Astra: Efficient Transformer Architecture and Contrastive Dynamics Learning for Embodied Instruction Following

Yueen Ma[1], **Dafeng Chi**[2], **Shiguang Wu**[2], **Yuecheng Liu**[2], **Yuzheng Zhuang**[2], **Irwin King**[1]
Department of Computer Science and Engineering, The Chinese University of Hong Kong[1]
Huawei Noah's Ark Lab[2]
{yema21, king}@cse.cuhk.edu.hk
{chidafeng1, wushiguang, liuyuecheng1, zhuangyuzheng}@huawei.com

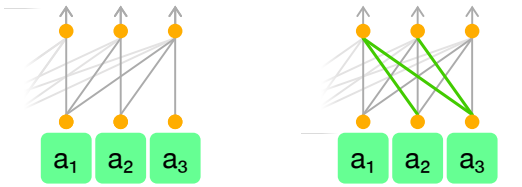
Abstract

Vision-language-action models have gained significant attention for their ability to model multimodal sequences in embodied instruction following tasks. However, most existing models rely on causal attention, which we find suboptimal for processing sequences composed of interleaved segments from different modalities. In this paper, we introduce Astra[1], a novel Transformer architecture featuring trajectory attention and learnable action queries, designed to efficiently process segmented multimodal trajectories and predict actions for imitation learning. Furthermore, we propose a contrastive dynamics learning objective to enhance the model's understanding of environment dynamics and multimodal alignment, complementing the primary behavior cloning objective. Through extensive experiments on three large-scale robot manipulation benchmarks, Astra demonstrates substantial performance improvements over previous models.

1 Introduction

Vision-language-action models (VLAs) (Brohan et al., 2023a) have recently emerged to address embodied instruction following tasks (EIF) (Lu et al., 2025). Previous multimodal models, such as vision-language models (VLMs), have demonstrated proficiency in handling both visual and textual inputs, successfully tackling a variety of tasks, such as visual question answering and image captioning (Zhang et al., 2024a). In contrast, VLAs differ from VLMs in that they can interpret language instructions, visually perceive their environment, and execute actions to fulfill specified embodied tasks. As a result, VLAs can empower embodied agents to interact with the physical world.

To accommodate multimodal inputs, previous Transformer-based VLMs (Ghosh et al., 2024) explored designing special types of self-attention to better suit the unique properties of different modalities. For example, in the task of image captioning, causal attention is not ideal for encoding images because there is no inherent causal relationship among image patches (Li et al., 2023a). Consequently, these VLMs allow bidirectional attention for image tokens while maintaining causal attention for text tokens.



(a) Causal attention. (b) Trajectory attention.

Figure 1: Comparison of information flow in an action segment. Squares represent tokens, while orange dots represent their embeddings. Three action tokens comprise an action "segment". The lines illustrate information flow from input embeddings (bottom) to output embeddings (top) through a Transformer self-attention layer. In trajectory attention, tokens attend not only to preceding tokens, as in causal attention, but also to subsequent tokens within the same segment, as indicated by the green lines.

We have similarly observed that multimodal sequences in EIF tasks, which are often referred to as trajectories, exhibit unique properties that can be more effectively captured by a novel type of self-attention, named trajectory attention, as illustrated in Figure 1 & 3. Specifically, each language prompt, state, or action consists of multiple tokens, which we collectively refer to as a "segment." For instance, embodied agents often utilize multiple camera angles, resulting in a state that comprises a segment of tokens, with each token corresponding to a different camera. These state tokens lack causal relationships within the same segment, as they are conditionally independent. The same holds for action tokens that correspond to action dimen-

[1] https://github.com/yueen-ma/Astra

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 13621–13639
November 4-9, 2025

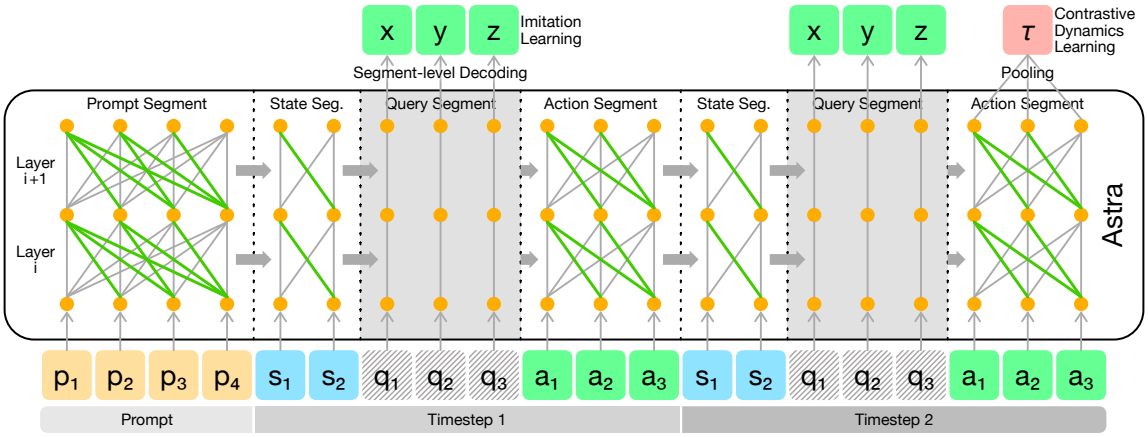


Figure 2: The architecture of Astra. A trajectory τ comprises a prompt segment $p_{1:4}$, state segments $s_{1:2,t}$, action segments $a_{1:3,t}$. Learnable action queries $q_{1:3,t}$ are inserted after state segments to extract information for action generation. Vertical dashed lines separate these segments. Token embeddings (orange dots) can attend to embeddings in all previous segments (thick horizontal arrows) and to all embeddings within the same segment (gray and green lines). Notably, action queries are hidden from other tokens and can only read from preceding tokens. To facilitate contrastive dynamics learning, Astra can also encode the entire trajectory by pooling the embeddings of the last segment (red box).

sions. Therefore, causal attention hinders full information flow within a segment because tokens are restricted from attending to subsequent tokens.

To overcome this limitation, we design trajectory attention with two key characteristics: inter-segment attention is causal, and intra-segment attention is bidirectional. Since a VLA model only needs to encode the language prompt and follow the corresponding instruction, we also apply bidirectional attention to the prompt segment. Consequently, our model processes EIF trajectories at the segment level. Accordingly, we also devise a segment-level decoding scheme that generates a segment as a whole. Drawing inspiration from DETR's object query (Carion et al., 2020; Chen et al., 2024) for object detection, we employ one learnable action query for each action dimension. Each action query extracts the most relevant information for its corresponding action dimension from preceding tokens and generates the optimal action independently of other action queries. By combining trajectory attention and action queries, we introduce an efficient Transformer architecture for EIF trajectories, which we name the **A**ction-predicting **S**egment-level **Tra**nsformer, or **Astra** for short. Figure 2 provides an overview of Astra.

The Astra architecture also possesses the capability to encode the entire sequence, which opens the possibility for contrastive dynamics learning (CDL), as shown in Figure 2. Numerous prior approaches have explored incorporating dynamics learning to bolster the main imitation learning task (Xu et al., 2024), but these efforts typically rely on decoding tasks: forward dynamics methods aim to predict the next state, while inverse dynamics methods attempt to reconstruct the action between two consecutive states. Such approaches often add considerable model complexity, such as requiring a video generator.

Our CDL objective instead leverages the encoding capabilities of Astra. As illustrated in Figure 4, we create positive samples using a novel action perturbation technique to augment action segments. Negative samples are constructed by mismatching segments with those from other trajectories, thereby violating the environment dynamics. By distinguishing positive samples from negative ones, it learns the correct dynamics, which in turn enhances performance on downstream EIF tasks. Due to the encoding nature of CDL, its implementation simply requires a classification head consisting of a pooling layer followed by a linear layer—a significantly lighter overhead compared to previous decoding-based dynamics learning methods. From another perspective, CDL also serves as a representation learning approach for multimodal alignment (Xiao et al., 2025), as it requires effectively encoding the multimodal trajectories.

The main contributions of this paper are:

- We introduce Astra, an efficient Transformer architecture featuring trajectory attention and action queries, designed to efficiently process

multimodal trajectories on the segment level;

- We propose a contrastive dynamics learning objective that enhances Astra's understanding of environment dynamics and its multimodal encoding capabilities, thereby complementing imitation learning;
- Extensive experiments across three large-scale robot manipulation benchmarks demonstrate that Astra significantly outperforms state-of-the-art methods, showcasing the effectiveness of our approach.

2 Related Work

Vision-Language-Action Models. VLAs (Ma et al., 2024b) are a new class of multimodal models designed to generate actions based on specified language prompts and perceived environments, first proposed by RT-2 (Brohan et al., 2023a). These models adapt pretrained large VLMs to predict actions for EIF tasks and are often referred to as "large VLAs." Representative works include RT-2 (O'Neill et al., 2024), OpenVLA (Kim et al., 2024), and π_0 (Black et al., 2024).

Another line of work does not utilize pretrained VLMs and instead builds VLAs from scratch, which are termed "generalized VLAs." These models predominantly draw upon the pioneering foundations laid by DT (Chen et al., 2021) and TT (Janner et al., 2021), where reinforcement learning is framed as sequence modeling problems. Gato (Reed et al., 2022) explored the use of a single Transformer model (Vaswani et al., 2017) for tasks spanning various domains. RT-1 (Brohan et al., 2023b) was the first robotics Transformer. VIMA (Jiang et al., 2022) studied multimodal prompts. Astra can also be categorized as a generalized VLA. However, distinct from these prior VLA models, which rely on causal or cross attention mechanisms, we propose a more efficient Transformer architecture for multimodal EIF tasks.

