

# Hume: Introducing System-2 Thinking in Visual-Language-Action Model

Haoming Song<sup>1,2\*</sup> Delin Qu<sup>2\*</sup> Yuanqi Yao<sup>2</sup> Qizhi Chen<sup>3,2</sup> Qi Lv<sup>2</sup> Yiwen Tang<sup>2</sup>  
Modi Shi<sup>4</sup> Guanghui Ren<sup>4</sup> Maoqing Yao<sup>4</sup> Bin Zhao<sup>2</sup> Dong Wang<sup>2†</sup> Xuelong Li<sup>2</sup>  
<sup>1</sup>Shanghai Jiao Tong University <sup>2</sup>Shanghai AI Laboratory <sup>3</sup>Zhejiang University <sup>4</sup>AgiBot

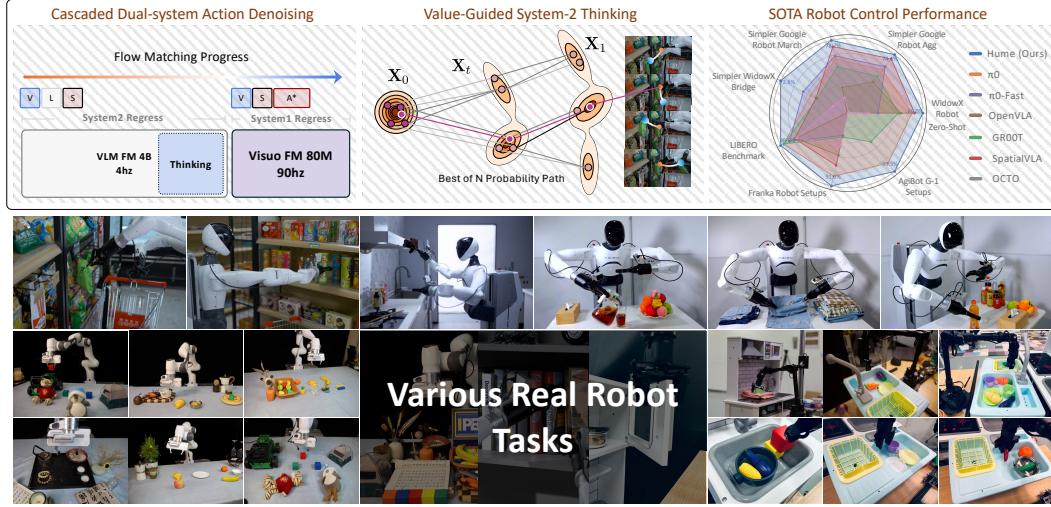


Figure 1: We present **Hume**, a dual-system vision-language-action model exploring human-like thinking capabilities for dexterous robot control. Equipped with value-guided System-2 thinking and cascaded action denoising, the model achieves superior complex reasoning and control capabilities. The model achieves state-of-the-art performance across a diverse range of evaluations and shows significantly advancement in complex robot control tasks.

## Abstract

Humans practice slow thinking before performing actual actions when handling complex tasks in the physical world. This thinking paradigm, recently, has achieved remarkable advancement in boosting Large Language Models (LLMs) to solve complex tasks in digital domains. However, the potential of slow thinking remains largely unexplored for robotic foundation models interacting with the physical world. In this work, we propose **Hume**: a dual-system Vision-Language-Action (VLA) model with value-guided System-2 thinking and cascaded action denoising, exploring human-like thinking capabilities of Vision-Language-Action models for dexterous robot control. System 2 of Hume implements value-guided thinking by extending a Vision-Language-Action Model backbone with a novel value-query head to estimate the state-action value of predicted actions. The value-guided thinking is conducted by repeat sampling multiple action candidates and selecting one according to state-action value. System 1 of Hume is a lightweight reactive visuomotor policy that takes System 2 selected action and performs cascaded action denoising for dexterous robot control. At deployment time, System 2 performs

\* Authors contributed equally: [haomingsong@sjtu.edu.cn](mailto:haomingsong@sjtu.edu.cn).

† Corresponding author: [dongwang.dw93@gmail.com](mailto:dongwang.dw93@gmail.com).

value-guided thinking at a low frequency while System 1 asynchronously receives the System 2 selected action candidate and predicts fluid actions in real time. We show that Hume outperforms the existing state-of-the-art Vision-Language-Action models across multiple simulation benchmark and real-robot deployments.

## 1 Introduction

*A wise man proportions his belief to the evidence.*

---

David Hume,  
*An Enquiry Concerning Human Understanding*

Creating generalist robots to perform various tasks like humans in the physical world has been a long-standing goal [31, 4, 24, 28, 56, 29, 55, 41, 47, 52]. Cognitive psychologists have revealed that humans conduct a deep, deliberate form of thinking when tackling complex problems, such as mathematical proofs or dinner making. This slow and reflective thought process is known as System-2 thinking, while the fast thinking process that relies on intuition is called System-1 thinking [21]. Inspired by the dual process theory of human cognition, thinking and reasoning steps have been introduced to enhance LLMs' capability to solve complex problems in digital domains and achieved significant results. Intuitively, generalist robots in the physical world also require similar System-2 slow thinking capabilities to perform dynamic, dexterous, and complex tasks, while intuition-based fast System-1 thinking is not capable of tackling delicate, tenuous, and fallible robot action prediction. Therefore, a crucial question for generalist robot policies designed to solve complex robotic tasks in diverse scenarios is: **how to enable effective System-2 slow thinking in generalist robot policy for accurate action prediction?**

However, equipping a generalist robot policy with System-2 thinking capabilities poses two primary challenges. First, thinking and reasoning techniques have mainly been demonstrated in text modality, while the delicate, tenuous robot actions lack of clear and consistent semantics, making it difficult to semantic Chain-of-Thought (CoT) [49] thinking as in LLMs. Second, the generalist robot policy needs to perform dexterous, complex tasks in real time. How to effectively balance the "slowness" of System-2 thinking against the "fastness" demanded by robot control is essential. Recently, embodied chain-of-thought reasoning (ECoT) [53] enable a VLA model to predict helpful intermediate reasoning text before choosing robot actions, improving the generalization and performance of VLA models. However, performing intermediate reasoning steps significantly slow down policy inference speed. Moreover, dual-system architectures have been incorporated into Vision-Language-Action models in several works [17, 54, 6, 50, 2, 44]. Typical work Helix [12] adopts a pretrained VLM backbone as System 2, while employing a smaller network as System 1 to outputs high-frequency actions for real time control. These approaches use either latent vectors or detailed language instructions as bridges to communicate between the two systems. Despite their faster inference speed, these models' System 2 does not conduct effective thinking and reasoning to guide the System-1 action prediction.

In this work, we introduce a dual-system Vision-Language-Action model Hume, which powers the VLA model with System-2 thinking capabilities through value-guided repeat sampling and cascaded action denoising. The System 2 of Hume is built on top of a pre-trained vision-language model (VLM) and attach it with two specialized heads: a flow matching denoising head to predict robot actions, and a value-query head to estimate the state-action value of predicted actions. It processes the robot's observation and language instruction to predict long-horizon action chunk through action denoising head. Subsequently, the corresponding state-action value are estimated conditioning on predicted action chunk. The value-guided thinking is conducted by repeat sampling multiple action chunks and selecting one with the highest value. For System 1 of Hume, it takes one short segment from the selected long-horizon action chunks from System 2, current visual observation, and robot states, then conducts cascaded action denoising to generate final fluid robot actions via a separate lightweight diffusion policy. At deployment time, System 2 performs value-guided thinking at a low frequency (4 Hz) while System 1 asynchronously receives the System 2 selected action chunk and predicts fluid actions in real time (90 Hz). Equipped with the proposed value-guided thinking and cascaded action denoising, Hume explores powerful System 2 slow thinking paradigm to enhance the VLA models. We extensively evaluate and ablate Hume on both standard simulation benchmarks and real-robot platforms, including 21 real-world robot settings and 3 simulation environments. To validate Hume's capability to solve complex robot control tasks with the assistance of System-2 Thinking, the test scenarios include variations in viewpoint, texture, lighting, layout, unseen objects,

unseen environments, as well as the most challenging humanoid robot control tasks. In summary, the main contributions of this work are three-folds:

- We propose Hume, a dual-system generalist robot policy that explores System-2 slow thinking paradigm for Vision-Language-Action models.
- We introduce novel value-guided thinking and cascaded action denoising to seamlessly combine low frequency System 2 and high frequency System 1, resulting in effective thinking and reasoning in various robot deployments.
- Hume achieves state-of-the-art performance on multiple benchmarks and real-robot tests, achieving +4.4% increase in success rate over  $\pi_0$  on the LIBERO [34] benchmark, +25.9% in Simpler benchmark [32], and +12.9% improvement in real-world deployments.

## 2 Related Work

**Dual-System Vision-Language-Action Models.** Recently, several studies [31, 4, 24, 28, 56, 29, 55, 41, 6, 12, 17] have extended VLMs for robot control. RT-2 [4] fine-tunes PaLI-X [9] with discretized action tokens, while OpenVLA [24] adapts Prismatic VLM [22] on the OXE dataset [11].  $\pi_0$  [3] integrates PaliGemma with flow-matching for continuous actions. To address efficiency and integration challenges, dual-system architectures have emerged. HiRT [54] runs VLMs at low frequencies while maintaining high-frequency vision-based control for real-time interaction. DexVLA [50] uses diffusion action experts with embodiment curriculum learning across multiple robot types. GR00T N1 [2] features an end-to-end trained dual-system specifically for humanoid robots. Gemini Robotics [47] builds on Gemini 2.0 with specialized models for control and reasoning. HiRobot [44] enables processing of complex instructions with situated feedback. These approaches improve efficiency, success rates, and adaptability compared to monolithic architectures.

**System 2 and System-2 Thinking.** Numerous studies [49, 51, 45, 18, 43, 37, 38, 35, 46] have explored System 2 reasoning approaches to enhance LLMs’ problem-solving capabilities. Chain-of-Thought [49] introduces intermediate reasoning steps before producing answers, while Tree-of-Thoughts [38] explores multiple solution paths with self-verification. Reflexion [45] enables verbal reflection on previous attempts. System-2 Thinking frameworks explicitly model human-like deliberative processes. SETS [8] combines self-critique with multiple reasoning paths and majority voting. SC-MCTS [14] combines multiple reward models for more robust tree search. Addressing efficiency concerns, O1-Pruner [36] introduces length-penalty loss to create concise reasoning processes. While these approaches have improved reasoning in language tasks, their application to visual-language-action models remains largely unexplored.

**Cascaded Denoising.** Cascaded Denoising [19] was first proposed as a diffusion model that generates higher-resolution images through a cascading approach. Starting with a diffusion model at low resolution, it continuously upsamples the generated images to obtain high-resolution results. f-DM [15] applies inter-stage transformations for progressive signal restoration through function learning. Cas-DM [1] cascades noise-prediction and image-prediction modules to integrate perceptual losses in diffusion models. SCDM [7] cascades generation across spectral dimensions, reconstructing hyperspectral bands progressively. HiFI [20] cascades consistent-resolution patches for memory-efficient high-resolution frame interpolation. CDM-VTON [27] employs two-stage cascading for virtual try-on applications. While these approaches have advanced capabilities in image domains, the application of cascaded denoising to integrate System 1 and System 2 remains largely unexplored.

## 3 Methodology

In this section, we describe Hume model architecture and its training and deployment strategy in detail. The process of Value-Guided System-2 Thinking with the help of state-action value estimator was described in detail in Sec. 3.1. Next, we detail how the System 1 module and System 2 module cooperate asynchronously through the proposed cascaded action denoising in Sec. 3.2. Finally, the multi-stage training and deployment strategy of the model is explained in Sec. 3.3.

