Horizon	**4**	Δ_{AUC}	**0.985**
	8	Δ_{AUC}	0.961
	16	Δ_{AUC}	0.919
Codebook Size	512	Δ_{AUC}	0.912
	1024	Δ_{AUC}	0.919
	2048	Δ_{AUC}	**0.921**
Hidden Dimension	128	Δ_{AUC}	0.897
	256	Δ_{AUC}	0.909
	384	Δ_{AUC}	0.911
	512	Δ_{AUC}	0.914
	768	Δ_{AUC}	**0.919**
	1024	Δ_{AUC}	0.917
Code Sequence Length	2	Δ_{AUC}	0.853
	4	Δ_{AUC}	0.877
	8	Δ_{AUC}	0.909
	16	Δ_{AUC}	**0.919**
	32	Δ_{AUC}	0.893

Table 12: Forward Dynamics Ablations.

Ablation Factor	Configuration	Metric	Performance
Prediction Horizon	**4**	Pixel Accuracy	**0.757**
	8	Pixel Accuracy	0.678
	16	Pixel Accuracy	0.613
Vision Encoder	**ResNet-18**	Pixel Accuracy	**0.613**
	ResNet-50	Pixel Accuracy	0.621
	DINOv2	Pixel Accuracy	0.621
	ViT	Pixel Accuracy	0.614
Token Pooling Strategy	**Patch Tokens**	Pixel Accuracy	**0.613**
	CLS Token	Pixel Accuracy	0.621
Transformer Depth	**4 Layers**	Δ_{AUC}	**0.930**
	8 Layers	Δ_{AUC}	0.929
Vision Encoder Tuning	**Frozen**	Δ_{AUC}	**0.929**
	Fine-tuned	Δ_{AUC}	0.929

Table 13: Inverse Dynamics Ablations.

Ablation Factor	Configuration	Metric	Performance
Prediction Horizon	4	Success Rate	0.36
	8	Success Rate	0.64
	16	Success Rate	**0.75**
Action Head	**Gaussian (Transformer, MLP)**	Success Rate	**0.74**
	Diffusion (U-Net)	Success Rate	**0.74**
	Flow Matching (DiT)	Success Rate	0.73

- **Prediction Horizon**: We observe a consistent improvement in task success as we increase the action prediction horizon. This makes intuitive sense: longer horizons provide more context for disambiguating latent trajectories and allow the inverse dynamics model to recover the intended actions more reliably. We thus adopt a 16-step horizon in our final setup.
- **Action Head**: We experiment with three different action heads. (1) a Gaussian Action head, consisting of a transformer decoder and MLP projection to output mean and log-std for a Gaussian Policy, trained with negatice likelihood loss as described in the main paper; (2) a Diffusion Policy head with U-net architecture (recommended for simple tasks), which we use in the real world setup for fair comparison. (3) A Flow Matching [98] head with the cross-attention DiT architecture from GR00T N1 [99], with optimal transport coupling and 10 integration steps with a midpoint ODE solver for inference. Our evaluations did not highlight significant differences in performance, so we opted for the simplest Gaussian Policy for the main results. This follows [62], who similarly found that a complex action head did not help on LIBERO tasks.

F Additional Results

F.1 Video Scaling Experiment

Table 14: We scale the amount of video data used AMPLIFY, showing that performance improves as we scale the amount of action-free video data. Note that these policies are trained on only 2 action-annotated trajectories.

# Videos	0	2	5	10	50
Ours	0.00	0.12	0.34	0.23	0.55

In order to investigate the impact of abundant action-free video data on policy performance, we conducted an experiment on the LIBERO-Object subset. In this setup, we vary the number of videos used to train the forward dynamics model while keeping the action-annotated dataset extremely limited (only 2 trajectories). The goal is to simulate a realistic scenario where acquiring action labels is expensive, yet large-scale video data is readily available.

Specifically, we trained the forward dynamics model using 0, 2, 5, 10, and 50 action-free video clips of LIBERO rollouts. Subsequently, the inverse dynamics model was trained using the outputs of the forward dynamics model as described in Section 2.3. The results summarized in Table 14 demonstrate that policy performance generally improves as the volume of video data increases. It is worth noting that while the overall trend is positive, there is a non-monotonic behavior (e.g., a drop from 0.34 with 5 videos to 0.23 with 10 videos). This variation could be due to inherent stochasticity in training or differences in video content quality. In addition, since all of the demonstrations are likely similar, the forward dynamics model may not gain significantly more information as the number of demonstrations increase. Overall, these findings suggest that augmenting limited action-annotated data with large-scale, action-free video data can effectively improve the learned forward dynamics, which in turn improves policy performance. However, more thorough investigation is needed before drawing definitive conclusions.