Multimodal Transformers & Learnable Queries. Several VLMs (Ghosh et al., 2024), such as UniLM (Dong et al., 2019), M6 (Lin et al., 2021), and PaliGemma (Beyer et al., 2024) have endeavored to optimize Transformer's self-attention for vision-language inputs by proposing various attention types, such as block attention. To the best of our knowledge, our architecture is the first VLA designed to accommodate multimodal EIF trajectories with a unique self-attention mechanism.

First introduced in DETR (Chen et al., 2024; Carion et al., 2020), learnable object queries have shown promising results in extracting information for object detection. BLIP-2 (Li et al., 2023a) used a similar strategy to extract visual embeddings for vision-language tasks. In our approach, we employ learnable action queries at the action-dimension level to extract information most relevant to individual action dimensions.

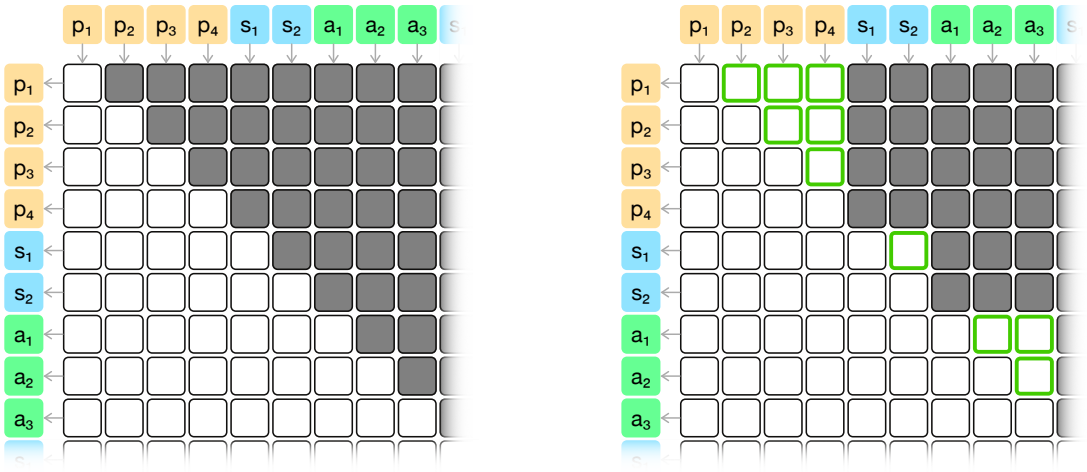
Dynamics Learning & Multimodal Contrastive Learning. Many recent dynamics learning approaches (Li et al., 2024; Sun et al., 2023; Liu et al., 2022) can be classified into two categories: forward dynamics learning and inverse dynamics learning. Most of these methods rely on extra generative modules, such as video generators (Cheang et al., 2024; Du et al., 2023). Our CDL leverages contrastive learning and involves only an encoding process using a lightweight linear head.

A series of VLMs have demonstrated the significance of contrastive learning in enhancing multimodal interaction (Zhang et al., 2024a). However, contrastive learning methods for EIF trajectories, such as R3M (Nair et al., 2022) and VIP (Ma et al., 2023a), primarily focus on improving visual representations. In contrast, our CDL task compels the model to align all three modalities, thereby enabling more effective encoding of EIF trajectories.

3 Method

3.1 Preliminaries

Multimodal sequences in embodied instruction following tasks are often referred to as trajectories (Lu et al., 2025). These trajectories consist of a language instruction (p), states (s), and actions (a). An trajectory is denoted as $\tau = (p, s_{t=1}, a_{t=1}, \ldots, s_{t=T}, a_{t=T})$, where t represents the timestep. Each element in the trajectory—p, s_t, or a_t—comprises a segment of tokens. For instance, a state s_t is a segment $s_{1:M,t} = (s_{1,t}, s_{2,t}, \ldots, s_{M,t})$, where each element is a token representing an image from a particular camera angle. Action tokens in a_t represent either $SE(2)$ actions or 6D pose actions. Tokens in p are standard language tokens. Therefore, a trajectory at the token level can be written as $\tau = (p_{1:L}, s_{1:M,t=1}, a_{1:N,t=1}, \ldots, s_{1:M,t=T}, a_{1:N,t=T})$. L, M, and N are the length of their corresponding segments. The goal is to obtain a policy that can generate an optimal action based on the past trajectory, expressed as $\pi(a_t|p, s_{\leq t}, a_{<t})$.



(a) Causal attention.

(b) Trajectory attention.

Figure 3: Attention matrices of causal and trajectory attention. The direction of attention is from the top (input) to the left (output). Dark cells represent attention masks. Green-bordered cells highlight additional information flow enabled by trajectory attention, corresponding to the green lines in Figure 1.

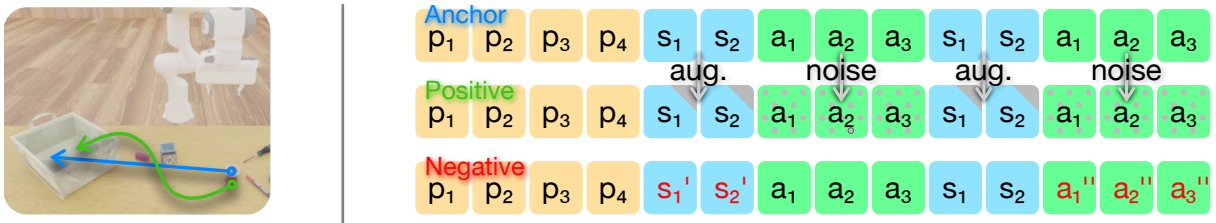
3.2 Architecture of Astra

Astra consists of two main novel components: trajectory attention and action queries, as illustrated in Figure 2. Most VLMs (Bai et al., 2025) utilize Transformer decoders as the backbone for natural language generation (NLG). To ensure that future tokens are not visible, these Transformer decoders typically use causal attention. Previous VLAs (O'Neill et al., 2024; Brohan et al., 2023b) have also adopted this approach for action generation. Although causal attention is well-suited for NLG, where language tokens are generated autoregressively, it is not the optimal attention mechanism for modeling multimodal trajectories in EIF tasks.

Trajectory attention. Images of the state s_t from multiple cameras arrive simultaneously, lacking causality among themselves. They are determined solely by the preceding action a_{t-1} and the environment. The same principle applies to actions: the action dimensions of a_t depend only on previous states and actions and are conditionally independent of each other. They do not exhibit a clear causal order. For instance, in a 3D coordinate (x, y, z), it is not evident whether x depends on y or vice versa. Regarding the language prompt, as it is provided by the user, the model's role is to encode and understand it rather than generate it, akin to BERT (Devlin et al., 2019). Vanilla causal attention might impede information flow within each segment of a multimodal EIF trajectory, prohibiting $s_{1,t}$ from attending to $s_{2:M,t}$, and $s_{2,t}$ from attending to $s_{3:M,t}$, and so forth. This phenomenon also manifests in the prompt and action segments.

To address the issue, we propose an efficient Transformer self-attention mechanism for multimodal EIF trajectories, termed trajectory attention. Trajectory attention exhibits two key properties: the inter-segment connections are causal, and the intra-segment connections are bidirectional. Its corresponding attention matrix is illustrated in Figure 3. Following the convention of Transformer attention matrices, we designate the column index (top) as the source of self-attention and the row index (left) as the destination. Consequently, the causal attention matrix has all its lower triangular entries, (i, j) for $i \geq j$, set to one, while the remaining entries are set to zero. Trajectory attention is achieved by unmasking the entries corresponding to (p_i, p_j), $(s_{i,t}, s_{j,t})$, or $(a_{i,t}, a_{j,t})$ for $i < j$. When compared with causal attention, there are $L(L-1)/2 + T\big(M(M-1)/2 + N(N-1)/2\big)$ additional entries joining the self-attention in every Transformer layer, which theoretically explains the effectiveness of trajectory attention. As a result, Astra is designed to process multimodal trajectories at the segment level, aligning well with the EIF setting, as it involves states and actions rather than individual tokens.

Action query. Adapting to the segment-level trajectory attention mechanism, we introduce a segment-level decoding scheme based on learnable action queries. Most prior VLAs generate action dimensions autoregressively, where each action dimension depends on its preceding token (Kim et al.,



(a) Action perturbation.

(b) Construction of positive and negative samples.

Figure 4: Contrastive dynamics learning. (a) In the anchor trajectory (blue arrow), the object on the right is picked up and placed into the bin on the left. A slightly deviated trajectory (green arrow) can still reach the desired destination, enabling action perturbation to be used in constructing positive samples. (b) Given the anchor, we construct a positive sample by applying image augmentation (aug.) and the proposed action perturbation. Negative samples are created by mismatching states and actions from other trajectories.

2024; Brohan et al., 2023a). However, this approach is suboptimal because the embedding of the preceding token is highly dependent on its input and may lack the most relevant information for the action dimension. For instance, when generating $a_{1,t}$, its preceding token is $s_{M,t}$. Although the embedding of $s_{M,t}$ can aggregate information from the past trajectory through self-attention, it remains significantly influenced by its corresponding input image due to various mechanisms in Transformers, such as residual connections. Consequently, it may fail to encapsulate sufficient information necessary for accurately predicting $a_{1,t}$.

To overcome this limitation, we adopt learnable action queries $q_{1:N}$ for individual action dimensions $a_{1:N}$, inspired by DETR (Chen et al., 2024; Carion et al., 2020). Each action query q_i is dedicated to one action dimension a_i and is shared across all timesteps: $q_{i,t=1} = q_{i,t=2} = \cdots = q_{i,t=T}$ for $i \in \{1 \dots N\}$. We argue that this approach can find more relevant information for each action dimension because the action query q_i can exclusively attend to information pertinent to $a_{i,t}$. Since action queries have no associated input token, their embeddings fully retain action dimension information. Moreover, distinct from autoregressive generation, the action queries are independent of each other and can therefore generate all dimensions of an action segment in parallel. Consequently, the decoding procedure operates at the segment level. As the action queries are solely used for information extraction and do not hold any trajectory information, they are masked out from the attention matrix, ensuring that other tokens cannot see them through the self-attention mechanism.