### 3.1 Value-Guided System-2 Thinking

As shown in Fig. 2, the System 2 module is instantiated as a vision-language-action model (VLA) built upon a pretrained Vision-Language Model. Formally, the inputs of System 2 module consists of RGB

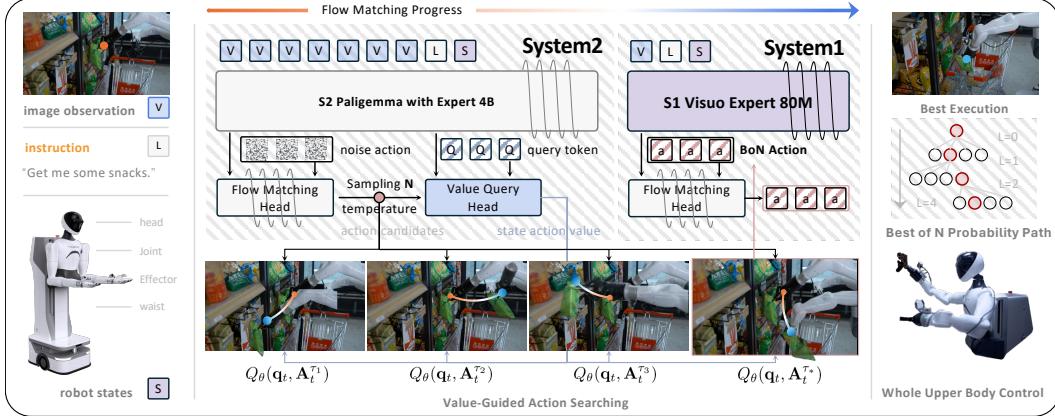


Figure 2: **Overview of Hume.** Hume contains two systems working asynchronously. Given the observation, System 2 of Hume first generates  $N$  candidate action chunks with different noise level, and the best-of- $N$  candidate with the highest  $Q$  value will be selected as the optimal candidate  $\mathbf{A}_t^{\tau^*}$ , which is segmented and conveyed to System 1 for continuous action denoising.

images  $\mathbf{i}_t = [\mathbf{I}_t^1, \dots, \mathbf{I}_t^n]$  at time step  $t$ , natural language instructions  $\ell_t$ , and robot state information  $s_t$ . Similar as VLA models, we first augment the VLM backbone with an action denoising head to learn a mapping function  $\mathcal{F}(\cdot)$  to generate candidate robot actions  $\mathbf{A}_t$  from the observation  $\mathbf{o}_t$ , *i.e.*,  $\mathbf{A}_t = \mathcal{F}(\mathbf{o}_t)$ . Moreover, to empower Hume with System-2 Thinking ability, we attach the VLM backbone another value-query head, which is designed to estimate the state-action value  $Q_\theta(\mathbf{q}_t, \mathbf{A}_t)$  of the candidate robot action  $\mathbf{A}_t$ .

**Candidate Actions Generation** The candidate robot actions are generated by a action denoising head that aims to model the data distribution  $p(\mathbf{A}_t | \mathbf{o}_t)$ , where  $\mathbf{o}_t$  consists of images  $\mathbf{i}_t$ , language instructions  $\ell_t$ , and robot state information  $s_t$ . It is implemented as a transformer-based flow matching denoising process that predicts the remaining action noise  $\mathbf{v}_\theta(\mathbf{A}_t^\tau, \mathbf{o}_t)$  in the “noisy action”  $\mathbf{A}_t^\tau$ , where  $\tau \in [0, 1]$  is the flow matching time step that represents the noise level of the action. Starting from a random noise  $\mathbf{A}_t^0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ , the denoising head generates actions by gradually removing the noise from  $\mathbf{A}_t^0$  to  $\mathbf{A}_t^1$  step by step using the forward Euler method:  $\mathbf{A}_t^{\tau+\delta} = \mathbf{A}_t^\tau + \delta \mathbf{v}_\theta(\mathbf{A}_t^\tau, \mathbf{o}_t)$ , where  $\delta$  is the size of the denoising step. In practice, we use 10 denoising steps, corresponding to  $\delta = 0.1$ . During training, for the ground truth action  $\mathbf{A}_t$  sampled from the dataset, the denoising head is optimized by minimizing the loss between the actual remaining noise  $\epsilon - \mathbf{A}_t$  and the network output  $\mathbf{v}_\theta(\mathbf{A}_t^\tau, \mathbf{o}_t)$  given the observation  $\mathbf{o}_t$  and the noisy action  $\mathbf{A}_t^\tau = \tau \mathbf{A}_t + (1 - \tau)\epsilon$  as input.

After training, conditioned on the same observation  $\mathbf{o}_t$  at timestep  $t$ , the action denoising head generates  $N$  candidate robot action chunks  $\mathbf{A}_t^{\tau_n} \in \{\mathbf{A}_t^{\tau_1}, \mathbf{A}_t^{\tau_2}, \dots, \mathbf{A}_t^{\tau_N}\}$  with different noise levels by integrating the learned vector field  $\mathbf{v}_\theta(\mathbf{A}_t^\tau, \mathbf{o}_t)$  from  $\tau = 0$  to  $\tau = 1 - (n - 1)\xi$  separately:

$$\mathbf{A}_t^{\tau_n} = \int_0^{1-(n-1)\xi} \mathbf{v}_\theta(\mathbf{A}_t^\tau, \mathbf{o}_t) d\tau + \mathbf{A}_t^0, \quad (1)$$

where  $\xi$  is used to control the noise gap between adjacent candidates, and  $\mathbf{A}_t^0$  is the initial action sampled from the normal distribution  $\mathbf{A}_t^0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ . Note that  $n \in \{1, 2, \dots, N\}$ , which leading to most of generated candidate actions  $\mathbf{A}_t^{\tau_n}$  is not fully denoised.

**State-Action Value Estimation** The state-action values are estimated with the proposed value-query head built on the same VLM backbone via learning a latent conditioned Q function. The value-query head is composed of two critic networks estimating state-action values and one actor network for assisting the training of critic networks. Specifically, a special query token  $\mathbf{q}_t$  is introduced and attached at the end of the VLM input sequence, which is a learnable token with the same embedding dimension as the language tokens. Then, for one action chunk  $\mathbf{A}_t$  (either ground-truth action  $\mathbf{A}_t$  or denoised candidate actions  $\mathbf{A}_t^{\tau_n}$ ), it is combined with this special query token  $\mathbf{q}_t$  to feed into the value-query head. Due to its last position of the input sequence, the query token  $\mathbf{q}_t$  attends to all previous tokens and aggregates necessary information from the VLM inputs, *i.e.*, current RGB images  $\mathbf{i}_t = [\mathbf{I}_t^1, \dots, \mathbf{I}_t^n]$  at time step  $t$ , natural language instructions  $\ell_t$ , and robot state information  $s_t$ . In this way, the value-query head estimate the state-action value  $Q_\theta(\mathbf{q}_t, \mathbf{A}_t^{\tau_n})$  of the action chunk  $\mathbf{A}_t^{\tau_n}$ .

conditioned on the input query token  $\mathbf{q}_t$ . We visualize the state-action value map of candidate actions in Fig. 8, and provide a detailed analysis in Sec. 6.2. This value-query head is trained on pre-collected robot demonstration dataset with ground-truth action  $\mathbf{A}_t$  via offline RL [26]. We construct the training dataset  $\mathcal{D}$  using the reward function following [25], where the rewards of last 3 transitions in one robot episode is defined as +1, and the rest is 0. During training, we use the calibrated Q-learning algorithm [39] to optimize the value-query head.

**Value-Guided Thinking** The System-2 value-guided thinking is implemented with Best-of-N selection strategy and the selected action chunk is conveyed to System 1 for cascaded action denoising. Specifically, conditioned on the same observation, the action denoising head generates  $N$  candidate action chunks  $\{\mathbf{A}_t^{\tau_1}, \mathbf{A}_t^{\tau_2}, \dots, \mathbf{A}_t^{\tau_N}\}$  with different noise levels. Then, these candidates are passed to the value-query head to estimate their state-action values. Guided by the estimated state-action values, we select the action with the highest value as the optimal candidate  $\mathbf{A}_t^{\tau^*}$  transferred to System 1, depicted as  $\mathbf{A}_t^{\tau^*} = \arg \max_{\mathbf{A}_t^{\tau_i}} Q(\mathbf{q}_t, \mathbf{A}_t^{\tau_i})$ , where  $i = 1 \dots N$ .

### 3.2 Cascaded Dual-system Action Denoising

In order to achieve rapid, reactive robot control, System 1 module needs to be lightweight and fast in inference. In detail, System-1 consists of a DINOv2-small visual encoder and a lightweight transformer for cascaded action denoising. Given the selected candidate action chunk  $\mathbf{A}_t^{\tau^*}$  from System 2, the System 1 module takes the observation  $\tilde{\mathbf{o}}_{t+kh}$  (including current image  $\mathbf{i}_t = \{\mathbf{I}_t^1, \dots, \mathbf{I}_t^n\}$ , robot state  $\mathbf{s}_t$ ), and sub-action chunks  $\tilde{\mathbf{A}}_t$  segmented from the selected candidate  $\mathbf{A}_t^{\tau^*}$  as input, and produces refined robot actions by continuously denoising on sub-action chunks  $\tilde{\mathbf{A}}_t$ .

Specifically, at timestep  $t$ , the selected action chunk from System 2 is  $\tilde{\mathbf{A}}_t = [\mathbf{a}_t, \mathbf{a}_{t+1}, \dots, \mathbf{a}_{t+H-1}]$ , then  $\tilde{\mathbf{A}}_t$  is segmented into  $K := H/h$  sub-action chunks  $\{\tilde{\mathbf{A}}_t, \tilde{\mathbf{A}}_{t+h}, \dots, \tilde{\mathbf{A}}_{t+(K-1)h}\}$  with a horizon of  $h$ . The System 1 sequentially performs cascaded denoising on these sub-action chunks with observation  $\tilde{\mathbf{o}}_{t+kh}$ . Note that System 1 is much faster than System 2, so that System 2 could finish cascaded denoising on all sub-action chunks before next action chunk  $\tilde{\mathbf{A}}_t$  arriving. System 1 module is trained with the same flow matching loss using the action denoising head of System 2:

$$L^\omega(\theta) = \mathbb{E}_{p(\tilde{\mathbf{A}}_{t+kh}|\tilde{\mathbf{o}}_{t+kh}), q(\tilde{\mathbf{A}}_{t+kh}^\omega|\tilde{\mathbf{A}}_{t+kh})} \|\mathbf{v}_\theta(\tilde{\mathbf{A}}_{t+kh}^\omega, \tilde{\mathbf{o}}_{t+kh}) - \mathbf{u}(\tilde{\mathbf{A}}_{t+kh}^\omega|\tilde{\mathbf{A}}_{t+kh})\|^2, \quad (2)$$

while the superscript  $\omega$  represents the flow matching timestep in System 1.

Note that, during training and inference, the generated candidate action chunks from System 2 are not fully denoised, *i.e.*,  $\tilde{\mathbf{A}}_{t+kh}^{\tau^*} \neq \tilde{\mathbf{A}}_{t+kh}^1$ , requiring the continuous denoising for accurate action prediction. Following the continuous denoising [19] from image generation, System 1 refine the action by integrating the learned vector field from  $\omega = 0$  to  $\omega = 1$ . Instead of starting with random noise, for the  $k$ -th sub-action chunk the integration process of System 1 starting with the sub-action chunk  $\tilde{\mathbf{A}}_{t+kh}^{\tau^*}$ :

$$\tilde{\mathbf{A}}_{t+kh}^\omega = \int_0^\omega \mathbf{v}_\theta \left( \tilde{\mathbf{A}}_{t+kh}^\omega, \tilde{\mathbf{o}}_{t+kh} \right) d\omega + \tilde{\mathbf{A}}_{t+kh}^{\tau^*}, \quad (3)$$

where  $\mathbf{v}_\theta \left( \tilde{\mathbf{A}}_{t+kh}^\omega, \tilde{\mathbf{o}}_{t+kh} \right)$  is the vector field learned by System 1. With the same forward Euler method used in System-2 action denoising head, System 1 produces the final denoised action  $\tilde{\mathbf{A}}_{t+kh}^1$  with 10 denoise steps (corresponding to  $\delta = 0.1$ ). After all  $K$  sub-action chunks have been processed by System 1, System 2 will generate a new selected action chunk  $\mathbf{A}_{t+H}^{\tau^*}$ , and System 1 will continue to refine segments from the new selected action chunk.