F.2 Detailed Tables of Main Results

We provide more detailed versions of tables provided in the main body.

Table 15: Prediction Performance on Track Prediction Metrics on all LIBERO subsets.

Method	Metric	LIBERO 90	LIBERO Long	LIBERO Object	LIBERO Spatial	LIBERO Goal	Aggregate
ATM [54]	MSE	0.012	0.008	0.008	0.074	0.009	0.022
	Δ_{AUC}	0.799	0.803	0.782	0.693	0.757	0.767
	Pixel Accuracy	0.339	0.349	0.222	0.146	0.195	0.250
AMPLIFY	MSE	0.004	0.002	0.001	0.019	0.002	0.006
	Δ_{AUC}	0.892	0.921	0.937	0.904	0.914	0.913
	Pixel Accuracy	0.612	0.633	0.656	0.613	0.630	0.629

Table 16: Few-shot Learning from Limited Action Data on LIBERO.

Method	LIBERO-Long			LIBERO-90			LIBERO-Object			LIBERO-Spatial			LIBERO-Goal		
# Demos	2	5	10	2	5	10	2	5	10	2	5	10	2	5	10
ATM [54]	0.16	0.37	0.39	–	–	–	0.51	0.58	0.68	0.51	0.66	0.68	0.38	0.64	**0.77**
AMPLIFY (inverse only)	0.00	0.04	0.07	0.06	0.18	0.28	0.04	0.09	0.38	0.17	0.16	0.37	0.00	0.04	0.16
AMPLIFY	**0.55**	**0.58**	**0.62**	**0.47**	**0.56**	**0.66**	**0.73**	**0.67**	**0.85**	**0.71**	**0.77**	**0.69**	**0.57**	**0.77**	0.75

Table 17: Real-World Task Performance with Partial and Full Success Rates.

Method	Open Box & Place Eggplant - Partial			Open Box & Place Eggplant - Full			Stack Cups Partial			Stack Cups Full			Place Cube			Avg
# Demos	5	10	All	5	10	All	5	10	All	5	10	All	5	10	All	
DP [100]	**0.5**	0.2	0.3	**0.1**	0.2	0.2	**0.8**	0.8	0.9	**0.3**	0.5	0.5	0.6	0.5	**0.9**	0.49
AMPLIFY	0.1	**0.4**	**0.5**	**0.1**	**0.3**	**0.4**	**0.8**	**0.9**	**1.0**	**0.3**	**0.6**	**1.0**	**0.7**	**0.9**	**0.9**	**0.59**

F.3 Qualitative Results

We provide qualitative results of predictions from AMPLIFY and samples from the video prediction model conditioned on predicted tracks. Within each frame, yellow indicates points at the current time, and red indicates the points in the future. Model outputs are for the full 400-point grid. To reduce clutter, however, we do not visualize points that are predicted to have no motion.