Astra. Combining trajectory attention and action queries, we introduce a novel Transformer architecture named Astra for imitation learning. The training process optimizes the standard behavior cloning objective on offline expert trajectories:

$$\mathcal{L}_{\text{BC}} = \min_{\theta} \sum_{t=1}^{T} -\log \pi_\theta(a_t | p, s_{\le t}, a_{<t}). \quad (1)$$

3.3 Contrastive Dynamics Learning

Dynamics learning encourages the model to learn how the environment transitions from one state to another based on the agent's action, enabling it to make more informed decisions for EIF. Our contrastive dynamics learning (CDL) introduces minimal overhead to the model architecture, requiring only an additional classification head composed of a pooling and linear layer. As illustrated in Figure 4b, we construct positive samples by augmenting the anchor trajectory using standard image augmentation and a novel action perturbation technique. Negative samples are created by mismatching states and actions from different trajectories.

Concretely, we assume that the anchor trajectory is $\tau = (p, s_{t=1}, a_{t=1}, \dots, s_{t=T}, a_{t=T})$. To construct a positive sample, τ^+, we first apply standard computer vision data augmentation techniques to state images, such as random cropping. Additionally, we introduce a novel approach for augmenting actions. The intuition is that a slightly deviated path can still lead the agent to the desired destination, as shown in Figure 4a. To achieve this, we perturb the actions by adding a small amount of random noise. By combining image augmentation and action perturbation, the positive sample is an augmented version of the anchor trajectory.

Subsequently, we create negative trajectories that violate the correct environment dynamics. Given different trajectories from the anchor, $\tau' = (p', s'_{t=1}, a'_{t=1}, \dots, s'_{t=T}, a'_{t=T})$ and

Table 1: Performance comparison of success rate (%) on the VIMA-Bench benchmark. "Attn" stands for attention. "Params" denotes the number of parameters. Gato$*$ modifies the original Gato model by incorporating object tokens.

	Configuration			Generalization Levels			
Model	**Attn Type**	**Visual Token**	**Params**	**L1**	**L2**	**L3**	**L4**
DT	Causal	Single Image	42.0M	56.15	55.38	44.17	12.50
Gato	Causal	Image Patches	42.0M	53.08	50.77	41.67	15.00
Flamingo	Cross	Image Perceiver	42.4M	51.54	52.31	43.33	10.00
Gato$*$	Causal	Object Tokens	42.0M	85.77	82.62	78.92	40.25
VIMA	Cross	Object Tokens	42.4M	87.69	86.92	84.17	47.50
Astra (ours)	Trajectory	Object Tokens	37.8M	**97.08**	**94.62**	**86.17**	**49.50**

$\tau'' = (p'', s''_{t=1}, a''_{t=1}, \ldots, s''_{t=T}, a''_{t=T})$, we mismatch their states and actions with those of the anchor trajectory to construct negative samples: $\tau^- = (p, s'_{t=1}, a_{t=1}, \ldots, s_{t=T}, a''_{t=T})$.

These strong negatives are constructed based on the following three principles discovered during the development of CDL. (1) We refrain from inserting entirely random actions or states, as these have not appeared in the dataset and can be easily identified as negatives. (2) Instead of mismatching only the original states and actions, we also use augmented ones. This prevents models from trivially identifying positive samples by detecting the presence of image augmentation or action perturbation. (3) We avoid merely shuffling states and actions along the time axis, as such negatives are also easily recognizable.

In CDL, Astra encodes the entire multimodal trajectory into a sequence of embeddings. Due to our trajectory attention mechanism, the action tokens at the final timestep attend to the entire trajectory. Their token embeddings are then aggregated into a single trajectory embedding using a simple classification head, consisting of a pooling layer (Li et al., 2023b) and a linear layer, as shown in Figure 2. We denote this process as $f(\cdot)$. Finally, we employ the standard InfoNCE objective (van den Oord et al., 2018) in contrastive learning to train the model to distinguish positive trajectories from negative ones:

$$\begin{aligned} &\mathcal{L}_{\text{CDL}}(\tau, \tau^+, \tau^-) \\ &= -\log \mathbb{E}\left[\frac{s(\tau, \tau^+)}{s(\tau, \tau^+) + \sum_i s(\tau, \tau_i^-)}\right], \end{aligned} \tag{2}$$

where $s(x, y) = \exp(f(x) \cdot f(y))$. Because Astra does not need to decode actions for trajectory encoding, action queries and their corresponding attention entries are not included during CDL.

During training, we incorporate CDL as an auxiliary objective alongside the primary behavior cloning objective to define the overall training loss: $\mathcal{L} = \mathcal{L}_{\text{BC}} + \alpha \mathcal{L}_{\text{CDL}}$.

4 Experiments

4.1 Experimental Setup

We compare our approach with various baseline models across three different benchmarks: VIMA-Bench (Jiang et al., 2022), ManiSkill (Tao et al., 2024; Gu et al., 2023), and CALVIN (Mees et al., 2022). Each benchmark emphasizes different aspects of robot learning. VIMA-Bench investigates multimodal robot learning, where the embodied instructions are multimodal, and evaluates generalization to novel adjectives, nouns, and even meta-tasks. ManiSkill targets everyday objects with complex geometries, testing generalization to objects with unseen geometric and visual attributes. CALVIN, on the other hand, examines long-horizon manipulation tasks, assessing how well models generalize to new environments.

4.2 Implementation Details

We introduce Astra with two model sizes. Astra (38M) is composed of 12 layers, 16 attention heads, and an embedding size of 512. Astra (198M) is composed of 10 layers, 20 attention heads, and an embedding size of 1280. As Astra is a novel architecture for Transformer backbones, it is compatible with various types of vision encoders, language encoders, and action types, as demonstrated by different configurations across the three benchmarks. Within each benchmark, the vision encoder and language encoder remain identical for Astra, DT, Gato, Flamingo, and VIMA, as we focus on comparing the core Transformer backbone. All models are trained using the AdamW optimizer (Loshchilov and Hutter, 2019). Baselines do not incorporate CDL. We provide benchmark-specific implementation details—such as vision encoder, language

Table 2: Performance comparison of success rate (%) on the ManiSkill benchmark. "Traj." is an abbreviation for trajectory attention. "Cont." stands for container. "0", "2-4", "6-8" indicate the number of distractor objects.

	Configuration		Unseen Tasks					Seen Tasks		
Model	**Attn**	**Params**	**Color**	**Size**	**Shape**	**Cont.**	**All**	**0**	**2-4**	**6-8**
RT-1	Causal	46M	27.03	6.36	20.30	0.79	1.27	61.09	39.17	23.40
VIMA	Cross	525M	26.00	26.00	17.20	30.75	19.33	47.93	41.47	36.33
Gato	Causal	198M	46.00	74.00	42.00	44.40	40.00	76.40	73.33	62.67
Astra (ours)	Traj.	198M	**72.00**	**91.00**	**52.40**	**63.43**	**70.67**	**90.93**	**90.53**	**79.07**

Table 3: Performance comparison on the CALVIN benchmark under the most challenging ABC→D setting.

	Configuration		Tasks completed in a row (%)					
Model	**Attn Type**	**Params**	**1**	**2**	**3**	**4**	**5**	**Avg. Len.**
MCIL	—	63.6M	30.4	1.3	0.2	0.0	0.0	0.31
OpenVLA	Causal	7B	32.4	4.2	1.4	0.5	0.3	0.39
RT-1	Causal	46M	53.3	22.2	9.4	3.8	1.3	0.90
MDT	Cross	75.1M	61.7	41.6	23.8	14.7	8.7	1.54
RoboFlamingo	Cross	1B	82.4	61.9	46.6	33.1	23.5	2.48
GR-1	Causal	21.3M	85.4	71.2	59.6	49.7	40.1	3.06
VIMA	Cross	42.4M	64.1	47.6	34.6	29.2	23.7	1.99
Flamingo	Cross	42.4M	69.9	53.2	39.1	31.2	24.6	2.18
DT	Causal	44.1M	74.1	56.1	42.0	32.2	25.6	2.30
Gato	Causal	44.1M	77.3	57.0	44.4	33.4	26.1	2.38
Astra (ours)	Trajectory	37.8M	**89.7**	**79.2**	**65.8**	**52.4**	**42.3**	**3.29**

encoder, and hyperparameters—in their respective sections.

4.3 VIMA-Bench

VIMA-Bench (Jiang et al., 2022) focuses on multimodal robot learning, where the instructions are multimodal. We summarize the results in Table 1. It evaluates generalization capabilities across four levels: placement generalization (L1), combinatorial generalization (L2), novel object generalization (L3), and novel task generalization (L4). Each level presents increasing difficulty, with placement generalization involving only the randomization of object positions, combinatorial generalization recombining seen adjectives and nouns, novel object generalization introducing unseen adjectives or nouns, and novel task generalization incorporating entirely new meta-tasks.

The vision encoder is ViT (Dosovitskiy et al., 2021) and the prompt encoder is T5 (Raffel et al., 2020). Baselines in VIMA-Bench explore different methods of encoding images, such as patch tokens and object tokens. Astra uses the best-performing object tokens as visual inputs. Discrete $SE(2)$ actions are used in this benchmark. All models are trained for 10 epochs with learning rate $= 1 \times 10^{-4}$ and weight decay $= 0.1$. Each task is evaluated using 100 trials.

According to the VIMA-Bench paper, models with cross-attention and self-attention achieve comparable performance only when the model size exceeds 42M parameters. Therefore, we adopt this configuration for all baselines to strike a balance between model size and performance. We intentionally reduce our model size by approximately 10%. Despite the smaller model size, Astra achieves improved performance across all four generalization levels, demonstrating its efficiency.