### 3.3 Training and Deployment Strategy

The training process of Hume contains two stages. In the first stage, the VLM backbone and the action denoising head of System 2 are trained first using the flow matching loss similar to eq. (2) to ensure System 2 can predict reliable actions. In the second stage of training, the VLM backbone and the action denoising head of System 2 are frozen, while System 1 and value-query head of System 2 are trained from scratch. The training objective of the Value-Query Head is to minimize the Bellman

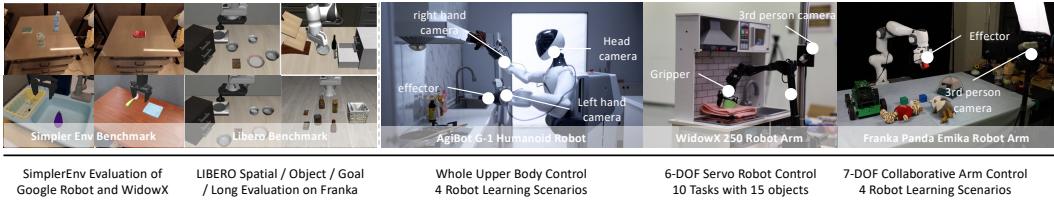


Figure 3: **Experiments setup on WidowX, AgiBot G-1 and Franka Robot.** We evaluate Hume across 3 simulation environments and 3 different real-world robotic platforms, covering 15 robot learning scenarios and 21 real-world manipulation tasks.

error with a regularization term, which is defined as:

$$\min_{\theta} \alpha \mathcal{R}(\theta) + \frac{1}{2} \mathbb{E}_{\mathbf{q}_t, \mathbf{A}_t, \mathbf{q}'_t \sim \mathcal{D}} \left[ (Q_{\theta}(\mathbf{q}_t, \mathbf{A}_t) - \mathcal{B}^{\pi} \bar{Q}(\mathbf{q}_t, \mathbf{A}_t))^2 \right], \quad (4)$$

where  $\mathcal{R}(\theta)$  is a calibrated conservative regularizer that aims to prevent overestimation in the Q-values,  $\mathcal{R}(\theta) := \mathbb{E}_{\mathbf{q}_t \sim \mathcal{D}, a \sim \pi} [\max(Q_{\theta}(\mathbf{q}_t, \mathbf{A}_t), Q^{\mu}(\mathbf{q}_t, \mathbf{A}_t))] - \mathbb{E}_{\mathbf{q}_t, \mathbf{A}_t \sim \mathcal{D}} [Q_{\theta}(\mathbf{q}_t, \mathbf{A}_t)]$ , and  $\mathcal{B}^{\pi} \bar{Q}(\mathbf{q}_t, \mathbf{A}_t)$  is the backup operator applied to the delayed target Q-network  $\bar{Q}$ .

During the inference phase, the System 2 and System 1 modules are cooperating at asynchronous mechanism to boost up the overall control frequency. Specifically, at the initial timestep  $t$ , the action denoising head of System 2 generates  $N = 5$  multiple action chunks  $\mathbf{A}_0^{\tau_n} \in \{\mathbf{A}_0^{\tau_1}, \mathbf{A}_0^{\tau_2}, \dots, \mathbf{A}_0^{\tau_N}\}$  with horizon of  $H = 30$  as candidates at 4 Hz. Then the selected optimal action candidate  $\mathbf{A}_t^{\tau^*}$  is stored in a shared queue. Then  $\tilde{\mathbf{A}}_t^{\tau^*}$ , a sub-action chunk with horizon  $h = 15$ , is segmented from the first  $h$  steps of  $\mathbf{A}_t^{\tau^*}$  and passed into System 1. System 1 removes the remaining noise from  $\tilde{\mathbf{A}}_t^{\tau^*}$ , and produces the fully denoised action  $\tilde{\mathbf{A}}_t$  at 6 Hz and execute all  $h = 15$  actions on real robot immediately, resulting in a overall 90 Hz robot action control frequency. After the robot executes all  $K = H/h = 2$  sub-action chunks in  $\tilde{\mathbf{A}}_t$ , System 1 repeatedly get the newest selected action chunks from the shared queue for subsequent action denoising. Due the different working frequencies of System 2 and System 1, they asynchronously cooperate to achieve a balance between slow, human-like thinking and fast, reactive real robot control.

## 4 Experiment

Our experiments aim to evaluate whether Hume, as a dual-system Vision Language Action Model, can effectively utilize System-2 Thinking to solve complex robot control tasks. Our extensive experiments include evaluating the model’s ability to perform complex manipulation tasks on various robotic platforms in both simulated and real-world environments, including humanoid robots. Hume is compared with previous state-of-the-art generalist policies and alternative designs of various model components. Specifically, our experiments aim to answer the following research questions:

1. How is Hume’s capability to learn multiple tasks on standard simulation benchmarks?
2. Can Hume effectively solve a variety of complex robot control tasks in the real world?
3. To what extent do value-guided thinking and cascaded denoising improve the performance?

To answer these questions, as shown in Fig. 3, we tested Hume’s capabilities across a diverse range of representative robot learning scenarios, including 3 simulation environments and 3 different real-world robotic platforms, covering 15 robot learning scenarios and 21 real-world manipulation tasks. First, we evaluated Hume’s capability to finish multiple tasks in SimplerEnv [32] and LIBERO [34] simulation benchmarks, validating that Hume’s design can effectively accomplish multiple tasks in simulated environments. Second, we extensively tested Hume’s capability to control 3 real-world robotic platforms, WidowX, Franka, and AgiBot G-1, in completing tasks of varying difficulty, effectively validating Hume’s generalization capability in aspects such as object positions, language descriptions, deformable objects, and long-horizon operations in the real world. Finally, we conducted comprehensive ablation experiments on the model design in both simulation and real-world environments to validate the design choices in Hume.

### 4.1 Multitask Evaluation on Simulation Benchmarks

**Evaluation Setups and Baselines.** To assess the robustness of Hume in diverse environmental variations, we employ the SimplerEnv [32] simulation benchmark to evaluate visual matching and

variant aggregation metrics. SimplerEnv features WidowX and Google Robot setups, providing diverse manipulation scenarios with varied lighting, color, textures, and robot camera pose conditions, bridging the visual appearance gap between real and simulated environments. We compare our model with the latest state-of-the-art generalist manipulation policies, including RT-1 [5], RT-1-X [11], RT-2-X [11], Octo [40], OpenVLA [24], HPT [48], TraceVLA [56], RoboVLM [30], SpatialVLA [42], GR00T [2],  $\pi_0$ -FAST [41], and  $\pi_0$  [3].

**Evaluation Results.** Tab. 1 present the LIBERO [34] experimental results. We observe that Hume can be effectively adapted to tasks in the LIBERO environments, as it obtains the highest average success rate of 98.6% and the first rank across all the policies. In particular, Hume achieves a remarkable 96.7% success rate (+11.5% over  $\pi_0$ , +6.1% over GR00T) on the LIBERO-Long task, which consists of long-horizon tasks, demonstrating the model’s strong long-term planning capabilities. Tab. 2 presents the SimplerEnv experimental results on WidowX and Google robot tasks. Hume also achieves state-of-the-art performance on WidowX multitasks, with an average success rate of 72.6%, representing a significant improvement compared to all current generalist manipulation policies (+32.5% over  $\pi_0$ , +39.6% over GR00T, +64.8% over OpenVLA). Similarly, Hume achieves an average success rate of 76.4% (+19.6% over  $\pi_0$ ) on Google robot tasks. In summary, Hume demonstrates its versatility as a generalist robot control policy, achieving better performance across various tasks.

Table 1: **LIBERO Benchmark Results.** We present the success rate (SR) and standard error for each method across four task suites, which are averaged over three random seeds with 500 trials. Hume achieve the highest average success rate and ranking, followed by OpenVLA-OFT and  $\pi_0$ .

	LIBERO-Spatial		LIBERO-Object		LIBERO-Goal		LIBERO-Long		Average	
	SR ( $\uparrow$ )	Rank ( $\downarrow$ )	SR ( $\uparrow$ )	Rank ( $\downarrow$ )	SR ( $\uparrow$ )	Rank ( $\downarrow$ )	SR ( $\uparrow$ )	Rank ( $\downarrow$ )	SR ( $\uparrow$ )	Rank ( $\downarrow$ )
Diffusion Policy [10]	78.5 $\pm$ 1.1%	6	87.5 $\pm$ 0.7%	6	73.5 $\pm$ 1.2%	6	64.8 $\pm$ 1.3%	5	76.1 $\pm$ 0.7%	6
OpenVLA-OFT [23]	97.6 $\pm$ 0.9%	2	98.4 $\pm$ 0.8%	3	97.9 $\pm$ 1.0%	2	94.5 $\pm$ 1.3%	2	97.1 $\pm$ 0.6%	2
$\pi_0$ [3]	96.8 $\pm$ 0.8%	3	98.8 $\pm$ 0.9%	2	95.8 $\pm$ 1.1%	3	85.2 $\pm$ 1.2%	4	94.2 $\pm$ 0.9%	3
$\pi_0$ -FAST [41]	96.4 $\pm$ 0.7%	4	96.8 $\pm$ 0.7%	5	88.6 $\pm$ 1.0%	5	60.2 $\pm$ 1.4%	6	85.5 $\pm$ 1.0%	4
GR00T N1 [2]	94.4 $\pm$ 0.9%	5	97.6 $\pm$ 1.0%	4	93.0 $\pm$ 1.2%	4	90.6 $\pm$ 1.0%	3	93.9 $\pm$ 1.1%	5
Hume	<b>98.6 <math>\pm</math> 0.2%</b>	<b>1</b>	<b>99.8 <math>\pm</math> 0.1%</b>	<b>1</b>	<b>99.4 <math>\pm</math> 0.3%</b>	<b>1</b>	<b>96.7 <math>\pm</math> 0.9%</b>	<b>1</b>	<b>98.6 <math>\pm</math> 0.7%</b>	<b>1</b>

Table 2: **SimplerEnv evaluation across different policies on robot tasks.** The zero-shot results denote performance of OXE dataset [11] pre-trained models.

Model	SimplerEnv on WidowX Robot Tasks						#Overall Average		
	Put Spoon on Towel Success	Put Carrot on Plate Success	Stack Green Block on Yellow Block Success	Put Eggplant in Yellow Basket Success	Grasp Spoon	Grasp Carrot			
RT-1-X [11]	16.7%	0%	20.8%	4.2%	8.3%	0%	0.0%	0%	6.3%
Octo-Base [40]	34.7%	12.5%	52.8%	8.3%	31.9%	0%	66.7%	43.1%	31.3%
Octo-Small [40]	<b>77.8%</b>	47.2%	27.8%	9.7%	40.3%	4.2%	87.5%	56.9%	43.9%
OpenVLA [24]	4.1%	0%	33.3%	0%	12.5%	0%	8.3%	4.1%	7.8%
RoboVLM [30]	54.2%	29.2%	25.0%	25.0%	45.8%	12.5%	58.3%	58.3%	38.5%
SpatialVLA [42]	25.0%	20.8%	41.7%	20.8%	58.3%	25.0%	79.2%	70.8%	42.7%
$\pi_0$ [3]	45.8%	29.1%	25.0%	0%	50.0%	16.6%	91.6%	62.5%	40.1%
$\pi_0$ -FAST [41]	62.5%	29.1%	58.5%	21.9%	54.0%	10.8%	83.3%	66.6%	48.3%
Hume	73.8%	<b>58.0%</b>	<b>83.3%</b>	<b>66.7%</b>	<b>83.2%</b>	<b>45.5%</b>	<b>97.8%</b>	<b>72.8%</b>	<b>72.6%</b>