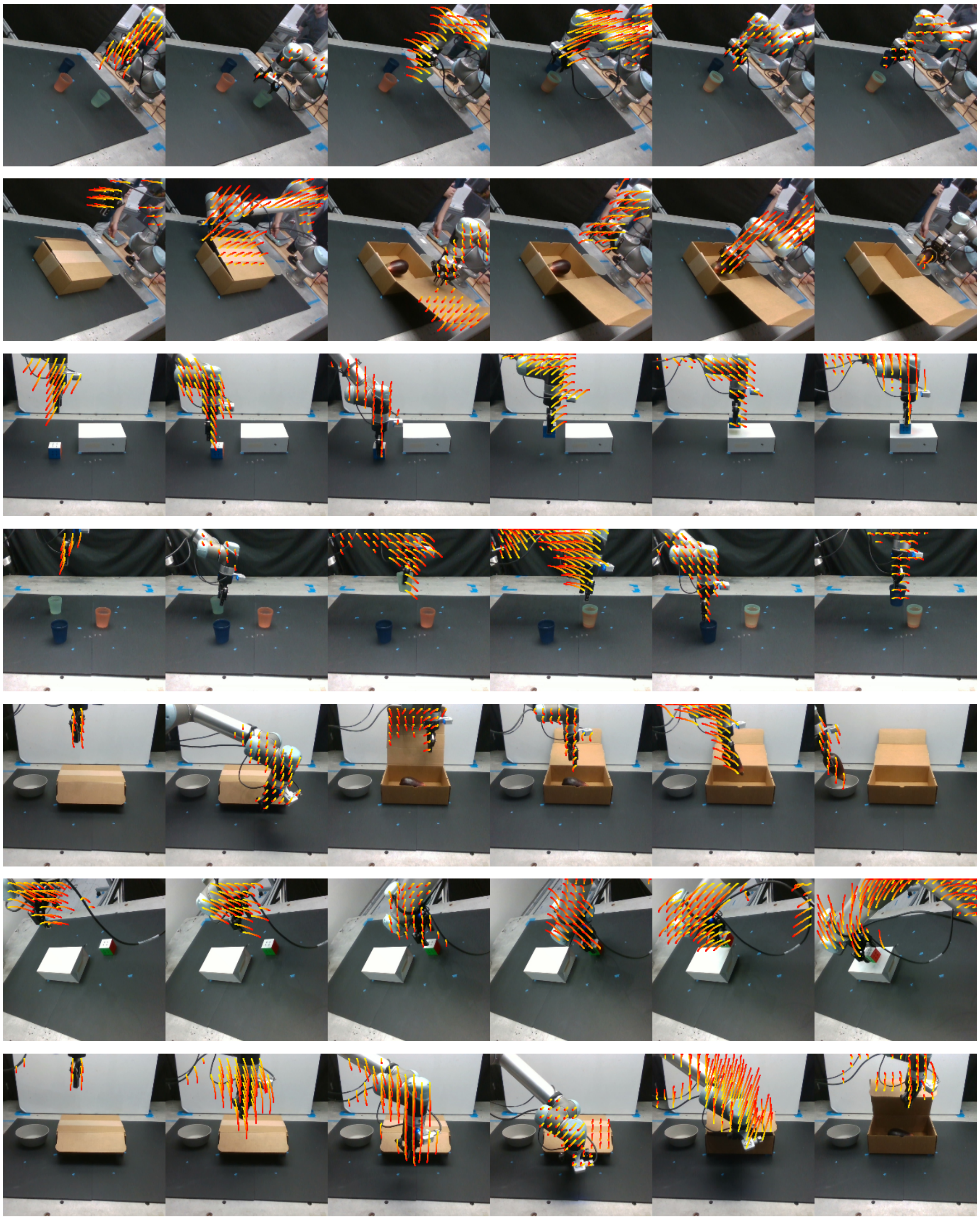


Figure 8: Track Predictions from AMPLIFY on Real-World Robot Data.