4.4 ManiSkill

In the ManiSkill2 environment (Tao et al., 2024; Gu et al., 2023), we evaluate one of the most commonly utilized skills, "pick and place", using everyday objects with complex geometries, as detailed in Table 2. Its goal is to pick up an object and place it into a container. This benchmark spans generalization levels L1 to L3 in VIMA-Bench: all items are randomly placed and the robot pose is randomly initialized, thereby including placement generalization; novel objects are also introduced as unseen tasks, facilitating both combinatorial and novel object generalization. To test these general-

Table 4: Ablation study of the components in Astra. In ManiSkill, we compare seen tasks with 2-4 distractors.

Configuration	VIMA-Bench				ManiSkill		
	L1	L2	L3	L4	Easy	Medium	Hard
Astra w/ CDL	**97.08**	**94.62**	**86.17**	49.50	**93.83**	**89.71**	**83.50**
Astra	94.69	92.15	85.83	**50.00**	91.33	88.57	75.00
w/o Trajectory Attention	91.23	89.31	84.42	47.50	86.33	83.71	76.00
w/o Action Query	89.08	84.85	82.58	45.25	86.00	80.86	69.00
w/o Both	85.77	82.62	78.92	40.25	72.33	76.00	67.00

ization capabilities, we limit the training set to 15 tasks and evaluate the models on 34 tasks. Training data includes "2-4" distractors.

This benchmark consists of five types of unseen tasks. The first three types involve picked objects with unseen colors, sizes, and shapes. For example, the apple is part of the training data, while the bowl, with its novel shape, is not. The fourth type introduces unseen containers. The fifth type composes all of the first four types. A distractor is an item that is neither the picked object nor the container. They are randomly sampled from a diverse pool of items. For all five types of "unseen tasks", we randomly sample and place 2-4 distractors. For seen objects, we explore whether the number of distractors can impact the models' performance.

We utilize ResNet (He et al., 2016) for images and T5 (Raffel et al., 2020) for language prompts. Actions are discrete 6D poses. The models are trained for 5 epochs with learning rate $= 1 \times 10^{-4}$ and weight decay $= 1 \times 10^{-4}$. We conduct 50 trials for each task, and each trial is limited to 100 timesteps before a timeout.

We experiment with the same number of Transformer layers for Astra (198M), VIMA, and Gato. RT-1 retains its original configuration with 46M parameters. Our model matches the model size of Gato while achieving superior performance. Due to the additional cross-attention layers in VIMA, its model size is significantly larger, which may have contributed to overfitting in this experiment.

4.5 CALVIN

The CALVIN benchmark (Mees et al., 2022) focuses on long-horizon manipulation tasks. Performance comparison results are presented in Table 3. During each evaluation session, the model is prompted with five random tasks. The session terminates as soon as a task fails, and the remaining tasks are not attempted. Performance is measured by the number of tasks successfully completed in a row. The benchmark provides three different experimental settings: D→D, ABCD→D, and ABC→D, where each letter represents a distinct environment. For example, in the ABC→D setting, the model is trained on data from environments A, B, and C, but evaluated in environment D. This setting thus assesses zero-shot generalization to new environments. We compare our model to baselines in this most challenging ABC→D experiment. Since only 1% of the training dataset for ABC→D is annotated with language prompts, we utilize this language-annotated subset for training, further increasing the difficulty of the task.

We use MAE-ViT vision encoder (He et al., 2022) and CLIP language encoder (Radford et al., 2021), following GR-1 (Wu et al., 2024). All models use continuous 6D pose actions and are trained from scratch for 20 epochs with learning rate $= 9 \times 10^{-4}$ and weight decay $= 1 \times 10^{-4}$.

Since the rollout of a trajectory terminates as soon as any of the five tasks fail, completing all five tasks is highly challenging. Our model can complete longer task sequences, highlighting its effectiveness in generalizing to new environments.

4.6 Ablation Study

We analyze the effects of the proposed components of Astra, including trajectory attention, action queries, and contrastive dynamics learning, as shown in Table 4. Since our trajectory data augmentation method is only applied in conjunction with CDL, it is not utilized for "Astra" (rows 2-5). In ManiSkill, we provided more granular results across three difficulty levels (Appendix A.6).

CDL proves effective in enhancing imitation learning, particularly for the first three generalization levels, as it enables the model to better learn environment dynamics. The lower performance on L4 may be attributed to CDL's training data, which includes only seen nouns and adjectives, thereby enhancing the performance on seen meta-tasks at

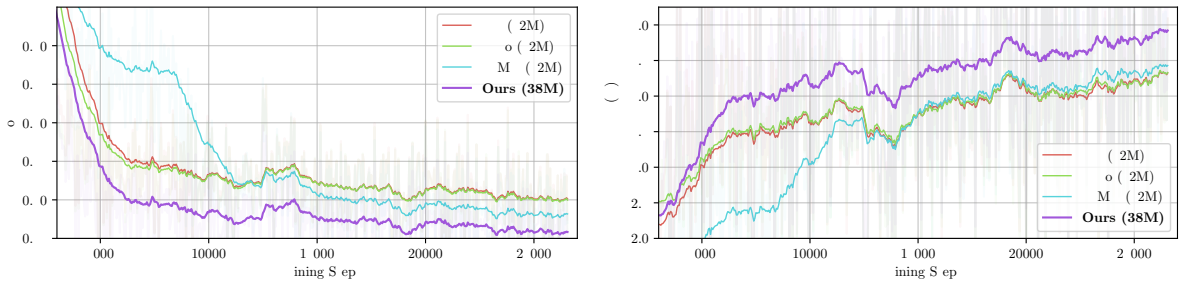


Figure 5: Loss and accuracy curves during training on VIMA-Bench.

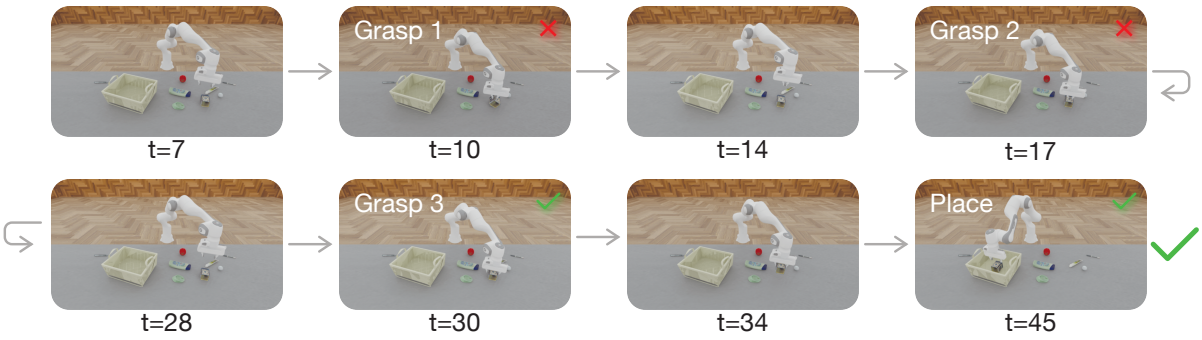


Figure 6: An example of instantaneous regrasp. The task is to "pick blue tea box and place into clear box." Three grasp attempts were completed within only 30 steps, a capability not observed in the baseline models.

the expense of generalizability to novel ones. The removal of trajectory attention results in a noticeable decrease in success rates across all levels, underscoring its crucial role in processing segmented multimodal trajectories. Similarly, the absence of action queries leads to reduced success rates, highlighting its importance in enhancing information extraction for action generation. When both components are removed, the model reverts to a typical token-level autoregressive model, akin to Gato.

4.7 Qualitative Analysis

We present the loss and accuracy curves for the models on VIMA-Bench in Figure 5. Despite having a model size approximately 10% smaller, our model exhibits a faster convergence rate. The substantially lower loss and higher accuracy account for Astra's superior performance. Specifically, trajectory attention facilitates improved information flow within each segment, while action queries more effectively extract information relevant to individual action dimensions. The Flamingo baseline is excluded from the figure due to its significantly worse performance.

In ManiSkill, we identified a crucial distinction between Astra's capabilities and those of the baselines: Astra masters "instantaneous regrasp". Figure 6 illustrates the most challenging task in ManiSkill. Baseline models often struggle to recognize failed grasps. In contrast, our Astra model promptly detects a failed grasp and repeatedly attempts to grasp the object until successful. A more detailed description is provided in Appendix A.7.

5 Conclusion

This paper introduces Astra, an efficient Transformer architecture designed for multimodal trajectories in embodied instruction following tasks. Astra distinguishes itself from standard Transformer decoders through two novel components: trajectory attention and action queries. Trajectory attention harnesses the unique characteristics of multimodal EIF trajectories, facilitating enhanced information flow among tokens within each segment. Combined with our action queries, which enable parallel information extraction for individual dimensions, Astra achieves segment-level decoding. Furthermore, we incorporate a contrastive dynamics learning objective to explicitly train the model to learn environment dynamics, which also improves multimodal alignment. This further elevates Astra's performance in imitation learning. Comprehensive experiments across three large-scale benchmarks demonstrate substantial performance gains by Astra. Detailed ablation studies and qualitative analyses further validate its effectiveness.

Limitations

Although real-world robot evaluation is a common practice, there is no widely accepted real-world benchmark for embodied instruction following tasks due to the difficulty of precisely replicating environmental setups across different institutions. Benchmarks based on simulated environments offer the advantage of highly controllable settings, eliminating variable factors that can lead to imprecise measurements. Therefore, we focus on well-established simulated benchmarks that evaluate various aspects of model performance and generalization. Because 3D information can be more informative than 2D image inputs for EIF tasks, Astra can also be extended to integrate 3D vision modules to further improve performance, as the Astra backbone is compatible with various vision, language, or action modules.

Acknowledgements

The work described in this paper was partially supported by the Research Grants Council of the Hong Kong Special Administrative Region, China (CUHK 2410072, RGC R1015-23) and Huawei Technology Investment, Limited (CUHK 6906061).