Model	SimplerEnv on Google Robot Tasks				#Overall Average			
	Pick Coke Can	Move Near	Open/Close Drawer	#Average				
RT-1 [5] (Begin)	2.7%	5.0%	13.9%	7.2%	2.2%	4.0%	6.9%	4.4%
RT-1 [5] (15%)	71.0%	35.4%	56.5%	54.3%	81.3%	44.6%	26.7%	56.2%
RT-1 [5] (Converged)	85.7%	44.2%	<b>73.0%</b>	74.6%	89.8%	50.0%	32.3%	63.3%
RT-1-X [11]	56.7%	31.7%	59.7%	53.4%	49.0%	32.3%	29.4%	39.6%
RT-2-X [11]	78.7%	77.9%	25.0%	60.7%	82.3%	79.2%	35.3%	65.6%
Octo-Base [40]	17.0%	4.2%	22.7%	16.8%	0.6%	3.1%	1.1%	1.1%
OpenVLA [24]	16.3%	46.2%	35.6%	27.7%	54.5%	47.7%	17.7%	39.8%
TraceVLA [56]	28.0%	53.7%	57.0%	42.0%	60.0%	56.4%	31.0%	45.0%
RoboVLM [30]	77.3%	61.7%	43.5%	63.4%	75.6%	60.0%	10.6%	51.3%
SpatialVLA [42]	86.0%	77.9%	57.4%	73.8%	88.0%	72.7%	41.8%	70.7%
HPT [48]	56.0%	60.0%	24.0%	46.0%				
$\pi_0$ [3]	72.7%	65.3%	38.3%	58.8%	75.2%	63.7%	25.6%	54.8%
$\pi_0$ -FAST [41]	75.3%	67.5%	42.9%	61.9%	77.6%	68.2%	31.3%	59.0%
Hume	<b>97.0%</b>	<b>80.4%</b>	58.8%	<b>78.7%</b>	<b>98.0%</b>	<b>79.7%</b>	<b>44.6%</b>	<b>74.1%</b>

## 4.2 Real-World Embodiment Control

**Real-world WidowX Evaluation.** Fig. 4 presents the results of the real-world evaluation in WidowX robot platform. We compared some representative single-system VLA models and dual-system VLA models on multiple tasks. We observe that, in simple task scenarios (#1-2), most policies exhibit some generalizability, successfully completing tasks in unseen environments. However, in more complex tasks (#3-7), policies such as GR00T,  $\pi_0$ -FAST, and OpenVLA struggle with manipulation, frequently encountering grasp failures issues, such as inability to accurately grasp or place target objects. Even when the policies attempt to recover from failures, they often fall in error states that prevent successful execution. In contrast, Hume leverages value-guided thinking to effectively recover

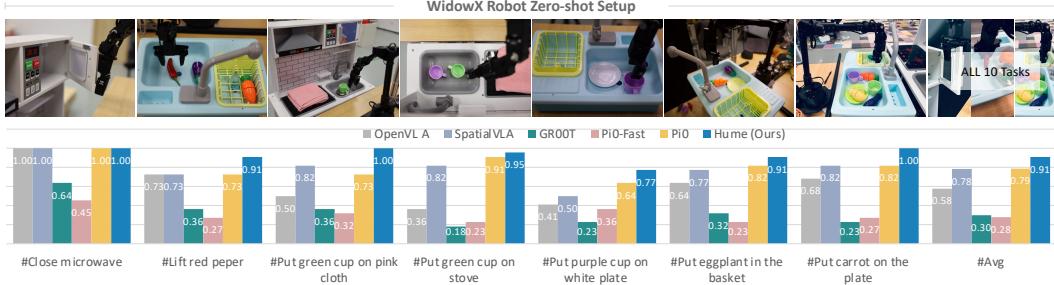


Figure 4: **Real-world evaluation on WidowX Robot tasks.** We evaluate Hume across 10 zero-shot tasks with varying backgrounds, poses, and motion distractors. Hume achieves the highest average success rate, surpassing  $\pi_0$  and all other generalist manipulation policies in comparative evaluations.

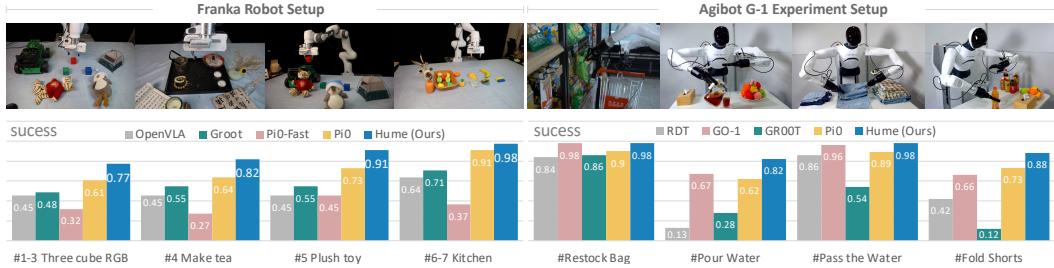


Figure 5: **Evaluation on Franka and Agibot G-1 Robot.** We evaluate Hume across 11 real-world common tasks on Franka and Agibot G-1 robot.

from failures, demonstrating superior performance on various complex tasks. That is, *when Hume falls into a wrong state, it can evaluate multiple candidate actions and select the another correct trajectory forward*. As a result, even if failures occur during initial attempts, Hume can adjust its trajectory and successfully complete the task on subsequent tries (please refer to supplementary video for more details), achieving strong robustness ability across various complex unseen tasks (91% average success rate), improves by +12% over  $\pi_0$ , and +33% over OpenVLA.

**Real-world Franka and Agibot G-1 Evaluation.** Fig. 5 presents the results of the real-world evaluation on Franka and Agibot G-1 robot platforms. The task design incorporates multiple real-world daily long-horizon tasks, deformable objects manipulation, tool-usage, and other challenging scenarios, with further validation conducted on both the Franka robot and the humanoid robot Agibot G-1. We observe that *the long-term planning capability of System-2 thinking employed by Hume helps us better solve long-horizon tasks*. For example, in Agibot’s long-horizon deformable objects manipulation task (#Fold Shorts), where the models need to make the robot fold two shorts, Hume achieves a success rate of 88%, improving by +15% over  $\pi_0$ . In the complex long-horizon task (#Pour Water), Hume achieves success rate of 82%, significantly improving by +20% over  $\pi_0$ , and +60% over GR0OT. Additionally, Hume also achieves an average success rate of 87% across various tasks on the Franka robot, improving by +14.75% over  $\pi_0$ , and +37.25% over OpenVLA.

### 4.3 Ablations on Design Decisions

In this section, we conduct value-guided thinking and cascaded denoising ablations across multiple tasks in both simulation and real-world environments , with results presented in Tab. 3 and Fig. 6.

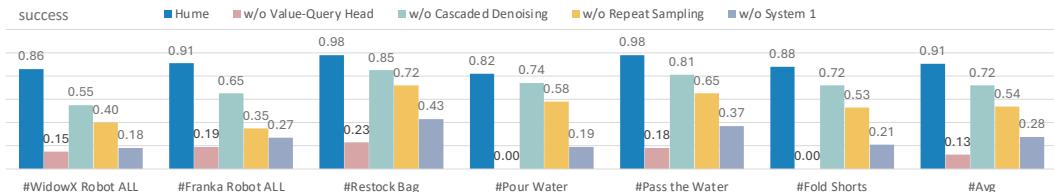


Figure 6: **Real-world Ablations on WidowX, Franka and Agibot G-1 Robot.** We conducted ablation studies of Hume across 3 different real-world robotic platforms, covering 15 robot learning scenarios and 21 real-world manipulation tasks.

**Table 3: Ablations in LIBERO and SimplerEnv tasks.** We conducted ablation studies of Hume across LIBERO [34] and SimplerEnv [32] on WidowX and Google Robot tasks. Models are trained with mixtures from the OXE dataset [11] in the SimplerEnv experiments.

LIBERO Tasks							
#setting	LIBERO-Spatial		LIBERO-Object		LIBERO-Goal		Average
[1]. Hume	98.6 ± 0.2%		99.8 ± 0.1%		99.4 ± 0.3%		98.6 ± 0.7%
[2]. w/o Cascaded Denoising	95.4 ± 0.8%		97.2 ± 0.5%		96.8 ± 0.6%		95.9 ± 0.5%
[3]. w/o Repeat Sampling	93.6 ± 0.4%		94.8 ± 0.2%		95.2 ± 0.3%		93.8 ± 0.5%
[4]. w/o System 1	90.2 ± 0.6%		91.8 ± 0.9%		92.4 ± 0.7%		89.8 ± 0.6%
[5]. w/o Value-Query Head	85.2 ± 0.2%		86.9 ± 0.4%		88.2 ± 0.5%		84.9 ± 0.5%
SimplerEnv on WidowX Robot Tasks							
#setting	Put Spoon on Towel		Put Carrot on Plate		Stack Green Block on Yellow Block		Overall
	Grasp Spoon	Success	Grasp Carrot	Success	Grasp Green Block	Success	Average
[1]. Hume	73.8%	58.0%	83.3%	66.7%	83.2%	45.5%	97.8%
[2]. w/o Cascaded Denoising	70.2%	55.6%	78.1%	62.5%	79.6%	42.1%	93.9%
[3]. w/o Repeat Sampling	68.8%	49.2%	76.8%	57.9%	75.4%	39.9%	90.2%
[4]. w/o System 1	64.7%	42.3%	74.3%	53.4%	71.2%	36.2%	87.3%
[5]. w/o Value-Query Head	58.2%	36.8%	67.6%	47.3%	66.9%	31.9%	83.6%
SimplerEnv on Google Robot Tasks.							
Variant Aggregation							
#setting	Visual Matching				Variant Aggregation		
	Pick Coke Can	Move Near	Open/Close Drawer	Average	Pick Coke Can	Move Near	Open/Close Drawer
[1]. Hume	97.0%	80.4%	58.8%	78.7%	98.0%	79.7%	44.5%
[2]. w/o Cascaded Denoising	95.3%	75.4%	57.5%	76.1%	94.8%	77.8%	42.5%
[3]. w/o Repeat Sampling	94.0%	70.8%	54.2%	73.0%	92.2%	74.6%	39.3%
[4]. w/o System 1	92.4%	65.4%	52.8%	70.2%	89.9%	70.8%	35.9%
[5]. w/o Value-Query Head	89.9%	63.8%	48.9%	67.5%	85.4%	65.9%	30.2%

**Effect of Cascaded Denoising.** According to the ablation results (#1v.s.#2), the proposed cascaded denoising *employs System 1 to remove the remaining noise, enabling the robot to perform precise and dexterous movements*. Models w/o cascaded denoising utilize System 2 to complete the entire denoising process, leading to all candidates being sampled from the same distribution. This consequently reduces the range of candidates that the model choose from, resulting in suboptimal candidates selection. The models suffer a significant performance drop in variant aggregation, showing an average decline of -3.2% across multiple SimplerEnv tasks, -2.7% across LIBERO tasks, and -19% in real-world robot tasks.

Furthermore, we compared Hume with models directly using the candidate with the highest value output by System 2 (#1v.s.#4). Without System 1 to remove the remaining noise, models w/o System 1 cannot perform precise and dexterous movements, ultimately leading to an average decrease of -9.8% across SimplerEnv tasks, -8.8% across LIBERO tasks, and -63% in real-world robot tasks.

**Effect of Value-Guided Thinking.** According to the ablation results (#1v.s.#5), the proposed value-guided thinking *enables the System 2 to select the most valuable candidate from multiple noisy candidates to pass to System 1, which effectively improves Hume’s performance in handling robot control tasks*. Models w/o value-guided thinking randomly select 1 out of 5 candidate actions generated by System 2 to pass to System 1. Since there are candidates with different levels of noise in the candidate actions, this random selection strategy may select harmful candidates for System 1, ultimately resulting in a significant decrease in success rate, showing an average decline of -14.95% across multiple SimplerEnv tasks, -13.7% across LIBERO tasks. Notably, this performance degradation is even more pronounced in more complex real-world robot scenarios, with an astonishing average decrease of -78% across various tasks in real-world environments.