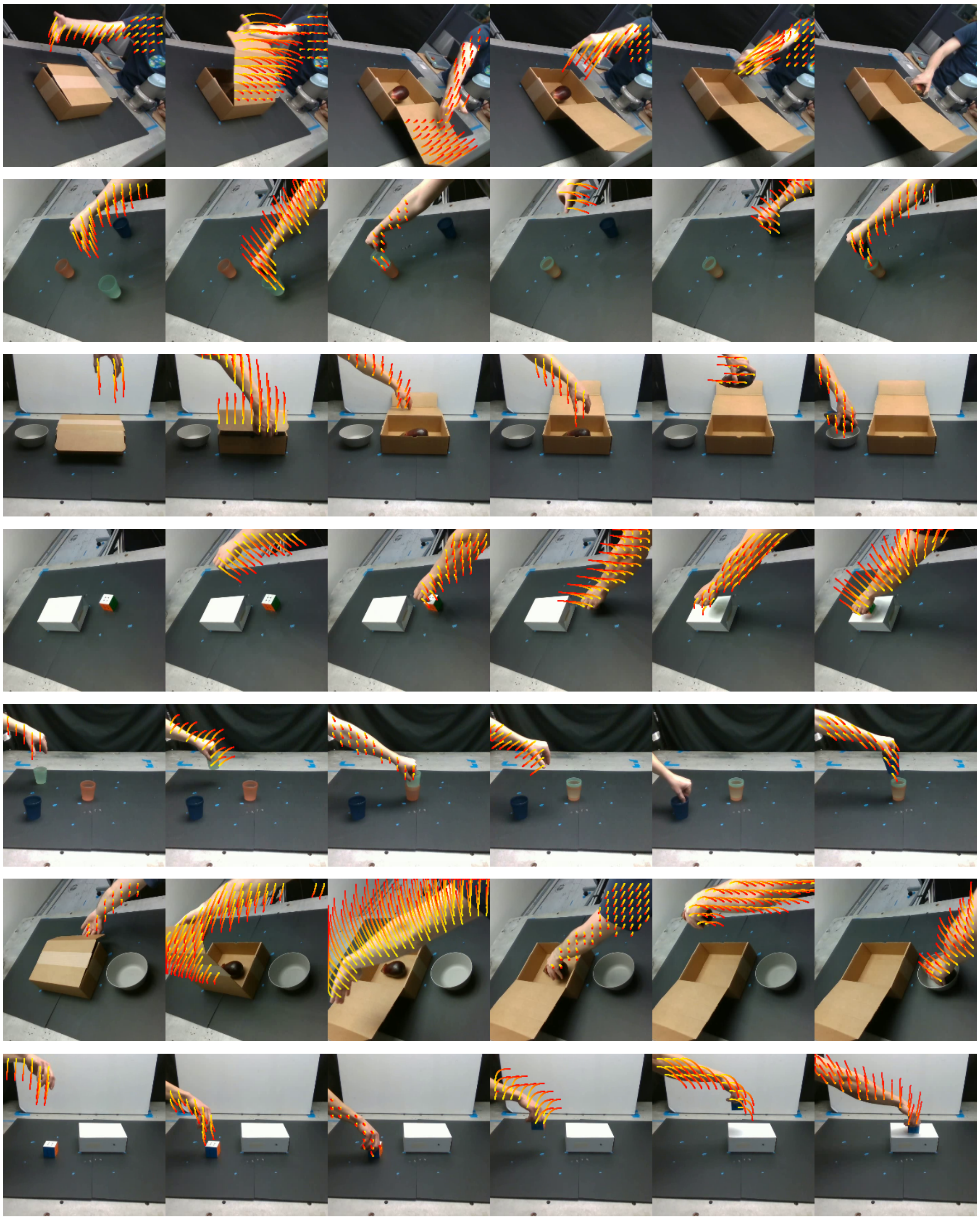


Figure 9: Track Predictions from AMPLIFY on Real-World Human Data.

Figure 10: Video Predictions from AVDC [44] conditioned on AMPLIFY, trained on Bridge Data.

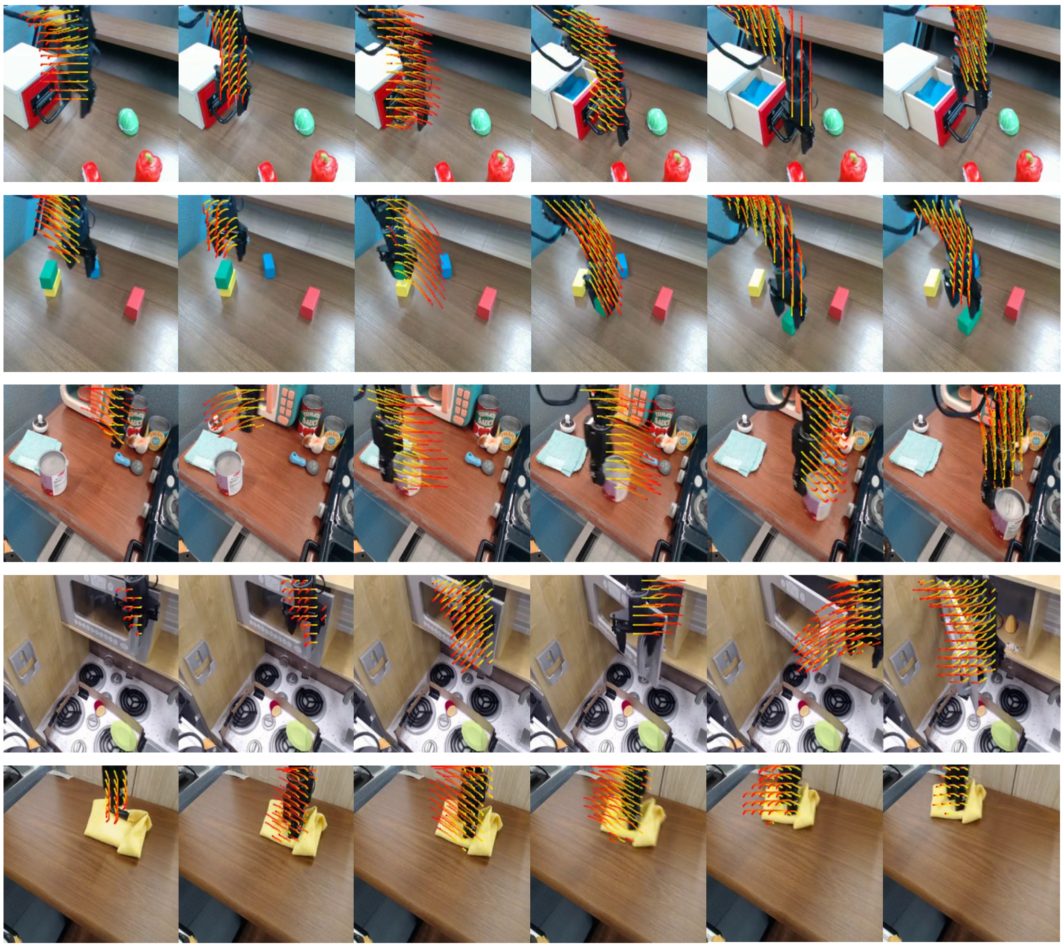


Figure 11: Track Predictions from AMPLIFY on Bridge Data.