References

Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob L. Menick, Sebastian Borgeaud, and 8 others. 2022. Flamingo: a visual language model for few-shot learning. In NeurIPS.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. 2021. Structured denoising diffusion models in discrete state-spaces. In NeurIPS, pages 17981–17993.

Anas Awadalla, Irena Gao, Josh Gardner, Jack Hessel, Yusuf Hanafy, Wanrong Zhu, Kalyani Marathe, Yonatan Bitton, Samir Yitzhak Gadre, Shiori Sagawa, Jenia Jitsev, Simon Kornblith, Pang Wei Koh, Gabriel Ilharco, Mitchell Wortsman, and Ludwig Schmidt. 2023. Openflamingo: An open-source framework for training large autoregressive vision-language models. CoRR, abs/2308.01390.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Ming-Hsuan Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, and 8 others. 2025. Qwen2.5-vl technical report. CoRR, abs/2502.13923.

Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, Thomas Unterthiner, Daniel Keysers, Skanda Koppula, Fangyu Liu, Adam Grycner, Alexey A. Gritsenko, Neil Houlsby, Manoj Kumar, Keran Rong, and 16 others. 2024. Paligemma: A versatile 3b VLM for transfer. CoRR, abs/2407.07726.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, and 5 others. 2024. π_0: A vision-language-action flow model for general robot control. CoRR, abs/2410.24164.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alexander Herzog, Jasmine Hsu, Brian Ichter, and 35 others. 2023a. RT-2: vision-language-action models transfer web knowledge to robotic control. CoRR, abs/2307.15818.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil J. Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, and 32 others. 2023b. RT-1: robotics transformer for real-world control at scale. In Robotics: Science and Systems.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. 2020. End-to-end object detection with transformers. In ECCV (1), volume 12346 of Lecture Notes in Computer Science, pages 213–229. Springer.

Chilam Cheang, Guangzeng Chen, Ya Jing, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Hongtao Wu, Jiafeng Xu, Yichu Yang, Hanbo Zhang, and Minzhao Zhu. 2024. GR-2: A generative video-language-action model with web-scale knowledge for robot manipulation. CoRR, abs/2410.06158.

Yevgen Chebotar, Quan Vuong, Alex Irpan, Karol Hausman, Fei Xia, Yao Lu, Aviral Kumar, Tianhe Yu, Alexander Herzog, Karl Pertsch, Keerthana Gopalakrishnan, Julian Ibarz, Ofir Nachum, Sumedh Sontakke, Grecia Salazar, Huong T. Tran, Jodilyn Peralta, Clayton Tan, Deeksha Manjunath, and 6 others. 2023. Q-transformer: Scalable offline reinforcement learning via autoregressive q-functions. CoRR, abs/2309.10150.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Michael Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. 2021. Decision transformer: Reinforcement learning via sequence modeling. In NeurIPS, pages 15084–15097.

Wei Chen, Jinjin Luo, Fan Zhang, and Zijian Tian. 2024. A review of object detection: Datasets, performance evaluation, architecture, applications and current trends. Multim. Tools Appl., 83(24):65603–65661.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In NAACL-HLT (1), pages 4171–4186. Association for Computational Linguistics.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In NeurIPS, pages 13042–13054.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2021. An image is worth 16x16 words: Transformers for image recognition at scale. In ICLR. OpenReview.net.

Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, and 3 others. 2023. Palm-e: An embodied multimodal language model. In ICML, volume 202 of Proceedings of Machine Learning Research, pages 8469–8488. PMLR.

Yilun Du, Sherry Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Josh Tenenbaum, Dale Schuurmans, and Pieter Abbeel. 2023. Learning universal policies via text-guided video generation. In NeurIPS.

Pete Florence, Corey Lynch, Andy Zeng, Oscar A. Ramirez, Ayzaan Wahid, Laura Downs, Adrian Wong, Johnny Lee, Igor Mordatch, and Jonathan Tompson. 2021. Implicit behavioral cloning. In CoRL, volume 164 of Proceedings of Machine Learning Research, pages 158–168. PMLR.

Akash Ghosh, Arkadeep Acharya, Sriparna Saha, Vinija Jain, and Aman Chadha. 2024. Exploring the frontier of vision-language models: A survey of current methodologies and future directions. CoRR, abs/2404.07214.

Jiayuan Gu, Fanbo Xiang, Xuanlin Li, Zhan Ling, Xiqiang Liu, Tongzhou Mu, Yihe Tang, Stone Tao, Xinyue Wei, Yunchao Yao, Xiaodi Yuan, Pengwei Xie, Zhiao Huang, Rui Chen, and Hao Su. 2023. Maniskill2: A unified benchmark for generalizable manipulation skills. In International Conference on Learning Representations.

Abhishek Gupta, Vikash Kumar, Corey Lynch, Sergey Levine, and Karol Hausman. 2019. Relay policy learning: Solving long-horizon tasks via imitation and reinforcement learning. In 3rd Annual Conference on Robot Learning, CoRL 2019, Osaka, Japan, October 30 - November 1, 2019, Proceedings, volume 100 of Proceedings of Machine Learning Research, pages 1025–1037. PMLR.

Danijar Hafner, Timothy P. Lillicrap, Jimmy Ba, and Mohammad Norouzi. 2020. Dream to control: Learning behaviors by latent imagination. In ICLR. OpenReview.net.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross B. Girshick. 2022. Masked autoencoders are scalable vision learners. In CVPR, pages 15979–15988. IEEE.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In CVPR, pages 770–778. IEEE Computer Society.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. In NeurIPS.

Brian Ichter, Anthony Brohan, Yevgen Chebotar, Chelsea Finn, Karol Hausman, Alexander Herzog, Daniel Ho, Julian Ibarz, Alex Irpan, Eric Jang, Ryan Julian, Dmitry Kalashnikov, Sergey Levine, Yao Lu, Carolina Parada, Kanishka Rao, Pierre Sermanet, Alexander Toshev, Vincent Vanhoucke, and 26 others. 2022. Do as I can, not as I say: Grounding language in robotic affordances. In CoRL, volume 205 of Proceedings of Machine Learning Research, pages 287–318. PMLR.

Michael Janner, Qiyang Li, and Sergey Levine. 2021. Offline reinforcement learning as one big sequence modeling problem. In NeurIPS, pages 1273–1286.

Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. 2022. VIMA: general robot manipulation with multimodal prompts. CoRR, abs/2210.03094.

Siddharth Karamcheti, Suraj Nair, Annie S. Chen, Thomas Kollar, Chelsea Finn, Dorsa Sadigh, and Percy Liang. 2023. Language-driven representation learning for robotics. In Robotics: Science and Systems.

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

Jiachen Li, Qiaozi Gao, Michael Johnston, Xiaofeng Gao, Xuehai He, Hangjie Shi, Suhaila Shakiah, Reza Ghanadan, and William Yang Wang. 2024. Mastering robot manipulation with multimodal prompts through pretraining and multi-task fine-tuning. In ICML. OpenReview.net.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven C. H. Hoi. 2023a. BLIP-2: bootstrapping language-image pre-training with frozen image encoders and large language models. In ICML, volume 202 of Proceedings of Machine Learning Research, pages 19730–19742. PMLR.

Xinghang Li, Minghuan Liu, Hanbo Zhang, Cunjun Yu, Jie Xu, Hongtao Wu, Chilam Cheang, Ya Jing, Weinan Zhang, Huaping Liu, Hang Li, and Tao Kong. 2023b. Vision-language foundation models as effective robot imitators. CoRR, abs/2311.01378.

Junyang Lin, Rui Men, An Yang, Chang Zhou, Ming Ding, Yichang Zhang, Peng Wang, Ang Wang, Le Jiang, Xianyan Jia, Jie Zhang, Jianwei Zhang, Xu Zou, Zhikang Li, Xiaodong Deng, Jie Liu, Jinbao Xue, Huiling Zhou, Jianxin Ma, and 6 others. 2021. M6: A chinese multimodal pretrainer. CoRR, abs/2103.00823.

Fangchen Liu, Hao Liu, Aditya Grover, and Pieter Abbeel. 2022. Masked autoencoding for scalable and generalizable decision making. In NeurIPS.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In ICLR (Poster). OpenReview.net.

Guanxing Lu, Ziwei Wang, Changliu Liu, Jiwen Lu, and Yansong Tang. 2025. Thinkbot: Embodied instruction following with thought chain reasoning. In ICLR. OpenReview.net.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In NeurIPS, pages 13–23.

Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. 2023a. VIP: towards universal visual reward and representation via value-implicit pre-training. In ICLR. OpenReview.net.

Yueen Ma, Dafeng Chi, Jingjing Li, Kai Song, Yuzheng Zhuang, and Irwin King. 2024a. VOLTA: improving generative diversity by variational mutual information maximizing autoencoder. In NAACL-HLT (Findings), pages 364–378. Association for Computational Linguistics.

Yueen Ma, Zixing Song, Xuming Hu, Jingjing Li, Yifei Zhang, and Irwin King. 2023b. Graph component contrastive learning for concept relatedness estimation. In AAAI, pages 13362–13370. AAAI Press.

Yueen Ma, Zixing Song, Yuzheng Zhuang, Jianye Hao, and Irwin King. 2024b. A survey on vision-language-action models for embodied AI. CoRR, abs/2405.14093.

Yueen Ma, Da Yan, Cheng Long, D. Rangaprakash, and Gopikrishna Deshpande. 2021. Predicting autism spectrum disorder from brain imaging data by graph convolutional network. In IJCNN, pages 1–8. IEEE.

Oier Mees, Lukás Hermann, Erick Rosete-Beas, and Wolfram Burgard. 2022. CALVIN: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. IEEE Robotics Autom. Lett., 7(3):7327–7334.

Vincent Micheli, Eloi Alonso, and François Fleuret. 2023. Transformers are sample-efficient world models. In ICLR. OpenReview.net.

Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. 2022. R3M: A universal visual representation for robot manipulation. In CoRL, volume 205 of Proceedings of Machine Learning Research, pages 892–909. PMLR.