Additionally, according to the ablation results (#1v.s.#3), we also compared Hume with model generate only one candidate from System 2 then passing it to System 1. Models w/o repeat sampling, due to having no additional candidates to choose from, cannot leverage the estimated value to provide more helpful action selection. This also led to performance degradation, with an average decrease of -6.2% across SimplerEnv tasks, -4.8% across LIBERO tasks, and -37% in real-world robot tasks.

## 5 Conclusion and Limitations

In this paper, we present Hume, a dual-system Vision-Language-Action (VLA) model to explore human-like thinking capabilities for generalist robot policy. Hume implements value-guided System-2 thinking by performing effective best-of-N selection with state-action estimation, and integrate System 1&2 with the proposed cascaded action denoising to achieve rapid and fluid control for dexterous tasks. With extensive experiments in both simulation and real robot platforms, we validated that Hume outperforms current state-of-the-art models, demonstrating the superiority of the Hume on various robot tasks, especially when failures occur in complex tasks during the deployment time, showing a promising research direction on generalist robot policy.

**Limitations.** First, the value-guided System-2 thinking is limited by the quality of sampled candidate action chunks, how to include more high-value candidates among the sampled candidates is a question worthy of discussion. Second, the estimated state-action value is not well-aligned with semantics, remaining a research direction for better value learning. Last, the System-2 thinking paradigm implemented in Hume is still relatively naive, and future work can explore more sophisticated paradigms such as tree search, self-correction, and reinforcement learning approaches.

# Hume: Introducing System-2 Thinking in Visual-Language-Action Model

## Supplementary Material

### Abstract

This supplementary material accompanies the main paper by providing more detailed visualization analysis of Hume’s workflow, as well as implementation details and additional experimental results:

- ▷ **Sec. 6:** Detail Hume’s workflow by visualize the value-guided thinking and cascaded action denoising processes.
- ▷ **Sec. 7:** Video demonstration and anonymous link: <https://hume-vla.github.io>.
- ▷ **Sec. 8:** Implementation details including loss functions and hyperparameters.
- ▷ **Sec. 9:** Experimental details in simulation and real world including experiment setup and detailed results.

## 6 Hume Visualization Analysis

In this section, we first demonstrate the detailed inference process of Hume through the dexterous Push-T task in Sec. 6.1, then visualize two key designs of Hume: value-guided System-2 thinking and cascaded action denoising in Sec. 6.2 and Sec. 6.3 to provide a comprehensive understanding of Hume.

### 6.1 Hume Workflow Visualization

To illustrate the detailed inference workflow of Hume, we evaluate Hume on the Push-T task that needs complex and contact-rich controls to push the T block precisely. The Push-T task requires the policy to control a blue dot on a two-dimensional plane to push a gray T-shaped block into the green area. Since the action space is two-dimensional, we can visualize the actions predicted by Hume as trajectories on a plane.

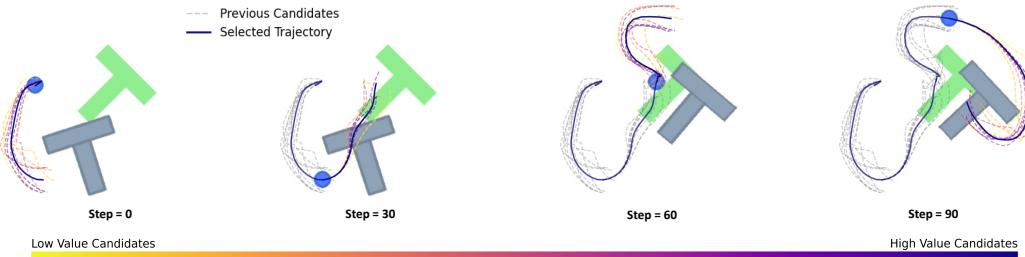
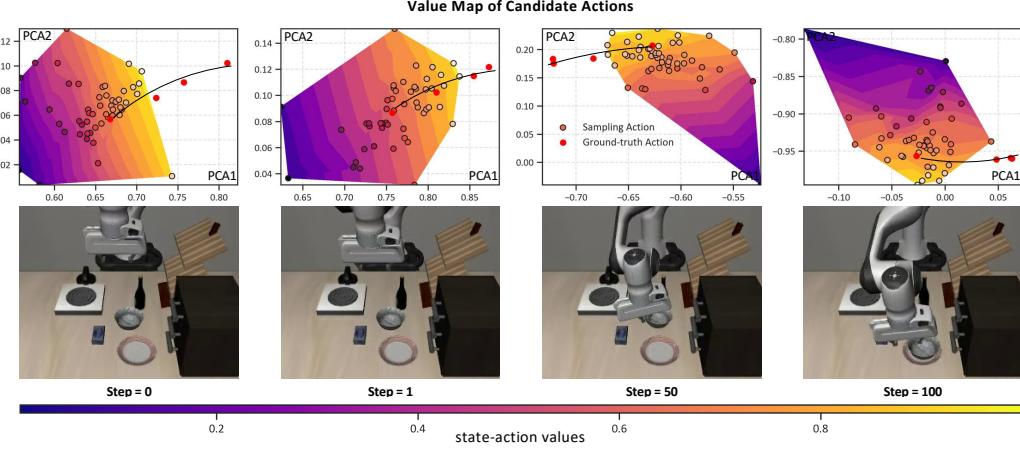


Figure 7: **Visualization of Hume in Push-T.** We visualize the candidate actions  $\mathbf{A}_t^{\tau_i}$  sampled from System 2 with dashed lines and the final executed action  $\hat{\mathbf{A}}_{t+kh}^1$  from System 1 with solid line. The intensity of colors of lines indicates the magnitude of state-action values  $Q(\mathbf{q}_t, \mathbf{A}_t^{\tau_i})$  of candidates.

As shown in Fig. 7, we illustrate the detailed inference process of Hume in the Push-T task. Specifically, in inference, we sample the candidate actions  $\mathbf{A}_t^{\tau_i}$  from System 2 at time steps  $t = 0, 30, 60, 90$  with a horizon of  $H = 30$ , and produce 10 candidate actions at each timestep. The selected action with highest value is conveyed to System 1 and continuously denoised to executed action as drew with solid line. We can see that the final denoised action from System 1 is smoother and more delicate for accomplishing the task.

### 6.2 Value-Guided Thinking Visualization

Fig. 8 visualizes estimated state-action values the candidate actions at different time steps in the LIBERO-GOAL setting. Since the action space in LIBERO is 7-dimensional, we used Principal Component Analysis (PCA) to project the high dimensional actions onto a two-dimensional plane. For



**Figure 8: Value Map of Candidate Actions.** The candidate actions  $\mathbf{A}_t^{\tau_i}$  sampled from System 2 and ground-truth actions  $\mathbf{A}_t^{\text{GT}}$  are projected into the same two-dimensional space through Principal Component Analysis (PCA). The intensity of colors indicates the magnitude of state-action values  $Q(\mathbf{q}_t, \mathbf{A}_t^{\tau_i})$  of candidate actions.

each projected candidate action, we use different colors to represent their corresponding state-action values estimated by the value-query head. These projected points together with their corresponding state-action values generate the value map shown in the figure. In the value map, yellow color represent actions with high state-action values, while purple color represent actions with low state-action values. Additionally, we also show the ground truth actions from collected demonstrations in the value map for comparison.

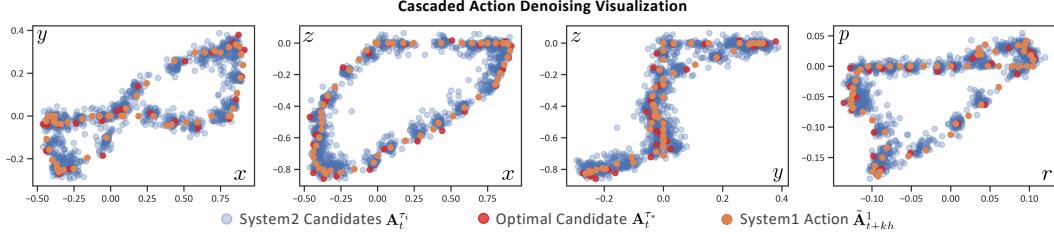
By observing the positions of ground truth actions in the value map, we find these ground truth actions are consistently located in high-value regions, which demonstrates that System 2’s value-query head is capable of making reasonable estimates of the state-action values. We also find ground truth actions are not located at the highest-value positions in the value map, which proves System 2’s value-query head has not been overfitted to ground truth actions, but is able to estimate appropriate state-action values in whole action spaces, guiding System 2 to select the optimal candidate action. Furthermore, by comparing value maps across different timesteps, we observe adjacent timesteps (step=0 and step=1) have similar value maps, while value maps at distant timesteps (step=50 and step=100) exhibit significant differences. This demonstrates the value-query head can reasonably adjust its estimation of state-action values by capturing nature world dynamics, guiding System 2 to make smooth choices for robot control.

### 6.3 Cascaded Action Denoising Visualization

Fig. 9 visualizes the cascaded action denoising process in LIBERO-OBJECT. For the 7 dimensions in action space (X, Y, Z, Roll, Pitch, Yaw, Gripper), we pair them into combinations for illustration, *i.e.*, X-Y, X-Z, Y-Z, and R-P. The drew points is down-sampled from the actual denoised action sequences for accomplishing one task, where the blue points represent candidate actions  $\mathbf{A}_t^{\tau_i}$  sampled from System 2’s denoising head, the red points represent optimal candidates  $\mathbf{A}_t^{\tau^*}$  selected by System 2, and the orange points represent final actions  $\tilde{\mathbf{A}}_{t+k_h}^1$  denoised by System 1. The red points and orange points are generally distributed within the region covered by blue points, while the distribution of orange points shows slight differences from the red points. This demonstrates that cascaded action denoising is actually refining selected actions from System 2 with higher-frequency new observation inputs in System 1, achieving accurate, fluid, and delicate robot control.

## 7 Video Demonstration and Anonymous Link

We provide a video of Hume and an anonymous link (Please refer to: <https://hume-vla.github.io> for more details) to demonstrate the deployment on real-world robot platforms.



**Figure 9: Cascaded Action Denoising on Different Action Dimension.** We visualize denoised actions grouped to coordinate pairs (X-Y, X-Z, Y-Z, and R-P) from 7-dimensional action space (X, Y, Z, Roll, Pitch, Yaw, Gripper). For each group, we display the candidate actions  $\mathbf{A}_t^{\tau_i}$  sampled from System 2, the optimal candidate  $\mathbf{A}_t^{\tau*}$ , and final action  $\tilde{\mathbf{A}}_{t+kh}^1$  denoised by System 1.

## 8 Implementation Details

In this section, we provide further implementation details of Hume, including the training details of the value-query head and the hyperparameters used by the model during training.

### 8.1 Value Objective Functions

Formally, the goal of the value-query Head is to learn  $Q_\theta(\mathbf{q}_t, \mathbf{A}_t)$ , which is the optimal estimate of the state-action value function  $Q^\pi(\mathbf{q}_t, \mathbf{A}_t) = \frac{1}{1-\gamma} \sum_t \mathbb{E}_{\mathbf{A}_t \sim \pi(\mathbf{q}_t)} [\gamma^t r(\mathbf{q}_t, \mathbf{A}_t) | \mathbf{q}_{t_0} = \mathbf{q}_t, \mathbf{A}_{t_0} = \mathbf{A}_t]$  in a Markov Decision Process  $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \rho, \gamma)$ . Here  $\mathcal{S}, \mathcal{A}$  denote the state and action spaces, while  $P(\mathbf{q}'|\mathbf{q}, \mathbf{A})$  and  $r(\mathbf{q}_t, \mathbf{A}_t)$  are the dynamics and reward functions.  $\rho(\mathbf{q})$  denotes the initial state distribution, and  $\gamma \in (0, 1)$  denotes the discount factor. The training objective of the Value-Query Head is to minimize the Bellman error with a regularization term  $\mathcal{R}(\theta)$ , which is defined as:

$$\min_{\theta} \alpha \mathcal{R}(\theta) + \frac{1}{2} \mathbb{E}_{\mathbf{q}_t, \mathbf{A}_t, \mathbf{q}'_t \sim \mathcal{D}} \left[ (Q_\theta(\mathbf{q}_t, \mathbf{A}_t) - \mathcal{B}^\pi \bar{Q}(\mathbf{q}_t, \mathbf{A}_t))^2 \right], \quad (5)$$

The second term in eq. (5) is the standard TD error [33, 13, 16], where  $Q_\theta(\mathbf{q}_t, \mathbf{A}_t)$  is the output of the value-query head, and  $\mathcal{B}^\pi \bar{Q}(\mathbf{q}_t, \mathbf{A}_t)$  is the backup operator applied to the delayed target Q-network  $\bar{Q}$ :  $\mathcal{B}^\pi \bar{Q}(\mathbf{q}_t, \mathbf{A}_t) := r(\mathbf{q}_t, \mathbf{A}_t) + \gamma \mathbb{E}_{\mathbf{A}'_t \sim \pi(\mathbf{A}'_t | \mathbf{q}_t)} [\bar{Q}(\mathbf{q}'_t, \mathbf{A}'_t)]$ , which can be calculated from the offline dataset  $\mathcal{D} = \{(\mathbf{q}_t, \mathbf{A}_t, r, \mathbf{q}'_t)\}$ . The  $\mathcal{R}(\theta)$  in eq. (5) is a calibrated conservative regularizer that aims to prevent overestimation in the Q-values for OOD actions by penalizing the Q-values, and compensating for this pessimism on actions seen in the training dataset, and  $\alpha$  is a hyperparameter to control the conservative penalty. Specifically, the regularization term  $\mathcal{R}(\theta)$  is defined as:

$$\mathcal{R}(\theta) := \mathbb{E}_{\mathbf{q}_t \sim \mathcal{D}, a \sim \pi} [\max(Q_\theta(\mathbf{q}_t, \mathbf{A}_t), Q^\mu(\mathbf{q}_t, \mathbf{A}_t))] - \mathbb{E}_{\mathbf{q}_t, \mathbf{A}_t \sim \mathcal{D}} [Q_\theta(\mathbf{q}_t, \mathbf{A}_t)] \quad (6)$$

where  $Q^\mu(\mathbf{q}_t, \mathbf{A}_t)$  is the value function of the calibrated policy  $\mu$ .

### 8.2 Training and Inference Hyperparameters

**LIBERO** In LIBERO, Hume takes images of third-person camera, wrist camera, and robot state as input. We set the chunk size of System 2's output to 16 and the chunk size of System 1's output to 8. The model was trained using 8 GPUs with a batch size of 16.

**SimplerEnv** In SimplerEnv, Hume takes image of third-person camera and robot state as input. In Bridge tasks, we set the chunk size of System 2's output to 8 and the chunk size of System 1's output to 4. In Google Robot tasks, we set the chunk size of System 2's output to 4 and the chunk size of System 1's output to 2. The model was trained using 8 GPUs with a batch size of 32.

**Franka** In real-world experiment on Franka-Emika-Panda, Hume takes images of third-person camera and robot state as input. We set the chunk size of System 2's output to 16 and the chunk size of System 1's output to 8. The model was trained using 4 GPUs with a batch size of 32.

**Widowx** In real-world experiment on Widowx 250s, Hume uses the same training settings as in the simulation environment of Bridge in SimplerEnv, which takes images of third-person camera and

Table 4: SimplerEnv evaluation results across different policies on Google Robot tasks.

	RT-1 (begin)	2.2%	1.3%	03.1%	02.2%	04.0%	00.5%	13.2%	06.9%	04.4%
	RT-1 (15%)	92.0%	70.4%	81.3%	81.2%	44.6%	21.2%	32.3%	26.8%	50.9%
	RT-1 (converged)	96.9%	76.0%	96.4%	89.8%	50.0%	27.0%	37.6%	32.3%	57.4%
	RT-1-X	56.9%	20.4%	69.8%	49.0%	32.3%	06.9%	51.9%	29.4%	36.9%
	RT-2-X	82.2%	75.4%	89.3%	82.3%	79.2%	33.3%	37.2%	35.3%	65.6%
Variant Aggregation	Octo-Base	0.5%	00.0%	01.3%	00.6%	03.1%	00.0%	02.1%	01.1%	01.6%
	OpenVLA	71.1%	27.1%	65.3%	54.5%	47.7%	15.8%	19.5%	17.7%	40.0%
	TraceVLA	—	—	—	60.0%	56.4%	—	—	31.0%	49.1%
	RoboVLM	93.8%	49.8%	83.1%	75.6%	60.0%	02.6%	18.5%	10.6%	48.7%
	SpatialVLA	93.3%	78.2%	92.4%	88.0%	72.7%	28.6%	55.0%	41.8%	67.5%
	$\pi_0$	82.0%	58.0%	85.6%	75.2%	63.7%	18.0%	33.2%	25.6%	54.8%
	$\pi_0$ -FAST	84.0%	63.0%	85.8%	77.6%	68.2%	24.0%	38.6%	31.3%	59.0%
	Hume	99.0%	96.0%	99.0%	98.0%	79.7%	38.0%	51.2%	44.6%	74.1%
	RT-1 (Begin)	5.0%	00.0%	03.0%	02.7%	05.0%	00.0%	27.8%	13.9%	07.2%
	RT-1 (15%)	86.0%	79.0%	48.0%	71.0%	35.4%	46.3%	66.7%	56.5%	54.3%
Visual Matching	RT-1 (Converged)	96.0%	90.0%	71.0%	85.7%	44.2%	60.1%	86.1%	73.1%	67.7%
	RT-1-X	82.0%	33.0%	55.0%	56.7%	31.7%	29.6%	89.1%	59.4%	49.3%
	RT-2-X	74.0%	74.0%	88.0%	78.7%	77.9%	15.7%	34.3%	25.0%	60.5%
	Octo-Base	21.0%	21.0%	09.0%	17.0%	04.2%	00.9%	44.4%	22.7%	14.6%
	OpenVLA	27.0%	03.0%	19.0%	16.3%	46.2%	19.4%	51.8%	35.6%	32.7%
	TraceVLA	—	—	—	28.0%	53.7%	—	—	57.0%	46.2%
	RoboVLM	94.0%	47.0%	91.0%	77.3%	61.7%	33.3%	53.1%	43.2%	60.7%
	SpatialVLA	85.0%	76.0%	97.0%	86.0%	77.9%	50.0%	64.8%	57.4%	73.8%
	HPT	—	—	—	56.0%	60.0%	—	—	24.0%	46.7%
	$\pi_0$	76.0%	57.0%	85.1%	72.7%	65.3%	30.0%	46.6%	38.3%	58.8%
	$\pi_0$ -FAST	79.0%	61.0%	85.9%	75.3%	67.5%	34.0%	51.8%	42.9%	61.9%
	Hume	99.0%	93.0%	99.0%	97.0%	80.4%	52.0%	65.6%	58.8%	78.7%

robot state as input, and the chunk size of System 2 is 8 and the chunk size of System 1 is 4, trained using 8 GPUs with a batch size of 32.

**Agibot G-1** In real-world experiment on Agibot G-1, Hume takes images from the head camera and wrist cameras on both arms along with robot state as input. We set the chunk size of System 2’s output to 30 and the chunk size of System 1’s output to 15. The model was trained using 8 GPUs with a batch size of 8.

## 9 Experiment Details

In this section, we will provide experiment details including evaluation setup and additional test results. In Sec. 9.1, we provide detailed descriptions of test setup for standard simulation benchmark and the implementation details of all comparison methods along with the detailed test results. In Sec. 9.2, we introduce the detailed setup of test tasks on real robot platforms and testing standards, and provide the detailed test results.

### 9.1 Simulation Benchmark Details

**Simulation Benchmark Setup.** In LIBERO, all methods use third-person camera, wrist camera, and robot state as input, where the results of Diffusion Policy and OpenVLA-OFT are from the technical report of OpenVLA-OFT [23], the results of  $\pi_0$  [3] and  $\pi_0$ -FAST [41] are provided by Physical Intelligence’s open-source repository, and the results of GR00T N1 are obtained by training and testing on the LIBERO dataset using its open source code. In SimplerEnv, test results of RT-1 [5], RT-1-X [11], RT-2-X [11], Octo [40], OpenVLA [24], HPT [48], TraceVLA [56], RoboVLM [30], SpatialVLA [42] come from their official technical reports. The results of GR00T [2],  $\pi_0$ -FAST [41], and  $\pi_0$  [3] are obtained by us fine-tuning and testing them on the corresponding datasets using their official open-source code.

**Detailed Results of Simulation Benchmark.** Since the Google Robot tasks in SimplerEnv include various test settings such as environment layout, object position, and texture variations, we provide more detailed test results in Tab. 4.



**Put eggplant in the basket:** Put the eggplant in the basket.



**Put carrot on the plate:** Put the carrot on the plate.



**Close microwave:** Close the microwave.



**Lift red pepper:** Lift the red pepper.



**Put green cup on the pink cloth :** Put the green cup on the pink cloth.



**Put purple cup on the white plate :** Put the purple cup on the white plate.

Figure 10: **Evaluation Setup of WidowX 250s.** We evaluated models with 9 tasks on WidowX 250s to verify the model’s learning ability on a large multi-task manipulation dataset.

## 9.2 Real-World Evaluation Details

**Real-World Evaluation Setup.** In this section, we provide detailed descriptions of task setups on three real-world robot platforms: WidowX 250s, Franka-Panda-Emika, and Agibot G-1.

As shown in Fig. 10, the detailed task specifications on WidowX 250s are:

- **Put eggplant in the basket:** A complex task requiring the robot to identify and pick an eggplant from a sink containing multiple vegetables, then place it in a yellow basket. This task evaluates object discrimination and spatial awareness.
- **Put carrot on the plate:** The robot needs to perform a pick-and-place task by grasping a carrot from the sink and placing it on a plate, assessing both grasping precision and placement accuracy.
- **Close microwave:** The robot must close a toy microwave door positioned at various angles ( $30^\circ$ ,  $45^\circ$ ,  $60^\circ$ , and  $90^\circ$ ), testing the model’s capability to manipulate articulated objects in different configurations.
- **Lift red pepper:** A basic pick task requiring the robot to grasp and lift a red pepper from the sink, designed to evaluate the model’s object localization accuracy.