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, Albert Tung, Alex Bewley, Alexander Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anchit Gupta, Andrew E. Wang, Anikait Singh, and 260 others. 2024. Open x-embodiment: Robotic learning datasets and RT-X models : Open x-embodiment collaboration. In ICRA, pages 6892–6903. IEEE.

Minting Pan, Xiangming Zhu, Yunbo Wang, and Xiaokang Yang. 2022. Iso-dream: Isolating and leveraging noncontrollable visual dynamics in world models. In NeurIPS.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision. In ICML, volume 139 of Proceedings of Machine Learning Research, pages 8748–8763. PMLR.

Ilija Radosavovic, Tete Xiao, Stephen James, Pieter Abbeel, Jitendra Malik, and Trevor Darrell. 2022. Real-world robot learning with masked visual pre-training. In CoRL, volume 205 of Proceedings of Machine Learning Research, pages 416–426. PMLR.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. J. Mach. Learn. Res., 21:140:1–140:67.

Scott E. Reed, Konrad Zolna, Emilio Parisotto, Sergio Gómez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky,

Jackie Kay, Jost Tobias Springenberg, Tom Eccles, Jake Bruce, Ali Razavi, Ashley Edwards, Nicolas Heess, Yutian Chen, Raia Hadsell, Oriol Vinyals, Mahyar Bordbar, and Nando de Freitas. 2022. A generalist agent. Trans. Mach. Learn. Res., 2022.

Jan Robine, Marc Höftmann, Tobias Uelwer, and Stefan Harmeling. 2023. Transformer-based world models are happy with 100k interactions. In ICLR. OpenReview.net.

Zixing Song, Yifei Zhang, and Irwin King. 2023a. No change, no gain: Empowering graph neural networks with expected model change maximization for active learning. In NeurIPS.

Zixing Song, Yifei Zhang, and Irwin King. 2023b. Optimal block-wise asymmetric graph construction for graph-based semi-supervised learning. In NeurIPS.

Austin Stone, Ted Xiao, Yao Lu, Keerthana Gopalakrishnan, Kuang-Huei Lee, Quan Vuong, Paul Wohlhart, Brianna Zitkovich, Fei Xia, Chelsea Finn, and Karol Hausman. 2023. Open-world object manipulation using pre-trained vision-language models. CoRR, abs/2303.00905.

Yanchao Sun, Shuang Ma, Ratnesh Madaan, Rogerio Bonatti, Furong Huang, and Ashish Kapoor. 2023. SMART: self-supervised multi-task pretraining with control transformers. In ICLR. OpenReview.net.

Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao, Xinsong Lin, Yulin Liu, Tse-kai Chan, Yuan Gao, Xuanlin Li, Tongzhou Mu, Nan Xiao, Arnav Gurha, Zhiao Huang, Roberto Calandra, Rui Chen, Shan Luo, and Hao Su. 2024. Maniskill3: GPU parallelized robotics simulation and rendering for generalizable embodied AI. CoRR, abs/2410.00425.

Aäron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. CoRR, abs/1807.03748.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In NIPS, pages 5998–6008.

Sai Vemprala, Rogerio Bonatti, Arthur Bucker, and Ashish Kapoor. 2023. Chatgpt for robotics: Design principles and model abilities. CoRR, abs/2306.17582.

Hongtao Wu, Ya Jing, Chilam Cheang, Guangzeng Chen, Jiafeng Xu, Xinghang Li, Minghuan Liu, Hang Li, and Tao Kong. 2024. Unleashing large-scale video generative pre-training for visual robot manipulation. In ICLR. OpenReview.net.

Xuan Xiao, Jiahang Liu, Zhipeng Wang, Yanmin Zhou, Yong Qi, Shuo Jiang, Bin He, and Qian Cheng. 2025. Robot learning in the era of foundation models: a survey. Neurocomputing, 638:129963.

Zhiyuan Xu, Kun Wu, Junjie Wen, Jinming Li, Ning Liu, Zhengping Che, and Jian Tang. 2024. A survey on robotics with foundation models: toward embodied AI. CoRR, abs/2402.02385.

Jingyi Zhang, Jiaxing Huang, Sheng Jin, and Shijian Lu. 2024a. Vision-language models for vision tasks: A survey. IEEE Trans. Pattern Anal. Mach. Intell., 46(8):5625–5644.

Yifei Zhang, Hao Zhu, Zixing Song, Yankai Chen, Xinyu Fu, Ziqiao Meng, Piotr Koniusz, and Irwin King. 2024b. Geometric view of soft decorrelation in self-supervised learning. In KDD, pages 4338–4349. ACM.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. 2023. Learning fine-grained bimanual manipulation with low-cost hardware. In Robotics: Science and Systems.

A Appendix

A.1 Notation

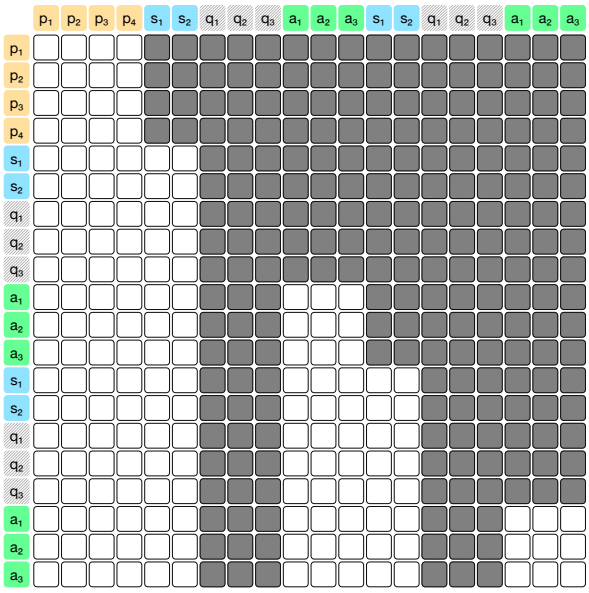
In this paper, we use the following notation:

Symbol	Description
Basic symbols	
τ	EIF trajectory
p	Language instruction/prompt
s	State (visual observation)
a	Action
q	Action queries (shared across timesteps)
Segment-level symbols	
s_t	State segment at timestep t
a_t	Action segment at timestep t
$s_{\leq t}$	All states up to and including timestep t
$a_{<t}$	All actions before timestep t
t	Timestep index ($1 \leq t \leq T$)
T	Total number of timesteps in a trajectory
Token-level symbols	
p_i	The i-th prompt token
$s_{i,t}$	The i-th state token at timestep t
$a_{i,t}$	The i-th action dimension at timestep t
q_i / $q_{i,t}$	The i-th action query (shared across timesteps)
Token slices	
$p_{1:L}$	Prompt tokens 1 through L
$s_{1:M,t}$	State tokens 1 through M at timestep t
$a_{1:N,t}$	Action dimensions 1 through N at timestep t
$q_{1:N}$	Action queries 1 through N (shared across timesteps)
L	Number of language instruction tokens
M	Number of state tokens per segment
N	Number of action dimensions/queries per segment
Contrastive dynamics learning	
τ^+	Positive trajectory sample
τ^-	Negative trajectory sample
τ', τ''	Alternative trajectories different from τ
Model	
π	Policy
π_θ	Policy parameterized by θ
θ	Model parameters
$f(\cdot)$	Trajectory encoding model

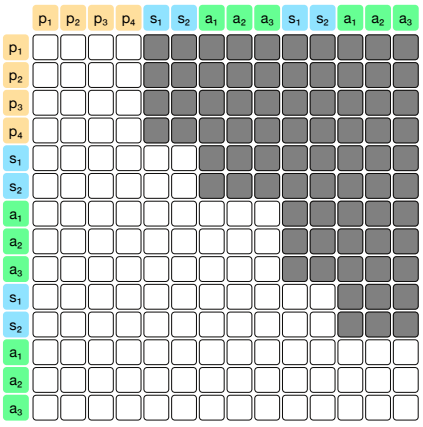
A.2 Trajectory Attention Matrix

The trajectory attention is implemented with its corresponding attention mask, as illustrated in Figure 7. Output tokens on the left attend to input tokens at the top. For example, in the first row, the token p_1 can attend to tokens (p_1, p_2, p_3, p_4) but not to any other subsequent tokens. Consequently, tokens starting from s_1 onward are masked out. During action generation in imitation learning, action queries can attend to all tokens up to the current state, while no other tokens can attend to action queries. Therefore, action query tokens are completely masked from the input.

Although we can use the decoding attention matrix for both decoding and encoding, utilizing the encoding attention matrix can avoid computational overhead for processing action queries. Regardless of whether the decoding or encoding matrix is employed, the resulting embeddings for the encoded trajectory remain identical in contrastive dynamics learning.



(a) Trajectory attention matrix for decoding.



(b) Trajectory attention matrix for encoding.

Figure 7: The attention matrices of trajectory attention. Input tokens are at the top, and output tokens are on the left. Dark boxes represent masked entries in the attention matrices. The decoding attention matrix is utilized in imitation learning, whereas the encoding attention matrix is employed in contrastive dynamics learning.