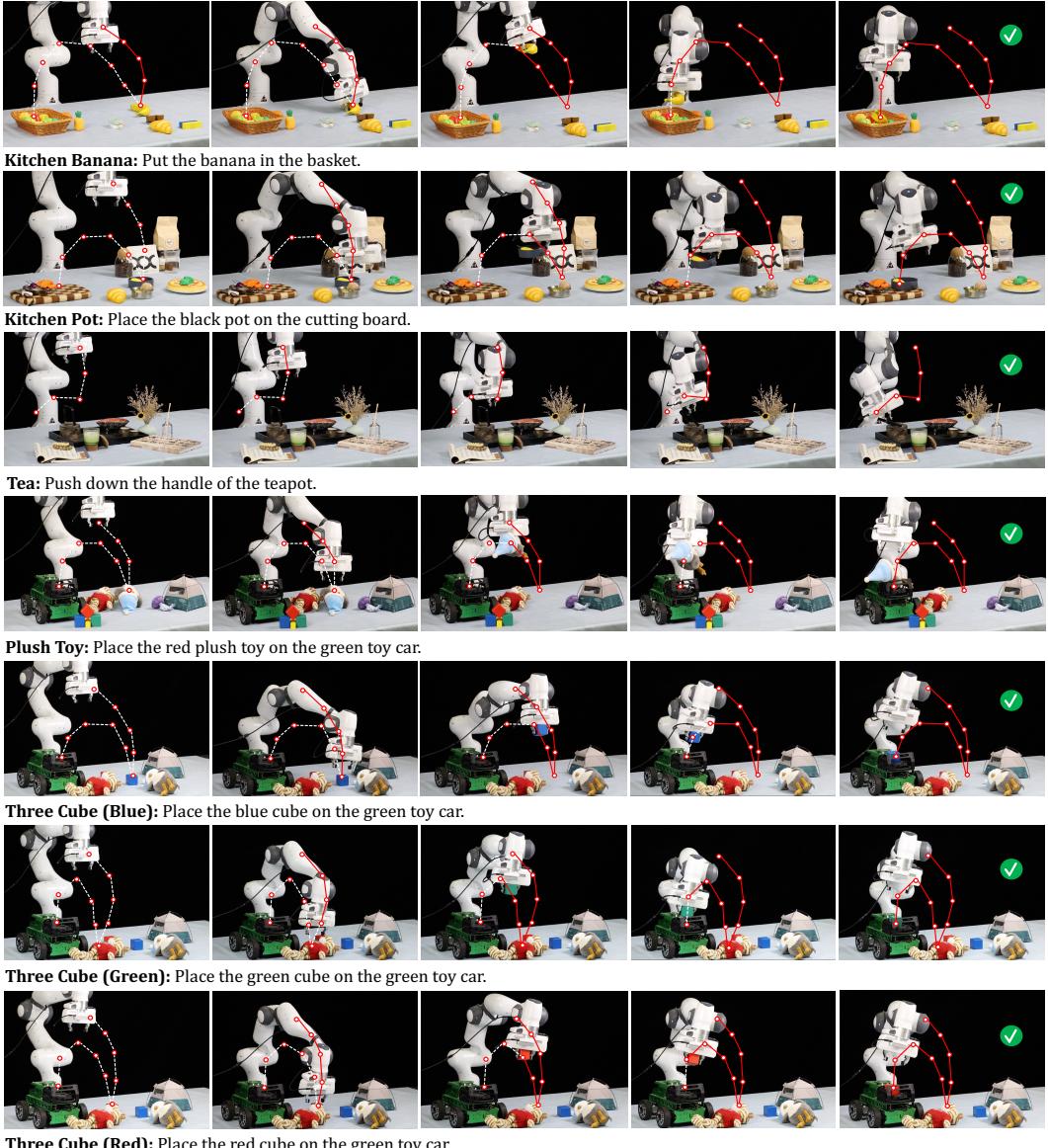


Figure 11: **Evaluation Setup of Frank-Emika-Panda.** We evaluated policies on Fnaka robot with 7 tasks, including instruction following, articulated manipulation, and pick and place tasks.

- **Put green cup on the pink cloth:** This task suite comprises two scenarios testing vertical spatial understanding. In the first scenario, the robot grasps a green cup positioned either on a stove or elevated on a yellow block. In the second scenario, the cup is placed either at the bottom of a sink or elevated on a bowl. This variation in object heights challenges the model’s ability to adapt its manipulation strategy according to spatial configurations.
- **Put purple cup on the white plate:** The robot must identify and transfer a purple cup to a white plate within a sink containing multiple plates, testing color recognition and precise manipulation.

As shown in Fig. 11, the detailed task specifications on Frank-Emila-Panda are:

- **Kitchen Banana:** A pick-and-place task where the robot must transfer a banana from the table to a basket. With only 50 human demonstrations, this task evaluates model performance with limited data.
- **Kitchen Pot:** Another pick-and-place task requiring the robot to grasp a white bowl from the right side of the table and place it on a cutting board, trained with 100 human demonstrations.



**Restock the hanging basket area:** The robot is in front of the snack shelf, with the shopping cart positioned between the snack shelf and the robot. The snacks that need to be restocked are in the box inside the shopping cart.



**Pour water:** Lift the kettle on the table with the right arm. Pour two-thirds of the water into the cup on the table with the right arm. Place the held kettle on the coaster on the table with the right arm.



**Fold Shorts:** Grasp the bottom legs and waist of the shorts with both hands and fold them over the top legs and waist. Grasp the waistband of the shorts and fold it to the leg with both arms.



**Pass the water:** Pick up the green mineral water on the table with the right arm. Pass the picked-up green mineral water to the guest with the right arm.

Figure 12: **Evaluation Setup of AgiBot G-1.** We evaluated policies on four challenging tasks on AgiBot G-1 to test ability of controlling a humanoid robot completing dexterous and long-horizon tasks.

- **Tea:** The robot must push a teapot handle from a perpendicular to a parallel position relative to the desktop, using its gripper tip. This task tests the manipulation of revolute joints and includes 50 human demonstrations.
- **Plush Toy:** The robot must identify and grasp the nearest plush toy among two options and place it on a green car. To rigorously assess spatial understanding, we systematically vary the relative positions of the plush toys during testing.
- **Three Cube:** An instruction-following task where the robot must identify and place a specifically colored cube (red, green, or blue) onto a green car. Training included 50 demonstrations for each color, totaling 150.

As shown in Fig. 12, the detailed task specifications on AgiBot G-1 are:

- **Restock the hanging basket area:** This task requires the robot to grab snacks from a cart and place them at a designated location on the shelf. This task includes different types of snacks, different placements on the shelf, and interfering objects in the cart to verify the generalization of the model.
- **Pour water:** This is a long-horizon task that requires the robot to first grasp the handle of the teapot, lift the teapot and accurately pour the water from the teapot into the cup, and then put the teapot back on the mat after the cup is full. This task involves changing the material of the cup, as well as the position of the teapot and the cup. To complete this challenging task, the robot needs to accurately identify the location of the cup and pour water into it, and it needs to correctly sense the water level in the cup to avoid spilling water on the table.
- **Fold Shorts:** This is a long-horizon task that requires the collaboration of both arms and involves the manipulation of flexible objects. In this task, the robot first needs to accurately identify the

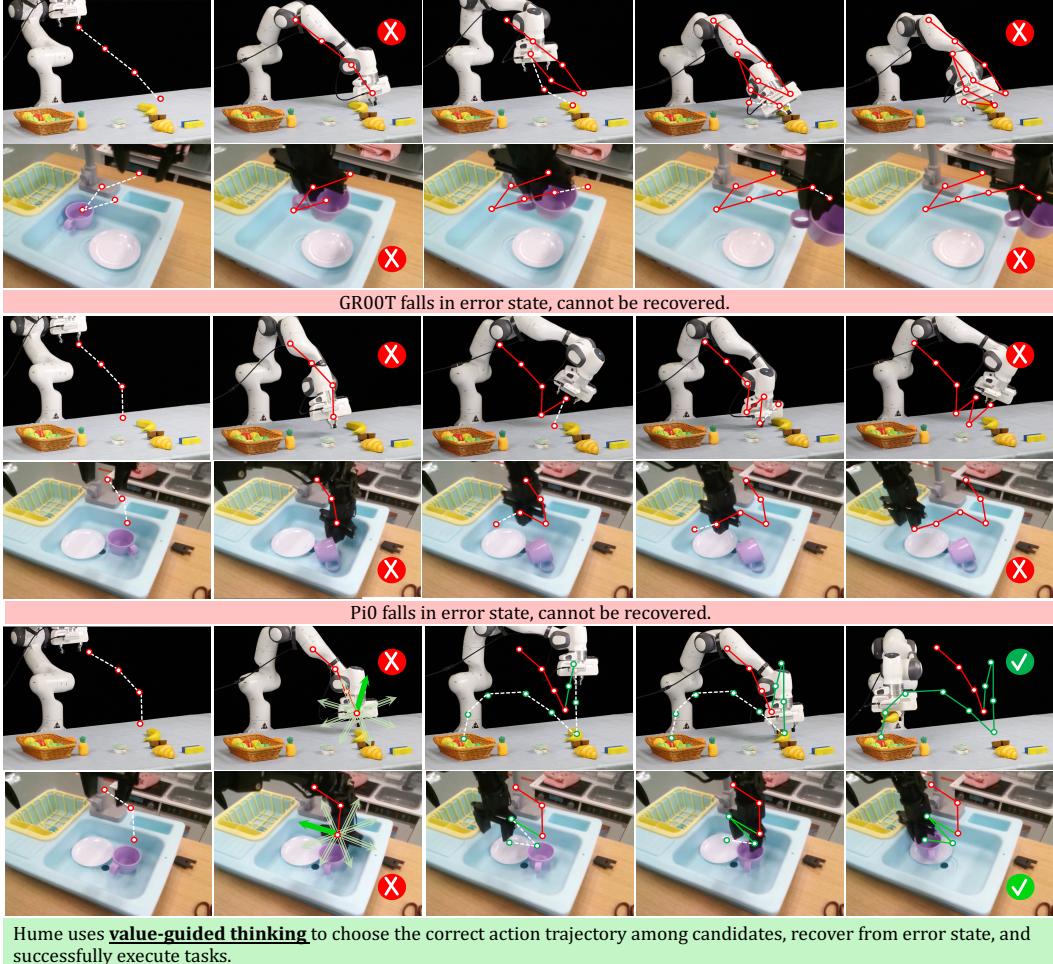


Figure 13: **Failure Recovery of Hume.** When a failure occurs, such as missing the grasping position, other policies fall into a failure state, and Hume selects the correct action through value-guided thinking to help it recover from the failure state and successfully complete the task.

position of the shorts and use the grippers on both arms to fold the shorts for the first time. After completing the first fold, the robot needs to confirm the current state of the shorts again and perform a second fold. During the testing of this task, we used shorts of different colors and materials to verify the model’s long-horizon manipulation capabilities for flexible objects of different shapes.

- **Pass the water:** This task is designed to test the model’s ability to follow instructions and collaborate with humans. In this task, the robot needs to grab the correct bottle according to the language instructions and hand it to the human. We used different types of bottles in the test, and we also arbitrarily adjusted the position of the bottles on the white table shown in the figure to verify the generalization of the robot to the position of objects.

**Detailed Results of Real-World Evaluation.** To ensure the reliability of real robot platform test results, we used standardized evaluation metrics. For simple tasks such as “Lift red pepper” on WidowX, we only tracked the overall task completion success rate. For more general tasks, such as “Kitchen Banana” on Franka, we tracked both partial and overall success rates, meaning the robot’s success rate in grasping the banana and placing it at the designated location. For more complex long-horizon tasks, such as “Pour water” on Agibot G-1, we tracked the success rate of each subtask and the overall task success rate, including the robot’s success in grasping the kettle, pouring water from the kettle into the cup, and placing the kettle on the pad. In addition, to verify the model’s generalization capability, we conducted tests under different lighting conditions, with various types of objects, different environmental layouts, and diverse language instructions. For example, in the “put green cup on the pink cloth” task on the WidowX robot, the initial position of the green cup was

significantly adjusted. In the “Fold Shorts” task on the Agibot G-1, shorts of different colors and materials were used for testing.

As shown in Fig. 13, we also observed Hume has the ability to recover from failures during evaluation. Compared to other methods, Hume can recover from failure states more often and complete the task successfully. Benefit from value-guided thinking, when the model falls into a failure state, such as missing the correct grasp position, Hume can select the correct action from a variety of candidate actions to recover from the failure and guide the robot to gradually recover from the failure state. For common imitation policies such as  $\pi_0$  and GR00T, when they enter an error state, since the observation of the error state does not appear in their training dataset, these models are easily trapped in the error state and cannot recover, resulting in the failure of the final task. For Hume, although the error state observation also does not appear in its training dataset, it can include the correct action that recovers from the error state by repeatedly sampling the candidate actions that are not completely denoised, and then select the correct candidate to guide the model to recover based on the state-action value estimated by the value-query head, thereby achieving a strong ability to recover from failures.