A.3 Trajectory Attention Visualization

We present a visualization of the attention matrices from the top layer of Astra, depicted in Figure 8. Prompt tokens exhibit similar attention values, while state and action tokens from more recent timesteps receive higher attention weights compared to those from earlier in the sequence. This observation aligns with the expectation that the latest timestep provides the most informative con-

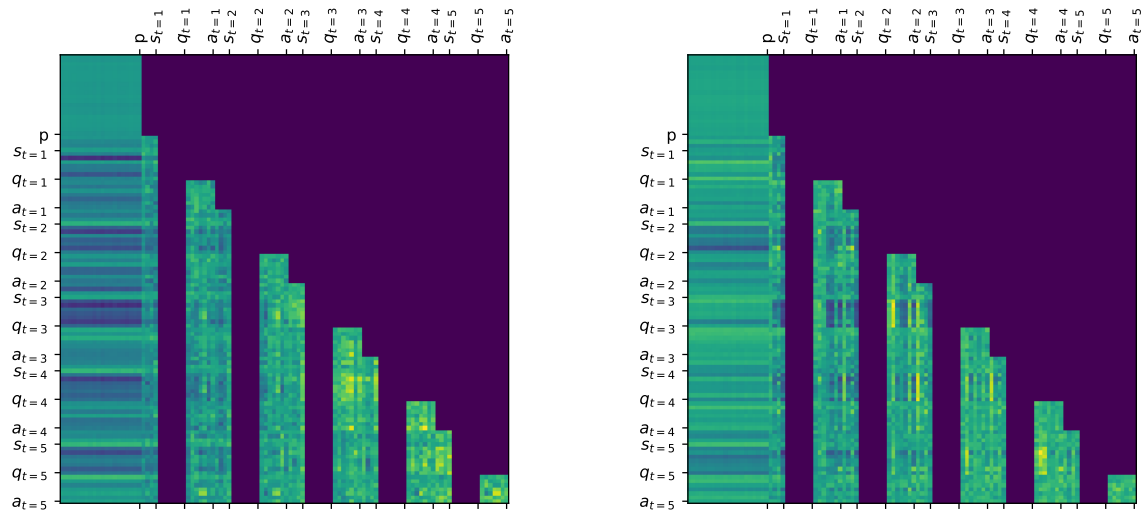


Figure 8: Visualization of two trained trajectory attention matrices over five timesteps. Brighter cells indicate higher attention weights. Tick labels are displayed for the last token of each segment.

text for generating the next action. Moreover, several attention weights above the main diagonal are strongly activated, suggesting that the additional attention connections enabled by our trajectory attention mechanism are beneficial. In the second matrix, a clear distinction emerges between the attention weights for the output state tokens and query tokens. This distinction highlights that action queries extract information differently from state tokens, elucidating their role in improving action generation.

A.4 Integration with Large VLAs

Due to the trajectory attention mechanism and action queries in Astra, large VLA models that rely on autoregressive modeling cannot directly adopt this architecture without significant retraining. However, there are alternative methods to leverage the Astra architecture in large VLAs. For instance, it can be incorporated as an action prediction head by attaching it to the top of pretrained VLMs. The integration of Astra with large VLA models presents a promising direction for future research.

A.5 ManiSkill Generalization Levels

For Astra, Gato, and VIMA in Table 2, unseen shape proves to be the most challenging generalization level of ManiSkill, followed by unseen containers. VIMA also exhibits volatility when dealing with small objects from the "Size" level. RT-1, in particular, struggles with identifying the container when an unseen container is introduced. It is important to note that not all seen target objects (the object to be picked) have a generalized version. For example, a strawberry is a seen target object that is difficult to grasp, but there is no oversized strawberry for the "Shape" level. Consequently, models may achieve a higher success rate on some generalization levels than on seen tasks. Furthermore, since unseen color and size are part of the mixture, the success rate in "All" is not as low as in "Shape" or "Container". The introduction of more distractors in the scene increases the likelihood of collisions and causes additional difficulty in grasping the objects. However, this negative effect is not severe enough to considerably degrade the performance.

A.6 ManiSkill Difficulty Levels

The ablation study in the ManiSkill environment is reported by difficulty level, as shown in Table 4. In simple terms, easy tasks involve spherical, regular-sized target objects, such as a baseball. The medium difficulty level includes elongated or small target objects, such as a banana or a strawberry. Hard tasks encompass oversized, non-spherical, or thin objects, such as a tea box or knife.

Easy tasks include spherical, regular-sized objects. Round objects are easier to pick because the robot arm can close the gripper in any direction. Size also has a significant impact on the success rate because oversized objects require more precise grasp poses. If a grasp is not precise, the two fingers of the gripper may collide with the object and not be able to reach down on the object. On the other hand, small objects can increase the difficulty

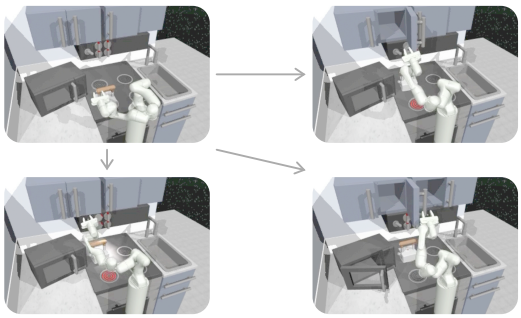


Figure 9: Astra in Franka Kitchen. The model completes four random tasks. The top left image shows the initial state and the other three images are three different final states.

Figure 10: Astra in Push-T. The gray T is the object, and the green T is its target position. The model controls the blue dot to push the T-shaped object towards the target position. The red cross is the cursor. The images on the left and right are two different initial states, and the middle image is the final state where the object perfectly overlaps with the target position.

because the gripper might miss them if the grasp is slightly off. An elongated object, such as a remote controller or a banana, must be picked up "across" the object rather than "along" the object. Consequently, we define the medium difficulty level as the objects that are too big, too small, or elongated. Hard tasks involve non-spherical, oversized, or thin objects, such as a tea box and a knife. A tea box, being both non-spherical and oversized, requires the robot arm to grasp it precisely parallel to its sides rather than diagonally. A knife can be hard since it is very thin and close to the desk. During grasping, the gripper may collide with the desk, further increasing the difficulty.

A.7 Instantaneous Regrasp

We provide a more detailed explanation for the example in Figure 6. Only three of the four grasp attempts are shown, with the third one omitted due to page limitations. Therefore, the grasp labeled as "Grasp 3" in the figure is, in fact, the fourth grasp attempt. The first grasp was unsuccessful because one finger of the gripper collided with the blue tea box, and the grasp slipped. Subsequently, Astra swiftly initiated two additional grasps; however, the gripper closed too early, resulting in collisions with the tea box again. Shortly after the second and third failures, the gripper's fingers successfully reached down to opposite sides of the blue tea box, completing the fourth regrasp. Remarkably, all four grasp attempts were executed within a mere 30 timesteps, a feat not achieved by any of the baseline methods.

A.8 Additional Related Work

VLAs (Ma et al., 2024b,b) have recently emerged as a new type of multimodal model for EIF tasks in the field of embodied AI. MOO (Stone et al., 2023) introduced multi-modal prompt capability to RT-1, while Q-Transformer (Chebotar et al., 2023) adapted RT-1 to the Q-learning setting. RoboFlamingo (Li et al., 2023b) constructed a VLA based on the existing Flamingo VLM (Alayrac et al., 2022; Awadalla et al., 2023). ACT (Zhao et al., 2023) adopts the DETR framework for robotics tasks but utilizes fixed position embeddings at the timestep level. VLAs can also be integrated with high-level planners to address long-horizon robotics tasks, as demonstrated by SayCan (Ichter et al., 2022), PaLM-E (Driess et al., 2023), and ChatGPT for Robotics (Vemprala et al., 2023). Similar to other multimodal models, the success of VLAs is predicated on a foundation of numerous prior unimodal models and a variety of deep learning techniques (Ma et al., 2023b, 2024a; Song et al., 2023b; Zhang et al., 2024b; Song et al., 2023a; Ma et al., 2021).

In addition to the primary learning objective, auxiliary or pretraining objectives have proven useful in further enhancing model performance. The success of masked language modeling, as initially proposed in BERT (Devlin et al., 2019), has prompted the adoption of similar objectives in various domains. In computer vision models and VLMs, representative works like MAE (He et al., 2022) and ViLBERT (Lu et al., 2019) have employed comparable strategies. VLAs have also utilized masked modeling objectives for their vision encoders, such as MVP (Radosavovic et al., 2022), Voltron (Karamcheti et al., 2023), GR-1 (Wu et al., 2024). While these approaches have proven beneficial for the vision encoder, they often overlook the crucial alignment between different modalities.

Dynamics learning has long been recognized as a powerful technique for improving the performance of robot learning models. Dreamer (Hafner

Table 5: Performance comparison (%) in Franka Kitchen.

Model	Configuration		Continuous Action					Discrete Action				
	Attn Type	Params	$p1$	$p2$	$p3$	$p4$	Mean	$p1$	$p2$	$p3$	$p4$	Mean
Diffusion	Causal	43M	**100**	94	72	34	76	54	18	10	2	21
VIMA	Cross	113M	92	90	80	66	82	90	72	48	16	57
Gato	Causal	43M	**100**	98	84	62	86	**100**	88	**68**	38	74
Astra (ours)	Traj.	43M	**100**	**100**	**94**	**70**	**91**	**100**	**94**	**68**	**52**	**79**

Table 6: Performance comparison in Push-T (Discrete Action).

Model	Attn	Params	Score
Diffusion	Causal	43M	83.89
VIMA	Cross	26M	91.09
Gato	Causal	19M	90.43
Astra (ours)	Traj.	19M	**94.11**

et al., 2020) was a pioneering work in this domain, inspiring several follow-up methods, including IsoDream (Pan et al., 2022), TWM (Robine et al., 2023), and IRIS (Micheli et al., 2023).

A.9 Additional Baselines

The "Diffusion" baseline in Appendix A.10 is based on a causal Transformer backbone and is trained with modified training objectives: DDPM (Ho et al., 2020) for continuous actions and D3PM (Austin et al., 2021) for discrete actions. The training losses are defined as follows:

$$\begin{aligned} \mathcal{L}_{\text{DDPM}} &= \text{MSE}\left(\varepsilon^k, \varepsilon_\theta(\mathbf{x}^0 + \varepsilon^k, k)\right), \\ \mathcal{L}_{\text{D3PM}} &= \text{CE}\left(\varepsilon^k, \varepsilon_\theta(\mathbf{x}^0 + \varepsilon^k, k)\right), \end{aligned} \tag{3}$$

where $\mathbf{x}^0$ is the original action and ε^k is the noise of the k-th iteration; ε_θ is the Transformer backbone.

We have also experimented with π_0 (Black et al., 2024) on the CALVIN benchmark. However, its performance was notably poor, and as a result, we decided not to include it in Table 3.

A.10 Additional Benchmarks

Franka Kitchen. Franka Kitchen (Gupta et al., 2019) includes five skills that span seven specific tasks within the scene. The "turn knob" skill involves turning the oven knob to activate either the top or bottom burner. "Toggle switch" involves turning on the light switch. "Slide door open" requires opening the slide cabinet, while "swing door open" involves opening either the left hinge cabinet or the microwave door by the door handle. The "lift by handle" skill entails moving the kettle by its handle. Figure 9 provides examples of executions by Astra in Franka Kitchen.

Performance is measured by the completion of multi-stage tasks, as summarized in Table 5. Results are averaged over 50 runs. In each run, the models are required to complete four random tasks within 280 steps. pi means the model has completed i tasks and thus reached the i-th stage. Since the scale of Franka Kitchen is relatively small, we compare the models using the Astra (43M) configuration. Astra (43.3M parameters) consists of 6 layers, 12 attention heads, and an embedding size of 768. We also compare the performance of models using continuous and discrete actions in this environment.

Push-T. In Push-T (Florence et al., 2021), the models need to push a T-shaped object until it aligns perfectly with the target position. This task requires precise control, as performance is measured by the overlapping area, with perfect alignment equating to a score of 1.0. Several push-T examples by Astra are included in Figure 10.

The performance of the models is compared in Table 6. Results are averaged over 30 trials. The maximum number of steps the models can take in each trial is 200, so they need to push the object precisely while maintaining adequate speed. Because this is a 2D task, we determined that the configuration of Astra (19M) is adequate for most models, except for the diffusion-based model, which utilizes the configuration of Astra (43M). Astra (19.4M parameters) consists of 6 layers, 8 attention heads, and an embedding size of 512. As the cross-attention layers in VIMA result in a model size of 50.8M parameters—exceeding the size of Astra (43M)—we instead employ three Transformer blocks for VIMA. We found that models fail to learn effective policies using continuous actions; therefore, we only report results of discrete actions.

A.11 Scalability

Astra is an efficient Transformer architecture designed for EIF tasks. Consequently, it also inherits the scalability of Transformers. Depending on the complexity of the EIF benchmarks, we select the most efficient configuration with sufficient capacity to address the demands of each benchmark. In this paper, we explore four configurations of Astra, with model sizes ranging from 19M to 198M, tailored to benchmarks of varying levels of difficulty:

- Astra (19M): Push-T (Table 6)
- Astra (38M): VIMA-Bench (Table 1) and CALVIN (Table 3)
- Astra (43M): Franka Kitchen (Table 5)
- Astra (198M): ManiSkill (Table 2)

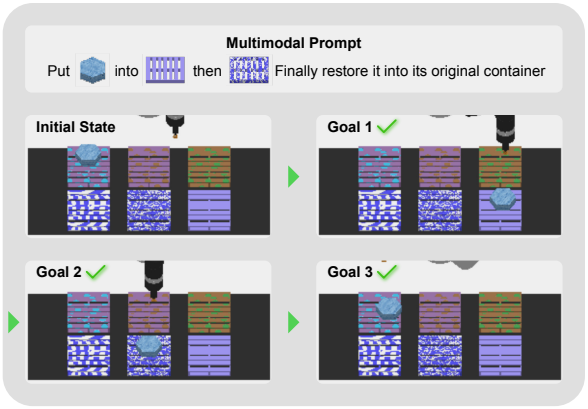
With the help of CDL, the performance of Astra (38M) and Astra (198M) can be further improved, as shown in Table 4. This suggests that CDL can serve as an effective large-scale pretraining method for VLAs.

A.12 Additional Qualitative Analysis

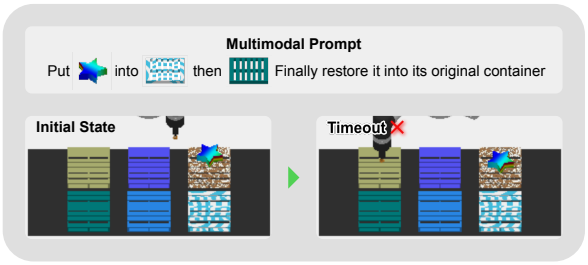
We provide two examples in Figure 11 that compare Astra with CDL to Astra without CDL, aiming to qualitatively demonstrate the effectiveness of contrastive dynamics learning. The failure case of Astra without CDL illustrates how CDL can contribute to improved generalization. In this comparison, CDL enables Astra to more effectively handle novel objects.

A.13 Additional Examples

We include a few execution examples of Astra in VIMA-Bench tasks in Figure 12 & 13 and CALVIN tasks in Figure 14 & 15 & 16.



(a) Astra with CDL.



(b) Astra without CDL.

Figure 11: Comparison between Astra with and without contrastive dynamics learning.

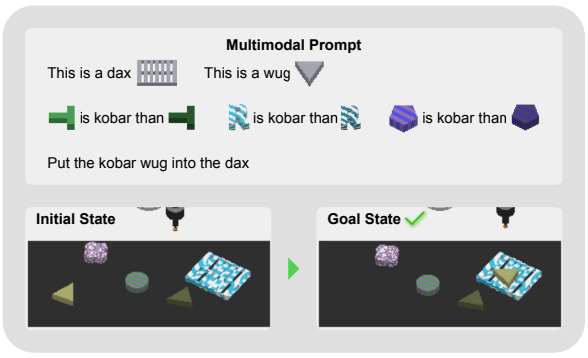


Figure 12: An example of an L4 generalization level task (novel task generalization of "novel adjective and noun") in VIMA-Bench.

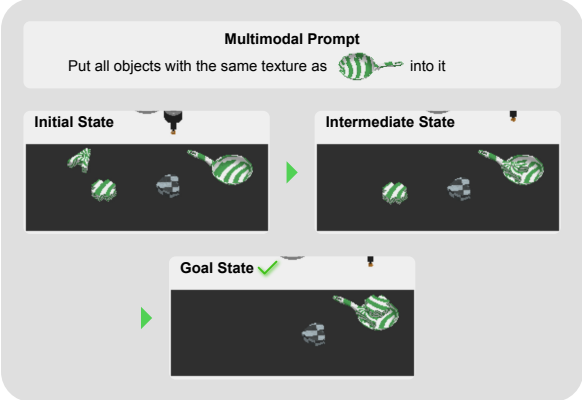


Figure 13: An example of an L4 generalization level task (novel task generalization of "same texture") in VIMA-Bench.

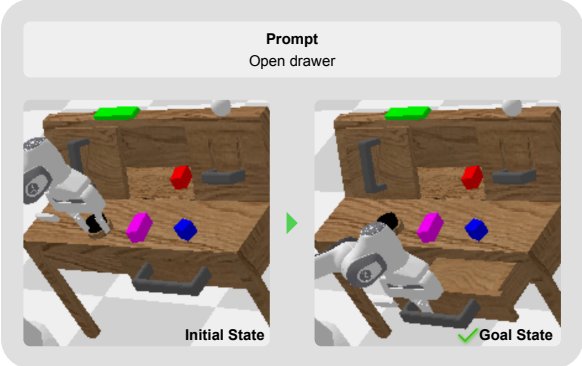


Figure 14: An example of a task in CALVIN.

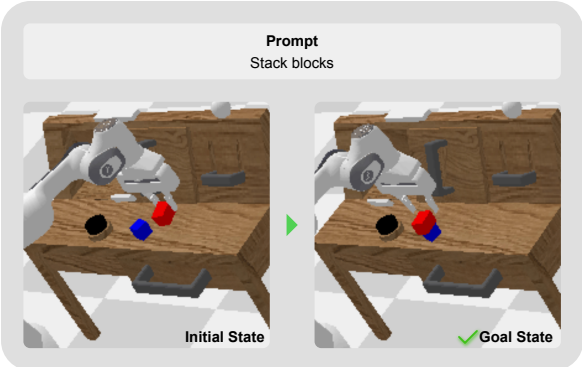


Figure 15: An example of a task in CALVIN.

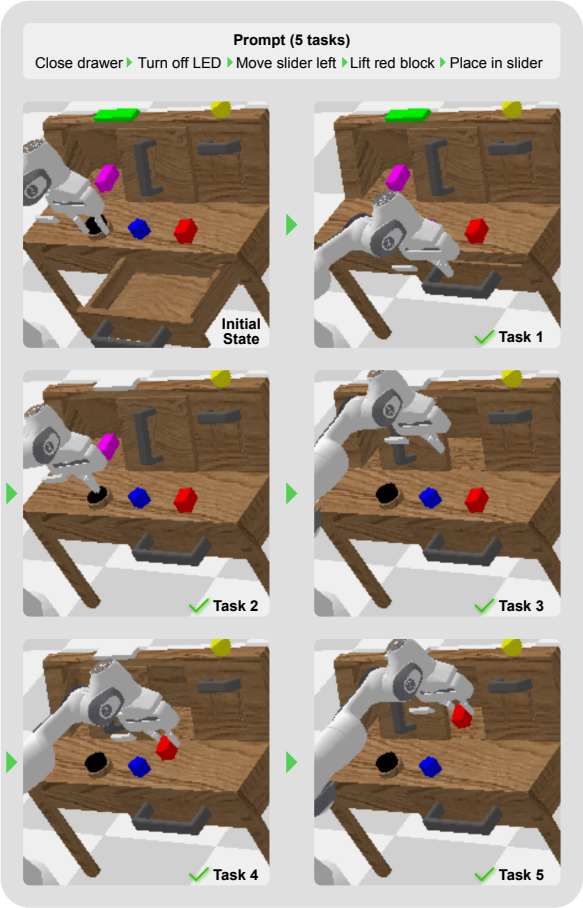


Figure 16: An example of a long-horizon trajectory in CALVIN.