## References

- [1] Jie An, Zhengyuan Yang, Jianfeng Wang, Linjie Li, Zicheng Liu, Lijuan Wang, and Jiebo Luo. Bring metric functions into diffusion models. *arXiv preprint arXiv:2401.02414*, 2024.
- [2] Johan Bjorck, Fernando Castañeda, Nikita Cherniakov, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. Gr0ot n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.
- [3] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolò Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.
- [4] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.
- [5] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.
- [6] Qingwen Bu, Hongyang Li, Li Chen, Jisong Cai, Jia Zeng, Heming Cui, Maoqing Yao, and Yu Qiao. Towards synergistic, generalized, and efficient dual-system for robotic manipulation. *arXiv preprint arXiv:2410.08001*, 2024.
- [7] Bowen Chen, Liqin Liu, Chenyang Liu, Zhengxia Zou, and Zhenwei Shi. Spectral-cascaded diffusion model for remote sensing image spectral super-resolution. *IEEE Transactions on Geoscience and Remote Sensing*, 2024.
- [8] Jiefeng Chen, Jie Ren, Xinyun Chen, Chengrun Yang, Ruoxi Sun, and Sercan Ö Arik. Sets: Leveraging self-verification and self-correction for improved test-time scaling, 2025.
- [9] Xi Chen, Josip Djolonga, Piotr Padlewski, Basil Mustafa, Soravit Changpinyo, Jialin Wu, Carlos Riquelme Ruiz, Sebastian Goodman, Xiao Wang, Yi Tay, et al. Pali-x: On scaling up a multilingual vision and language model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.
- [10] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023.
- [11] Open X-Embodiment Collaboration, Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2024.
- [12] Figure. Helix: A vision-language-action model for generalist humanoid control, 2025.
- [13] Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning (ICML)*, pages 1587–1596, 2018.
- [14] Zitian Gao, Boye Niu, Xuzheng He, Haotian Xu, Hongzhang Liu, Aiwei Liu, Xuming Hu, and Lijie Wen. Interpretable contrastive monte carlo tree search reasoning, 2024.
- [15] Jiatao Gu, Shuangfei Zhai, Yizhe Zhang, Miguel Angel Bautista, and Josh Susskind. f-dm: A multi-stage diffusion model via progressive signal transformation. *arXiv preprint arXiv:2210.04955*, 2022.
- [16] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications. Technical report, 2018.
- [17] ByungOk Han, Jaehong Kim, and Jinhyeok Jang. A dual process vla: Efficient robotic manipulation leveraging vlm. In *Conference on Robot Learning (CoRL)*, 2024.
- [18] Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. Reasoning with language model is planning with world model, 2023.
- [19] Jonathan Ho, Chitwan Saharia, William Chan, David J Fleet, Mohammad Norouzi, and Tim Salimans. Cascaded diffusion models for high fidelity image generation. *arXiv preprint arXiv:2106.15282*, 2021.

- [20] Junhwa Hur, Charles Herrmann, Saurabh Saxena, Janne Kontkanen, Wei-Sheng Lai, Yichang Shih, Michael Rubinstein, David J Fleet, and Deqing Sun. High-resolution frame interpolation with patch-based cascaded diffusion. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 3868–3876, 2025.
- [21] Daniel Kahneman. Thinking, fast and slow. *Farrar, Straus and Giroux*, 2011.
- [22] Siddharth Karamcheti, Suraj Nair, Ashwin Balakrishna, Percy Liang, Thomas Kollar, and Dorsa Sadigh. Prismatic vlms: Investigating the design space of visually-conditioned language models. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2024.
- [23] Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025.
- [24] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.
- [25] Aviral Kumar, Anikait Singh, Frederik Ebert, Mitsuhiro Nakamoto, Yanlai Yang, Chelsea Finn, and Sergey Levine. Pre-training for robots: Offline rl enables learning new tasks from a handful of trials. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, 2023.
- [26] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems, 2020.
- [27] Guangyuan Li, Yongkang Wang, Junsheng Luan, Lei Zhao, Wei Xing, Huaizhong Lin, and Binkai Ou. Cascaded diffusion models for virtual try-on: Improving control and resolution. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 4689–4697, 2025.
- [28] Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, et al. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. *arXiv preprint arXiv:2411.19650*, 2024.
- [29] Xinghang Li, Peiyan Li, Minghuan Liu, Dong Wang, Jirong Liu, Bingyi Kang, Xiao Ma, Tao Kong, Hanbo Zhang, and Huaping Liu. Towards generalist robot policies: What matters in building vision-language-action models. *arXiv preprint arXiv:2412.14058*, 2024.
- [30] Xinghang Li, Peiyan Li, Minghuan Liu, Dong Wang, Jirong Liu, Bingyi Kang, Xiao Ma, Tao Kong, Hanbo Zhang, and Huaping Liu. Towards generalist robot policies: What matters in building vision-language-action models. *arXiv preprint arXiv:2412.14058*, 2024.
- [31] Xinghang Li, Minghuan Liu, Hanbo Zhang, Cunjun Yu, Jie Xu, Hongtao Wu, Chilam Cheang, Ya Jing, Weinan Zhang, Huaping Liu, et al. Vision-language foundation models as effective robot imitators. In *Proceedings of International Conference on Learning Representations (ICLR)*, 2024.
- [32] Xuanlin Li, Kyle Hsu, Jiayuan Gu, Karl Pertsch, Oier Mees, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, Sergey Levine, Jiajun Wu, Chelsea Finn, Hao Su, Quan Vuong, and Ted Xiao. Evaluating real-world robot manipulation policies in simulation. In *Proceedings of the Conference on Robot Learning (CoRL)*, 2024.
- [33] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.
- [34] Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *arXiv preprint arXiv:2306.03310*, 2023.
- [35] Tongxuan Liu, Xingyu Wang, Weizhe Huang, Wenjiang Xu, Yuting Zeng, Lei Jiang, Hailong Yang, and Jing Li. Groupdebate: Enhancing the efficiency of multi-agent debate using group discussion, 2024.
- [36] Haotian Luo, Li Shen, Haiying He, Yibo Wang, Shiwei Liu, Wei Li, Naiqiang Tan, Xiaochun Cao, and Dacheng Tao. O1-pruner: Length-harmonizing fine-tuning for o1-like reasoning pruning, 2025.
- [37] Ning Miao, Yee Why Teh, and Tom Rainforth. Selfcheck: Using llms to zero-shot check their own step-by-step reasoning. *arXiv preprint arXiv:2308.00436*, 2023.
- [38] Jananee Muralidharan and Tiju Thomas. Deliberate Problem-solving with a Large Language Model as a Brainstorm Aid Using a Checklist for Prompt Generation. *The Journal of the Association of Physicians of India*, 72(5):89–90, 2024.

- [39] Mitsuhiro Nakamoto, Yuexiang Zhai, Anikait Singh, Max Sobol Mark, Yi Ma, Chelsea Finn, Aviral Kumar, and Sergey Levine. Cal-ql: Calibrated offline rl pre-training for efficient online fine-tuning. 2023.
- [40] Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Charles Xu, Jianlan Luo, Tobias Kreiman, You Liang Tan, Lawrence Yunliang Chen, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy. In *Proceedings of Robotics: Science and Systems (RSS)*, 2024.
- [41] Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. Fast: Efficient action tokenization for vision-language-action models. *arXiv preprint arXiv:2501.09747*, 2025.
- [42] Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Yan Ding, Zhigang Wang, JiaYuan Gu, Bin Zhao, Dong Wang, and Xuelong Li. Spatialvla: Exploring spatial representations for visual-language-action model, 2025.
- [43] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Huawei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models, 2024.
- [44] Lucy Xiaoyang Shi, Brian Ichter, Michael Equi, Liyiming Ke, Karl Pertsch, Quan Vuong, James Tanner, Anna Walling, Haohuan Wang, Niccolò Fusai, Adrian Li-Bell, Danny Driess, Lachy Groom, Sergey Levine, and Chelsea Finn. Hi robot: Open-ended instruction following with hierarchical vision-language-action models, 2025.
- [45] Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning, 2023.
- [46] Xiaoshuai Song, Yanan Wu, Weixun Wang, Jiaheng Liu, Wenbo Su, and Bo Zheng. Progco: Program helps self-correction of large language models, 2025.
- [47] Gemini Robotics Team, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Travis Armstrong, Ashwin Balakrishna, Robert Baruch, Maria Bauza, Michiel Blokzijl, Steven Bohez, Konstantinos Bousmalis, Anthony Brohan, Thomas Buschmann, Arunkumar Byravan, Serkan Cabi, Ken Caluwaerts, Federico Casarini, Oscar Chang, Jose Enrique Chen, Xi Chen, Hao-Tien Lewis Chiang, Krzysztof Choromanski, David D’Ambrosio, Sudeep Dasari, Todor Davchev, Coline Devin, Norman Di Palo, Tianli Ding, Adil Dostmohamed, Danny Driess, Yilun Du, Debidatta Dwibedi, Michael Elabd, Claudio Fantacci, Cody Fong, Erik Frey, Chuyuan Fu, Marissa Giustina, Keerthana Gopalakrishnan, Laura Graesser, Leonard Hasenclever, Nicolas Heess, Brandon Hernaez, Alexander Herzog, R. Alex Hofer, Jan Humplik, Atil Iscen, Mithun George Jacob, Deepali Jain, Ryan Julian, Dmitry Kalashnikov, M. Emre Karagozler, Stefani Karp, Chase Kew, Jerad Kirkland, Sean Kirmani, Yuheng Kuang, Thomas Lampe, Antoine Laurens, Isabel Leal, Alex X. Lee, Tsang-Wei Edward Lee, Jacky Liang, Yixin Lin, Sharath Maddineni, Anirudha Majumdar, Assaf Hurwitz Michaely, Robert Moreno, Michael Neunert, Francesco Nori, Carolina Parada, Emilio Parisotto, Peter Pastor, Acorn Pooley, Kanishka Rao, Krista Reymann, Dorsa Sadigh, Stefano Saliceti, Pannag Sanketi, Pierre Sermanet, Dhruv Shah, Mohit Sharma, Kathryn Shea, Charles Shu, Vikas Sindhwani, Sumeet Singh, Radu Soricu, Jost Tobias Springenberg, Rachel Sterneck, Razvan Surdulescu, Jie Tan, Jonathan Tompson, Vincent Vanhoucke, Jake Varley, Grace Vesom, Giulia Vezzani, Oriol Vinyals, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Fei Xia, Ted Xiao, Annie Xie, Jinyu Xie, Peng Xu, Sichun Xu, Ying Xu, Zhuo Xu, Yuxiang Yang, Rui Yao, Sergey Yaroshenko, Wenhao Yu, Wentao Yuan, Jingwei Zhang, Tingnan Zhang, Allan Zhou, and Yuxiang Zhou. Gemini robotics: Bringing ai into the physical world, 2025.
- [48] Lirui Wang, Xinlei Chen, Jialiang Zhao, and Kaiming He. Scaling proprioceptive-visual learning with heterogeneous pre-trained transformers. In *Proceedings of the Conference on Neural Information Processing System (NeurIPS)*, 2024.
- [49] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, volume 35, pages 24824–24837, 2022.
- [50] Junjie Wen, Yichen Zhu, Jinming Li, Zhibin Tang, Chaomin Shen, and Feifei Feng. Dexvla: Vision-language model with plug-in diffusion expert for general robot control. *arXiv preprint arXiv:2502.05855*, 2025.
- [51] Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models, 2023.

- [52] Yuanqi Yao, Siao Liu, Haoming Song, Delin Qu, Qizhi Chen, Yan Ding, Bin Zhao, Zhigang Wang, Xuelong Li, and Dong Wang. Think small, act big: Primitive prompt learning for lifelong robot manipulation, 2025.
- [53] Michał Zawalski, William Chen, Karl Pertsch, Oier Mees, Chelsea Finn, and Sergey Levine. Robotic control via embodied chain-of-thought reasoning. *arXiv preprint arXiv:2407.08693*, 2024.
- [54] Jianke Zhang, Yanjiang Guo, Xiaoyu Chen, Yen-Jen Wang, Yucheng Hu, Chengming Shi, and Jianyu Chen. Hirt: Enhancing robotic control with hierarchical robot transformers. *arXiv preprint arXiv:2410.05273*, 2024.
- [55] Jinliang Zheng, Jianxiong Li, Dongxiu Liu, Yinan Zheng, Zhihao Wang, Zhonghong Ou, Yu Liu, Jingjing Liu, Ya-Qin Zhang, and Xianyuan Zhan. Universal actions for enhanced embodied foundation models. *arXiv preprint arXiv:2501.10105*, 2025.
- [56] Ruijie Zheng, Yongyuan Liang, Shuaiyi Huang, Jianfeng Gao, Hal Daumé III, Andrey Kolobov, Furong Huang, and Jianwei Yang. Tracevla: Visual trace prompting enhances spatial-temporal awareness for generalist robotic policies. *arXiv preprint arXiv:2412.10345*, 2024.